Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What Is AWS Cloud9?</td>
<td>1</td>
</tr>
<tr>
<td>How Does AWS Cloud9 Work?</td>
<td>1</td>
</tr>
<tr>
<td><strong>AWS Cloud9 Environments</strong></td>
<td>2</td>
</tr>
<tr>
<td>Environments and Computing Resources</td>
<td>2</td>
</tr>
<tr>
<td>What Can I Do with AWS Cloud9? (A Summary)</td>
<td>2</td>
</tr>
<tr>
<td>How Do I Get Started?</td>
<td>3</td>
</tr>
<tr>
<td>Additional Topics</td>
<td>3</td>
</tr>
<tr>
<td>What Can I Do with It?</td>
<td>3</td>
</tr>
<tr>
<td><strong>Additional Information</strong></td>
<td>4</td>
</tr>
<tr>
<td>Related Videos</td>
<td>4</td>
</tr>
<tr>
<td>Related Topics on the AWS Site</td>
<td>5</td>
</tr>
<tr>
<td>Pricing</td>
<td>5</td>
</tr>
<tr>
<td>I Have Additional Questions or Need Help</td>
<td>5</td>
</tr>
<tr>
<td><strong>Setting Up</strong></td>
<td>6</td>
</tr>
<tr>
<td>Individual User Setup</td>
<td>6</td>
</tr>
<tr>
<td>Step 1: Create an AWS Account</td>
<td>6</td>
</tr>
<tr>
<td>Step 2: Sign In to the AWS Cloud9 Console with the AWS Account Root User</td>
<td>7</td>
</tr>
<tr>
<td>Next Steps</td>
<td>7</td>
</tr>
<tr>
<td>Team Setup</td>
<td>8</td>
</tr>
<tr>
<td>Step 1: Create an AWS Account</td>
<td>8</td>
</tr>
<tr>
<td>Step 2: Create an IAM Group and User, and Add the User to the Group</td>
<td>9</td>
</tr>
<tr>
<td>Step 3: Add AWS Cloud9 Access Permissions to the Group</td>
<td>12</td>
</tr>
<tr>
<td>Step 4: Sign In to the AWS Cloud9 Console</td>
<td>15</td>
</tr>
<tr>
<td>Next Steps</td>
<td>15</td>
</tr>
<tr>
<td>Enterprise Setup</td>
<td>15</td>
</tr>
<tr>
<td>Step 1: Create a Master Account for the Organization</td>
<td>17</td>
</tr>
<tr>
<td>Step 2: Create an Organization for the Master Account</td>
<td>18</td>
</tr>
<tr>
<td>Step 3: Add Member Accounts to the Organization</td>
<td>18</td>
</tr>
<tr>
<td>Step 4: Enable AWS SSO across the Organization</td>
<td>18</td>
</tr>
<tr>
<td>Step 5: Set Up Groups and Users within the Organization</td>
<td>19</td>
</tr>
<tr>
<td>Step 6: Enable Groups and Users within the Organization to Use AWS Cloud9</td>
<td>20</td>
</tr>
<tr>
<td>Step 7: Start Using AWS Cloud9</td>
<td>21</td>
</tr>
<tr>
<td>Next Steps</td>
<td>22</td>
</tr>
<tr>
<td>Additional Setup Options (Team and Enterprise)</td>
<td>22</td>
</tr>
<tr>
<td>Step 1: Create a Customer-Managed Policy</td>
<td>22</td>
</tr>
<tr>
<td>Step 2: Add Customer-Managed Policies to a Group</td>
<td>23</td>
</tr>
<tr>
<td>Customer-Managed Policy Examples for Teams Using AWS Cloud9</td>
<td>24</td>
</tr>
<tr>
<td>Next Steps</td>
<td>28</td>
</tr>
<tr>
<td>Individual student setup</td>
<td>28</td>
</tr>
<tr>
<td>Step 1: Create an AWS Educate Starter Account</td>
<td>28</td>
</tr>
<tr>
<td>Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9</td>
<td>29</td>
</tr>
<tr>
<td>Next steps</td>
<td>30</td>
</tr>
<tr>
<td>Classroom Setup</td>
<td>30</td>
</tr>
<tr>
<td>Step 1: Create an AWS Account and Apply for AWS Educate</td>
<td>30</td>
</tr>
<tr>
<td>Step 2: Set Up Your Virtual Classroom for AWS Cloud9</td>
<td>32</td>
</tr>
<tr>
<td>Step 3: Enter Your Virtual Classroom</td>
<td>32</td>
</tr>
<tr>
<td>Next Steps</td>
<td>34</td>
</tr>
<tr>
<td>Getting Started: Basic Tutorials</td>
<td>35</td>
</tr>
<tr>
<td>Hello AWS Cloud9 (console)</td>
<td>35</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>35</td>
</tr>
<tr>
<td>Steps</td>
<td>35</td>
</tr>
<tr>
<td>Step 1: Create an Environment</td>
<td>35</td>
</tr>
<tr>
<td>Step 2: Basic Tour</td>
<td>38</td>
</tr>
<tr>
<td>Step 3: Clean Up</td>
<td>43</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Step 9: Outline Window</td>
<td>108</td>
</tr>
<tr>
<td>Step 10: Go Window</td>
<td>109</td>
</tr>
<tr>
<td>Step 11: Immediate Tab</td>
<td>111</td>
</tr>
<tr>
<td>Step 12: Process List</td>
<td>112</td>
</tr>
<tr>
<td>Step 13: Preferences</td>
<td>113</td>
</tr>
<tr>
<td>Step 14: Terminal</td>
<td>114</td>
</tr>
<tr>
<td>Step 15: Debugger Window</td>
<td>115</td>
</tr>
<tr>
<td>Final Thoughts</td>
<td>120</td>
</tr>
<tr>
<td>Language Support</td>
<td>120</td>
</tr>
<tr>
<td>Menu Commands Reference</td>
<td>122</td>
</tr>
<tr>
<td>AWS Cloud9 Menu</td>
<td>122</td>
</tr>
<tr>
<td>File Menu</td>
<td>123</td>
</tr>
<tr>
<td>Edit Menu</td>
<td>124</td>
</tr>
<tr>
<td>Find Menu</td>
<td>126</td>
</tr>
<tr>
<td>View Menu</td>
<td>126</td>
</tr>
<tr>
<td>Go Menu</td>
<td>127</td>
</tr>
<tr>
<td>Run Menu</td>
<td>128</td>
</tr>
<tr>
<td>Tools Menu</td>
<td>128</td>
</tr>
<tr>
<td>Window Menu</td>
<td>129</td>
</tr>
<tr>
<td>Support Menu</td>
<td>130</td>
</tr>
<tr>
<td>Preview Menu</td>
<td>131</td>
</tr>
<tr>
<td>Other Menu Bar Commands</td>
<td>131</td>
</tr>
<tr>
<td>Finding and Replacing Text</td>
<td>131</td>
</tr>
<tr>
<td>Find Text in a Single File</td>
<td>132</td>
</tr>
<tr>
<td>Replace Text in a Single File</td>
<td>132</td>
</tr>
<tr>
<td>Find Text in Multiple Files</td>
<td>132</td>
</tr>
<tr>
<td>Replace Text in Multiple Files</td>
<td>133</td>
</tr>
<tr>
<td>Find and Replace Options</td>
<td>134</td>
</tr>
<tr>
<td>Previewing Files</td>
<td>134</td>
</tr>
<tr>
<td>Open a File for Preview</td>
<td>135</td>
</tr>
<tr>
<td>Reload a File Preview</td>
<td>135</td>
</tr>
<tr>
<td>Change the File Preview Type</td>
<td>135</td>
</tr>
<tr>
<td>Open a File Preview in a Separate Web Browser Tab</td>
<td>136</td>
</tr>
<tr>
<td>Switch to a Different File Preview</td>
<td>136</td>
</tr>
<tr>
<td>Previewing Running Applications</td>
<td>136</td>
</tr>
<tr>
<td>Run an Application</td>
<td>136</td>
</tr>
<tr>
<td>Preview a Running Application</td>
<td>138</td>
</tr>
<tr>
<td>Reload an Application Preview</td>
<td>138</td>
</tr>
<tr>
<td>Change the Application Preview Type</td>
<td>138</td>
</tr>
<tr>
<td>Open an Application Preview in a Separate Web Browser Tab</td>
<td>139</td>
</tr>
<tr>
<td>Switch to a Different Preview URL</td>
<td>139</td>
</tr>
<tr>
<td>Share a Running Application over the Internet</td>
<td>139</td>
</tr>
<tr>
<td>Working with File Revisions</td>
<td>143</td>
</tr>
<tr>
<td>Working with Image Files</td>
<td>144</td>
</tr>
<tr>
<td>View or Edit an Image</td>
<td>144</td>
</tr>
<tr>
<td>Resize an Image</td>
<td>144</td>
</tr>
<tr>
<td>Crop an Image</td>
<td>145</td>
</tr>
<tr>
<td>Rotate an Image</td>
<td>145</td>
</tr>
<tr>
<td>Flip an Image</td>
<td>145</td>
</tr>
<tr>
<td>Zoom an Image</td>
<td>145</td>
</tr>
<tr>
<td>Smooth an Image</td>
<td>146</td>
</tr>
<tr>
<td>Working with Builders, Runners, and Debuggers</td>
<td>146</td>
</tr>
<tr>
<td>Built-In Build, Run, and Debug Support</td>
<td>146</td>
</tr>
<tr>
<td>Build Your Project's Files</td>
<td>146</td>
</tr>
<tr>
<td>Run Your Code</td>
<td>147</td>
</tr>
<tr>
<td>Debug Your Code</td>
<td>147</td>
</tr>
<tr>
<td>Change a Built-In Runner</td>
<td>150</td>
</tr>
<tr>
<td>Sample</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Java Sample</td>
<td>400</td>
</tr>
<tr>
<td>LAMP Sample</td>
<td>383</td>
</tr>
<tr>
<td>AWS CDK Sample</td>
<td>377</td>
</tr>
<tr>
<td>GitHub Sample</td>
<td>361</td>
</tr>
<tr>
<td>AWS Cloud9 User Guide</td>
<td></td>
</tr>
<tr>
<td>Step 1: Install Required Tools</td>
<td>391</td>
</tr>
<tr>
<td>Step 2: Set Up the Tools</td>
<td>392</td>
</tr>
<tr>
<td>Step 3: Set Up a Website</td>
<td>393</td>
</tr>
<tr>
<td>Step 4: Clean Up</td>
<td>394</td>
</tr>
<tr>
<td>Step 5: Add Files to the Repository</td>
<td>395</td>
</tr>
<tr>
<td>Step 6: Clean Up</td>
<td>396</td>
</tr>
<tr>
<td>Step 7: Clean Up</td>
<td>397</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>398</td>
</tr>
<tr>
<td>Step 1: Install the Required Tools</td>
<td>399</td>
</tr>
<tr>
<td>Step 2: Set Up MySQL</td>
<td>400</td>
</tr>
<tr>
<td>Step 3: Set Up the WordPress Website</td>
<td>401</td>
</tr>
<tr>
<td>Step 4: Share the WordPress Website Over the Internet</td>
<td>402</td>
</tr>
<tr>
<td>Step 5: Clean Up</td>
<td>403</td>
</tr>
<tr>
<td>Step 6: Clean Up</td>
<td>404</td>
</tr>
<tr>
<td>Step 7: Clean Up</td>
<td>405</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>406</td>
</tr>
<tr>
<td>Step 1: Install Required Tools</td>
<td>407</td>
</tr>
<tr>
<td>Step 2: Add Code</td>
<td>408</td>
</tr>
<tr>
<td>Step 3: Run the Code</td>
<td>409</td>
</tr>
<tr>
<td>Step 4: Clean Up</td>
<td>410</td>
</tr>
<tr>
<td>Step 5: Add Files to the Repository</td>
<td>411</td>
</tr>
<tr>
<td>Step 6: Clean Up</td>
<td>412</td>
</tr>
<tr>
<td>Step 7: Clean Up</td>
<td>413</td>
</tr>
<tr>
<td>Prerequisites</td>
<td>414</td>
</tr>
<tr>
<td>Step 1: Install Required Tools</td>
<td>415</td>
</tr>
<tr>
<td>Step 2: Add Code</td>
<td>416</td>
</tr>
<tr>
<td>Step 3: Run the Code</td>
<td>417</td>
</tr>
<tr>
<td>Step 4: Install and Configure the AWS SDK for C++</td>
<td>418</td>
</tr>
</tbody>
</table>
AWS Cloud9 User Guide

Step 5: Add AWS SDK Code ................................................................................................. 450
Step 6: Run the AWS SDK Code .......................................................................................... 452
Step 7: Clean Up .................................................................................................................. 452

TypeScript Sample .................................................................................................................. 452
Prerequisites .......................................................................................................................... 453
Step 1: Install Required Tools .............................................................................................. 453
Step 2: Add Code .................................................................................................................... 454
Step 3: Run the Code .............................................................................................................. 455
Step 4: Install and Configure the AWS SDK for JavaScript in Node.js ......................... 455
Step 5: Add AWS SDK Code .................................................................................................. 456
Step 6: Run the AWS SDK Code .......................................................................................... 458
Step 7: Clean Up .................................................................................................................. 458

Docker Sample ....................................................................................................................... 458
Prerequisites .......................................................................................................................... 459
Step 1: Install and Run Docker .............................................................................................. 459
Step 2: Build the Image .......................................................................................................... 460
Step 3: Run the Container ....................................................................................................... 462
Step 4: Create the Environment ............................................................................................. 463
Step 5: Run the Code .............................................................................................................. 464
Step 6: Clean Up .................................................................................................................. 464

Related Samples .................................................................................................................. 465

Advanced Topics .................................................................................................................. 466
EC2 Environments versus SSH Environments ........................................................................ 466
Amazon VPC Settings ............................................................................................................. 467
Amazon VPC Requirements for AWS Cloud9 ........................................................................ 467
Create an Amazon VPC for AWS Cloud9 ............................................................................. 477
Create a Subnet for AWS Cloud9 .......................................................................................... 478
SSH Environment Host Requirements .................................................................................. 479
When and how to create an SSH Environment ..................................................................... 479
SSH host requirements ......................................................................................................... 480
AWS Cloud9 Installer ............................................................................................................. 481
Download and Run the AWS Cloud9 Installer ..................................................................... 482
Troubleshooting the AWS Cloud9 Installer .......................................................................... 482
Inbound SSH IP address ranges ............................................................................................ 483
IP addresses not in ip-ranges.json ...................................................................................... 484

AMI Contents ........................................................................................................................ 485
Amazon Linux ....................................................................................................................... 485
Ubuntu Server ....................................................................................................................... 486

Authentication and Access Control ....................................................................................... 486
Overview ............................................................................................................................... 487
AWS Managed (Predefined) Policies for AWS Cloud9 ......................................................... 490
Creating Customer-Managed Policies for AWS Cloud9 ....................................................... 493
AWS Managed Temporary Credentials ................................................................................. 501

Using Service-Linked Roles ................................................................................................. 504
Service-Linked Role Permissions for AWS Cloud9 ............................................................... 505
Creating a Service-Linked Role for AWS Cloud9 ............................................................... 506
Editing a Service-Linked Role for AWS Cloud9 .................................................................. 506
Deleting a Service-Linked Role for AWS Cloud9 ............................................................... 507
Supported Regions for AWS Cloud9 Service-Linked Roles ................................................. 507

Logging API Calls with CloudTrail ....................................................................................... 507
AWS Cloud9 Information in CloudTrail .............................................................................. 507
Understanding AWS Cloud9 Log File Entries ..................................................................... 508

Tags ...................................................................................................................................... 516
Propagating Tag Updates to Underlying Resources ............................................................. 516

Security .................................................................................................................................. 518
Data protection ...................................................................................................................... 518
Data encryption ..................................................................................................................... 519
What Is AWS Cloud9?

AWS Cloud9 is an integrated development environment, or IDE.

The AWS Cloud9 IDE offers a rich code-editing experience with support for several programming languages and runtime debuggers, and a built-in terminal. It contains a collection of tools that you use to code, build, run, test, and debug software, and helps you release software to the cloud.

You access the AWS Cloud9 IDE through a web browser. You can configure the IDE to your preferences. You can switch color themes, bind shortcut keys, enable programming language-specific syntax coloring and code formatting, and more.

(Got it! I'm ready to try AWS Cloud9. How Do I Get Started? (p. 3))

How Does AWS Cloud9 Work?

The following diagram shows a high-level overview of how AWS Cloud9 works.

From the diagram (starting at the bottom), you use the AWS Cloud9 IDE, running in a web browser on your local computer, to interact with your AWS Cloud9 environment. A computing resource (for example, an Amazon EC2 instance or your own server) connects to that environment. Finally, your work is stored in an AWS CodeCommit repository or other type of remote repository.
AWS Cloud9 Environments

An AWS Cloud9 environment is a place where you store your project's files and where you run the tools to develop your applications.

Using the AWS Cloud9 IDE, you can:

- Store your project's files locally on the instance or server.
- Clone a remote code repository—such as a repo in AWS CodeCommit—into your environment.
- Work with a combination of local and cloned files in the environment.

You can create and switch between multiple environments, with each environment set up for a specific development project. By storing the environment in the cloud, your projects no longer need to be tied to a single computer or server setup. This enables you to do things such as easily switch between computers and more quickly onboard developers to your team.

Environments and Computing Resources

Behind the scenes, there are a couple of ways you can connect your environments to computing resources:

- You can instruct AWS Cloud9 to create an Amazon EC2 instance, and then connect the environment to that newly created EC2 instance. This type of setup is called an EC2 environment.
- You can instruct AWS Cloud9 to connect an environment to an existing cloud compute instance or to your own server. This type of setup is called an SSH environment.

EC2 environments and SSH environments have some similarities and some differences. If you're new to AWS Cloud9, we recommend that you use an EC2 environment because AWS Cloud9 takes care of much of the configuration for you. As you learn more about AWS Cloud9, and want to understand these similarities and differences better, see EC2 Environments versus SSH Environments in AWS Cloud9 (p. 466).

For more information about how AWS Cloud9 works, see these related videos (p. 4) and webpages (p. 5).

What Can I Do with AWS Cloud9? (A Summary)

With AWS Cloud9, you can code, build, run, test, debug, and release software in many exciting scenarios and variations. These include (but are not limited to):

- Working with code in several programming languages and the AWS Cloud Development Kit (AWS CDK).
- Working with code in a running Docker container.
- Using online code repositories.
- Collaborating with others in real time.
- Interacting with various database and website technologies.
- Targeting AWS Lambda, Amazon API Gateway, and AWS Serverless Applications.
- Taking advantage of other AWS products such as Amazon Lightsail, AWS CodeStar, and AWS CodePipeline.

For a more detailed list, see What Can I Do with AWS Cloud9? (p. 3)
How Do I Get Started?

To start using AWS Cloud9, follow the steps in Setting Up AWS Cloud9 (p. 6), and then go through the basic tutorial (p. 35).

Additional Topics

- What Can I Do with AWS Cloud9? (p. 3)
- Additional Information about AWS Cloud9 (p. 4)

What Can I Do with AWS Cloud9?

Explore the following resources to learn about using AWS Cloud9 for some common scenarios.

Key Scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create, run, and debug code in AWS Lambda functions, APIs in Amazon API Gateway, and serverless applications.</td>
<td>AWS Lambda Tutorial for AWS Cloud9 (p. 325), Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342), and Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 284)</td>
</tr>
<tr>
<td>Work with Amazon Lightsail instances preconfigured with popular applications and frameworks such as WordPress, LAMP (Linux, Apache, MySQL, and PHP), Node.js, Nginx, Drupal, and Joomla, and Linux distributions such as Amazon Linux, Ubuntu, Debian, FreeBSD, and openSUSE.</td>
<td>Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)</td>
</tr>
<tr>
<td>Automate AWS services by using the AWS CLI and the aws-shell.</td>
<td>AWS Command Line Interface and aws-shell Sample for AWS Cloud9 (p. 352)</td>
</tr>
<tr>
<td>Work with source code repositories in AWS CodeCommit.</td>
<td>AWS CodeCommit Sample for AWS Cloud9 (p. 356)</td>
</tr>
<tr>
<td>Work with source code repositories in GitHub.</td>
<td>GitHub Sample for AWS Cloud9 (p. 361)</td>
</tr>
<tr>
<td>Work with NoSQL databases in Amazon DynamoDB.</td>
<td>Amazon DynamoDB Sample for AWS Cloud9 (p. 366)</td>
</tr>
<tr>
<td>Work with LAMP (Linux, Apache HTTP Server, MySQL, and PHP) stacks.</td>
<td>LAMP Sample for AWS Cloud9 (p. 383)</td>
</tr>
</tbody>
</table>
### Scenario | Resources
---|---
Work with code for Java and the AWS SDK for Java. | Java Sample for AWS Cloud9 (p. 400)
Work with code for C++ and the AWS SDK for C++. | C++ Sample for AWS Cloud9 (p. 410)
Work with code for Python and the AWS SDK for Python (Boto). | Python Tutorial for AWS Cloud9 (p. 418)
Work with code for .NET Core and the AWS SDK for .NET. | .NET Core Sample for AWS Cloud9 (p. 423)
Work with code for Node.js and the AWS SDK for JavaScript. | Node.js Sample for AWS Cloud9 (p. 430)
Work with code for PHP and the AWS SDK for PHP. | PHP Sample for AWS Cloud9 (p. 435)
Work with code for Ruby and the AWS SDK for Ruby. | Tutorial: Ruby in AWS Cloud9 (p. 440)
Work with code for Go and the AWS SDK for Go. | Go Sample for AWS Cloud9 (p. 446)
Work with code for TypeScript and the AWS SDK for JavaScript. | TypeScript Sample for AWS Cloud9 (p. 452)
Work with code for the AWS Cloud Development Kit (AWS CDK). | AWS CDK Sample for AWS Cloud9 (p. 377)
Work with code in a running Docker container. | Docker Sample for AWS Cloud9 (p. 458)
Invite others to use an environment with you, in real time and with text chat support. | Working with Shared Environments in AWS Cloud9 (p. 80)
Work with code for intelligent robotics applications in AWS RoboMaker. | Developing with AWS Cloud9 in the AWS RoboMaker Developer Guide

### Additional Information about AWS Cloud9

This topic provides more information to help you learn about AWS Cloud9.

**Topics**
- Related Videos (p. 4)
- Related Topics on the AWS Site (p. 5)
- Pricing (p. 5)
- I Have Additional Questions or Need Help (p. 5)

**Related Videos**
- AWS re:Invent 2017 - Introducing AWS Cloud9: Werner Vogels Keynote (9 minutes, YouTube website)
- AWS re:Invent Launchpad 2017 - AWS Cloud9, (15 minutes, YouTube website)
- Introducing AWS Cloud9 - AWS Online Tech Talks (33 minutes, YouTube website)
Pricing

There is no additional charge for AWS Cloud9. If you use an Amazon EC2 instance for your AWS Cloud9 development environment, you pay only for the compute and storage resources (for example, an Amazon EC2 instance, an Amazon EBS volume) that are used to run and store your code. You can also connect your environment to an existing Linux server (for example, an on-premises server) through SSH for no additional charge.

You only pay for what you use, as you use it; there are no minimum fees and no upfront commitments. You are charged the normal AWS rates for any AWS resources (for example, AWS Lambda functions) that you create or use within your environment.

New AWS customers who are eligible for the AWS Free Tier can use AWS Cloud9 for free. If your environment makes use of resources beyond the AWS Free Tier, you are charged the normal AWS rates for those resources.

For more information, see the following.

- **AWS Cloud9 pricing:** See [AWS Cloud9 Pricing](#).
- **AWS service pricing:** See [Amazon EC2 Pricing](#), [Amazon EBS Pricing](#), [AWS Lambda Pricing](#), and [AWS Pricing](#).
- **The AWS Free Tier:** See [Using the AWS Free Tier](#) and [Tracking Your Free Tier Usage](#) in the [AWS Billing and Cost Management User Guide](#).
- **Educational pricing:** See the [AWS Educate](#) program.

I Have Additional Questions or Need Help

To ask questions or seek help from the AWS Cloud9 community, see the [AWS Cloud9 Discussion Forum](#). (When you enter this forum, AWS might require you to sign in.)

See also our frequently asked questions (FAQs), or contact us directly.
Setting Up AWS Cloud9

To start using AWS Cloud9, follow one of these sets of procedures, depending on how you plan to use AWS Cloud9.

<table>
<thead>
<tr>
<th>Usage pattern</th>
<th>Follow these procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am the only <strong>individual</strong> using my AWS account, and I am not a student.</td>
<td>Individual User Setup (p. 6)</td>
</tr>
<tr>
<td>I belong to a <strong>team</strong> that has multiple users within a single AWS account.</td>
<td>Team Setup (p. 8)</td>
</tr>
<tr>
<td>I belong to an <strong>enterprise</strong> that has one or more AWS accounts within a single organization.</td>
<td>Enterprise Setup (p. 15)</td>
</tr>
<tr>
<td>I am a <strong>student</strong>¹, and I am the only individual using my account with AWS.</td>
<td>Individual Student Setup (p. 28)</td>
</tr>
<tr>
<td>I am an <strong>educator</strong>¹, and I want my students to work together in a virtual classroom.</td>
<td>Classroom Setup (p. 30)</td>
</tr>
</tbody>
</table>

¹ To learn about who qualifies as a student or educator, see Who can join AWS Educate on the AWS Educate Frequently Asked Questions website.

For general information about AWS Cloud9, see What Is AWS Cloud9? (p. 1).

Topics
- Individual User Setup for AWS Cloud9 (p. 6)
- Team Setup for AWS Cloud9 (p. 8)
- Enterprise Setup for AWS Cloud9 (p. 15)
- Additional Setup Options for AWS Cloud9 (Team and Enterprise) (p. 22)
- Individual student setup for AWS Cloud9 (p. 28)
- Classroom Setup for AWS Cloud9 (p. 30)

Individual User Setup for AWS Cloud9

This topic explains how to set up to use AWS Cloud9 as the only individual in your AWS account, and you are not a student. To set up to use AWS Cloud9 for any other usage pattern, see Setting Up AWS Cloud9 (p. 6) for the correct instructions. To learn about who qualifies as a student, see Who can join AWS Educate on the AWS Educate Frequently Asked Questions website.

To use AWS Cloud9 as the only individual in your AWS account, create an AWS account if you don’t already have one, and then sign in to the AWS Cloud9 console.

Step 1: Create an AWS Account

If you already have an AWS account, skip ahead to Step 2: Sign in to the AWS Cloud9 Console with the AWS Account Root User (p. 7).

To watch a 4-minute video related to the following procedure, see Creating an Amazon Web Services Account on the YouTube website.
To create an AWS account

2. Choose Sign In to the Console.
3. Choose Create a new AWS account.
4. Complete the process by following the on-screen directions. This includes giving AWS your email address and credit card information. You must also use your phone to enter a code that AWS gives you.

After you finish creating the account, AWS will send you a confirmation email. Do not go to the next step until you get this confirmation.

Step 2: Sign In to the AWS Cloud9 Console with the AWS Account Root User

After you complete the previous step, you're ready to sign in to the AWS Cloud9 console with an AWS account root user and start using AWS Cloud9.

2. Enter the email address for your AWS account, and then choose Next.
   
   Note
   
   If an email address is already displayed and it’s the wrong one, choose Sign in to a different account. Enter the correct email address, and then choose Next.
3. Enter the password for your AWS account, and then choose Sign In.

The AWS Cloud9 console is displayed, and you can now start using AWS Cloud9.

Important
Although you can sign in to the AWS Cloud9 console with the email address and password that you used when you created your AWS account (we call this an AWS account root user), this isn’t an AWS security best practice. In the future, we recommend that you sign in as an administrator user in AWS Identity and Access Management (IAM) in your AWS account instead. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide and AWS Tasks That Require AWS Account Root User Credentials in the Amazon Web Services General Reference.

Next Steps

<table>
<thead>
<tr>
<th>Task for learning</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>More advanced task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>
Team Setup for AWS Cloud9

This topic explains how to use AWS Identity and Access Management (IAM) to enable multiple users within a single AWS account to use AWS Cloud9. To set up to use AWS Cloud9 for any other usage pattern, see Setting Up AWS Cloud9 (p. 6) for the correct instructions.

These instructions assume that you have (or will have) administrative access to a single AWS account. For more information, see The AWS Account Root User and Creating Your First IAM Admin User and Group in the IAM User Guide. If you already have an AWS account but you do not have administrative access to it, see your AWS account administrator.

Note
You can use AWS Single Sign-On (SSO) instead of IAM to enable multiple users within a single AWS account to use AWS Cloud9. In this usage pattern, the single AWS account serves as the master account for an organization in AWS Organizations, and that organization has no member accounts. To use AWS SSO, skip this topic and follow the instructions in Enterprise Setup (p. 15) instead. For related information, see the following resources:

- What Is AWS Organizations in the AWS Organization User Guide (AWS SSO requires the use of AWS Organizations)
- The 4-minute video AWS Knowledge Center Videos: How do I get started with AWS Organizations on the YouTube website
- The 7-minute video Manage User Access to Multiple AWS Accounts Using AWS Single Sign-on on the YouTube website
- The 9-minute video How to Setup AWS Single Sign On for Your On-Premise Active Directory Users on the YouTube website

To enable multiple users in a single AWS account to start using AWS Cloud9, start with one of the following steps, depending on which AWS resources you already have.

<table>
<thead>
<tr>
<th>Do you have an AWS account?</th>
<th>Do you have at least one IAM group and user in that account?</th>
<th>Start with this step</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>—</td>
<td>Step 1: Create an AWS Account (p. 8)</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Step 2: Create an IAM Group and User, and Add the User to the Group (p. 9)</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Step 3: Add AWS Cloud9 Access Permissions to the Group (p. 12)</td>
</tr>
</tbody>
</table>

Step 1: Create an AWS Account

Note
Your organization might already have an AWS account set up for you. If your organization has an AWS account administrator, check with that person before starting the following procedure. If you already have an AWS account, skip ahead to Step 2: Create an IAM Group and User, and Add the User to the Group (p. 9).

To watch a 4-minute video related to the following procedure, see Creating an Amazon Web Services Account on the YouTube website.
To create an AWS account

2. Choose Sign In to the Console.
3. Choose Create a new AWS account.
4. Complete the process by following the on-screen directions. This includes giving AWS your email address and credit card information. You must also use your phone to enter a code that AWS gives you.

After you finish creating the account, AWS will send you a confirmation email. Do not go to the next step until you get this confirmation.

Step 2: Create an IAM Group and User, and Add the User to the Group

In this step, you create a group and a user in AWS Identity and Access Management (IAM), add the user to the group, and then use the user to access AWS Cloud9. This is an AWS security best practice. For more information, see IAM Best Practices in the IAM User Guide.

If you already have all of the IAM groups and users that you need, skip ahead to Step 3: Add AWS Cloud9 Access Permissions to the Group (p. 12).

Note

Your organization might already have an IAM group and user set up for you. If your organization has an AWS account administrator, check with that person before starting the following procedures.

You can complete these tasks using the AWS Management Console (p. 9) or the AWS Command Line Interface (AWS CLI) (p. 10).

To watch a 9-minute video related to the following console procedures, see How do I set up an IAM user and sign in to the AWS Management Console using IAM credentials on the YouTube website.

Step 2.1: Create an IAM Group with the Console

1. Sign in to the AWS Management Console, if you are not already signed in, at https://console.aws.amazon.com.

   Note
   Although you can sign in to the AWS Management Console with the email address and password that was provided when the AWS account was created (we call this an AWS account root user), this isn’t an AWS security best practice. In the future, we recommend you sign in using credentials for an IAM administrator user in the AWS account. An IAM administrator user has similar AWS access permissions to an AWS account root user and avoids some of the associated security risks. If you cannot sign in as an IAM administrator user, check with your AWS account administrator. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.

2. Open the IAM console. To do this, in the AWS navigation bar, choose Services. Then choose IAM.
3. In the IAM console’s navigation pane, choose Groups.
5. On the Set Group Name page, for Group Name, enter a name for the new group.
6. Choose Next Step.
7. On the Attach Policy page, choose Next Step without attaching any policies. (You will attach a policy in Step 3: Add AWS Cloud9 Access Permissions to the Group (p. 12).)
8. Choose Create Group.
Note
We recommend that you repeat this procedure to create at least two groups: one group for AWS Cloud9 users, and another group for AWS Cloud9 administrators. This AWS security best practice can help you better control, track, and troubleshoot issues with AWS resource access.

Skip ahead to Step 2.2: Create an IAM User and Add the User to the Group with the AWS CLI (p. 11).

Step 2.1: Create an IAM Group with the AWS CLI

Note
If you're using AWS managed temporary credentials (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

1. Install and configure the AWS CLI on your computer, if you haven't done so already. To do this, see the following in the AWS Command Line Interface User Guide:
   - Installing the AWS Command Line Interface
   - Quick Configuration

   Note
   Although you can configure the AWS CLI using the credentials associated with the email address and password that was provided when the AWS account was created (we call this an AWS account root user), this isn't an AWS security best practice. Instead, we recommend you configure the AWS CLI using credentials for an IAM administrator user in the AWS account. An IAM administrator user has similar AWS access permissions to an AWS account root user and avoids some of the associated security risks. If you cannot configure the AWS CLI as an IAM administrator user, check with your AWS account administrator. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.

2. Run the IAM `create-group` command, specifying the new group's name (for example, `MyCloud9Group`).
   
   ```
   aws iam create-group --group-name MyCloud9Group
   ```

   Note
   We recommend that you repeat this procedure to create at least two groups: one group for AWS Cloud9 users, and another group for AWS Cloud9 administrators. This AWS security best practice can help you better control, track, and troubleshoot issues with AWS resource access.

Skip ahead to Step 2.2: Create an IAM User and Add the User to the Group with the AWS CLI (p. 11).

Step 2.2: Create an IAM User and Add the User to the Group with the Console

1. With the IAM console open from the previous procedure, in the navigation pane, choose Users.
2. Choose Add user.
3. For User name, enter a name for the new user.

   Note
   You can create multiple users at the same time by choosing Add another user. The other settings in this procedure apply to each of these new users.

4. Select the Programmatic access and AWS Management Console access check boxes. This allows the new user to use various AWS developer tools and service consoles.
5. Leave the default choice of **Autogenerated password**. This creates a random password for the new user to sign in to the console. Or choose **Custom password** and enter a specific password for the new user.

6. Leave the default choice of **Require password reset**. This prompts the new user to change their password after they sign in to the console for the first time.

7. Choose **Next: Permissions**.

8. Leave the default choice of **Add user to group** (or **Add users to group** for multiple users).

9. In the list of groups, select the check box (not the name) next to the group you want to add the user to.

10. Choose **Next: Review**.

11. Choose **Create user** (or **Create users** for multiple users).

12. On the last page of the wizard, do one of the following:
   - Next to each new user, choose **Send email**, and follow the on-screen directions to email the new user their console sign-in URL and user name. Then communicate to each new user their console sign-in password, AWS access key ID, and AWS secret access key separately.
   - Choose **Download .csv**. Then communicate to each new user their console sign-in URL, console sign-in password, AWS access key ID, and AWS secret access key that is in the downloaded file.
   - Next to each new user, choose **Show** for both **Secret access key** and **Password**. Then communicate to each new user their console sign-in URL, console sign-in password, AWS access key ID, and AWS secret access key.

   **Note**
   If you do not choose **Download .csv**, this is the only time you can view the new user's AWS secret access key and console sign-in password. To generate a new AWS secret access key or console sign-in password for the new user, see the following in the IAM User Guide:
   - Creating, Modifying, and Viewing Access Keys (Console)
   - Creating, Changing, or Deleting an IAM User Password (Console)

13. Repeat this procedure for each additional IAM user that you want to create, and then skip ahead to **Step 3: Add AWS Cloud9 Access Permissions to the Group** (p. 12).

### Step 2.2: Create an IAM User and Add the User to the Group with the AWS CLI

**Note**
If you're using **AWS managed temporary credentials** (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

1. Run the IAM `create-user` command to create the user, specifying the new user's name (for example, `MyCloud9User`).

   ```bash
   aws iam create-user --user-name MyCloud9User
   ```

2. Run the IAM `create-login-profile` command to create a new console sign-in password for the user, specifying the user's name and initial sign-in password (for example, `MyCloud9Us3r!`). After the user signs in, AWS asks the user to change their sign-in password.

   ```bash
   aws iam create-login-profile --user-name MyCloud9User --password MyCloud9Us3r! --password-reset-required
   ```
Step 3: Add AWS Cloud9 Access Permissions to the Group

By default, most IAM groups and users don't have access to any AWS services, including AWS Cloud9. (An exception is IAM administrator groups and IAM administrator users, which have access to all AWS services in their AWS account by default.) In this step, you use IAM to add AWS Cloud9 access permissions directly to an IAM group to which one or more users belong, so that you can ensure those users can access AWS Cloud9.

**Note**
Your organization might already have a group set up for you with the appropriate access permissions. If your organization has an AWS account administrator, check with that person before starting the following procedure.

You can complete this task using the AWS Management Console (p. 12) or the AWS CLI (p. 13).

Add AWS Cloud9 Access Permissions to the Group with the Console

1. Sign in to the AWS Management Console, if you are not already signed in, at https://console.aws.amazon.com/.

   **Note**
   Although you can sign in to the AWS Management Console with the email address and password that was provided when the AWS account was created (we call this an AWS account root user), this isn't an AWS security best practice. In the future, we recommend you sign in using credentials for an IAM administrator user in the AWS account. An IAM administrator user has similar AWS access permissions to an AWS account root user and avoids some of the associated security risks. If you cannot sign in as an IAM administrator user, check with your AWS account administrator. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.

2. Open the IAM console. To do this, in the AWS navigation bar, choose Services. Then choose IAM.

3. Choose Groups.
4. Choose the group’s name.

5. Decide whether you want to add AWS Cloud9 user or AWS Cloud9 administrator access permissions to the group. These permissions will apply to each user in the group.

AWS Cloud9 user access permissions allow each user in the group to do the following things within their AWS account:

- Create their own AWS Cloud9 development environments.
- Get information about their own environments.
- Change the settings for their own environments.

AWS Cloud9 administrator access permissions allow each user in the group to do additional things within their AWS account, such as:

- Create environments for themselves or others.
- Get information about environments for themselves or others.
- Delete environments for themselves or others.
- Change the settings of environments for themselves or others.

Note
We recommend that you add only a limited number of users to the AWS Cloud9 administrators group. This AWS security best practice can help you better control, track, and troubleshoot issues with AWS resource access.

6. On the Permissions tab, for Managed Policies, choose Attach Policy.

7. In the list of policy names, choose the box next to AWSCloud9User for AWS Cloud9 user access permissions or AWSCloud9Administrator for AWS Cloud9 administrator access permissions. (If you don't see either of these policy names in the list, enter the policy name in the Filter box to display it.)

8. Choose Attach Policy.

Note
If you have more than one group you want to add AWS Cloud9 access permissions to, repeat this procedure for each of those groups.

To see the list of access permissions that these AWS managed policies give to a group, see AWS Managed (Predefined) Policies (p. 524).

To learn about AWS access permissions that you can add to a group in addition to access permissions that are required by AWS Cloud9, see Managed Policies and Inline Policies and Understanding Permissions Granted by a Policy in the IAM User Guide.

Skip ahead to Step 4: Sign in to the AWS Cloud9 Console (p. 15).

Add AWS Cloud9 Access Permissions to the Group with the AWS CLI

Note
If you're using AWS managed temporary credentials (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

1. Install and configure the AWS CLI on your computer, if you haven't done so already. To do this, see the following in the AWS Command Line Interface User Guide:

- Installing the AWS Command Line Interface
- Quick Configuration
Note
Although you can configure the AWS CLI using the credentials associated with the email address and password that was provided when the AWS account was created (we call this an AWS account root user), this isn't an AWS security best practice. Instead, we recommend you configure the AWS CLI using credentials for an IAM administrator user in the AWS account. An IAM administrator user has similar AWS access permissions to an AWS account root user and avoids some of the associated security risks. If you cannot configure the AWS CLI as an IAM administrator user, check with your AWS account administrator. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.

2. Decide whether to add AWS Cloud9 user or AWS Cloud9 administrator access permissions to the group. These permissions will apply to each user in the group.

AWS Cloud9 user access permissions allow each user in the group to do the following things within their AWS account:

- Create their own AWS Cloud9 development environments.
- Get information about their own environments.
- Change the settings for their own environments.

AWS Cloud9 administrator access permissions allow each user in the group to do additional things within their AWS account, such as the following:

- Create environments for themselves or others.
- Get information about environments for themselves or others.
- Delete environments for themselves or others.
- Change the settings of environments for themselves or others.

Note
We recommend that you add only a limited number of users to the AWS Cloud9 administrators group. This AWS security best practice can help you better control, track, and troubleshoot issues with AWS resource access.

3. Run the IAM attach-group-policy command, specifying the group's name and the Amazon Resource Name (ARN) for the AWS Cloud9 access permissions policy to add.

For AWS Cloud9 user access permissions, specify the following ARN.

```
aws iam attach-group-policy --group-name MyCloud9Group --policy-arn arn:aws:iam::aws:policy/AWSCloud9User
```

For AWS Cloud9 administrator access permissions, specify the following ARN.

```
aws iam attach-group-policy --group-name MyCloud9Group --policy-arn arn:aws:iam::aws:policy/AWSCloud9Administrator
```

Note
If you have more than one group you want to add AWS Cloud9 access permissions to, repeat this procedure for each of those groups.

To see the list of access permissions that these AWS managed policies give to a group, see AWS Managed (Predefined) Policies (p. 524).

To learn about AWS access permissions that you can add to a group in addition to access permissions that are required by AWS Cloud9, see Managed Policies and Inline Policies and Understanding Permissions Granted by a Policy in the IAM User Guide.
Step 4: Sign In to the AWS Cloud9 Console

After you complete the previous steps in this topic, you and your users are ready to sign in to the AWS Cloud9 console and start using it.

1. If you are already signed in to the AWS Management Console as an AWS account root user, sign out of the console.
3. Enter the AWS account number for the IAM user you created or identified earlier, and then choose Next.
   Note
   If you don't see an option for entering the AWS account number, choose Sign in to a different account. Enter the AWS account number on the next page, and then choose Next.
4. Enter the user name and password of the IAM user you created or identified earlier, and then choose Sign In.
5. If prompted, follow the on-screen directions to change your user's initial sign-in password. Save your new sign-in password in a secure location.

The AWS Cloud9 console is displayed, and you can begin using AWS Cloud9.

Next Steps

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Restrict AWS Cloud9 usage for others in your AWS account, to control costs.</td>
<td>Additional Setup Options (p. 22)</td>
</tr>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>

Enterprise Setup for AWS Cloud9

This topic explains how to use AWS Single Sign-On (SSO) to enable one or more AWS accounts to use AWS Cloud9 within an enterprise. To set up to use AWS Cloud9 for any other usage pattern, see Setting Up AWS Cloud9 (p. 6) for the correct instructions.

These instructions assume that you have (or will have) administrative access to the organization in AWS Organizations. If you don't already have administrative access to the organization in AWS Organizations, see your AWS account administrator. For more information, see the following resources:

- Managing Access Permissions for Your AWS Organization in the AWS Organizations User Guide (AWS SSO requires the use of AWS Organizations)
- Overview of Managing Access Permissions to Your AWS SSO Resources in the AWS Single Sign-On User Guide
For introductory information related to this topic, see the following resources:

- **What Is AWS Organizations** in the *AWS Organization User Guide* (AWS SSO requires the use of AWS Organizations)
- The 4-minute video AWS Knowledge Center Videos: How do I get started with AWS Organizations on the YouTube website
- The 7-minute video Manage User Access to Multiple AWS Accounts Using AWS Single Sign-on on the YouTube website
- The 9-minute video How to Setup AWS Single Sign On for Your On-Premise Active Directory Users on the YouTube website

The following conceptual diagram shows what you'll end up with.

![Diagram showing AWS Cloud9 setup within an enterprise]

To enable one or more AWS accounts to start using AWS Cloud9 within an enterprise, start with one of the following steps, depending on which AWS resources you already have.

<table>
<thead>
<tr>
<th>Do you have an AWS account that can or does serve as the master account for the organization in AWS Organizations?</th>
<th>Do you have an organization in AWS Organizations for that master account?</th>
<th>Are all of the wanted AWS accounts members of that organization?</th>
<th>Is that organization set up to use AWS SSO?</th>
<th>Is that organization set up with all of the wanted groups and users who want to use AWS Cloud9?</th>
<th>Start with this step</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Step 1: Create a Master Account for the Organization (p. 17)</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Step 2: Create an Organization for the Master Account (p. 18)</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>—</td>
<td>—</td>
<td>Step 3: Add Member Accounts to the Organization (p. 18)</td>
</tr>
</tbody>
</table>
Step 1: Create a Master Account for the Organization

Note
Your enterprise might already have a master account set up for you. If your enterprise has an AWS account administrator, check with that person before starting the following procedure. If you already have a master account, skip ahead to Step 2: Create an Organization for the Master Account (p. 18).

To use AWS Single Sign-On (AWS SSO), you must have an AWS account that will serve as the master account for an organization in AWS Organizations. For more information, see the discussion about master accounts in AWS Organizations Terminology and Concepts in the AWS Organizations User Guide.

To watch a 4-minute video related to the following procedure, see Creating an Amazon Web Services Account on the YouTube website.

To create a master account:

2. Choose Sign In to the Console.
3. Choose Create a new AWS account.
4. Complete the process by following the on-screen directions. This includes giving AWS your email address and credit card information. You must also use your phone to enter a code that AWS gives you.
After you finish creating the account, AWS will send you a confirmation email. Do not go to the next step until you get this confirmation.

**Step 2: Create an Organization for the Master Account**

**Note**
Your enterprise might already have AWS Organizations set up to use the master account. If your enterprise has an AWS account administrator, check with that person before starting the following procedure. If you already have AWS Organizations set up to use the master account, skip ahead to **Step 3: Add Member Accounts to the Organization (p. 18)**.

To use AWS SSO, you must have an organization in AWS Organizations that uses the master account. For more information, see the discussion about *organizations* in [AWS Organizations Terminology and Concepts](https://docs.aws.amazon.com/organizations/latest/userguide/org-concepts.html) in the *AWS Organizations User Guide*.

To create an organization in AWS Organizations for the master AWS account, follow these instructions in the *AWS Organizations User Guide*:

1. Creating an Organization
2. Enabling All Features in Your Organization

To watch a 4-minute video related to these procedures, see [AWS Knowledge Center Videos: How do I get started with AWS Organizations](https://aws.amazon.com/solutions/library/aws-knowledge-center/how-do-i-get-started-with-aws-organizations/) on the YouTube website.

**Step 3: Add Member Accounts to the Organization**

**Note**
Your enterprise might already have AWS Organizations set up with the wanted member accounts. If your enterprise has an AWS account administrator, check with that person before starting the following procedure. If you already have AWS Organizations set up with the wanted member accounts, skip ahead to **Step 4: Enable AWS SSO Across the Organization (p. 18)**.

In this step, you add any AWS accounts that will serve as member accounts for the organization in AWS Organizations. For more information, see the discussion about *member accounts* in [AWS Organizations Terminology and Concepts](https://docs.aws.amazon.com/organizations/latest/userguide/org-concepts.html) in the *AWS Organizations User Guide*.

**Note**
You don't have to add any member accounts to the organization. You can use AWS SSO with just the single master account in the organization. Later, you can add member accounts to the organization, if you want. If you don't want to add any member accounts now, skip ahead to **Step 4: Enable AWS SSO Across the Organization (p. 18)**.

To add member accounts to the organization in AWS Organizations, follow one or both of the following sets of instructions in the *AWS Organizations User Guide*. Repeat these instructions as many times as needed until you have all of the AWS accounts you want as members of the organization:

- Creating an AWS Account in Your Organization
- Inviting an AWS Account to Join Your Organization

**Step 4: Enable AWS SSO across the Organization**

**Note**
Your enterprise might already have AWS Organizations set up to use AWS SSO. If your enterprise has an AWS account administrator, check with that person before starting the following
Step 5. Set Up Groups and Users within the Organization

In this step, you either create groups and users in an AWS SSO directory for the organization, or you connect to an AWS Managed Microsoft AD or AD Connector directory that is managed in AWS Directory Service for the organization. In a later step, you give groups the necessary access permissions to use AWS Cloud9.

- If you're using an AWS SSO directory for the organization, follow these sets of instructions in the AWS Single Sign-On User Guide. Repeat these steps as many times as needed until you have all of the groups and users you want:
  1. Add Groups. We recommend creating at least one group for any AWS Cloud9 administrators across the organization, and then repeating this step to create another group for all AWS Cloud9 users across the organization. Optionally, you might also repeat this step to create a third group for all users across the organization with whom you want to share existing AWS Cloud9 development environments, but not allow them to create environments on their own. For ease of use, we recommend naming these groups AWSCloud9Administrators, AWSCloud9Users, and AWSCloud9EnvironmentMembers, respectively. For more information, see AWS Managed (Predefined) Policies for AWS Cloud9 (p. 524).
  2. Add Users.
  3. Add Users to Groups. Add any AWS Cloud9 administrators to the AWSCloud9Administrators group, repeat this step to add AWS Cloud9 users to the AWSCloud9Users group, and optionally repeat this step to add any remaining users to the AWSCloud9EnvironmentMembers group. Adding users to groups is an AWS security best practice that can help you better control, track, and troubleshoot issues with AWS resource access.

- If you're using an AWS Managed Microsoft AD or AD Connector directory that you manage in AWS Directory Service for the organization, follow the instructions in Connect to Your Microsoft AD Directory in the AWS Single Sign-On User Guide.
Step 6. Enable Groups and Users within the Organization to Use AWS Cloud9

By default, most users and groups in an organization in AWS Organizations don't have access to any AWS services, including AWS Cloud9. In this step, you use AWS SSO to allow groups and users across an organization in AWS Organizations to use AWS Cloud9 within any combination of participating accounts.

1. In the AWS SSO console, choose AWS accounts in the service navigation pane.
2. Choose the Permission sets tab.
3. Choose Create permission set set.
4. Select Create a custom permission set.
5. Enter a Name for this permission set. We recommend creating at least one permission set for any AWS Cloud9 administrators across the organization, and then repeating steps 3 through 10 in this procedure to create another permission set for all AWS Cloud9 users across the organization. Optionally, you might also repeat steps 3 through 10 in this procedure to create a third permission set for all users across the organization with whom you want to share existing AWS Cloud9 development environments, but not allow them to create environments on their own. For ease of use, we recommend naming these permission sets AWSCloud9AdministratorsPerms, AWSCloud9UsersPerms, and AWSCloud9EnvironmentMembersPerms, respectively. For more information, see AWS Managed (Predefined) Policies for AWS Cloud9 (p. 524).
6. Enter an optional Description for the permission set.
7. Choose a Session duration for the permission set, or leave the default session duration of 1 hour.
8. Select Attach AWS managed policies.
9. In the list of policies, select one of the following boxes next to the correct Policy name entry. (Don't choose the policy name itself. If you don't see a policy name in the list, enter the policy name in the Search box to display it.)
   - For the AWSCloud9AdministratorsPerms permission set, select AWSCloud9Administrator.
   - For the AWSCloud9UsersPerms permission set, select AWSCloud9User.
   - Optionally, for the AWSCloud9EnvironmentMembersPerms permission set, select AWSCloud9EnvironmentMember.
   
   Note
   To learn about policies that you can add in addition to the policies that are required by AWS Cloud9, see Managed Policies and Inline Policies and Understanding Permissions Granted by a Policy in the IAM User Guide.
10 Choose Create.
11 After you finish creating all of the permission sets you want, on the AWS organization tab, choose the AWS account that you want to assign AWS Cloud9 access permissions to. (If the AWS organization tab isn't visible, then in the service navigation pane, choose AWS accounts. This displays the AWS organization tab).
12 Choose Assign users.
13 On the Groups tab, select the box next to the name of the group that you want to assign AWS Cloud9 access permissions to, as follows. (Don't choose the group name itself.)
   - If you're using an AWS SSO directory for the organization, you might have a created a group named AWSCloud9Administrators for AWS Cloud9 administrators.
   - If you're using an AWS Managed Microsoft AD or AD Connector directory that you manage in AWS Directory Service for the organization, choose the directory's ID, enter part or all of the group's name, and choose Search connected directory. Then select the box next to the name of the group that you want to assign AWS Cloud9 access permissions to.
Step 7: Start Using AWS Cloud9

After you complete the previous steps in this topic, you and your users are ready to sign in to AWS SSO and start using AWS Cloud9.

1. If you are already signed in to an AWS account or to AWS SSO, sign out. To do this, see How do I sign out of my AWS account on the AWS Support website or How to Sign Out of the User Portal in the AWS Single Sign-On User Guide.

2. To sign in to AWS SSO, follow the instructions in How to Accept the Invitation to Join AWS SSO in the AWS Single Sign-On User Guide. This includes going to a unique sign-in URL and signing in with a unique user name and password. Your AWS account administrator will either email you this information or otherwise provide it to you.

   **Note**
   Be sure to bookmark the unique sign-in URL that you were provided, so that you can easily return to it later. Also be sure to store the unique user name and password for this URL in a secure location.
   This combination of URL, user name, and password might change depending on different levels of AWS Cloud9 access permissions that your AWS account administrator gives you. For example, you might use one URL, user name, and password to get AWS Cloud9 administrator access to one account, and you might use a different URL, user name, and password that allows only AWS Cloud9 user access to a different account.

3. After you sign in to AWS SSO, choose the **AWS Account** tile.

4. Choose your user’s display name from the drop-down list that is displayed. If more than one name is displayed, choose the name that you want to start using AWS Cloud9. If you’re not sure which of these names to choose, see your AWS account administrator.

5. Choose the **Management console** link next to your user’s display name. If more than one **Management console** link is displayed, choose the link next to the correct permission set. If you’re not sure which of these links to choose, see your AWS account administrator.

6. From the AWS Management Console, do one of the following:
   - Choose **Cloud9**, if it’s already displayed.
   - Expand **All services**, and then choose **Cloud9**.
   - In the **Find services** box, type **Cloud9**, and then press **Enter**.
   - In the AWS navigation bar, choose **Services**, and then choose **Cloud9**.

The AWS Cloud9 console is displayed, and you can begin using AWS Cloud9.
Next Steps

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>

Additional Setup Options for AWS Cloud9 (Team and Enterprise)

This topic assumes you have already completed the setup steps in Team Setup (p. 8) or Enterprise Setup (p. 15).

In Team Setup (p. 8) or Enterprise Setup (p. 15), you created groups and added AWS Cloud9 access permissions directly to those groups, to ensure that users in those groups can access AWS Cloud9. In this topic, you will add more access permissions to restrict the kinds of environments that users in those groups can create. This can help control costs related to AWS Cloud9 in AWS accounts and organizations.

To add these access permissions, you create your own set of policies that define the AWS access permissions you want to enforce. (We call each of these a customer-managed policy.) Then you attach those customer-managed policies to the groups that the users belong to. (In some scenarios, you must also detach existing AWS managed policies that are already attached to those groups.) To set this up, follow the procedures in this topic.

Note
The following procedures cover attaching and detaching policies for AWS Cloud9 users only. These procedures assume you already have a separate AWS Cloud9 users group and AWS Cloud9 administrators group and that you have only a limited number of users in the AWS Cloud9 administrators group. This AWS security best practice can help you better control, track, and troubleshoot issues with AWS resource access.

- Step 1: Create a Customer-Managed Policy (p. 22)
- Step 2: Add Customer-Managed Policies to a Group (p. 23)
- Customer-Managed Policy Examples for Teams Using AWS Cloud9 (p. 24)

Step 1: Create a Customer-Managed Policy

You can create a customer-managed policy using the AWS Management Console (p. 23) or the AWS Command Line Interface (AWS CLI) (p. 23).

Note
This step covers creating a customer-managed policy for IAM groups only. To create a custom permission set for groups in AWS Single Sign-On (SSO), skip this step and follow the instructions in Create Permission Set in the AWS Single Sign-On User Guide instead. In this topic, follow the
instructions to create a custom permission set. For related custom permissions policies, see Customer-Managed Policy Examples for Teams Using AWS Cloud9 (p. 24) later in this topic.

Create a Customer-Managed Policy Using the Console

1. Sign in to the AWS Management Console, if you are not already signed in.
   We recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.
2. Open the IAM console. To do this, in the console's navigation bar, choose Services. Then choose IAM.
3. In the service's navigation pane, choose Policies.
4. Choose Create policy.
5. In the JSON tab, paste one of our suggested Customer-Managed Policy Examples (p. 24).
   Note
   You can also create your own customer-managed policies. For more information, see the IAM JSON Policy Reference in the IAM User Guide and the AWS services' documentation.
7. On the Review policy page, type a Name and an optional Description for the policy, and then choose Create policy.

Repeat this step for each additional customer-managed policy that you want to create, then skip ahead to Add Customer-Managed Policies to a Group Using the Console (p. 24).

Create a Customer-Managed Policy Using the AWS CLI

1. On the computer where you run the AWS CLI, create a file to describe the policy (for example, policy.json).
   If you create the file with a different file name, substitute it throughout this procedure.
2. Paste one of our suggested Customer-Managed Policy Examples (p. 24) into the policy.json file.
   Note
   You can also create your own customer-managed policies. For more information, see the IAM JSON Policy Reference in the IAM User Guide and the AWS services' documentation.
3. From the terminal or command prompt, switch to the directory that contains the policy.json file.
4. Run the IAM create-policy command, specifying a name for the policy and the policy.json file.

   ```bash
   aws iam create-policy --policy-document file://policy.json --policy-name MyPolicy
   ```

   In the preceding command, replace MyPolicy with a name for the policy.

Skip ahead to Add Customer-Managed Policies to a Group Using the AWS CLI (p. 24).

Step 2: Add Customer-Managed Policies to a Group

You can add customer-managed policies to a group using the AWS Management Console (p. 24) or the AWS Command Line Interface (AWS CLI) (p. 24).

Note
This step covers adding customer-managed policies to IAM groups only. To add custom permission sets to groups in AWS Single Sign-On (SSO), skip this step and follow the instructions in Assign User Access in the AWS Single Sign-On User Guide instead.
Add Customer-Managed Policies to a Group Using the Console

1. With the IAM console open from the previous procedure, in the service's navigation pane, choose Groups.
2. Choose the group's name.
3. On the Permissions tab, for Managed Policies, choose Attach Policy.
4. In the list of policy names, choose the box next to each customer-managed policy you want to attach to the group. (If you don't see a specific policy name in the list, type the policy name in the Filter box to display it.)
5. Choose Attach Policy.

Add Customer-Managed Policies to a Group Using the AWS CLI

Note
If you're using AWS managed temporary credentials (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

Run the IAM attach-group-policy command, specifying the group's name and the Amazon Resource Name (ARN) of the policy.

```
aws iam attach-group-policy --group-name MyGroup --policy-arn arn:aws:iam::123456789012:policy/MyPolicy
```

In the preceding command, replace MyGroup with the name of the group. Replace 123456789012 with the AWS account ID, and replace MyPolicy with the name of the customer-managed policy.

Customer-Managed Policy Examples for Teams Using AWS Cloud9

Following are some examples of policies you can use to restrict the kinds of environments that users in a group can create in an AWS account.

- Prevent Users in a Group from Creating Environments (p. 24)
- Prevent Users in a Group from Creating EC2 Environments (p. 25)
- Allow Users in a Group to Create EC2 Environments Only with Specific Amazon EC2 Instance Types (p. 25)
- Allow Users in a Group to Create Only a Single EC2 Environment Per AWS Region (p. 26)

Prevent Users in a Group from Creating Environments

The following customer-managed policy, when attached to an AWS Cloud9 users group, prevents those users from creating environments in an AWS account. This is useful if you want an IAM administrator user in your AWS account to manage creating environments instead of users in an AWS Cloud9 users group.

```
{
  "Version": "2012-10-17",
  "Statement": [
```
Customer-Managed Policy Examples for Teams Using AWS Cloud9

"Effect": "Deny",
"Action": [
   "cloud9:CreateEnvironmentEC2",
   "cloud9:CreateEnvironmentSSH"
],
"Resource": "*"
}

Note that the preceding customer-managed policy explicitly overrides "Effect": "Allow" for "Action": "cloud9:CreateEnvironmentEC2" and "cloud9:CreateEnvironmentSSH" on "Resource": "*" in the AWSCloud9User managed policy that is already attached to the AWS Cloud9 users group.

Prevent Users in a Group from Creating EC2 Environments

The following customer-managed policy, when attached to an AWS Cloud9 users group, prevents those users from creating EC2 environments in an AWS account. This is useful if you want an IAM administrator user in your AWS account to manage creating EC2 environments instead of users in an AWS Cloud9 users group. This assumes you haven't also attached a policy that prevents users in that group from creating SSH environments. Otherwise, those users won't be able to create environments at all.

{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Deny",
         "Action": "cloud9:CreateEnvironmentEC2",
         "Resource": "*"
      }
   ]
}

Allow Users in a Group to Create EC2 Environments Only with Specific Amazon EC2 Instance Types

The following customer-managed policy, when attached to an AWS Cloud9 users group, allows those users to create EC2 environments that only use instance types starting with `t2` in an AWS account. This policy assumes you haven't also attached a policy that prevents users in that group from creating EC2 environments. Otherwise, those users won't be able to create EC2 environments at all.

You can replace "t2.*" in the following policy with a different instance class (for example, "m4.*"). Or you can restrict it to multiple instance classes or instance types (for example, [ "t2.*", "m4.*" ] or [ "t2.micro", "m4.large" ]).

For an AWS Cloud9 users group, detach the AWSCloud9User managed policy from the group, and then add the following customer-managed policy in its place. (If you do not detach the AWSCloud9User managed policy, the following customer-managed policy will have no effect.)

{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "cloud9:CreateEnvironmentEC2",
         "Resource": "*"
      }
   ]
}
"Effect": "Allow",
"Action": [
    "cloud9:CreateEnvironmentSSH",
    "cloud9:ValidateEnvironmentName",
    "cloud9:GetUserPublicKey",
    "cloud9:GetUserSettings",
    "cloud9:GetUserSettings",
    "iam:GetUser",
    "iam:ListUsers",
    "ec2:DescribeVpcs",
    "ec2:DescribeSubnets"
],
"Resource": "*",
"Condition": {
    "StringLike": {
        "cloud9:InstanceType": "t2.*"
    }
}
},
{
"Effect": "Allow",
"Action": "cloud9:CreateEnvironmentEC2",
"Resource": "*",
"Condition": {
    "StringLike": {
        "cloud9:InstanceType": "t2.*"
    }
}
},
{
"Effect": "Allow",
"Action": [
    "cloud9:DescribeEnvironmentMemberships"
],
"Resource": "*",
"Condition": {
    "Null": {
        "cloud9:UserArn": "true",
        "cloud9:EnvironmentId": "true"
    }
}
},
{
"Effect": "Allow",
"Action": ["iam:CreateServiceLinkedRole"],
"Resource": "*",
"Condition": {
    "StringLike": {
        "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
}
]
}

Note that the preceding customer-managed policy also allows those users to create SSH environments. To prevent those users from creating SSH environments altogether, remove "cloud9:CreateEnvironmentSSH", from the preceding customer-managed policy.

**Allow Users in a Group to Create Only a Single EC2 Environment Per AWS Region**

The following customer-managed policy, when attached to an AWS Cloud9 users group, allows each of those users to create a maximum of one EC2 environment per AWS Region that AWS Cloud9 is available
in. This is done by restricting the name of the environment to one specific name in that AWS Region (in this example, my-demo-environment).

**Note**
AWS Cloud9 doesn't enable restricting the creation of environments to specific AWS Regions. Nor does it enable restricting the overall number of environments that can be created (other than the published service limits (p. 558)).

For an AWS Cloud9 users group, detach the AWSCloud9User managed policy from the group, and then add the following customer-managed policy in its place. (If you do not detach the AWSCloud9User managed policy, the following customer-managed policy will have no effect.)

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentSSH",
        "cloud9:ValidateEnvironmentName",
        "cloud9:GetUserPublicKey",
        "cloud9:UpdateUserSettings",
        "cloud9:GetUserSettings",
        "iam:GetUser",
        "iam:ListUsers",
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:CreateEnvironmentEC2"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloud9:EnvironmentName": "my-demo-environment"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:DescribeEnvironmentMemberships"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "Null": {
          "cloud9:UserArn": "true",
          "cloud9:EnvironmentId": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:EnvironmentName": "my-demo-environment"
        }
      }
    }
  ]
}
```
Note that the preceding customer-managed policy allows those users to create SSH environments. To prevent those users from creating SSH environments altogether, remove "cloud9:CreateEnvironmentSSH", from the preceding customer-managed policy.

For additional examples, see Customer-Managed Policy Examples (p. 493).

**Next Steps**

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>

**Individual student setup for AWS Cloud9**

This topic explains how to set up to use AWS Cloud9 as a student, with you as the only individual using your account with AWS. To set up to use AWS Cloud9 for any other usage pattern, see Setting Up AWS Cloud9 (p. 6) for the correct instructions. To learn about who qualifies as a student, see Who can join AWS Educate on the AWS Educate Frequently Asked Questions website.

- Step 1: Create an AWS Educate Starter Account (p. 28)
- Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29)
- Next steps (p. 30)

**Step 1: Create an AWS Educate Starter Account**

In this step, you create an AWS Educate Starter Account (p. 29), which you can then use to access AWS Cloud9.

For an AWS Educate Starter account, note the following:

- You must be an eligible student.
- You don't need a credit card.
- There are some restrictions on AWS usage in the account.
- The account isn't eligible for the AWS Free Tier.
- After you graduate, you can no longer keep using the account.
- You can't link the account to any other account.
• You can't apply any additional AWS promotional credit to the account.
• AWS resources that you create in the account will stop working after all AWS promotional credit is used.
• If the account runs out of AWS promotional credit, the account is deactivated.
• You can't turn the account into a regular AWS account.

Create an AWS Educate Starter Account

If you already have an AWS Educate Starter Account, skip ahead to Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29).

1. Go to the Apply for AWS Educate website, at https://www.awseducate.com/registration#INFO-Student.
2. Choose the Student role.
3. On the Tell us about yourself page, follow the on-screen instructions. Provide information such as the educational institution that you attend and your student email address. Leave Promo Code blank.
4. Complete the security challenge, and then choose Next.
5. Review and agree to the AWS Educate program’s terms and conditions. AWS will send you an email with the title Email Verification - AWS Educate Application. Choose the link in that email to confirm your email address.
6. After you complete the application process, AWS will send you an email with the title Thank You for Applying for AWS Educate. This email includes details such as how to check the status of your application and how to contact AWS Educate support with any questions.

After AWS Educate approves your application, AWS will send you an email with the title AWS Educate Application Approved. Note that the approval process might take a day or longer. Do not go past this step in this procedure until you get this approval.

7. After you get the approval email, follow the instructions in the email to set your AWS Educate Student Portal password, and then sign in to your AWS Educate Student Portal. After you set your AWS Educate Student Portal password, store that password in a secure location.
8. Go to Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29).

Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console

1. Go to the AWS Educate Site Login website, at https://www.awseducate.com/signin/.
2. Enter the email address and password that you use to sign in to your AWS Educate Student Portal, and then choose Sign In.
3. Select AWS Account from the top navigation banner.
4. Select AWS Educate Starter Account.

   Note
   If this is your first time logging in, select Create Starter Account to activate your AWS Educate Starter Account, and then select AWS Educate Starter Account.
   First-time users of the AWS Educate Starter Account also need to agree to the Vocareum terms and conditions.

5. Choose AWS Console.
6. Do one of the following:
   • Choose Cloud9, if it's already displayed.
- Expand All services, and then choose Cloud9.
- In the Find services box, type Cloud9, and then press Enter.
- In the AWS navigation bar, choose Services, and then choose Cloud9.

You can now start using AWS Cloud9. Go to Next Steps (p. 30).

## Next steps

<table>
<thead>
<tr>
<th>Task for learning</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>More advanced task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>

## Classroom Setup for AWS Cloud9

This topic explains how an educator can set up for students to use AWS Cloud9 in a virtual classroom. To set up to use AWS Cloud9 for any other usage pattern, see Setting Up AWS Cloud9 (p. 6) for the correct instructions. To learn about who qualifies as an educator, see Who can join AWS Educate on the AWS Educate Frequently Asked Questions website.

- Step 1: Create an AWS Account and Apply for AWS Educate (p. 30)
- Step 2: Set Up Your Virtual Classroom for AWS Cloud9 (p. 32)
- Step 3: Enter Your Virtual Classroom (p. 32)
- Next Steps (p. 34)

### Step 1: Create an AWS Account and Apply for AWS Educate

In this step, you create an AWS account. Then you apply for AWS Educate. During the application process, you provide AWS Educate with your AWS account ID.

If you already have an AWS account, skip ahead to step 5 of this procedure to apply for AWS Educate. If you're not sure whether your department or institution already has an existing AWS account for you, or whether you should create a new one on your own, check with your department's or institution's AWS account administrator.

If you are already a member of AWS Educate, skip ahead to Step 2: Set Up Your Virtual Classroom for AWS Cloud9 (p. 32).
2. Choose **Sign In to the Console**.
3. Choose **Create a new AWS account**.
4. Complete the process by following the on-screen directions. This includes giving AWS your email address and credit card information. You must also use your phone to enter a code that AWS gives you.

   After you finish creating the account, AWS sends you a confirmation email. Do not go past this step in this procedure until you get this confirmation.
5. Go to the Apply for AWS Educate website, at https://aws.amazon.com/education/awseducate/apply/.
6. Choose **Apply for AWS Educate for Educators**.
7. On the **Tell us about yourself** page, follow the on-screen instructions. This includes giving AWS Educate information such as the educational institution that you work for and your email address there. Leave **Promo Code** blank.
8. Complete the security challenge, and then choose **Next**.
9. On the **Choose one of the following** page, select **Click here to enter an AWS Account ID**, and then choose **Next**.
10. In the box next to **Click here to enter an AWS Account ID**, enter your AWS account ID, and then choose **Next**. (To get your AWS account ID, see Finding Your AWS Account ID in the Amazon Web Services General Reference.)
11. AWS will send you an email with the title **Email Verification - AWS Educate Application**. Choose the link in that email to confirm your email address, review the AWS Educate program terms and conditions, and complete the application process.
12. After you complete the application process, AWS will send you an email with the title **Thank You for Applying for AWS Educate**. This email includes details such as how to check the status of your application and how to contact AWS Educate support with any questions.

   After AWS Educate approves your application, AWS will send you an email with the title **AWS Educate Application Approved**. Note that the approval process might take a few days. Do not go past this step in this procedure until you get this approval.

   After you get the approval email, follow the instructions in the email to set your AWS Educate Educator Collaboration Portal (ECP) password, and then sign in to your AWS Educate ECP. After you set your AWS Educate ECP password, store that password in a secure location. The approval email contains information about AWS promotional credit that you can apply to your AWS account. AWS promotional credit can help offset the cost of your individual use of qualifying AWS services, some of which can be used by or with AWS Cloud9. To apply AWS promotional credit to your AWS account, do the following:

   a. With your AWS Educate ECP already open, choose **AWS Account**.

      **Note**
      To return to your AWS Educate ECP, go to the AWS Educate Site Login website, at https://www.awseducate.com/signin/. Enter the email address and password that you used when you initially signed in to your AWS Educate ECP, and then choose **Sign In**.

   b. In the AWS navigation bar, choose **My Account, Billing & Cost Management**.

   c. Enter the email address for your AWS account, and then choose **Next**.

      **Note**
      If an email address is already displayed and it’s the wrong one, choose **Sign in to a different account**. Enter the correct email address, and then choose **Next**.

   d. Enter the password for your AWS account, and then choose **Sign in**.

   **Important**
   Enter the password for your AWS account, not your password for your AWS Educate ECP.
Step 2: Set Up Your Virtual Classroom for AWS Cloud9

In this step, you use your AWS Educate Educator Collaboration Portal (ECP) to set up a virtual classroom for your students to use AWS Cloud9.

1. With your AWS Educate ECP already open, choose Classrooms & Credits.

   **Note**
   To open or return to your AWS Educate ECP, go to the AWS Educate Site Login website, at https://www.awseducate.com/signin/. Enter the email address and password that you used when you initially signed in to your AWS Educate ECP, and then choose Sign In.

2. Choose Request or go to an AWS Educate Classroom.

3. For Select Classroom Template, choose the Cloud9 tile.

4. For Enter Classroom Details, choose Next.

5. Provide information about your class, such as the class’s start and end dates and the estimated number of students.

6. For Credit Amount Requested Per Student, enter the amount of AWS promotional credit that you want each student to have available to use for the class. If you request more than maximum default AWS promotional credit per student, provide additional details to support your request.

7. For Upload a List of Email Addresses, choose Next.

8. Follow the on-screen instructions to download the email address upload template, fill in the template, and then upload your completed template.

9. After you upload your completed template, choose Submit.

After AWS Educate approves your application, AWS will send you an email with the title Your AWS Educate Classroom has been created. Note that the approval process might take a few days. Do not go past this step in this procedure until you get this approval. After you get the approval email, AWS Educate will start sending email invitations to each of the email addresses that you provided earlier in step 8 of this procedure.

Step 3: Enter Your Virtual Classroom

After you complete the previous steps in this topic, you and your students are ready to enter your virtual classroom and start using AWS Cloud9.

- Enter a Virtual Classroom as a Student (p. 33)
- Return to a Virtual Classroom as a Student (p. 33)
- Enter a Virtual Classroom as the Instructor (p. 34)
Enter a Virtual Classroom as a Student

1. AWS Educate will send you an email with the title Your AWS Educate Application. In that email, choose the link to complete the AWS Educate application process and to accept your instructor’s virtual classroom invitation.

2. On the Tell us about yourself page, follow the on-screen instructions. This includes giving AWS Educate information, such as the city where your educational institution is located and when you expect to graduate. Leave Promo Code blank.

3. Complete the security challenge, and then choose Next.

4. On the Choose one of the following page, select Click here to select an AWS Educate Starter Account, and then choose Next.

5. AWS will send you an email with the title Email Verification - AWS Educate Application. Choose the link in that email to confirm your email address, review the AWS Educate program terms and conditions, and complete the application process.

6. After you complete the application process, AWS will send you an email with the title Thank You for Applying for AWS Educate. This email includes details such as how to contact AWS Educate support with any questions.

   After AWS Educate approves your application, AWS will send you an email with the title AWS Educate Application Approved. Note that the approval process might take a day or longer. Do not go past this step in this procedure until you get this approval.

   When you get the approval email, follow the instructions in it to set your AWS Educate Student Portal password, and then sign in to your AWS Educate Student Portal. After you set your AWS Educate Student Portal password, store that password in a secure location.

7. With your AWS Educate Student Portal open, choose My Classrooms.

8. Next to the name of the course that you want to enter, choose Go to classroom.

9. Choose Continue.

10. If any terms and conditions are displayed, read them, and then choose I Agree.

11. Choose AWS Console.

12. Do one of the following:
    • Choose Cloud9, if it’s already displayed.
    • Expand All services, and then choose Cloud9.
    • In the Find services box, type Cloud9, and then press Enter.
    • In the AWS navigation bar, choose Services, and then choose Cloud9.

You can now start using AWS Cloud9. Skip ahead to Next Steps (p. 30).

Return to a Virtual Classroom as a Student

1. Go to the AWS Educate Site Login website, at https://www.awseducate.com/signin/.

2. Enter the email address and password that you use to sign in to your AWS Educate Student Portal, and then choose Sign In.

3. Choose My Classrooms.

4. Next to the name of the course that you want to return to, choose Go to classroom.

5. Choose Continue.

6. Choose AWS Console.

7. Do one of the following:
   • Choose Cloud9, if it’s already displayed.
   • Expand All services, and then choose Cloud9.
Next Steps

- In the **Find services** box, type **Cloud9**, and then press **Enter**.
- In the AWS navigation bar, choose **Services**, and then choose **Cloud9**.

You can now continue using AWS Cloud9. Skip ahead to **Next Steps (p. 30)**.

**Enter a Virtual Classroom as the Instructor**

1. With your AWS Educate ECP already open, choose **Classrooms & Credits**.
   
   **Note**
   
   To return to your AWS Educate ECP, go to the AWS Educate Site Login website, at [https://www.awseducate.com/signin/](https://www.awseducate.com/signin/). Enter the email address and password that you used when you initially signed in to your AWS Educate ECP, and then choose **Sign In**.

2. Choose **Request or go to an AWS Educate Classroom**.
3. Choose **Go to my classrooms**.
4. Next to your course's name, choose **Go to classroom**.
5. Choose **Continue**.
6. If any terms and conditions are displayed, read them, and then choose **I Agree**.
7. If your course name isn't displayed, on the site navigation bar, choose **My Classes**, and then choose your course's name.
8. To start or continue using AWS Cloud9, in the **Select Student** list for your course's name, choose your instructor email address.
9. For **View**, choose **Workarea**.
10. Choose **AWS Console**.
11. Do one of the following:
    - Choose **Cloud9**, if it's already displayed.
    - Expand **All services**, and then choose **Cloud9**.
    - In the **Find services** box, type **Cloud9**, and then press **Enter**.
    - In the AWS navigation bar, choose **Services**, and then choose **Cloud9**.

**Next Steps**

<table>
<thead>
<tr>
<th>Task for learning</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn how to use the AWS Cloud9 IDE.</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>More advanced task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create an AWS Cloud9 development environment, and then use the AWS Cloud9 IDE to work with code in your new environment.</td>
<td>Creating an Environment (p. 55)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
</tbody>
</table>
Getting Started: Basic Tutorials for AWS Cloud9

Are you new to AWS Cloud9? If you haven’t done so already, take a look at the general information about AWS Cloud9 in What Is AWS Cloud9 (p. 1).

In the following tutorials, you create an environment in AWS Cloud9 and then use that environment to create a simple application. Both tutorials have the same input and results, but one uses the AWS Cloud9 console and the other uses the AWS Command Line Interface (AWS CLI). You can choose to perform either or both.

When you are finished with these tutorials, you can learn more about the AWS Cloud9 IDE in Tour the AWS Cloud9 IDE (p. 100).

Topics
- Tutorial: Hello AWS Cloud9 (console) (p. 35)
- Tutorial: Hello AWS Cloud9 (CLI) (p. 45)

Tutorial: Hello AWS Cloud9 (console)

This tutorial provides a first look at AWS Cloud9. It uses the AWS Cloud9 console, which enables you to perform the steps of the tutorial in a graphical user interface instead of a command line tool (p. 45).

In this tutorial, you set up an AWS Cloud9 development environment and then use the AWS Cloud9 IDE to code, run, and debug your first application.

This tutorial should take approximately an hour.

Warning
Completing this tutorial might result in charges to your AWS account. These include possible charges for Amazon EC2. For more information, see Amazon EC2 Pricing.

Prerequisites
To successfully complete this tutorial, you must first complete the steps in Setting Up AWS Cloud9 (p. 6).

Steps
- Step 1: Create an Environment (p. 35)
- Step 2: Basic Tour of the IDE (p. 38)
- Step 3: Clean Up (p. 43)
- Related Information (p. 44)

Step 1: Create an Environment
(First step of Tutorial: Hello AWS Cloud9 (console) (p. 35))
In this step, you use the AWS Cloud9 console to create and then open an AWS Cloud9 development environment.

**Note**
If you have already created the environment that you want to use for this tutorial, open that environment and skip ahead to Step 2: Basic Tour of the IDE (p. 38).

In AWS Cloud9, a development environment (or just environment) is a place where you store your development project's files and where you run the tools to develop your applications. In this tutorial, you create a special kind of environment called an EC2 environment, and then work with the files and tools in that environment.

### Create an EC2 Environment with the Console

1. Sign in to the AWS Cloud9 console as follows:

   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to [https://console.aws.amazon.com/cloud9/](https://console.aws.amazon.com/cloud9/).
   - If your organization uses AWS Single Sign-On (AWS SSO), see your AWS account administrator for sign-in instructions.
   - If you're using an AWS Educate Starter Account, see Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29) in Individual Student Signup.
   - If you're a student in a classroom, see your instructor for sign-in instructions.

2. After you sign in to the AWS Cloud9 console, in the top navigation bar, choose an AWS Region to create the environment in. For a list of available AWS Regions, see AWS Cloud9 in the AWS General Reference.

3. Choose the large Create environment button in one of the locations shown below.

   If you have no AWS Cloud9 environments yet, the button is shown on a welcome page.
If you already have AWS Cloud9 environments, the button is shown as follows.

4. On the Name environment page, for Name, enter a name for your environment. For this tutorial, use my-demo-environment.
5. For Description, enter something about your environment. For this tutorial, use This environment is for the AWS Cloud9 tutorial.
6. Choose Next step.
7. On the Configure settings page, for Environment type, choose Create a new instance for environment (EC2).

   Warning
   Choosing Create a new instance for environment (EC2) might result in possible charges to your AWS account for Amazon EC2.

8. For Instance type, leave the default choice. This choice has relatively low RAM and vCPUs, which is sufficient for this tutorial.

   Warning
   Choosing instance types with more RAM and vCPUs might result in additional charges to your AWS account for Amazon EC2.

9. For Platform, choose the type of Amazon EC2 instance that you want: Amazon Linux or Ubuntu. AWS Cloud9 creates the instance and then connects the environment to it.
10. Choose a value for Cost-saving setting. When all web browser instances that are connected to the IDE for the environment are closed, AWS Cloud9 waits this amount of time and then shuts down the Amazon EC2 instance for the environment.

    Warning
    Choosing a longer time period might result in more charges to your AWS account.


    AWS Cloud9 uses Amazon Virtual Private Cloud (Amazon VPC) to communicate with the newly created Amazon EC2 instance. For this tutorial, we recommend that you don't change the preselected default settings. With the default settings, AWS Cloud9 attempts to automatically use the default VPC with its single subnet in the same AWS account and AWS Region as the new environment.

    You can find more information about Amazon VPC choices in Create an EC2 Environment with the Console (p. 58), and in VPC Settings for AWS Cloud9 Development Environments (p. 467).
12. Add up to 50 tags by supplying a **Key** and a **Value** for each tag. The tags will be attached to the AWS Cloud9 environment as resource tags, and are propagated to the following underlying resources: the AWS CloudFormation stack, the Amazon EC2 instance, and Amazon EC2 security groups. You can find information about tags in Control Access Using AWS Resource Tags in the **IAM User Guide**. Also see the advanced information (p. 516) about tags.

   **Warning**
   If you update these tags after you create them, the changes are NOT automatically propagated to the underlying resources. For more information, see Propagating Tag Updates to Underlying Resources (p. 516) in the advanced information about tags (p. 516).

13. Choose **Next step**.

14. On the **Review** page, choose **Create environment**. Wait while AWS Cloud9 creates your environment. This can take several minutes.

   **Note**
   If account creation fails, a banner is displayed at the top of the console page. Additionally, the card for the environment, if it exists, indicates that environment creation failed.

After AWS Cloud9 creates your environment, it displays the AWS Cloud9 IDE for the environment.

If AWS Cloud9 doesn't display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see Cannot Open an Environment (p. 544) in **Troubleshooting**.

**Next Step**

Step 2: Basic Tour of the IDE (p. 38)

**Step 2: Basic Tour of the IDE**

(Previous step: Step 1: Create an Environment (p. 35))

This part of the tutorial introduces some of the ways in which you can use the AWS Cloud9 IDE to create and test applications.

- You can use an **editor** window to create and edit code.
- You can use a **terminal** window or a **Run Configuration** window to run your code without debugging it.
- You can use the **Debugger** window to debug your code.

You will perform these three tasks using JavaScript and the Node.js engine. See Tutorials and Samples (p. 325) for other programming languages.

**Topics**

- Get Your Environment Ready (p. 39)
- Write Some Code (p. 39)
- Run Your Code (p. 39)
- Debug Your Code (p. 41)
- Next Step (p. 43)
Get Your Environment Ready

Most of the tools you need to run and debug JavaScript code are already installed for you. However, you need one additional Node.js package for this tutorial. Install it as follows.

1. On the menu bar (at the top of the AWS Cloud9 IDE), choose **Window, New Terminal** (or use an existing terminal window).
2. In the terminal window (one of the tabs in the bottom portion of the IDE), enter the following.

   ```bash
   npm install readline-sync
   ```

   Verify that the result is similar to the following.

   ```
   + readline-sync@1.4.10
   added 1 package from 1 contributor and audited 5 packages in 0.565s
   found 0 vulnerabilities
   ```

Write Some Code

Begin by writing some code.

1. On the menu bar, choose **File, New File**.
2. Add the following JavaScript to the new file.

   ```javascript
   var readline = require('readline-sync');
   var i = 10;
   var input;

   console.log("Hello Cloud9!");
   console.log("i is " + i);

   do {
     input = readline.question("Enter a number (or 'q' to quit): ");
     if (input === 'q') {
       console.log('OK, exiting.');
     } else{
       i += Number(input);
       console.log("i is now " + i);
     }
   } while (input != 'q');

   console.log("Goodbye!");
   ```

3. Choose **File, Save**, and then save the file as **hello-cloud9.js**.

Run Your Code

Next, you can run your code.

Depending on the programming language you are using, there might be multiple ways in which you can run code. For this tutorial, we are using JavaScript, which you can run using a terminal window or a **Run Configuration** window.
To Run the Code Using a Run Configuration Window

1. On the menu bar, choose Run, Run Configurations, New Run Configuration.
2. In the new Run Configuration window (one of the tabs in the bottom portion of the IDE), enter hello-cloud9.js in the Command field, and then choose Run.
3. Be sure that the Run Configuration prompt is active, and then interact with the application by entering a number at the prompt.
4. View the output from your code in the Run Configuration window. It should be similar to the following.

```
Hello Cloud9!
i is 10
Enter a number (or 'q' to quit): 5
i is now 15
Enter a number (or 'q' to quit): q
Exiting.
Goodbye!
Waiting for the debugger to disconnect...
```

To Run the Code Using a Terminal Window

1. Go to the terminal window that you used earlier (or open a new one).
2. In the terminal window, enter `ls` at the terminal prompt, and verify that your code file is in the list of files.
3. Enter `node hello-cloud9.js` at the prompt to start the application.
4. Interact with the application by entering a number at the prompt.
5. View the output from your code in the terminal window. It should be similar to the following.
Finally, you can debug your code by using the Debugger window.

1. Add a breakpoint to your code at line 10 (`if (input === 'q')`) by choosing the margin next to line 10. A red circle is displayed next to that line number, as follows.

2. Open the Debugger window by choosing the Debugger button on the right side of the IDE. Alternatively, choose Window, Debugger on the menu bar.

Then, put a watch on the `input` variable by choosing Type an expression here in the Watch Expressions section of the Debugger window.
3. Go to the Run Configuration window that you used earlier to run the code. Choose Run.

Alternately, you can open a new Run Configuration window and start running the code by choosing Run, Run With, Node.js, from the menu bar.

4. Enter a number at the Run Configuration prompt and see that the code pauses at line 10. The Debugger window shows the value you entered in Watch Expressions.

5. In the Debugger window, choose Resume, which is the blue arrow icon that is highlighted in the previous screenshot.

6. Select Stop in the Run Configuration window to stop the debugger.
Next Step

Step 3: Clean Up (p. 43)

Step 3: Clean Up

(Previous step: Step 2: Basic Tour of the IDE (p. 38))

To prevent ongoing charges to your AWS account related to this tutorial, you should delete the environment.

Warning
Deleting an environment cannot be undone.

Delete the Environment by Using the AWS Cloud9 Console

1. To open the dashboard, on the menu bar in the IDE, choose AWS Cloud9, Go To Your Dashboard.
2. Do one of the following:
   - Choose the title inside of the my-demo-environment card, and then choose Delete.
   - Select the my-demo-environment card, and then choose Delete.
In the **Delete** dialog box, enter **Delete**, and then choose **Delete**. The delete operation will take a few minutes.

**Note**

If you followed this tutorial exactly, then the environment was an EC2 environment and AWS Cloud9 also terminates the Amazon EC2 instance that was connected to that environment. However, if you used an SSH environment instead of following the tutorial, and that environment was connected to an Amazon EC2 instance, AWS Cloud9 doesn't terminate that instance. If you don't terminate that instance later, your AWS account might continue to have ongoing charges for Amazon EC2 related to that instance.

**Next Step**

[Related Information (p. 44)]

**Related Information**

The following is additional information for Tutorial: Hello AWS Cloud9 (console) (p. 35).

- When you create an EC2 environment, the environment doesn't contain any sample code by default. To create an environment with sample code, see one of the following topics:
  - Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)
- During creation of the AWS Cloud9 development environment, you directed AWS Cloud9 to create an Amazon EC2 instance. AWS Cloud9 created the instance and then connected the environment to it. As an alternative, you can use an existing cloud compute instance or your own server (which is called an **SSH environment**). For more information, see Creating an Environment in AWS Cloud9 (p. 55).

**Optional Next Steps**

Explore any or all of the following topics to continue getting familiar with AWS Cloud9.
<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more about what you can do with an environment.</td>
<td>Working with Environments in AWS Cloud9 (p. 55)</td>
</tr>
<tr>
<td>Try other computer languages.</td>
<td>Tutorials and Samples for AWS Cloud9 (p. 325)</td>
</tr>
<tr>
<td>Learn more about the AWS Cloud9 IDE.</td>
<td>Tour the AWS Cloud9 IDE (p. 100) in Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real time and with text chat support.</td>
<td>Working with Shared Environments in AWS Cloud9 (p. 80)</td>
</tr>
<tr>
<td>Create SSH environments (environments that use cloud compute instances or servers that you create, instead of an Amazon EC2 instances that AWS Cloud9 creates for you).</td>
<td>Creating an Environment in AWS Cloud9 (p. 55) and AWS Cloud9 SSH Development Environment host requirements (p. 479)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with Lambda.</td>
<td>AWS Lambda Tutorial for AWS Cloud9 (p. 325), Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342), and Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 284)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with Amazon Lightsail.</td>
<td>Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with the AWS CLI, the aws-shell, AWS CodeCommit, the AWS Cloud Development Kit (AWS CDK), GitHub, or Amazon DynamoDB, as well as Node.js, Python, or other programming languages.</td>
<td>Tutorials and Samples (p. 325)</td>
</tr>
<tr>
<td>Work with code for intelligent robotics applications in AWS RoboMaker.</td>
<td>Developing with AWS Cloud9 in the AWS RoboMaker Developer Guide</td>
</tr>
</tbody>
</table>

To get help with AWS Cloud9 from the community, see the AWS Cloud9 Discussion Forum. (When you enter this forum, AWS might require you to sign in.)

To get help with AWS Cloud9 directly from AWS, see the support options on the AWS Support page.

**Tutorial: Hello AWS Cloud9 (CLI)**

This tutorial provides a first look at AWS Cloud9. It uses the AWS Command Line Interface (AWS CLI), which enables you to set up and tear down the resources you need by using a command line instead of a graphical user interface (p. 35).
In this tutorial, you set up an AWS Cloud9 development environment and then use the AWS Cloud9 IDE to code, run, and debug your first application.

This tutorial should take approximately an hour.

**Warning**
Completing this tutorial might result in charges to your AWS account. These include possible charges for Amazon EC2. For more information, see Amazon EC2 Pricing.

**Prerequisites**

To successfully complete this tutorial, you must first complete the steps in Setting Up AWS Cloud9 (p. 6).

**Steps**

- **Step 1: Create an Environment** (p. 46)
- **Step 2: Basic Tour of the IDE** (p. 47)
- **Step 3: Clean Up** (p. 52)
- **Related Information** (p. 53)

**Step 1: Create an Environment**

(First step of Tutorial: Hello AWS Cloud9 (CLI) (p. 45))

In this step, you use the AWS CLI to create an AWS Cloud9 development environment.

In AWS Cloud9, a *development environment* (or just *environment*) is a place where you store your development project's files and where you run the tools to develop your applications. In this tutorial, you create a special kind of environment called an *EC2 environment*, and then work with the files and tools in that environment.

**Create an EC2 Environment with the AWS CLI**

**Note**
Currently, you cannot use the AWS CLI to create an Ubuntu Server-based EC2 environment—only Amazon Linux. Support for Ubuntu Server is expected in the future.

1. Install and configure the AWS CLI, if you have not done so already. To do this, see the following in the AWS Command Line Interface User Guide:
   - Installing the AWS Command Line Interface
   - Quick Configuration

   We recommend you configure the AWS CLI using credentials for one of the following:

   - The IAM user you created in Team Setup for AWS Cloud9 (p. 8).
   - An IAM administrator user in your AWS account, if you will be working regularly with AWS Cloud9 resources for multiple users across the account. If you cannot configure the AWS CLI as an IAM administrator user, check with your AWS account administrator. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.
   - An AWS account root user, but only if you will always be the only one using your own AWS account, and you don't need to share your environments with anyone else. For more information, see Creating, Disabling, and Deleting Access Keys for Your AWS Account in the Amazon Web Services General Reference.
2. In the following AWS Cloud9 command, provide a value for --region and --subnet-id. Then run the command and make note of the "environmentId" value for later cleanup.

```
aws cloud9 create-environment-ec2 --name my-demo-environment --description "This environment is for the AWS Cloud9 tutorial." --instance-type t2.micro --region MY-REGION --subnet-id subnet-12a3456b
```

In the preceding command:

- **--name** represents the name of the environment. In this tutorial, we use the name *my-demo-environment*.

- **--description** represents an optional description for the environment.

- **--instance-type** represents the type of Amazon EC2 instance AWS Cloud9 will launch and connect to the new environment. This example specifies *t2.micro*, which has relatively low RAM and vCPUs and is sufficient for this tutorial. Specifying instance types with more RAM and vCPUs might result in additional charges to your AWS account for Amazon EC2. For a list of available instance types, see the create environment wizard in the AWS Cloud9 console.

- **--region** represents the ID of the AWS Region for AWS Cloud9 to create the environment in. For a list of available AWS Regions, see AWS Cloud9 in the Amazon Web Services General Reference.

- **--subnet-id** represents the subnet you want AWS Cloud9 to use. Replace *subnet-12a3456b* with the ID of the subnet of an Amazon Virtual Private Cloud (VPC), which must be compatible with AWS Cloud9. For more information, see Create an Amazon VPC for AWS Cloud9 (p. 477) in VPC Settings for AWS Cloud9 Development Environments (p. 467).

- By default, AWS Cloud9 shuts down the Amazon EC2 instance for the environment 30 minutes after all web browser instances that are connect to the IDE for the environment have been closed. To change this, add **--automatic-stop-time-minutes** and the number of minutes. A shorter time period might result in fewer charges to your AWS account. Likewise, a longer time might result in more charges.

- By default, the entity that calls this command owns the environment. To change this, add **--owner-id** and the Amazon Resource Name (ARN) of the owning entity.

3. After you successfully run this command, open the AWS Cloud9 IDE for the newly created environment. To do this, see Opening an Environment in AWS Cloud9 (p. 67). Then return to this topic and continue with Step 2: Basic Tour of the IDE (p. 38) to learn how to use the AWS Cloud9 IDE to work with your new environment.

If you try to open the environment, but AWS Cloud9 doesn't display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see Cannot Open an Environment (p. 544) in Troubleshooting.

**Next Step**

Step 2: Basic Tour of the IDE (p. 47)
• You can use a **terminal** window or a **Run Configuration** window to run your code without debugging it.
• You can use the **Debugger** window to debug your code.

You will perform these three tasks using JavaScript and the Node.js engine. See Tutorials and Samples (p. 325) for other programming languages.

**Topics**
- Get Your Environment Ready (p. 48)
- Write Some Code (p. 48)
- Run Your Code (p. 49)
- Debug Your Code (p. 50)
- Next Step (p. 52)

**Get Your Environment Ready**

Most of the tools you need to run and debug JavaScript code are already installed for you. However, you need one additional Node.js package for this tutorial. Install it as follows.

1. On the menu bar (at the top of the AWS Cloud9 IDE), choose **Window, New Terminal** (or use an existing terminal window).
2. In the terminal window (one of the tabs in the bottom portion of the IDE), enter the following.

```
npm install readline-sync
```

Verify that the result is similar to the following.

```
+ readline-sync@1.4.10
added 1 package from 1 contributor and audited 5 packages in 0.565s
found 0 vulnerabilities
```

**Write Some Code**

Begin by writing some code.

1. On the menu bar, choose **File, New File**.
2. Add the following JavaScript to the new file.

```javascript
var readline = require('readline-sync');
var i = 10;
var input;

console.log("Hello Cloud9!");
console.log("i is " + i);

do {
    input = readline.question("Enter a number (or 'q' to quit): ");
    if (input === 'q') {
        console.log('OK, exiting.')
    } else{
        i += Number(input);
```
```
    console.log("i is now " + i);
} while (input != 'q');
console.log("Goodbye!");
```

3. Choose File, Save, and then save the file as `hello-cloud9.js`.

### Run Your Code

Next, you can run your code.

Depending on the programming language you are using, there might be multiple ways in which you can run code. For this tutorial, we are using JavaScript, which you can run using a terminal window or a Run Configuration window.

#### To Run the Code Using a Run Configuration Window

1. On the menu bar, choose Run, Run Configurations, New Run Configuration.
2. In the new Run Configuration window (one of the tabs in the bottom portion of the IDE), enter `hello-cloud9.js` in the Command field, and then choose Run.
3. Be sure that the Run Configuration prompt is active, and then interact with the application by entering a number at the prompt.
4. View the output from your code in the Run Configuration window. It should be similar to the following.

![Run Configuration Window](image)

#### To Run the Code Using a Terminal Window

1. Go to the terminal window that you used earlier (or open a new one).
2. In the terminal window, enter `ls` at the terminal prompt, and verify that your code file is in the list of files.
3. Enter `node hello-cloud9.js` at the prompt to start the application.
4. Interact with the application by entering a number at the prompt.
5. View the output from your code in the terminal window. It should be similar to the following.

![Terminal Output](image)

**Debug Your Code**

Finally, you can debug your code by using the **Debugger** window.

1. Add a breakpoint to your code at line 10 (if (input === 'q')) by choosing the margin next to line 10. A red circle is displayed next to that line number, as follows.

![Breakpoint Added](image)

2. Open the **Debugger** window by choosing the **Debugger** button on the right side of the IDE. Alternatively, choose **Window**, **Debugger** on the menu bar.

Then, put a watch on the `input` variable by choosing **Type an expression here** in the **Watch Expressions** section of the **Debugger** window.
3. Go to the Run Configuration window that you used earlier to run the code. Choose Run.

Alternately, you can open a new Run Configuration window and start running the code by choosing Run, Run With, Node.js, from the menu bar.

4. Enter a number at the Run Configuration prompt and see that the code pauses at line 10. The Debugger window shows the value you entered in Watch Expressions.

5. In the Debugger window, choose Resume, which is the blue arrow icon that is highlighted in the previous screenshot.

6. Select Stop in the Run Configuration window to stop the debugger.
Step 3: Clean Up

(Previous step: Step 2: Basic Tour of the IDE (p. 47))

To prevent ongoing charges to your AWS account related to this tutorial, you should delete the environment.

**Warning**
Deleting an environment cannot be undone.

**Delete the Environment with the AWS CLI**

1. Run the AWS Cloud9 `delete-environment` command, specifying the ID of the environment to delete.

   ```bash
   aws cloud9 delete-environment --region MY-REGION --environment-id 12a34567b8cd9012345ef67abcd890e1
   ```

   In the preceding command, replace `MY-REGION` with the AWS Region in which the environment was created and `12a34567b8cd9012345ef67abcd890e1` with the ID of the environment to delete.

   If you didn’t save the ID when you created the environment, the ID can be found by using the AWS Cloud9 console. Select the name of the environment in the console, then find the last part of the Environment ARN.

2. If you created an Amazon VPC for this tutorial and you no longer need it, delete the VPC using the Amazon VPC console at [https://console.aws.amazon.com/vpc](https://console.aws.amazon.com/vpc).

**Next Step**

Related Information (p. 53)
Related Information

The following is additional information for Tutorial: Hello AWS Cloud9 (CLI) (p. 45).

- When you create an EC2 environment, the environment doesn't contain any sample code by default. To create an environment with sample code, see one of the following topics:
  - Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)
- During creation of the AWS Cloud9 development environment, you directed AWS Cloud9 to create an Amazon EC2 instance. AWS Cloud9 created the instance and then connected the environment to it. As an alternative, you can use an existing cloud compute instance or your own server (which is called an SSH environment). For more information, see Creating an Environment in AWS Cloud9 (p. 55).

Optional Next Steps

Explore any or all of the following topics to continue getting familiar with AWS Cloud9.

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more about what you can do with an environment.</td>
<td>Working with Environments in AWS Cloud9 (p. 55)</td>
</tr>
<tr>
<td>Try other computer languages.</td>
<td>Tutorials and Samples for AWS Cloud9 (p. 325)</td>
</tr>
<tr>
<td>Learn more about the AWS Cloud9 IDE.</td>
<td>Tour the AWS Cloud9 IDE (p. 100) in Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your new environment along with you, in real</td>
<td>Working with Shared Environments in AWS Cloud9 (p. 80)</td>
</tr>
<tr>
<td>time and with text chat support.</td>
<td></td>
</tr>
<tr>
<td>Create SSH environments (environments that use cloud compute instances</td>
<td>Creating an Environment in AWS Cloud9 (p. 55) and AWS Cloud9 SSH Development Environment host requirements (p. 479)</td>
</tr>
<tr>
<td>or servers that you create, instead of an Amazon EC2 instances that</td>
<td></td>
</tr>
<tr>
<td>AWS Cloud9 creates for you).</td>
<td></td>
</tr>
<tr>
<td>Use AWS Cloud9 with Lambda.</td>
<td>AWS Lambda Tutorial for AWS Cloud9 (p. 325), Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342), and Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 284)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with Amazon Lightsail.</td>
<td>Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with the AWS CLI, the aws-shell, AWS CodeCommit, the</td>
<td>Tutorials and Samples (p. 325)</td>
</tr>
<tr>
<td>AWS Cloud Development</td>
<td></td>
</tr>
</tbody>
</table>
Related Information

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kit (AWS CDK), GitHub, or Amazon DynamoDB, as well as Node.js, Python, or other programming languages.</td>
<td>Developing with AWS Cloud9 in the AWS RoboMaker Developer Guide</td>
</tr>
<tr>
<td>Work with code for intelligent robotics applications in AWS RoboMaker.</td>
<td></td>
</tr>
</tbody>
</table>

To get help with AWS Cloud9 from the community, see the AWS Cloud9 Discussion Forum. (When you enter this forum, AWS might require you to sign in.)

To get help with AWS Cloud9 directly from AWS, see the support options on the AWS Support page.
Working with Environments in AWS Cloud9

A *development environment* is a place in AWS Cloud9 where you store your project's files and where you run the tools to develop your applications.

AWS Cloud9 provides two types of development environments: *EC2 environments* and *SSH environments*. To understand the key similarities and differences between these development environments, see EC2 Environments versus SSH Environments in AWS Cloud9 (p. 466).

Learn how to work with an environment in AWS Cloud9 by reading one or more of these topics.

**Topics**
- Creating an Environment in AWS Cloud9 (p. 55)
- Opening an Environment in AWS Cloud9 (p. 67)
- Calling AWS Services from an Environment in AWS Cloud9 (p. 70)
- Changing Environment Settings in AWS Cloud9 (p. 77)
- Working with Shared Environments in AWS Cloud9 (p. 80)
- Moving an environment or resizing an Amazon EBS volume (p. 92)
- Deleting an Environment in AWS Cloud9 (p. 95)

Creating an Environment in AWS Cloud9

To create an AWS Cloud9 development environment, follow one of these sets of procedures, depending on how you plan to use AWS Cloud9.

If you're not sure what to choose, we recommend Creating an EC2 Environment (p. 56).

Creating an EC2 environment is the easiest option. AWS Cloud9 automatically creates and sets up a new Amazon EC2 instance in your AWS account. AWS Cloud9 then automatically connects that new instance to the environment for you.

To understand the key similarities and differences between the development environments, see EC2 Environments versus SSH Environments in AWS Cloud9 (p. 466).

<table>
<thead>
<tr>
<th>Source code provided by</th>
<th>Development environment host provided by</th>
<th>Follow these procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>You</td>
<td>AWS Cloud9</td>
<td>This topic (create an EC2 environment (p. 56))</td>
</tr>
<tr>
<td>You</td>
<td>You</td>
<td>This topic (create an SSH environment (p. 65))</td>
</tr>
<tr>
<td>Amazon Lightsail or you</td>
<td>You, by using Lightsail</td>
<td>Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)</td>
</tr>
</tbody>
</table>
Creating an EC2 Environment

Note
Completing this procedure might result in charges to your AWS account. These include possible charges for Amazon EC2. For more information, see Amazon EC2 Pricing.

In this procedure, AWS Cloud9 creates an EC2 environment, creates a new Amazon EC2 instance, and then connects the environment to this newly-created instance. AWS Cloud9 manages this instance's lifecycle, including starting, stopping, and restarting the instance as needed. If you ever delete this environment, AWS Cloud9 automatically terminates this instance.

You can create an AWS Cloud9 EC2 development environment with the AWS Cloud9 console (p. 56) or with code (p. 64).

Prerequisites
Complete the steps in Setting Up AWS Cloud9 (p. 6) so that you can sign in to the AWS Cloud9 console and create environments.

Create an EC2 Environment with the Console

1. Sign in to the AWS Cloud9 console as follows:
   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to https://console.aws.amazon.com/cloud9/.
• If your organization uses AWS Single Sign-On (AWS SSO), see your AWS account administrator for sign-in instructions.

• If you're using an AWS Educate Starter Account, see Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29) in Individual Student Signup.

• If you're a student in a classroom, see your instructor for sign-in instructions.

2. After you sign in to the AWS Cloud9 console, in the top navigation bar, choose an AWS Region to create the environment in. For a list of available AWS Regions, see AWS Cloud9 in the AWS General Reference.

3. Choose the large Create environment button in one of the locations shown below.

   If you have no AWS Cloud9 environments yet, the button is shown on a welcome page.

   If you already have AWS Cloud9 environments, the button is shown as follows.

4. On the Name environment page, for Name, enter a name for your environment.

5. To add a description to your environment, enter it in Description.

6. Choose Next step.

7. On the Configure settings page, for Environment type, choose Create a new instance for environment (EC2).
**Warning**
Choosing **Create a new instance for environment (EC2)** might result in possible charges to your AWS account for Amazon EC2.

8. For **Instance type**, choose an instance type with the amount of RAM and vCPUs you think you need for the kinds of tasks you want to do.

**Warning**
Choosing instance types with more RAM and vCPUs might result in additional charges to your AWS account for Amazon EC2.

9. For **Platform**, choose the type of Amazon EC2 instance that you want: **Amazon Linux** or **Ubuntu**. AWS Cloud9 creates the instance and then connects the environment to it.

10. Choose a value for **Cost-saving setting**. When all web browser instances that are connected to the IDE for the environment are closed, AWS Cloud9 waits this amount of time and then shuts down the Amazon EC2 instance for the environment.

**Warning**
Choosing a longer time period might result in more charges to your AWS account.

11. Expand **Network settings (advanced)**.

12. AWS Cloud9 uses Amazon Virtual Private Cloud (Amazon VPC) to communicate with the newly created Amazon EC2 instance. Depending on how Amazon VPC is set up, do one of the following.

<table>
<thead>
<tr>
<th>Does the AWS account have access to an Amazon VPC?</th>
<th>Is that VPC in the same AWS account and AWS Region as the new environment?</th>
<th>Is that VPC the default VPC for its AWS account?</th>
<th>Does that VPC contain a single subnet?</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>If no VPC exists, create one.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>To create a VPC in the same AWS account and AWS Region as the new environment, choose <strong>Create new VPC</strong>, and then follow the on-screen directions. For more information, see <strong>Create an Amazon VPC for AWS Cloud9</strong> (p. 477).</td>
</tr>
<tr>
<td>Does the AWS account have access to an Amazon VPC?</td>
<td>Is that VPC in the same AWS account and AWS Region as the new environment?</td>
<td>Is that VPC the default VPC for its AWS account?</td>
<td>Does that VPC contain a single subnet?</td>
<td>Do this</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>-------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Skip ahead to the next step in this procedure. When you skip past Network settings (advanced) and leave the preselected default settings, AWS Cloud9 attempts to automatically use the default VPC with its single subnet in the same account and region as the new environment.</td>
</tr>
</tbody>
</table>

To create a VPC in a different AWS account than the new environment, see Working with Shared VPCs in the Amazon VPC User Guide.
<table>
<thead>
<tr>
<th>Does the AWS account have access to an Amazon VPC?</th>
<th>Is that VPC in the same AWS account and AWS Region as the new environment?</th>
<th>Is that VPC the default VPC for its AWS account?</th>
<th>Does that VPC contain a single subnet?</th>
<th>Do this</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>If the default VPC has multiple subnets, expand <strong>Network settings (advanced)</strong>. For <strong>Subnet</strong>, choose the subnet you want AWS Cloud9 to use in the preselected default VPC. If the default VPC has no subnets, create one. To do this, choose <strong>Create new subnet</strong>, and then follow the on-screen directions. For more information, see Create a Subnet for AWS Cloud9 (p. 478).</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Expand <strong>Network settings</strong>. For <strong>Network (VPC)</strong>, choose the VPC that you want AWS Cloud9 to use.</td>
</tr>
<tr>
<td>Does the AWS account have access to an Amazon VPC?</td>
<td>Is that VPC in the same AWS account and AWS Region as the new environment?</td>
<td>Is that VPC the default VPC for its AWS account?</td>
<td>Does that VPC contain a single subnet?</td>
<td>Do this</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
<td>------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Expand <strong>Network settings</strong>. For <strong>Network (VPC)</strong>, choose the VPC that you want AWS Cloud9 to use.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>If the chosen VPC has multiple subnets, expand <strong>Network settings (advanced)</strong>. For <strong>Subnet</strong>, choose the subnet you want AWS Cloud9 to use in the chosen VPC.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>If the chosen VPC has no subnets, create one. To do this, choose <strong>Create new subnet</strong>, and then follow the on-screen directions. For more information, see <strong>Create a Subnet for AWS Cloud9</strong> (p. 478).</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>—</td>
<td>AWS Cloud9 cannot use a default VPC in an AWS account than is different than the AWS account for the new environment. Choose a different option in this list.</td>
</tr>
<tr>
<td>Does the AWS account have access to an Amazon VPC?</td>
<td>Is that VPC in the same AWS account and AWS Region as the new environment?</td>
<td>Is that VPC the default VPC for its AWS account?</td>
<td>Does that VPC contain a single subnet?</td>
<td>Do this</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Expand <strong>Network settings</strong>. For <strong>Network (VPC)</strong>, choose the VPC that you want AWS Cloud9 to use. <strong>Note</strong> The VPC must be in the same AWS Region as the new environment, even if the VPC is in a different AWS account.</td>
</tr>
<tr>
<td>Does the AWS account have access to an Amazon VPC?</td>
<td>Is that VPC in the same AWS account and AWS Region as the new environment?</td>
<td>Is that VPC the default VPC for its AWS account?</td>
<td>Does that VPC contain a single subnet?</td>
<td>Do this</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Expand <strong>Network settings</strong>. For <strong>Network (VPC)</strong>, choose the VPC that you want AWS Cloud9 to use. For <strong>Subnet</strong>, choose the subnet you want AWS Cloud9 to use in the chosen VPC. If the chosen VPC has no subnets, to create a subnet for a VPC in a different AWS account than the new environment, see <strong>Working with Shared VPCs</strong> in the <em>Amazon VPC User Guide</em>. <strong>Note</strong> The VPC and subnet must be in the same AWS Region as the new environment, even if the VPC and subnet are in a different AWS account.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

For more information about these choices, see [VPC Settings for AWS Cloud9 Development Environments](#) (p. 467).
13. Add up to 50 tags by supplying a **Key** and a **Value** for each tag. The tags will be attached to the AWS Cloud9 environment as resource tags, and are propagated to the following underlying resources: the AWS CloudFormation stack, the Amazon EC2 instance, and Amazon EC2 security groups. You can find information about tags in *Control Access Using AWS Resource Tags* in the *IAM User Guide*. Also see the advanced information (p. 516) about tags.

**Warning**
If you update these tags after you create them, the changes are NOT automatically propagated to the underlying resources. For more information, see *Propagating Tag Updates to Underlying Resources* (p. 516) in the advanced information about tags (p. 516).

14. Choose **Next step**.

15. On the **Review** page, choose **Create environment**. Wait while AWS Cloud9 creates your environment. This can take several minutes.

**Note**
If account creation fails, a banner is displayed at the top of the console page. Additionally, the card for the environment, if it exists, indicates that environment creation failed.

After AWS Cloud9 creates your environment, it displays the AWS Cloud9 IDE for the environment.

If AWS Cloud9 doesn’t display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see *Cannot Open an Environment* (p. 544) in *Troubleshooting*.

### Creating an EC2 Environment with Code

To use code to create an EC2 environment in AWS Cloud9, call the AWS Cloud9 create EC2 environment operation, as follows.

**Note**
Currently, you cannot use code to create an Ubuntu Server-based EC2 environment, for example by using the AWS CLI, AWS CloudFormation, the AWS SDKs, the Tools for Windows PowerShell, or the AWS Cloud9 API. Currently, you can only use code to create an EC2 environment that is connected to Amazon Linux. Using code to create an Ubuntu Server-based EC2 environment is expected in the future.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>create-environment-ec2</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>CreateEnvironmentEC2Request, CreateEnvironmentEC2Result</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>CreateEnvironmentEC2, CreateEnvironmentEC2Request, CreateEnvironmentEC2WithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>CreateEnvironmentEC2Request, CreateEnvironmentEC2Result</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>createEnvironmentEC2</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>CreateEnvironmentEC2Request, CreateEnvironmentEC2Response</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>createEnvironmentEC2</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>create_environment_ec2</td>
</tr>
</tbody>
</table>
Creating an SSH Environment

You create an AWS Cloud9 SSH development environment with the AWS Cloud9 console. (You cannot create an SSH environment with code.)

Prerequisites

- Make sure you completed the steps in Setting Up AWS Cloud9 (p. 6) first, so that you can sign in to the AWS Cloud9 console and create environments.
- Identify an existing cloud compute instance (for example an Amazon EC2 instance in your AWS account), or your own server, that you want AWS Cloud9 to connect to the environment.
- Make sure that the existing instance or your own server meets all of the SSH host requirements (p. 480). This includes having specific versions of Python, Node.js, and other components installed; setting specific permissions on the directory that you want AWS Cloud9 to start from after login; and setting up any associated Amazon Virtual Private Cloud.

Create the SSH Environment

1. Make sure you completed the preceding prerequisites.
2. Connect to your existing instance or your own server by using an SSH client, if you are not already connected to it. You must do this so that you can add the necessary public SSH key value to the instance or server, as described later in this procedure.

   Note
   To connect to an existing AWS cloud compute instance, see one or more of the following resources:
   - For Amazon EC2, see Connect to Your Linux Instance in the Amazon EC2 User Guide for Linux Instances.
   - For Amazon Lightsail, see Connect to your Linux/Unix-based Lightsail instance in the Amazon Lightsail Documentation.
   - For AWS Elastic Beanstalk, see Listing and Connecting to Server Instances in the AWS Elastic Beanstalk Developer Guide.
   - For AWS OpsWorks, see Using SSH to Log In to a Linux Instance in the AWS OpsWorks User Guide.
   - For other AWS services, see the service's documentation.

   To connect to your own server, you could search the internet using a phrase such as "connect to a server by using the ssh command" (from macOS or Linux) or "connect to a server by using PuTTY" (from Windows).
4. After you sign in to the AWS Cloud9 console, in the top navigation bar, choose an AWS Region to create the environment in. For a list of available AWS Regions, see AWS Cloud9 in the AWS General Reference.
5. If a welcome page is displayed, for New AWS Cloud9 environment, choose Create environment. Otherwise, choose Create environment.

Or:

6. On the Name environment page, for Name, type a name for your environment.
7. To add a description to your environment, type it in Description.
8. Choose Next step.
9. For Environment type, choose Connect and run in remote server (SSH).
10. For User, type the login name you used to connect to the instance or server earlier in this procedure. For example, for an AWS cloud compute instance, it might be ec2-user, ubuntu, or root.

Note
For best results, we recommend that the login name is associated with administrative privileges or an administrator user on the instance or server. Specifically, this login name should own the Node.js installation on the instance or server. To check this, from your instance's or server's terminal, run the command `ls -l $(which node)` (or `ls -l $(nvm which node)` if you're using nvm). This command displays the Node.js installation's owner's name (along with the installation's permissions, group name, and location).
11. For **Host**, type the public IP address (preferred) or the hostname of the instance or server.
12. For **Port**, type the port that you want AWS Cloud9 to use to try to connect to the instance or server, or leave the default port.
13. To specify the path to the directory on the instance or server that you want AWS Cloud9 to start from after login, which you identified earlier in this procedure's prerequisites, expand **Advanced settings**, and then type the path in **Environment path**. If you leave this blank, AWS Cloud9 uses the directory that your instance or server typically starts with after login. This is usually a home or default directory.
14. To specify the path to the Node.js binary on the instance or server, expand **Advanced settings**, and then type the path in **Node.js binary path**. To get the path, you can run the command `which node` (or `nvm which node` if you're using nvm) on your instance or server. For example, the path might be `/usr/bin/node`. If you leave this blank, AWS Cloud9 will try to guess where the Node.js binary is when it tries to connect.
15. To specify a jump host that the instance or server uses, expand **Advanced settings**, and then type information about the jump host in **SSH jump host**, using the format `USER_NAME@HOSTNAME:PORT_NUMBER` (for example, `ec2-user@ip-192-0-2-0:22`).

The jump host must meet the following requirements.

- It must be reachable over the public Internet using SSH.
- It must allow inbound access by any IP address over the specified port.
- The public SSH key value that was copied into the `~/.ssh/authorized_keys` file on the existing instance or server must also be copied into the `~/.ssh/authorized_keys` file on the jump host.
- Netcat must be installed.
16. Choose **Copy key to clipboard**. (This is between **View public SSH key** and **Advanced settings**.) Paste the public SSH key value that was copied, into the `~/.ssh/authorized_keys` file on the existing instance or server that you connected to earlier in this procedure. (~ represents the home directory for the login name that you specified for **User** earlier in this procedure.)

**Note**
To see the public SSH key value that was copied, expand **View public SSH key**.

17. Choose **Next step**.
18. On the **Review** page, choose **Create environment**. Wait while AWS Cloud9 creates your environment. This can take several minutes.

If AWS Cloud9 doesn't display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated network. For possible fixes, see **Cannot Open an Environment (p. 544)** in Troubleshooting.

### Opening an Environment in AWS Cloud9

This procedure describes how to open an environment in AWS Cloud9.

**Note**
This procedure assumes you have already created an AWS Cloud9 development environment. To create an environment, see **Creating an Environment (p. 55)**.

1. Sign in to the AWS Cloud9 console as follows:
   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to [https://console.aws.amazon.com/cloud9/](https://console.aws.amazon.com/cloud9/).
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
• If you’re using an AWS Educate Starter Account, see Step 2: Sign in to the AWS Cloud9 Console (p. 29) in Individual Student Signup.

• If you’re a student in a classroom, see your instructor for sign-in instructions.

  **Important**
  If you sign out of your AWS Account, the AWS Cloud9 IDE can still be accessed for up to five minutes afterwards. Access is then denied when the required permissions expire.

2. In the top navigation bar, choose the AWS Region where the environment is located.

3. In the list of environments, for the environment you want to open, do one of the following.
   • Inside of the card, choose the **Open IDE** link.
   • Select the card, and then choose the **Open IDE** button.
If your environment is not displayed in the console, try doing one or more of the following actions to try to display it.

- In the side navigation bar, choose one or more of the following.
  - Choose **Your environments** to display all environments that your AWS entity owns within the selected AWS Region and AWS account.
  - Choose **Shared with you** to display all environments your AWS entity has been invited to within the selected AWS Region and AWS account.
  - Choose **Account environments** to display all environments within the selected AWS Region and AWS account that your AWS entity has permissions to display.

- Choose the previous arrow, next arrow, or page number button to display more environments in the current scope.

- If you think you should be a member of an environment, but the environment is not displayed in the **Shared with you** list, check with the environment owner.
- In the top navigation bar, choose a different AWS Region.
Calling AWS Services from an Environment in AWS Cloud9

You can call AWS services from an AWS Cloud9 development environment. For example, you can:

- Upload and download data in Amazon Simple Storage Service (Amazon S3) buckets.
- Send broadcast notifications through Amazon Simple Notification Service (Amazon SNS) topics.
- Read and write data in Amazon DynamoDB (DynamoDB) databases.

You can call AWS services from your environment in several ways. For example, you can use the AWS Command Line Interface (AWS CLI) or the aws-shell to run commands from a terminal session. You can also call AWS services from code you run within your environment, using AWS SDKs for programming languages such as JavaScript, Python, Ruby, PHP, Go, and C++. For more information, see the AWS CLI and aws-shell Sample (p. 352), the AWS Command Line Interface User Guide, and the AWS SDKs.

Each time the AWS CLI, the aws-shell, or your code calls an AWS service, the AWS CLI, the aws-shell, or your code must provide a set of AWS access credentials along with the call. These credentials determine whether the caller has the appropriate permissions to make the call. If the credentials don't cover the appropriate permissions, the call will fail.

There are several ways to provide credentials to your environment. The following table describes some approaches.

<table>
<thead>
<tr>
<th>Environment type</th>
<th>Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2</td>
<td>Use AWS managed temporary credentials. We recommend this approach for an EC2 environment. AWS managed temporary credentials manage AWS access credentials in an EC2 environment on your behalf, while also following AWS security best practices. If you are using an EC2 environment, you can skip the rest of this topic, as AWS managed</td>
</tr>
</tbody>
</table>
Create and Use an Instance Profile to Manage Temporary Credentials

<table>
<thead>
<tr>
<th>Environment type</th>
<th>Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>temporary credentials are already set up for you in the environment.</td>
<td></td>
</tr>
<tr>
<td>For more information, see AWS Managed Temporary Credentials (p. 536).</td>
<td></td>
</tr>
<tr>
<td>EC2</td>
<td>Attach an IAM instance profile to the instance.</td>
</tr>
<tr>
<td>You should only use this approach if for some reason you can't use AWS managed temporary credentials. Similar to AWS managed temporary credentials, an instance profile manages AWS access credentials on your behalf. However, you must create, manage, and attach the instance profile to the Amazon EC2 instance yourself.</td>
<td></td>
</tr>
<tr>
<td>For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71).</td>
<td></td>
</tr>
<tr>
<td>EC2 or SSH</td>
<td>Store your permanent AWS access credentials within the environment.</td>
</tr>
<tr>
<td>This approach is less secure than using temporary AWS access credentials. However, it is the only supported approach for an SSH environment.</td>
<td></td>
</tr>
<tr>
<td>For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).</td>
<td></td>
</tr>
<tr>
<td>EC2 or SSH</td>
<td>Insert your permanent AWS access credentials directly into your code.</td>
</tr>
<tr>
<td>We discourage this approach because it doesn't follow AWS security best practices.</td>
<td></td>
</tr>
<tr>
<td>Because we discourage this approach, we do not cover it in this topic.</td>
<td></td>
</tr>
</tbody>
</table>

Create and Use an Instance Profile to Manage Temporary Credentials

**Note**
You cannot use this procedure for an AWS Cloud9 SSH development environment. Instead, skip ahead to Create and Store Permanent Access Credentials in an Environment (p. 75).
We recommend using AWS managed temporary credentials instead of an instance profile. Follow these instructions only if for some reason you cannot use AWS managed temporary credentials. For more information, see AWS Managed Temporary Credentials (p. 536).

In this procedure, you will use IAM and Amazon EC2 to create and attach an IAM instance profile to the Amazon EC2 instance that connects to your environment. This instance profile will manage temporary credentials on your behalf. This procedure assumes you have already created a environment in AWS Cloud9. To create a environment, see Create an Environment (p. 55).
Create and Use an Instance Profile to Manage Temporary Credentials

You can complete these tasks with the IAM and Amazon EC2 consoles (p. 72) or the AWS Command Line Interface (AWS CLI) (p. 72).

Create an Instance Profile with the IAM Console

Note
If you already have an IAM role that contains an instance profile, skip ahead to Attach an Instance Profile to an Instance with the Amazon EC2 Console (p. 73).


   For this step, we recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.

2. In the navigation bar, choose Roles.

   Note
   You cannot use the IAM console to create an instance profile by itself. You must create an IAM role, which contains an instance profile.

3. Choose Create role.

4. On the Select type of trusted entity page, with AWS service already chosen, for Choose the service that will use this role, choose EC2.

5. For Select your use case, choose EC2.

6. Choose Next: Permissions.

7. On the Attach permissions policies page, in the list of policies, select the box next to AdministratorAccess, and then choose Next: Review.

   Note
   The AdministratorAccess policy allows unrestricted access to all AWS actions and resources across your AWS account. It should be used only for experimentation purposes. For more information, see IAM Policies in the IAM User Guide.

8. On the Review page, for Role Name, type a name for the role (for example my-demo-cloud9-instance-profile).

9. Choose Create Role.

Skip ahead to Attach an Instance Profile to an Instance with the Amazon EC2 Console (p. 73).

Create an Instance Profile with the AWS CLI

Note
If you already have an IAM role that contains an instance profile, skip ahead to Attach an Instance Profile to an Instance with the AWS CLI (p. 74).

For this topic, we recommend you configure the AWS CLI using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.

Note
If you’re using AWS managed temporary credentials (p. 501), you can’t use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don’t allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

1. Define a trust relationship in AWS for the instance profile’s required IAM role. To do this, create and then save a file with the following contents (for example, as my-demo-cloud9-instance-profile-role-trust.json).
2. Using the terminal or command prompt, switch to the directory where you just saved this file.

3. Create an IAM role for the instance profile. To do this, run the IAM `create-role` command, specifying a name for the new IAM role (for example, `my-demo-cloud9-instance-profile-role`), and the name of the file you just saved.

   ```bash
   ```

4. Attach AWS access permissions to the instance profile's IAM role. To do this, run the IAM `attach-role-policy` command, specifying the name of the existing IAM role and the Amazon Resource Name (ARN) of the AWS managed policy named `AdministratorAccess`.

   ```bash
   aws iam attach-role-policy --role-name my-demo-cloud9-instance-profile-role --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
   ```

   **Note**
   The `AdministratorAccess` policy allows unrestricted access to all AWS actions and resources across your AWS account. It should be used only for experimentation purposes. For more information, see IAM Policies in the IAM User Guide.

5. Create the instance profile. To do this, run the IAM `create-instance-profile` command, specifying a name for the new instance profile (for example, `my-demo-cloud9-instance-profile`).

   ```bash
   aws iam create-instance-profile --instance-profile-name my-demo-cloud9-instance-profile
   ```

6. Attach the IAM role to the instance profile. To do this, run the IAM `add-role-to-instance-profile` command, specifying the names of the existing IAM role and instance profile.

   ```bash
   aws iam add-role-to-instance-profile --role-name my-demo-cloud9-instance-profile-role --instance-profile-name my-demo-cloud9-instance-profile
   ```

   Skip ahead to Create an Instance Profile with the AWS CLI (p. 72).

**Attach an Instance Profile to an Instance with the Amazon EC2 Console**


   For this step, we recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.
2. In the navigation bar, be sure the region selector displays the AWS Region that matches the one for your environment. For example, if you created your environment in the US East (Ohio) region, choose **US East (Ohio)** in the region selector here as well.

3. Choose the **Running Instances** link or, in the navigation pane, expand **Instances**, and then choose **Instances**.

4. In the list of instances, choose the instance with the **Name** that includes your environment name. For example, if your environment name is `my-demo-environment`, choose the instance with the **Name** that includes `my-demo-environment`.

5. Choose **Actions, Instance Settings, Attach/Replace IAM Role**.

   **Note**
   Although you are attaching a role to the instance, the role contains an instance profile.

6. On the **Attach/Replace IAM Role** page, for **IAM role**, choose the name of the role you identified or that you created in the previous procedure, and then choose **Apply**.

7. Back in the environment, use the AWS CLI to run the `aws configure` command or the aws-shell to run the `configure` command. Do not specify any values for **AWS Access Key ID** or **AWS Secret Access Key** (press `Enter` after each of these prompts). For **Default region name**, specify the AWS Region closest to you or the region where your AWS resources are located. For example, `us-east-2` for the US East (Ohio) Region. For a list of regions, see **AWS Regions and Endpoints** in the **Amazon Web Services General Reference**. Optionally, specify a value for **Default output format** (for example, `json`).

You can now start calling AWS services from your environment. To use the AWS CLI, the aws-shell, or both to call AWS services, see the **AWS CLI and aws-shell Sample** (p. 352). To call AWS services from your code, see our other tutorials and samples (p. 325).

**Attach an Instance Profile to an Instance with the AWS CLI**

**Note**
If you're using **AWS managed temporary credentials** (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

1. Run the Amazon EC2 `associate-iam-instance-profile` command, specifying the name of the instance profile and the ID and AWS Region ID of the Amazon EC2 instance for the environment.

   ```
   aws ec2 associate-iam-instance-profile --iam-instance-profile Name=my-demo-cloud9-instance-profile --region us-east-2 --instance-id i-12a3b45678cdef9a0
   ```

   In the preceding command, replace `us-east-2` with the AWS Region ID for the instance and `i-12a3b45678cdef9a0` with the instance's ID.

   To get the instance's ID, you could for example run the Amazon EC2 `describe-instances` command, specifying the name and AWS Region ID of the environment.

   ```
   aws ec2 describe-instances --region us-east-2 --filters Name=tag:Name,Values=*my-environment* --query "Reservations[*].Instances[*].InstanceId" --output text
   ```

   In the preceding command, replace `us-east-2` with the AWS Region ID for the instance and `my-environment` with the name of the environment.

2. Back in the environment, use the AWS CLI to run the `aws configure` command or the aws-shell to run the `configure` command. Do not specify any values for **AWS Access Key ID** or **AWS Secret Access Key** (press `Enter` after each of these prompts). For **Default region name**, specify the AWS Region closest to you or the region where your AWS resources are located. For example, `us-east-2`
for the US East (Ohio) Region. For a list of regions, see AWS Regions and Endpoints in the Amazon Web Services General Reference. Optionally, specify a value for Default output format (for example, json).

You can now start calling AWS services from your environment. To use the AWS CLI, the aws-shell, or both to call AWS services, see the AWS CLI and aws-shell Sample (p. 352). To call AWS services from your code, see our other tutorials and samples (p. 325).

Create and Store Permanent Access Credentials in an Environment

Note
If you are using an AWS Cloud9 EC2 development environment, we recommend you use AWS managed temporary credentials instead of AWS permanent access credentials. To work with AWS managed temporary credentials, see AWS managed temporary credentials (p. 536).

In this section, you use AWS Identity and Access Management (IAM) to generate a set of permanent credentials that the AWS CLI, the aws-shell, or your code can use when calling AWS services. This set includes an AWS access key ID and an AWS secret access key, which are unique to your user in your AWS account. If you already have an AWS access key ID and an AWS secret access key, note those credentials, and then skip ahead to Store Permanent Access Credentials in an Environment (p. 76).

You can create a set of permanent credentials with the IAM console (p. 75) or the AWS CLI (p. 75).

Create Permanent Access Credentials with the Console

   - For this step, we recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.
2. In the navigation bar, choose Users.
3. In the list of users, choose the name of the user you created or identified in Team Setup (p. 8).
5. For Access keys, choose Create access key.
6. In the Create access key page, choose Show, and make a note of the Access key ID and Secret access key values. We recommend you also choose Download .csv file and save these credentials in a secure location.

Skip ahead to Store Permanent Access Credentials in an Environment (p. 76).

Create Permanent Access Credentials with the AWS CLI

Note
For this section, we recommend you configure the AWS CLI using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.

Note
If you’re using AWS managed temporary credentials (p. 501), you can’t use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don’t allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

Run the IAM create-access-key command to create a new AWS access key and corresponding AWS secret access key for the user.
aws iam create-access-key --user-name MyUser

In the preceding command, replace MyUser with the name of the user.

In a secure location, save the AccessKeyId and SecretAccessKey values that are displayed. After you run the IAM create-access-key command, this is the only time you can use the AWS CLI to view the user’s AWS secret access key. To generate a new AWS secret access key for the user later if needed, see Creating, Modifying, and Viewing Access Keys (API, CLI, PowerShell) in the IAM User Guide.

Store Permanent Access Credentials in an Environment

In this procedure, you use the AWS Cloud9 IDE to store your permanent AWS access credentials in your environment. This procedure assumes you have already created an environment in AWS Cloud9, opened the environment, and are displaying the AWS Cloud9 IDE in your web browser. For more information, see Creating an Environment (p. 55) and Opening an Environment (p. 67).

Note
The following procedure shows how to store your permanent access credentials by using environment variables. If you have the AWS CLI or the aws-shell installed in your environment, you can use the `aws configure` command for the AWS CLI or the `configure` command for the aws-shell to store your permanent access credentials instead. For instructions, see Quick Configuration in the AWS Command Line Interface User Guide.

1. With your environment open, in the AWS Cloud9 IDE, start a new terminal session, if one is not already started. To start a new terminal session, on the menu bar, choose Window, New Terminal.
2. Run each of the following commands, one command at a time, to set local environment variables representing your permanent access credentials. In these commands, after `AWS_ACCESS_KEY_ID`, type your AWS access key ID. After `AWS_SECRET_ACCESS_KEY`, type your AWS secret access key. After `AWS_DEFAULT_REGION_ID`, type the AWS Region identifier associated with the AWS Region closest to you (or your preferred AWS Region). For a list of available identifiers, see AWS Regions and Endpoints in the Amazon Web Services General Reference. For example, for the US East (Ohio) Region, you would use `us-east-2`.

   ```
   export AWS_ACCESS_KEY_ID=
   export AWS_SECRET_ACCESS_KEY=
   export AWS_DEFAULT_REGION=
   ```

3. Note that the preceding environment variables are valid only for the current terminal session. To make these environment variables available across terminal sessions, you must add them to your shell profile file as user environment variables, as follows.
   a. In the Environment window of the IDE, choose the gear icon, and then choose Show Home in Favorites. Repeat this step and choose Show Hidden Files as well.
   b. Open the `~/.bashrc` file.
   c. Type or paste the following code at the end of the file. In these commands, after `AWS_ACCESS_KEY_ID`, type your AWS access key ID. After `AWS_SECRET_ACCESS_KEY`, type your AWS secret access key. After `AWS_DEFAULT_REGION_ID`, type the AWS Region identifier associated with the AWS Region closest to you (or your preferred AWS Region). For a list of available identifiers, see AWS Regions and Endpoints in the Amazon Web Services General Reference. For example, for the US East (Ohio) Region, you would use `us-east-2`.

   ```
   export AWS_ACCESS_KEY_ID=
   export AWS_SECRET_ACCESS_KEY=
   export AWS_DEFAULT_REGION=
   ```

   d. Save the file.
   e. Source the `~/.bashrc` file to load these new environment variables.
Changing Environment Settings in AWS Cloud9

You can change the preferences or settings for an AWS Cloud9 development environment.

- Change Environment Preferences (p. 77)
- Change Environment Settings with the Console (p. 77)
- Change Environment Settings with Code (p. 80)

Change Environment Preferences

1. Open the environment you want to change settings for. To open an environment, see Opening an Environment (p. 67).
2. In the AWS Cloud9 IDE, on the menu bar, choose AWS Cloud9, Preferences.
3. In the Preferences window, choose Project Settings.
4. Change any of the available project settings as you want. These include settings such as Code Editor (Ace) and Find in Files.

Note
For more information, see Project Setting Changes You Can Make (p. 160).

Change Environment Settings with the Console

1. Sign in to the AWS Cloud9 console as follows:
   - If you’re the only individual using your AWS account or you are an IAM user in a single AWS account, go to https://console.aws.amazon.com/cloud9/.
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
   - If you’re using an AWS Educate Starter Account, see Step 2: Sign in to the AWS Cloud9 Console (p. 29) in Individual Student Signup.
   - If you’re a student in a classroom, see your instructor for sign-in instructions.
2. In the top navigation bar, choose the AWS Region where the environment is located.
3. In the list of environments, for the environment whose settings you want to change, do one of the following.

- Choose the title of the card for the environment. Then choose **Edit** on the next page.

- Select the card for the environment, and then choose the **Edit** button.

4. Make your changes, and then choose **Save changes**.
You can use the AWS Cloud9 console to change the following settings.

- For EC2 environments, **Name** and **Description**.
- For SSH environments: **Name**, **Description**, **User**, **Host**, **Port**, **Environment path**, **Node.js binary path**, and **SSH jump host**.

To change other settings, do the following.

- For EC2 environments, do the following.
  - You cannot change **Type**, **Security groups**, **VPC**, **Subnet**, **Environment path**, or **Environment ARN**.
  - For **Permissions** or **Number of members**, see Change the Access Role of an Environment Member (p. 89), Remove Your User (p. 89), Invite an IAM User (p. 84), and Remove Another Environment Member (p. 91).
  - For **EC2 instance type**, **Memory**, or **vCPU**, see Moving or Resizing an Environment (p. 92).
- For SSH environments, do the following.
  - You cannot change **Type** or **Environment ARN**.
  - For **Permissions** or **Number of members**, see Change the Access Role of an Environment Member (p. 89), Remove Your User (p. 89), Invite an IAM User (p. 84), and Remove Another Environment Member (p. 91).

If your environment is not displayed in the console, try doing one or more of the following actions to try to display it.

- In the side navigation bar, choose one or more of the following.
  - Choose **Your environments** to display all environments that your AWS entity owns within the selected AWS Region and AWS account.
  - Choose **Shared with you** to display all environments your AWS entity has been invited to within the selected AWS Region and AWS account.
  - Choose **Account environments** to display all environments within the selected AWS Region and AWS account that your AWS entity has permissions to display.

- Choose the previous arrow, next arrow, or page number button to display more environments in the current scope.

- If you think you should be a member of an environment, but the environment is not displayed in the **Shared with you** list, check with the environment owner.
- In the top navigation bar, choose a different AWS Region.
Change Environment Settings with Code

To use code to change the settings of an environment in AWS Cloud9, call the AWS Cloud9 update environment operation, as follows.

<table>
<thead>
<tr>
<th>Platform</th>
<th>Command/Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CLI</td>
<td>update-environment</td>
</tr>
<tr>
<td>AWS SDK for C++</td>
<td>UpdateEnvironmentRequest, UpdateEnvironmentResponse</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>UpdateEnvironment, UpdateEnvironmentRequest, UpdateEnvironmentWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>UpdateEnvironmentRequest, UpdateEnvironmentResponse</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>updateEnvironment</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>UpdateEnvironmentRequest, UpdateEnvironmentResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>updateEnvironment</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>update_environment</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>update_environment</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>UpdateEnvironment</td>
</tr>
</tbody>
</table>

Working with Shared Environments in AWS Cloud9

A shared environment is an AWS Cloud9 development environment that multiple users have been invited to participate in. This topic provides instructions for sharing an environment in AWS Cloud9 and how to participate in a shared environment.
To invite a user to participate in an environment you own, follow one of these sets of procedures, depending on the type of user you want to invite.

<table>
<thead>
<tr>
<th>User type</th>
<th>Follow these procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>A user in the same AWS account as the environment.</td>
<td>Invite a User in the Same Account as the Environment (p. 84)</td>
</tr>
<tr>
<td>An AWS Cloud9 administrator in the same AWS account as the environment,</td>
<td>To invite the AWS Cloud9 administrator yourself, see Invite a User in the Same Account as</td>
</tr>
<tr>
<td>specifically:</td>
<td>the Environment (p. 84).</td>
</tr>
<tr>
<td>• The AWS account root user.</td>
<td>To have the AWS Cloud9 administrator invite themself (or others in the same AWS account),</td>
</tr>
<tr>
<td>• An IAM administrator user.</td>
<td>see Have an AWS Cloud9 Administrator in the Same Account as the Environment Invite Themself or Others (p. 85).</td>
</tr>
<tr>
<td>• A user with the AWS managed policy AWSCloud9Administrator attached.</td>
<td></td>
</tr>
</tbody>
</table>

**Contents**

- Shared Environment Usage Scenarios (p. 81)
- About Environment Member Access Roles (p. 82)
- Invite a User in the Same Account as the Environment (p. 84)
- Have an AWS Cloud9 Administrator in the Same Account as the Environment Invite Themself or Others (p. 85)
- Open a Shared Environment (p. 86)
- See a List of Environment Members (p. 86)
- Open the Active File of an Environment Member (p. 87)
- Open the Open File of an Environment Member (p. 88)
- Go to the Active Cursor of an Environment Member (p. 88)
- Chat with Other Environment Members (p. 88)
- View Chat Messages in a Shared Environment (p. 88)
- Delete a Chat Message from a Shared Environment (p. 89)
- Delete All Chat Messages from a Shared Environment (p. 89)
- Change the Access Role of an Environment Member (p. 89)
- Remove Your User From a Shared Environment (p. 90)
- Remove Another Environment Member (p. 91)
- Environment Sharing Best Practices (p. 91)

**Shared Environment Usage Scenarios**

A shared environment is good for the following.

- Pair programming (also known as peer programming). This is where two users work together on the same code in a single environment. In pair programming, typically one user writes code while the other user observes the code being written. The observer gives immediate input and feedback to the code writer. These positions frequently switch during a project. Without a shared environment, teams of pair programmers typically sit in front of a single machine, and only one user at a time can write code. With
About Environment Member Access Roles

Before you share an environment or participate in a shared environment in AWS Cloud9, you should understand the access permission levels for a shared environment. We call these permission levels environment member access roles.

A shared environment in AWS Cloud9 offers three environment member access roles: owner, read/write, and read-only.

- An owner has full control over an environment. Each environment has one and only one owner, who is the environment creator. An owner can do the following.
  - Add, change, and remove members for the environment
  - Open, view, and edit files
  - Run code
  - Change environment settings
  - Chat with other members
  - Delete existing chat messages

In the AWS Cloud9 IDE, an environment owner is displayed with Read+Write access.

- A read/write member can do the following.
  - Open, view, and edit files
  - Run code
  - Change various environment settings from within the AWS Cloud9 IDE
  - Chat with other members
  - Delete existing chat messages

In the AWS Cloud9 IDE, read/write members are displayed with Read+Write access.

- A read-only member can do the following.
  - Open and view files
  - Chat with other members
  - Delete existing chat messages

In the AWS Cloud9 IDE, read-only members are displayed with Read Only access.

Before a user can become an environment owner or member, that user must meet one of the following criteria.

- The user is an AWS account root user.
- The user is an IAM administrator user. For more information, see Creating Your First IAM Admin User and Group in the IAM User Guide.
- The user is a user who belongs to an IAM group, a user who assumes a role, or a federated user who assumes a role, and that group or role has the AWS managed policy AWSCloud9Administrator or
AWS Cloud9 User Guide
About Environment Member Access Roles

AWSCloud9User (or AWSCloud9EnvironmentMember, to be a member only) attached. For more information, see AWS Managed (Predefined) Policies (p. 524).

- To attach one of the preceding managed policies to an IAM group, you can use the AWS Management Console (p. 83) or the AWS Command Line Interface (AWS CLI) (p. 83) as described in the following procedures.
- To create a role in IAM with one of the preceding managed policies for a user or a federated user to assume, see Creating Roles in the IAM User Guide. To have a user or a federated user assume the role, see coverage of assuming roles in Using IAM Roles in the IAM User Guide.

Attach an AWS Managed Policy for AWS Cloud9 to a Group Using the Console

1. Sign in to the AWS Management Console, if you are not already signed in.
   
   For this step, we recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.

2. Open the IAM console. To do this, in the console's navigation bar, choose Services. Then choose IAM.

3. Choose Groups.

4. Choose the group's name.

5. On the Permissions tab, for Managed Policies, choose Attach Policy.

6. In the list of policy names, choose one of the following boxes.
   
   - AWSCloud9User (preferred) or AWSCloud9Administrator to enable each user in the group to be an environment owner
   - AWSCloud9EnvironmentMember to enable each user in the group to be a member only

   (If you don't see one of these policy names in the list, type the policy name in the Search box to display it.)

7. Choose Attach policy.

Attach an AWS Managed Policy for AWS Cloud9 to a Group Using the AWS CLI

Note
If you're using AWS managed temporary credentials (p. 501), you can't use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don't allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

Run the IAM attach-group-policy command to attach the AWS managed policy for AWS Cloud9 to the group, specifying the group's name and the Amazon Resource Name (ARN) of the policy, for example:

```bash
aws iam attach-group-policy --group-name MyGroup --policy-arn arn:aws:iam::aws:policy/POLICY_NAME
```

In the preceding command, replace MyGroup with the name of the group. Replace POLICY_NAME with the name of one of the following AWS managed policies.

- AWSCloud9User (preferred) or AWSCloud9Administrator to enable each user in the group to be an environment owner
- AWSCloud9EnvironmentMember to enable each user in the group to be a member only
Invite a User in the Same Account as the Environment

Use the instructions in this section to share an AWS Cloud9 development environment that you own in your AWS account with a user in that same account.

1. If the user you want to invite is not one of the following types of users, be sure the user you want to invite already has the corresponding environment member access role. For instructions, see About Environment Member Access Roles (p. 82).
   - The AWS account root user.
   - An IAM administrator user.
   - A user who belongs to an IAM group, a user who assumes a role, or a federated user who assumes a role, and that group or role has the AWS managed policy AWSCloud9Administrator attached.

2. Open the environment that you own and want to invite the user to, if the environment is not already open.

3. In the menu bar in the AWS Cloud9 IDE, do one of the following.
   - Choose Window, Share.
   - Choose Share (located next to the Preferences gear icon).

4. In the Share this environment dialog box, for Invite Members, type one of the following.
   - To invite an IAM user, type the user's name.
   - To invite the AWS account root user, type `arn:aws:iam::123456789012:root`, replacing `123456789012` with your AWS account ID.
   - To invite a user with an assumed role or a federated user with an assumed role, type `arn:aws:sts::123456789012:assumed-role/MyAssumedRole/MyAssumedRoleSession`, replacing `123456789012` with your AWS account ID, `MyAssumedRole` with the name of the assumed role, and `MyAssumedRoleSession` with the session name for the assumed role.

5. To make this user a read-only member, choose R. To make this user read/write, choose RW.

6. Choose Invite.

   Note
   If you make this user a read/write member, a dialog box is displayed, containing information about possibly putting your AWS security credentials at risk. The following information provides more background about this issue.
   You should share an environment only with those you trust.
   A read/write member may be able to use the AWS CLI, the aws-shell, or AWS SDK code in your environment to take actions in AWS on your behalf. Furthermore, if you store your permanent AWS access credentials within the environment, that member could potentially copy those credentials and use them outside of the environment.
   Removing your permanent AWS access credentials from your environment and using temporary AWS access credentials instead does not fully address this issue. It lessens the
opportunity of the member to copy those temporary credentials and use them outside of the environment (as those temporary credentials will work only for a limited time). However, temporary credentials still enable a read/write member to take actions in AWS from the environment on your behalf.

7. Contact the user to let them know they can open this environment and begin using it.

Have an AWS Cloud9 Administrator in the Same Account as the Environment Invite Themself or Others

Note
If you’re using AWS managed temporary credentials (p. 501), you can’t use a terminal session in the AWS Cloud9 IDE to run some or all of the commands in this section. To address AWS security best practices, AWS managed temporary credentials don’t allow some commands to be run. Instead, you can run those commands from a separate installation of the AWS Command Line Interface (AWS CLI).

The following types of users can invite themselves (or other users in the same AWS account) to any environment in the same account.

- The **AWS account root user**.
- An **IAM administrator user**.
- A **user who belongs to an IAM group**, a **user who assumes a role**, or a **federated user who assumes a role**, and that group or role has the AWS managed policy **AWSCloud9Administrator** attached.

If the invited user is not one of the preceding types of users, be sure that user already has the corresponding environment member access role. For instructions, see About Environment Member Access Roles (p. 82).

To invite the user, use the AWS CLI or the aws-shell to run the AWS Cloud9 `create-environment-membership` command, as follows.

```
aws cloud9 create-environment-membership --environment-id 12a34567b8cd9012345ef67abcd890e1 --user-arn USER_ARN --permissions PERMISSION_LEVEL
```

In the preceding command, replace `12a34567b8cd9012345ef67abcd890e1` with the ID of the environment, and `PERMISSION_LEVEL` with read-write or read-only. Replace `USER_ARN` with one of the following:

- To invite an **IAM user**, type `arn:aws:iam::123456789012:user/MyUser`, replacing `123456789012` with your AWS account ID and `MyUser` with the user’s name.
- To invite the **AWS account root user**, type `arn:aws:iam::123456789012:root`, replacing `123456789012` with your AWS account ID.
- To invite a **user with an assumed role** or a **federated user with an assumed role**, type `arn:aws:sts::123456789012:assumed-role/MyAssumedRole/MyAssumedRoleSession`, replacing `123456789012` with your AWS account ID, `MyAssumedRole` with the name of the assumed role, and `MyAssumedRoleSession` with the session name for the assumed role.

For example, to invite the AWS account root user for account ID `123456789012` to an environment with ID `12a34567b8cd9012345ef67abcd890e1` as a read/write member, run the following command.
Open a Shared Environment

To open a shared environment, you use your AWS Cloud9 dashboard. You then use the AWS Cloud9 IDE to do things in a shared environment such as work with files and chat with other members.

1. Be sure the corresponding access policy is attached to the group or role for your user. For more information, see About Environment Member Access Roles (p. 82).

2. Sign in to the AWS Cloud9 console as follows:
   - If you’re the only individual using your AWS account or you are an IAM user in a single AWS account, go to https://console.aws.amazon.com/cloud9/.
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
   - If you’re a student in a classroom, see your instructor for sign-in instructions.

3. Open the shared environment from your AWS Cloud9 dashboard. For more information, see Opening an Environment in AWS Cloud9 (p. 67).

You use the Collaborate window to interact with other members, as described in the rest of this topic.

See a List of Environment Members

With the shared environment open, in the Collaborate window, expand Environment Members, if the list of members is not visible.

A circle next to each member indicates their online status, as follows.

- Active members have a green circle.
- Offline members have a gray circle.
- Idle members have an orange circle.
To use code to get a list of environment members, call the AWS Cloud9 describe environment memberships operation, as follows.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>describe-environment-memberships</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResult</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DescribeEnvironmentMemberships, DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>describeEnvironmentMemberships</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DescribeEnvironmentMembershipsRequest, DescribeEnvironmentMembershipsResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>describeEnvironmentMemberships</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>describe_environment_memberships</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>describe_environment_memberships</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Get-C9EnvironmentMembershipList</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>DescribeEnvironmentMemberships</td>
</tr>
</tbody>
</table>

Open the Active File of an Environment Member

With the shared environment open, in the menu bar, choose the member name. Then choose **Open Active File**.
Open the Open File of an Environment Member

1. With the shared environment open, in the **Collaborate** window, expand **Environment Members**, if the list of members is not visible.
2. Expand the name of the user whose open file you want to open in your environment.
3. Double-click the name of the file you want to open.

Go to the Active Cursor of an Environment Member

1. With the shared environment open, in the **Collaborate** window, expand **Environment Members**, if the list of members is not visible.
2. Right-click the member name, and then choose **Show Location**.

Chat with Other Environment Members

With the shared environment open, at the bottom of the **Collaborate** window, for **Enter your message here**, type your chat message, and then press **Enter**.

View Chat Messages in a Shared Environment

With the shared environment open, in the **Collaborate** window, expand **Group Chat**, if the list of chat messages is not visible.
Delete a Chat Message from a Shared Environment

With the shared environment open, in the **Collaborate** window, right-click the chat message in **Group Chat**, and then choose **Delete Message**.

**Note**
When you delete a chat message, it is deleted from the environment for all members.

Delete All Chat Messages from a Shared Environment

With the shared environment open, in the **Collaborate** window, right-click anywhere in **Group Chat**, and then choose **Clear history**.

**Note**
When you delete all chat messages, they are deleted from the environment for all members.

Change the Access Role of an Environment Member

1. Open the environment that you own and that contains the member whose access role you want to change, if the environment is not already open. For more information, see Opening an Environment in AWS Cloud9 (p. 67).
2. In the **Collaborate** window, expand **Environment Members**, if the list of members is not visible.
3. Do one of the following:
   - Next to the member name whose access role you want to change, choose **R** or **RW** to make this member owner or read/write, respectively.
   - To change a read/write member to read-only, right-click the member name, and then choose **Revoke Write Access**.
   - To change a read-only member to read/write, right-click the member name, and then choose **Grant Read+Write Access**.

**Note**
If you make this user a read/write member, a dialog box is displayed, containing information about possibly putting your AWS security credentials at risk. Do not make a user a read/write member unless you trust that user to take actions in AWS on your behalf. For more information, see the related note in Invite a User in the Same Account as the Environment (p. 84).

To use code to change the access role of a environment member, call the AWS Cloud9 update environment membership operation, as follows.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>update-environment-membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>UpdateEnvironmentMembership, UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>updateEnvironmentMembership</td>
</tr>
</tbody>
</table>

89
### Remove Your User From a Shared Environment

**Note**
You cannot remove your user from an environment if you are the environment owner. Removing your user from an environment does not remove your user from IAM.

1. With the shared environment open, in the **Collaborate** window, expand **Environment Members**, if the list of members is not visible.
2. Do one of the following.
   - Next to **You**, choose the trash can icon.
   - Right-click **You**, and then choose **Leave environment**.
3. When prompted, choose **Leave**.

To use code to remove your user from a shared environment, call the AWS Cloud9 delete environment membership operation, as follows.

<table>
<thead>
<tr>
<th>AWS SDK for .NET</th>
<th>UpdateEnvironmentMembershipRequest, UpdateEnvironmentMembershipResponse</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for PHP</td>
<td>updateEnvironmentMembership</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>update_environment_membership</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>update_environment_membership</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Update-C9EnvironmentMembership</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>UpdateEnvironmentMembership</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>delete-environment-membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DeleteEnvironmentMembership, DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>deleteEnvironmentMembership</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>deleteEnvironmentMembership</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>delete_environment_membership</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>delete_environment_membership</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Remove-C9EnvironmentMembership</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>DeleteEnvironmentMembership</td>
</tr>
</tbody>
</table>
Remove Another Environment Member

**Note**
To remove any member other than your user from an environment, you must be signed in to AWS Cloud9 using the credentials of the environment owner. Removing a member does not remove the user from IAM.

1. Open the environment that contains the member you want to remove, if the environment is not already open. For more information, see Opening an Environment in AWS Cloud9 (p. 67).
2. In the **Collaborate** window, expand **Environment Members**, if the list of members is not visible.
3. Do one of the following.
   - Next to the name of the member you want to delete, choose the trash can icon.
   - Right-click the name of the member you want to delete, and then choose **Revoke Access**.
4. When prompted, choose **Remove Member**.

To use code to remove a member from an environment, call the AWS Cloud9 delete environment membership operation, as follows.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>delete-environment-membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DeleteEnvironmentMembership, DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>deleteEnvironmentMembership</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DeleteEnvironmentMembershipRequest, DeleteEnvironmentMembershipResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>deleteEnvironmentMembership</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>delete_environment_membership</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>delete_environment_membership</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Remove-C9EnvironmentMembership</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>DeleteEnvironmentMembership</td>
</tr>
</tbody>
</table>

**Environment Sharing Best Practices**

We recommend the following practices when sharing environments.

- Only invite read/write members you trust to your environments.
- For EC2 environments, read/write members can use the environment owner’s AWS access credentials, instead of their own credentials, to make calls from the environment to AWS services. To prevent this, the environment owner can disable AWS managed temporary credentials for the environment. However, this also prevents the environment owner from making calls. For more information, see **AWS Managed Temporary Credentials (p. 536)**.
Moving an environment or resizing an Amazon EBS volume

You can move an AWS Cloud9 development environment from one Amazon EC2 instance to another. For example, you might want to do one of the following.

• Transfer an environment from an Amazon EC2 instance that is broken, or behaving in unexpected ways, to a healthy instance.
• Transfer an environment from an older instance to an instance that has the latest system updates.
• Increase an instance's compute resources, because the environment is over-utilized on the current instance.
• Decrease an instance's compute resources, because the environment is under-utilized on the current instance.

You can also resize the Amazon Elastic Block Store (Amazon EBS) volume that is associated with an Amazon EC2 instance for an environment. For example, you might want to do one of the following.

• Increase a volume's size, because you are running out of storage space on the instance.
• Decrease a volume's size, because you don't want to pay for extra storage space that you aren't using.

Before you move or resize an environment, you might want to try stopping some running processes in the environment or adding a swap file to the environment. For more information, see IDE Warning: “This Environment is Running Low on Memory” or “This Environment Has High CPU Load” (p. 553) in Troubleshooting.

This topic only covers moving an environment from one Amazon EC2 instance to another or resizing an Amazon EBS volume. To resize an environment from one of your own servers to another or to change the storage space for one of your own servers, refer to your server’s documentation.

Topics

- Moving an Environment (p. 92)
- Resizing an Amazon EBS volume used by an Environment (p. 94)

Moving an Environment

Before you start the move process, note the following.

• You cannot move an environment to an Amazon EC2 instance of the same type. When you move, you must choose a different Amazon EC2 instance type for the new instance.
• You must stop the Amazon EC2 instance that is associated with an environment before you can change the instance type. While the instance is stopped, neither you nor any members will be able to use the environment that are associated with the stopped instance.
Moving an Environment

- We move the instance to new hardware; however, the instance's ID does not change.
- If the instance is running in an Amazon VPC and has a public IPv4 address, we release the address and give it a new public IPv4 address. The instance retains its private IPv4 addresses, any Elastic IP addresses, and any IPv6 addresses.
- Ensure that you plan for downtime while your instance is stopped. The process might take several minutes.

To move an environment, do the following.

1. (Optional) If the new instance type requires drivers that are not installed on the existing instance, you must connect to your instance and install the drivers first. For more information, see Compatibility for Resizing Instances in the Amazon EC2 User Guide for Linux Instances.
2. Close all web browser tabs that are currently displaying the environment.
   **Important**
   If you do not close all of the web browser tabs that are currently displaying the environment, AWS Cloud9 might interfere with allowing you to fully complete the procedure. Specifically, AWS Cloud9 might try at the wrong time during this procedure to restart the Amazon EC2 instance that is associated with the environment. The instance must stay stopped until the very last step in this procedure.
   We recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.
4. Open the Amazon EC2 console. To do this, in the Services list, choose EC2.
5. In the AWS navigation bar, choose the AWS Region that contains the environment that you want to move (for example, US East (Ohio)).
6. In the service navigation pane, expand Instances if it is not already expanded, and then choose Instances.
7. In the list of instances, choose the instance that is associated with the environment that you want to move. For an EC2 environment, the instance name starts with aws-cloud9- followed by the environment name. For example, if the environment is named my-demo-environment, the instance name will start with aws-cloud9-my-demo-environment.
8. If the Instance State is not stopped, choose Actions, Instance State, Stop. When prompted, choose Yes, Stop. It can take a few minutes for the instance to stop.
9. After the Instance State is stopped, with the instance still selected, choose Actions, Instance Settings, Change Instance Type.
10. In the Change Instance Type dialog box, for Instance Type, choose the new instance type that you want the environment to use.
    **Note**
    If the instance type that you want does not appear in the list, then it is not compatible with the instance's configuration (for example, because of its virtualization type).
11. (Optional) If the instance type that you chose supports EBS-optimization, select EBS-optimized to enable EBS-optimization, or clear EBS-optimized to disable EBS-optimization.
    **Note**
    If the instance type that you chose is EBS-optimized by default, EBS-optimized is selected and you can't clear it.
12. Choose Apply to accept the new settings.
    **Note**
    If you did not choose a different instance type for Instance Type earlier in this procedure, nothing happens after you choose Apply.
13. Reopen the environment. For more information, see Opening an Environment in AWS Cloud9 (p. 67).

For more information about the preceding procedure, see Changing the Instance Type in the Amazon EC2 User Guide for Linux Instances.

Resizing an Amazon EBS volume used by an Environment

1. Open the environment that's associated with the Amazon EC2 instance for the Amazon EBS volume that you want to resize.
2. In the AWS Cloud9 IDE for the environment, create a file with the following contents, and then save the file with the extension .sh (resize.sh, for example).

   ```bash
   #!/bin/bash
   # Specify the desired volume size in GiB as a command-line argument. If not specified, default to 20 GiB.
   SIZE=${1:-20}
   # Get the ID of the environment host Amazon EC2 instance.
   # Get the ID of the Amazon EBS volume associated with the instance.
   VOLUMEID=$(aws ec2 describe-instances --instance-id $INSTANCEID --query "Reservations[0].Instances[0].BlockDeviceMappings[0].Ebs.VolumeId" --output text)
   # Resize the EBS volume.
   aws ec2 modify-volume --volume-id $VOLUMEID --size $SIZE
   # Wait for the resize to finish.
   while [ $(aws ec2 describe-volumes-modifications --volume-id $VOLUMEID --filters Name=modification-state,Values="optimizing","completed" --query "length(VolumesModifications)" --output text) != "1" ]; do
     sleep 1
   done
   if [ $(readlink -f /dev/xvda) = "/dev/xvda" ]; then
     # Rewrite the partition table so that the partition takes up all the space that it can.
     sudo growpart /dev/xvda 1
     # Expand the size of the file system.
     sudo resize2fs /dev/xvda1
   else
   ```
# Rewrite the partition table so that the partition takes up all the space that it can.
```
sudo growpart /dev/nvme0n1 1
```

# Expand the size of the file system.
```
sudo resize2fs /dev/nvme0n1p1
fi
```

3. From a terminal session in the IDE, switch to the directory that contains the `resize.sh` file. Then run the following command, replacing 20 with the desired size in GiB to resize the Amazon EBS volume to.
```
sh resize.sh 20
```

Deleting an Environment in AWS Cloud9

To prevent any ongoing charges to your AWS account related to an AWS Cloud9 development environment that you’re no longer using, you should delete the environment.

- Deleting an Environment with the Console (p. 95)
- Deleting an Environment with Code (p. 98)

Deleting an Environment with the Console

**Warning**
When you delete an environment, AWS Cloud9 deletes the environment permanently. This includes permanently deleting all related settings, user data, and uncommitted code. Deleted environments cannot be recovered.

1. Sign in to the AWS Cloud9 console as follows:
   - If you’re the only individual using your AWS account or you are an IAM user in a single AWS account, go to [https://console.aws.amazon.com/cloud9/](https://console.aws.amazon.com/cloud9/).
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
   - If you’re using an AWS Educate Starter Account, see Step 2: Sign in to the AWS Cloud9 Console (p. 29) in *Individual Student Signup*.
   - If you’re a student in a classroom, see your instructor for sign-in instructions.

2. In the top navigation bar, choose the AWS Region where the environment is located.
3. In the list of environments, for the environment you want to delete, do one of the following.
   - Choose the title of the card for the environment. Then choose **Delete** on the next page.
   - Select the card for the environment, and then choose the **Delete** button.

4. In the **Delete** dialog box, type **Delete**, and then choose **Delete**.
   - **EC2 environment**
AWS Cloud9 also terminates the Amazon EC2 instance that was connected to that environment.

**Note**
If account deletion fails, a banner is displayed at the top of the console webpage. Additionally, the card for the environment, if it exists, indicates that environment deletion failed.

- **SSH environment**

  If the environment was connected to an Amazon EC2 instance, AWS Cloud9 doesn't terminate that instance. If you don't terminate that instance later, your AWS account might continue to have ongoing charges for Amazon EC2 related to that instance.

5. If the environment was an SSH environment, AWS Cloud9 leaves behind a hidden subdirectory on the cloud compute instance or your own server that was connected to that environment. You can now safely delete that subdirectory if you want to. The subdirectory is named `.c9`. It is located in the **Environment path** directory that you specified when you created the environment.

If your environment is not displayed in the console, try doing one or more of the following actions to try to display it.

- In the side navigation bar, choose one or more of the following.
  - Choose **Your environments** to display all environments that your AWS entity owns within the selected AWS Region and AWS account.
  - Choose **Shared with you** to display all environments your AWS entity has been invited to within the selected AWS Region and AWS account.
  - Choose **Account environments** to display all environments within the selected AWS Region and AWS account that your AWS entity has permissions to display.

- Choose the previous arrow, next arrow, or page number button to display more environments in the current scope.

- If you think you should be a member of an environment, but the environment is not displayed in the **Shared with you** list, check with the environment owner.

- In the top navigation bar, choose a different AWS Region.
Deleting an Environment with Code

Warning
When you delete an environment, AWS Cloud9 deletes the environment permanently. This includes permanently deleting all related settings, user data, and uncommitted code. Deleted environments cannot be recovered.

To use code to delete an environment in AWS Cloud9, call the AWS Cloud9 delete environment operation, as follows.

<table>
<thead>
<tr>
<th>SDK Type</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS CLI</td>
<td>delete-environment</td>
</tr>
<tr>
<td>AWS SDK for C++</td>
<td>DeleteEnvironmentRequest, DeleteEnvironmentResult</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DeleteEnvironment, DeleteEnvironmentRequest,</td>
</tr>
<tr>
<td></td>
<td>DeleteEnvironmentWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DeleteEnvironmentRequest,DeleteEnvironmentResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>deleteEnvironment</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DeleteEnvironmentRequest, DeleteEnvironmentResult</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>deleteEnvironment</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>delete_environment</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>delete_environment</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Remove-C9Environment</td>
</tr>
<tr>
<td>AWS Cloud9 API</td>
<td>DeleteEnvironment</td>
</tr>
</tbody>
</table>

AWS Cloud9 User Guide
Deleting an Environment with Code
Working with the AWS Cloud9 Integrated Development Environment (IDE)

An integrated development environment (IDE) provides a set of coding productivity tools such as a source code editor, a debugger, and build tools.

Learn how to work with the AWS Cloud9 IDE by reading one or more of these topics.

Topics

- Tour the AWS Cloud9 IDE (p. 100)
- Language Support in the AWS Cloud9 Integrated Development Environment (IDE) (p. 120)
- Menu Bar Commands Reference for the AWS Cloud9 Integrated Development Environment (IDE) (p. 122)
- Finding and Replacing Text in the AWS Cloud9 Integrated Development Environment (IDE) (p. 131)
- Previewing Files in the AWS Cloud9 Integrated Development Environment (IDE) (p. 134)
- Previewing Running Applications in the AWS Cloud9 Integrated Development Environment (IDE) (p. 136)
- Working with File Revisions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 143)
- Working with Images Files in the AWS Cloud9 Integrated Development Environment (IDE) (p. 144)
- Working with Builders, Runners, and Debuggers in the AWS Cloud9 Integrated Development Environment (IDE) (p. 146)
- Working with Language Projects in the AWS Cloud9 Integrated Development Environment (IDE) (p. 154)
- Working with Custom Environment Variables in the AWS Cloud9 Integrated Development Environment (IDE) (p. 157)
- Working with Project Settings in the AWS Cloud9 Integrated Development Environment (IDE) (p. 159)
- Working with User Settings in the AWS Cloud9 IDE (p. 165)
- Working with AWS Project and User Settings in the AWS Cloud9 Integrated Development Environment (IDE) (p. 172)
- Working with Keybindings in the AWS Cloud9 Integrated Development Environment (IDE) (p. 173)
- Working with Themes in the AWS Cloud9 Integrated Development Environment (IDE) (p. 176)
- Working with Initialization Scripts in the AWS Cloud9 Integrated Development Environment (IDE) (p. 178)
- MacOS Default Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE) (p. 178)
- MacOS Vim Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE) (p. 189)
Tour the AWS Cloud9 IDE

This topic provides a basic tour of the AWS Cloud9 integrated development environment (IDE). To take full advantage of this tour, follow the steps shown below in sequence.

**Topics**

- Prerequisites (p. 100)
- Step 1: Menu Bar (p. 101)
- Step 2: Dashboard (p. 102)
- Step 3: Environment Window (p. 103)
- Step 4: Editor, Tabs, and Panes (p. 103)
- Step 5: Console (p. 105)
- Step 6: Open Files Section (p. 106)
- Step 7: Gutter (p. 106)
- Step 8: Status Bar (p. 107)
- Step 9: Outline Window (p. 108)
- Step 10: Go Window (p. 109)
- Step 11: Immediate Tab (p. 111)
- Step 12: Process List (p. 112)
- Step 13: Preferences (p. 113)
- Step 14: Terminal (p. 114)
- Step 15: Debugger Window (p. 115)
- Final Thoughts (p. 120)

**Prerequisites**

To go on this tour, you must have an AWS account and an open AWS Cloud9 development environment. To learn how to do these things, you can follow the steps in Getting Started: Basic Tutorials for AWS Cloud9 (p. 35). Alternatively, you can explore separate related topics such as Setting Up AWS Cloud9 (p. 6) and Working with Environments in AWS Cloud9 (p. 55).
Warning

Having an AWS Cloud9 development environment might result in charges to your AWS account. These include possible charges for Amazon EC2 if you are using an EC2 environment. For more information, see Amazon EC2 Pricing.

Step 1: Menu Bar

The menu bar, at the top edge of the IDE, contains common commands for working with files and code and changing IDE settings. You can also preview and run code from the menu bar.

You can hide the menu bar by choosing the arrow at its edge, as follows.

You can show the menu bar again by choosing the arrow in the middle of where the menu bar was earlier, as follows.

Compare your results to the following.

You can use the IDE to work with a set of files in the next several sections in this tutorial. To set up these files, choose File, New File.

Next, copy the following text into the Untitled1 editor tab.

```
fish.txt
```
Step 2: Dashboard

The dashboard gives you quick access to each of your environments. From the dashboard, you can create, open, and change the setting for an environment.

To open the dashboard, on the menu bar, choose AWS Cloud9, Go To Your Dashboard.

To view the settings for your environment, choose the title inside of the my-demo-environment card.

To go back to the dashboard, use your web browser's back button or the navigation breadcrumb called Environments.

To open to the IDE for your environment, choose Open IDE inside of the my-demo-environment card.

Note
It can take a few moments for the IDE to display again.
Step 3: Environment Window

The Environment window shows a list of your folders and files in the environment. You can also show different types of files, such as hidden files.

To show or hide the contents of the Environment window, choose the Environment button.

To show or hide the Environment window and the Environment button, choose Window, Environment on the menu bar.

![Environment Window](image)

To show or hide hidden files, in the Environment window, choose the gear icon, and then choose Show Hidden Files.

![Environment Window with gear icon](image)

Step 4: Editor, Tabs, and Panes

The editor is where you can do things such as write code, run a terminal session, and change IDE settings. Each instance of an open file, terminal session, and so on is represented by a tab. Tabs can be grouped into panes. Tabs are shown at the edge of their pane.
To show or hide tabs, choose View, Tab Buttons on the menu bar.

To open a new tab, choose the + icon at the edge of the row of tabs. Then choose one of the available commands, for example, New File, as follows.

To display two panes, choose the icon that looks like a drop-down menu, which is at the edge of the row of tabs. Then choose Split Pane in Two Rows, as follows.

To return to a single pane, choose the drop-down menu icon again, and then choose the single square icon, as follows.
Step 5: Console

The console is an alternate place for creating and managing tabs. By default, it contains a Terminal tab, but can also contain other types of tabs.

To show or hide the console, choose View, Console on the menu bar.

To expand or shrink the console, choose the resize icon, which is at the edge of the console, as follows.
Step 6: Open Files Section

The **Open Files** section shows a list of all files that are currently open in the editor. **Open Files** is part of the **Environment** window.

To show or hide the **Open Files** section, choose **View, Open Files** on the menu bar.

To switch between open files, choose the file of interest from the list.

**Step 7: Gutter**

The **gutter**, at the edge of each file in the editor, shows things like line numbers and contextual symbols as you work with files.

To show or hide the gutter, choose **View, Gutter** on the menu bar.
Step 8: Status Bar

The status bar, at the edge of each file in the editor, shows things like line and character numbers, file type preference, space and tab settings, and related editor settings.

To show or hide the status bar, choose View, Status Bar on the menu bar.

To go to a specific line number, choose a tab with the file of interest. Then in the status bar, choose the line and character number (it should be something like 7:45). Type a line number (like 4), and then press Enter, as follows.

To change the file type preference, in the status bar, choose a different file type. For example, for cat.txt, choose Ruby to see the syntax colors change. To go back to plain text colors, choose Plain Text, as follows.
Step 9: Outline Window

You can use the **Outline** window to quickly go to a specific file location.

To show or hide the **Outline** window and the **Outline** button, choose **Window, Outline** on the menu bar.

To see how the **Outline** window works, create a file named `hello.rb`. Copy the following code into the file and save it.

```ruby
def say_hello(i)
  puts "Hello!"
  puts "i is #{i}"
end

def say_goodbye(i)
  puts "i is now #{i}".
  puts "Goodbye!"
end

i = 1
say_hello(i)
i += 1
say_goodbye(i)
```

To show or hide the contents of the **Outline** window, choose the **Outline** button.

In the **Outline** window, choose `say_hello(i)`, and then choose `say_goodbye(i)`, as follows.
Step 10: Go Window

You can use the Go window to open a file in the editor, go to a symbol's definition, run a command, or go to a line in the active file in the editor.
To show the contents of the Go window, choose the Go button (the magnifying glass icon).

To show or hide the Go window and the Go button, choose Window, Go on the menu bar.

With the Go window open, you can:

- Type a forward slash (/) followed by part or all of a file name. In the list of matching files that displays, choose a file to open it in the editor. For example, typing /fish lists fish.txt, while typing /.txt lists both fish.txt and cat.txt.

  **Note**
  File search is scoped only to non-hidden files and non-hidden folders in the Environment window.

- Type an at symbol (@) followed by the name of a symbol. In the list of matching symbols that displays, choose a symbol to go to it in the editor. For example, with the hello.rb file open and active in the editor, type @hello to list say_hello(i), or type @say to list both say_hello(i) and say_goodbye(i).

  **Note**
  If the active file in the editor is part of a supported language project, symbol search is scoped to the current project. Otherwise, symbol search is scoped only to the active file in the editor. For more information, see Working with Language Projects in the AWS Cloud9 Integrated Development Environment (IDE) (p. 154).

- Type a dot (.) followed by the name of a command. In the list of commands that displays, choose a command to run it. For example, typing .closetab and then pressing Enter closes the current tab in the editor. For a list of available commands, see the Commands Reference for the AWS Cloud9 Integrated Development Environment (IDE) (p. 264).

- Type a colon (:) followed by a number to go to that line number in the active file in the editor. For example, with the hello.rb file open and active in the editor, type :11 to go to line 11 in that file.
To see the keybindings for each of these actions based on the current keyboard mode and operating system, see each of the available Go To commands on the Go menu in the menu bar.

**Step 11: Immediate Tab**

The **Immediate** tab enables you to test small snippets of JavaScript code. To see how the **Immediate** tab works, do the following.

1. Open an **Immediate** tab by choosing **Window, New Immediate Window** on the menu bar.
2. Run some code in the **Immediate** tab. To try this, type the following code into the window, pressing Shift-Enter after typing line 1 and again after line 2. Press Enter after line 3. (If you press Enter instead of Shift-Enter after you type line 1 or line 2, the code will run earlier than you want it to.)

```javascript
for (i = 0; i <= 10; i++) { // Press Shift-Enter after typing this line.
    console.log(i)        // Press Shift-Enter after typing this line.
}                           // Press Enter after typing this line. The numbers 0 to 10 will be printed.
```
Step 12: Process List

The Process List shows all of the running processes. You can stop or even forcibly stop processes that you don't want to run anymore. To see how the Process List window works, do the following.

2. Find a process. In the Process List, type the name of the process.
3. Stop or forcibly stop a process. In the list of processes, choose the process, and then choose Kill or Force Kill.
Step 13: Preferences

Preferences include the following settings.

- Settings for the current environment only, such as whether to use soft tabs in the editor, the file types to ignore, and code completion behaviors for languages such as PHP and Python.
- Your user settings across each of your environments, such as colors, fonts, and editor behaviors.
- Your keybindings, such as which shortcut key combinations you prefer to use to work with files and the editor.
- The IDE's overall theme.

To show preferences, choose AWS Cloud9, Preferences on the menu bar. Something like the following is displayed.
Step 14: Terminal

You can run one or more terminal sessions in the IDE. To start a terminal session, choose Window, New Terminal on the menu bar. Or, choose the “plus” icon next to the Console tabs and choose New Terminal.

You can try running a command in the terminal. For example, in the terminal, type `echo $PATH` and then press Enter to print the value of the PATH environment variable.

You can also try running additional commands. For example, try commands such as the following.

- `pwd` to print the path to the current directory.
- `aws --version` to print version information about the AWS CLI.
- `ls -l` to print information about the current directory.
Step 15: Debugger Window

You can use the Debugger window to debug your code. For example, you can step through running code a portion at a time, watch the values of variables over time, and explore the call stack.

**Note**

This procedure is similar to Step 2: Basic Tour of the IDE (p. 38) from either of the basic IDE tutorials (p. 35).

To show or hide the Debugger window and the Debugger button, choose Window, Debugger on the menu bar.

For this tutorial, you can experiment with the Debugger window and some JavaScript code by doing the following.

1. Check the Node.js installation in your environment by running the following command in a terminal session: `node --version`. If Node.js is installed, the Node.js version number is shown in the output, and you can skip ahead to step 3 in this procedure ("Write some JavaScript code...").
2. If you need to install Node.js, do the following.
   a. Run the following two commands, one at a time, to be sure your environment has the latest updates and then download Node Version Manager (nvm). (nvm is a simple Bash shell script that
is useful for installing and managing Node.js versions. For more information, see Node Version Manager on GitHub.)

For Amazon Linux:

```bash
sudo yum -y update
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
```

For Ubuntu Server:

```bash
sudo apt update
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
```

b. Use a text editor to update your shell profile file (for example, ~/.bashrc) to enable nvm to load. For example, in the Environment window of the IDE, choose the gear icon, and then choose Show Home in Favorites. Repeat this step and choose Show Hidden Files as well.

c. Open the ~/.bashrc file.

d. Type or paste the following code at the end of the file to enable nvm to load.

For Amazon Linux:

```bash
export NVM_DIR="/home/ec2-user/.nvm" [ -s "$NVM_DIR/nvm.sh" ] && ". "$NVM_DIR/nvm.sh" # This loads nvm.
```

For Ubuntu Server:

```bash
export NVM_DIR="/home/ubuntu/.nvm" [ -s "$NVM_DIR/nvm.sh" ] && ". "$NVM_DIR/nvm.sh" # This loads nvm.
```

e. Save the file.

f. Close that terminal session and start a new one. Then run the following command to install the latest version of Node.js.

```bash
nvm install node
```

3. Write some JavaScript code to debug. For example, create a file, add the following code to the file, and save it as hello.js.

```javascript
var i;
i = 10;

console.log("Hello!");
console.log("i is " + i);
i += 1;

console.log("i is now " + i);
console.log("Goodbye!");
```

4. Add some breakpoints to the code. For example, in the gutter, choose the margin next to lines 6 and 10. A red circle is displayed next to each of these line numbers, as follows.
5. Now you're ready to debug the JavaScript code. To try this, do the following.

   a. To show or hide the contents of the Debugger window, choose the Debugger button, as shown in the next step.

   b. Watch the value of the variable named `i` while the code is running. In the Debugger window, for Watch Expressions, choose Type an expression here. Type the letter `i`, and then press Enter, as follows.

   c. Begin running the code. Choose Run, Run With, Node.js, as follows.
d. The code pauses running on line 6. The **Debugger** window shows the value of `i` in **Watch Expressions**, which is currently 10.

![Debugger window showing the value of i](image)

e. In the **Debugger** window, choose **Resume**, which is the blue arrow icon, as follows.
f. The code pauses running on line 10. The Debugger window now shows the new value of \( i \), which is currently 11.

g. Choose Resume again. The code runs to the end. The output is printed to the console's hello.js tab, as follows.

Compare your results to the following.
Final Thoughts

Warning

Remember that having an AWS Cloud9 development environment might result in charges to your AWS account. These include possible charges for Amazon EC2 if you are using an EC2 environment. For more information, see Amazon EC2 Pricing.

There are additional topics in the parent section (Working with the IDE (p. 99)) that you might want to explore. However, when you are finished touring the AWS Cloud9 IDE and no longer need the environment, be sure to delete it and its associated resources, as described in Deleting an Environment (p. 95).

Language Support in the AWS Cloud9 Integrated Development Environment (IDE)

The AWS Cloud9 IDE supports many programming languages. The following table lists the languages that are supported and to what level.

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax highlighting</th>
<th>Run UI</th>
<th>Outline view</th>
<th>Code hints and linting</th>
<th>Code completion</th>
<th>Debugging</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X^5</td>
<td>X^4</td>
</tr>
<tr>
<td>C#</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
<td>X^5</td>
<td></td>
</tr>
</tbody>
</table>
### Language Support

<table>
<thead>
<tr>
<th>Language</th>
<th>Syntax highlighting</th>
<th>Run UI</th>
<th>Outline view</th>
<th>Code hints and linting</th>
<th>Code completion</th>
<th>Debugging</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoffeeScript</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSS</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dart</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Haskell</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HTML</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Java</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>JavaScript</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Node.js</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>PHP</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Python</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Ruby</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Shell script</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>TypeScript</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

### Notes

1. The AWS Cloud9 IDE provides syntax highlighting for many more languages. For a complete list, in the menu bar of the IDE, choose **View, Syntax**.

2. You can run programs or scripts at the click of a button for languages marked with an X, without using the command line. For languages not marked with an X or not displayed on the **Run, Run With** menu bar in the IDE, you can create a runner for that language. For instructions, see **Create a Builder or Runner** (p. 151).

3. You can use the IDE's built-in tools to debug programs or scripts for languages marked with an X. For instructions, see **Debug Your Code** (p. 147).

4. This feature is in an experimental state for this language. It is not fully implemented and is not documented or supported.

5. This feature supports only local functions for this language.

6. To specify paths for AWS Cloud9 to use for completion of custom PHP code, in the AWS Cloud9 IDE turn on the **Project, PHP Support, Enable PHP code completion** setting in **Preferences**, and then add the paths to the custom code to the **Project, PHP Support, PHP Completion Include Paths** setting.

7. To specify paths for AWS Cloud9 to use for completion of custom Python code, in the AWS Cloud9 IDE turn on the **Project, Python Support, Enable Python code completion** setting in **Preferences**, and then add the paths to the custom code to the **Project, Python Support, PYTHONPATH** setting.

8. The AWS Cloud9 IDE provides additional support for some programming languages, such as TypeScript (version 3.7.5 supported in the AWS Cloud9 IDE), within the context of a language project. For more information, see **Working with Language Projects** (p. 154).
Menu Bar Commands Reference for the AWS Cloud9 Integrated Development Environment (IDE)

The following lists describe the default menu bar commands in the AWS Cloud9 IDE. If the menu bar isn't visible, choose the thin bar along the top edge of the IDE to show it.

- AWS Cloud9 Menu (p. 122)
- File Menu (p. 123)
- Edit Menu (p. 124)
- Find Menu (p. 126)
- View Menu (p. 126)
- Go Menu (p. 127)
- Run Menu (p. 128)
- Tools Menu (p. 128)
- Window Menu (p. 129)
- Support Menu (p. 130)
- Preview Menu (p. 131)
- Other Menu Bar Commands (p. 131)

AWS Cloud9 Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preferences</td>
<td>Do one of the following:</td>
</tr>
<tr>
<td></td>
<td>• Open the Preferences tab if it isn't open.</td>
</tr>
<tr>
<td></td>
<td>• Make the Preferences tab active if it is open but not active.</td>
</tr>
<tr>
<td></td>
<td>• Hide the Preferences tab if it is active.</td>
</tr>
<tr>
<td></td>
<td>See Working with Project Settings (p. 159), Working with User Settings (p. 165), Working with Keybindings (p. 173), Working with Themes (p. 176), and Working with Initialization Scripts (p. 178).</td>
</tr>
<tr>
<td>Go To Your Dashboard</td>
<td>Open the AWS Cloud9 console in a separate web browser tab. See Creating an Environment (p. 55), Opening an Environment (p. 67), Changing Environment Settings (p. 77), and Deleting an Environment (p. 95).</td>
</tr>
<tr>
<td>Welcome Page</td>
<td>Open the Welcome tab.</td>
</tr>
<tr>
<td>Open Your Project Settings</td>
<td>Open the project.settings file for the current environment. See Working with Project Settings (p. 159).</td>
</tr>
<tr>
<td>Open Your User Settings</td>
<td>Open the user.settings file for the current user. See Working with User Settings (p. 165).</td>
</tr>
</tbody>
</table>
## File Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Your Keymap</td>
<td>Open the <code>keybindings.settings</code> file for the current user. See <em>Working with Keybindings</em> (p. 173).</td>
</tr>
<tr>
<td>Open Your Init Script</td>
<td>Open the <code>init.js</code> file for the current user. See <em>Working with Initialization Scripts</em> (p. 178).</td>
</tr>
<tr>
<td>Open Your Stylesheet</td>
<td>Open the <code>styles.css</code> file for the current user. See <em>Working with Themes</em> (p. 176).</td>
</tr>
</tbody>
</table>

### File Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>New File</td>
<td>Create a new file.</td>
</tr>
<tr>
<td>New From Template</td>
<td>Create a new file, based on the chosen file template.</td>
</tr>
<tr>
<td>Open</td>
<td>Show and go to the <strong>Navigate</strong> window.</td>
</tr>
<tr>
<td>Open Recent</td>
<td>Open the chosen file.</td>
</tr>
<tr>
<td>Save</td>
<td>Save the current file.</td>
</tr>
<tr>
<td>Save As</td>
<td>Save the current file with a different file name, location, or both.</td>
</tr>
<tr>
<td>Save All</td>
<td>Save all unsaved files.</td>
</tr>
<tr>
<td>Revert to Saved</td>
<td>Discard changes for current file since it was last saved.</td>
</tr>
<tr>
<td>Revert All to Saved</td>
<td>Discard changes for all unsaved files since they were last saved.</td>
</tr>
<tr>
<td>Show File Revision History</td>
<td>View and manage changes to the current file in the editor. See <em>Working with File Revisions</em> (p. 143).</td>
</tr>
<tr>
<td>Upload Local Files</td>
<td>Show the <strong>Upload Files</strong> dialog box, which enables you to drag files from your local computer into the environment.</td>
</tr>
<tr>
<td>Download Project</td>
<td>Combine the files in the environment into a .zip file, which you can download to your local computer.</td>
</tr>
<tr>
<td>Line Endings</td>
<td>Use <strong>Windows</strong> (carriage return plus line feed) or <strong>Unix</strong> (line feed only) line endings.</td>
</tr>
<tr>
<td>Close File</td>
<td>Close the current file.</td>
</tr>
<tr>
<td>Close All Files</td>
<td>Close all open files.</td>
</tr>
</tbody>
</table>
## Edit Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>Undo the last action.</td>
</tr>
<tr>
<td>Redo</td>
<td>Redo the last undone action.</td>
</tr>
<tr>
<td>Cut</td>
<td>Move the selection to the clipboard.</td>
</tr>
<tr>
<td>Copy</td>
<td>Copy the selection to the clipboard.</td>
</tr>
<tr>
<td>Paste</td>
<td>Copy the clipboard's contents to the selection point.</td>
</tr>
<tr>
<td>Keyboard Mode</td>
<td>The set of keybindings to use, such as Default, Vim, Emacs, or Sublime. See Working with Keybindings (p. 173).</td>
</tr>
<tr>
<td>Selection, Select All</td>
<td>Select all selectable content.</td>
</tr>
<tr>
<td>Selection, Split Into Lines</td>
<td>Add a cursor at the end of the current line.</td>
</tr>
<tr>
<td>Selection, Single Selection</td>
<td>Clear all previous selections.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Add Cursor Up</td>
<td>Add a cursor one line above the active cursor. If a cursor is already added, add another cursor above that one.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Add Cursor Down</td>
<td>Add a cursor one line below the active cursor. If a cursor is already added, add another cursor below that one.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Move Active Cursor Up</td>
<td>Add a second cursor one line above the active cursor. If a second cursor is already added, move the second cursor up one line.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Move Active Cursor Down</td>
<td>Add a second cursor one line below the active cursor. If a second cursor is already added, move the second cursor down one line.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Add Next Selection Match</td>
<td>Include more matching selections that are after the selection.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Add Previous Selection Match</td>
<td>Include more matching selections that are before the selection.</td>
</tr>
<tr>
<td>Selection, Multiple Selections, Merge Selection Range</td>
<td>Add a cursor at the end of the current line.</td>
</tr>
<tr>
<td>Selection, Select Word Right</td>
<td>Include the next word to the right of the cursor in the selection.</td>
</tr>
<tr>
<td>Selection, Select Word Left</td>
<td>Include the next word to the left of the cursor in the selection.</td>
</tr>
<tr>
<td>Selection, Select to Line End</td>
<td>Include from the cursor to the end of the current line in the selection.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Selection, Select to Line Start</td>
<td>Include from the beginning of the current line to the cursor in the selection.</td>
</tr>
<tr>
<td>Selection, Select to Document End</td>
<td>Include from the cursor down to the end of the current file in the selection.</td>
</tr>
<tr>
<td>Selection, Select to Document Start</td>
<td>Include from the cursor up to the beginning of the current file in the selection.</td>
</tr>
<tr>
<td>Line, Indent</td>
<td>Indent the selection one tab.</td>
</tr>
<tr>
<td>Line, Outdent</td>
<td>Outdent the selection one tab.</td>
</tr>
<tr>
<td>Line, Move Line Up</td>
<td>Move the selection up one line.</td>
</tr>
<tr>
<td>Line, Move Line Down</td>
<td>Move the selection down one line.</td>
</tr>
<tr>
<td>Line, Copy Lines Up</td>
<td>Copy the contents of the line, and paste the copied contents one line up.</td>
</tr>
<tr>
<td>Line, Copy Lines Down</td>
<td>Copy the contents of the line, and paste the copied contents one line down.</td>
</tr>
<tr>
<td>Line, Remove Line</td>
<td>Delete the contents of the current line.</td>
</tr>
<tr>
<td>Line, Remove to Line End</td>
<td>Delete from the cursor to the end of the current line.</td>
</tr>
<tr>
<td>Line, Remove to Line Start</td>
<td>Delete from the beginning of the current line up to the cursor.</td>
</tr>
<tr>
<td>Line, Split Line</td>
<td>Move the contents of the cursor to the end of the line, to its own line.</td>
</tr>
<tr>
<td>Text, Remove Word Right</td>
<td>Delete the word to the right of the cursor.</td>
</tr>
<tr>
<td>Text, Remove Word Left</td>
<td>Delete the word to the left of the cursor.</td>
</tr>
<tr>
<td>Text, Align</td>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned.</td>
</tr>
<tr>
<td>Text, Transpose Letters</td>
<td>Transpose the selection.</td>
</tr>
<tr>
<td>Text, To Upper Case</td>
<td>Change the selection to all uppercase.</td>
</tr>
<tr>
<td>Text, To Lower Case</td>
<td>Change the selection to all lowercase.</td>
</tr>
<tr>
<td>Comment, Toggle Comment</td>
<td>Add line comment characters at the start of each selected line, or remove them if they are there.</td>
</tr>
<tr>
<td>Code Folding, Toggle Fold</td>
<td>Fold code, or remove code folding if it is there.</td>
</tr>
<tr>
<td>Code Folding, Unfold</td>
<td>Unfold the selected code.</td>
</tr>
<tr>
<td>Code Folding, Fold Other</td>
<td>Fold all possibly foldable elements, except for the current selection scope.</td>
</tr>
<tr>
<td>Code Folding, Fold All</td>
<td>Fold all possibly foldable elements.</td>
</tr>
</tbody>
</table>
Find Menu

For more information, see Finding and Replacing Text (p. 131).

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Folding, Unfold All</td>
<td>Unfold code folding for the entire file.</td>
</tr>
<tr>
<td>Code Formatting, Apply Code Formatting</td>
<td>Reformat the selected JavaScript code.</td>
</tr>
<tr>
<td>Code Formatting, Open Language &amp; Formatting Preferences</td>
<td>Open the Project Settings section of the Preferences tab to programming language settings.</td>
</tr>
</tbody>
</table>

Find Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find</td>
<td>Show the find and replace bar for the current document, with focus on the Find expression.</td>
</tr>
<tr>
<td>Find Next</td>
<td>Go to the next match in the current document for the find query you entered last.</td>
</tr>
<tr>
<td>Find Previous</td>
<td>Go to the previous match in the current document for the find query you entered last.</td>
</tr>
<tr>
<td>Replace</td>
<td>Show the find and replace bar for the current document, with focus on the Replace With expression.</td>
</tr>
<tr>
<td>Replace Next</td>
<td>Replace the next match for Find with Replace With in the find and replace bar for the current document.</td>
</tr>
<tr>
<td>Replace Previous</td>
<td>Replace the previous match for Find with Replace With in the find and replace bar for the current document.</td>
</tr>
<tr>
<td>Replace All</td>
<td>Replace all matches for Find with Replace With in the find and replace bar for the current document.</td>
</tr>
<tr>
<td>Find in Files</td>
<td>Show the find and replace bar for multiple files.</td>
</tr>
</tbody>
</table>

View Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Editors</td>
<td>Show the chosen editor.</td>
</tr>
<tr>
<td>Open Files</td>
<td>Show the Open Files list in the Environment window, or hide if shown.</td>
</tr>
<tr>
<td>Menu Bar</td>
<td>Show the menu bar, or hide if shown.</td>
</tr>
<tr>
<td>Tab Buttons</td>
<td>Show tabs, or hide if shown.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Gutter</td>
<td>Show the gutter, or hide if shown.</td>
</tr>
<tr>
<td>Status Bar</td>
<td>Show the status bar, or hide if shown.</td>
</tr>
<tr>
<td>Console</td>
<td>Show the Console window, or hide if shown.</td>
</tr>
<tr>
<td>Layout, Single</td>
<td>Show a single pane.</td>
</tr>
<tr>
<td>Layout, Vertical Split</td>
<td>Show two panes, top and bottom.</td>
</tr>
<tr>
<td>Layout, Horizontal Split</td>
<td>Show two panes, side by side.</td>
</tr>
<tr>
<td>Layout, Cross Split</td>
<td>Show four panes of equal size.</td>
</tr>
<tr>
<td>Layout, Split 1:2</td>
<td>Show one pane on the left and two panes on the right.</td>
</tr>
<tr>
<td>Layout, Split 2:1</td>
<td>Show two panes on the left and one pane on the right.</td>
</tr>
<tr>
<td>Font Size, Increase Font Size</td>
<td>Increase the font size.</td>
</tr>
<tr>
<td>Font Size, Decrease Font Size</td>
<td>Decrease the font size.</td>
</tr>
<tr>
<td>Syntax</td>
<td>Show the syntax type for the current document.</td>
</tr>
<tr>
<td>Themes</td>
<td>Show the IDE theme type.</td>
</tr>
<tr>
<td>Wrap Lines</td>
<td>Wrap words to the edge of the current pane, or stop wrapping words if they are already wrapping.</td>
</tr>
<tr>
<td>Wrap To Print Margin</td>
<td>Wrap words to the edge of the current print margin, or stop wrapping words if they are already wrapping.</td>
</tr>
</tbody>
</table>

## Go Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go To Anything</td>
<td>Show the Go window in Go to Anything mode.</td>
</tr>
<tr>
<td>Go To Symbol</td>
<td>Show the Go window in Go to Symbol mode.</td>
</tr>
<tr>
<td>Go To File</td>
<td>Show the Go window in Go to File mode.</td>
</tr>
<tr>
<td>Go To Command</td>
<td>Show the Go window in Go to Command mode.</td>
</tr>
<tr>
<td>Go To Line</td>
<td>Show the Go window in Go to Line mode.</td>
</tr>
<tr>
<td>Next Error</td>
<td>Go to the next error.</td>
</tr>
<tr>
<td>Previous Error</td>
<td>Go to the previous error.</td>
</tr>
<tr>
<td>Word Right</td>
<td>Go one word to the right.</td>
</tr>
<tr>
<td>Word Left</td>
<td>Go one word to the left.</td>
</tr>
</tbody>
</table>
**Run Menu**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line End</td>
<td>Go to the end of the current line.</td>
</tr>
<tr>
<td>Line Start</td>
<td>Go to the start of the current line.</td>
</tr>
<tr>
<td>Jump to Definition</td>
<td>Go to the definition of the variable or function at the cursor.</td>
</tr>
<tr>
<td>Jump to Matching Brace</td>
<td>Go to the matching symbol in the current scope.</td>
</tr>
<tr>
<td>Scroll to Selection</td>
<td>Scroll the selection into better view.</td>
</tr>
</tbody>
</table>

**Run Menu**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run</td>
<td>Run or debug the current application.</td>
</tr>
<tr>
<td>Run Last</td>
<td>Run or debug the last run file.</td>
</tr>
<tr>
<td>Run With</td>
<td>Run or debug using the chosen runner. See <em>Working with Builders, Runners, and Debuggers</em> (p. 146).</td>
</tr>
<tr>
<td>Run History</td>
<td>View run history.</td>
</tr>
<tr>
<td>Run Configurations</td>
<td>Choose a run configuration to run or debug with, or create or manage run configurations. See <em>Working with Builders, Runners, and Debuggers</em> (p. 146).</td>
</tr>
<tr>
<td>Show Debugger at Break</td>
<td>When running code reaches a breakpoint, show the Debugger window.</td>
</tr>
<tr>
<td>Build</td>
<td>Build the current file.</td>
</tr>
<tr>
<td>Cancel Build</td>
<td>Stop building the current file.</td>
</tr>
<tr>
<td>Build System</td>
<td>Build using the chosen build system.</td>
</tr>
<tr>
<td>Show Build Result</td>
<td>Show the related build result.</td>
</tr>
<tr>
<td>Automatically Build Supported Files</td>
<td>Automatically build supported files.</td>
</tr>
<tr>
<td>Save All on Build</td>
<td>When building, save all related unsaved files.</td>
</tr>
</tbody>
</table>

**Tools Menu**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strip Trailing Space</td>
<td>Trim whitespace at the ends of lines.</td>
</tr>
<tr>
<td>Preview, Preview File</td>
<td>Preview the current document in a preview tab.</td>
</tr>
</tbody>
</table>
### Window Menu

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preview, Preview Running Application</td>
<td>Preview the current application in a separate web browser tab.</td>
</tr>
<tr>
<td>Preview, Configure Preview URL</td>
<td>Open the Project Settings section of the Preferences tab to the Run &amp; Debug, Preview URL box.</td>
</tr>
<tr>
<td>Preview, Show Active Servers</td>
<td>Show a list of available active server addresses in the Process List dialog box.</td>
</tr>
<tr>
<td>Process List</td>
<td>Show the Process List dialog box.</td>
</tr>
<tr>
<td>Show Autocomplete</td>
<td>Show the code completion context menu.</td>
</tr>
<tr>
<td>Rename Variable</td>
<td>Start a rename refactor for the selection.</td>
</tr>
<tr>
<td>Toggle Macro Recording</td>
<td>Start keystroke recording, of stop if it is already recording.</td>
</tr>
<tr>
<td>Play Macro</td>
<td>Play previously recorded keystrokes.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go</td>
<td>Show the Go window, or hide if shown.</td>
</tr>
<tr>
<td>New Terminal</td>
<td>Open a new Terminal tab.</td>
</tr>
<tr>
<td>New Immediate Window</td>
<td>Open a new Immediate tab.</td>
</tr>
<tr>
<td>Share</td>
<td>Show the Share this environment dialog box.</td>
</tr>
<tr>
<td>Installer</td>
<td>Show the AWS Cloud9 Installer dialog box.</td>
</tr>
<tr>
<td>Collaborate</td>
<td>Show the Collaborate window, or hide if shown.</td>
</tr>
<tr>
<td>Outline</td>
<td>Show the Outline window, or hide if shown.</td>
</tr>
<tr>
<td>AWS Resources</td>
<td>Show the AWS Resources window, or hide if shown.</td>
</tr>
<tr>
<td>Environment</td>
<td>Show the Environment window, or hide if shown.</td>
</tr>
<tr>
<td>Debugger</td>
<td>Show the Debugger window, or hide if shown.</td>
</tr>
<tr>
<td>Navigation, Tab to the Right</td>
<td>Go one tab right.</td>
</tr>
<tr>
<td>Navigation, Tab to the Left</td>
<td>Go one tab left.</td>
</tr>
<tr>
<td>Navigation, Next Tab in History</td>
<td>Go to the next tab.</td>
</tr>
<tr>
<td>Navigation, Previous Tab in History</td>
<td>Go to the previous tab.</td>
</tr>
<tr>
<td>Navigation, Move Tab to Right</td>
<td>Move the current tab right. If the tab is already at the far right, create a split tab there.</td>
</tr>
</tbody>
</table>
### Command | Description
---|---
Navigation, Move Tab to Left | Move the current tab left. If the tab is already at the far left, create a split tab there.
Navigation, Move Tab to Up | Move the current tab up one pane. If the tab is already at very top, create a split tab there.
Navigation, Move Tab to Down | Move the current tab down one pane. If the tab is already at the very bottom, create a split tab there.
Navigation, Go to Pane to Right | Go one pane right.
Navigation, Go to Pane to Up | Go one pane left.
Navigation, Go to Pane to Up | Go one pane up.
Navigation, Go to Pane to Down | Go one pane down.
Navigation, Switch Between Editor and Terminal | Switch between the editor and the Terminal tab.
Saved Layouts, Save | Save the current layout. To switch to this layout later, choose Saved Layouts, LAYOUT-ID.
Saved Layouts, Save and Close All | Save the current layout, and then close all tabs and panes.
Saved Layouts, Show Saved Layouts in File Tree | Show all saved layouts in the Environment window.
Tabs, Close Pane | Close the current pane.
Tabs, Close All Tabs In All Panes | Close all open tabs in all panes.
Tabs, Close All But Current Tab | Close all open tabs in the current pane, except the current tab.
Tabs, Split Pane in Two Rows | Split the current pane into two panes, top and bottom.
Tabs, Split Pane in Two Columns | Split the current pane into two panes, left and right.
Presets, Full IDE | Switch to full IDE mode.
Presets, Minimal Editor | Switch to minimal editor mode.
Presets, Sublime Mode | Switch to Sublime mode.

### Support Menu

| Command | Description |
---|---|
Welcome Page | Open the Welcome tab. |
Finding and Replacing Text in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the find and replace bar in the AWS Cloud9 IDE to find and replace text in a single file or multiple files.

- Find Text in a Single File (p. 132)
- Replace Text in a Single File (p. 132)
- Find Text in Multiple Files (p. 132)
- Replace Text in Multiple Files (p. 132)
- Find and Replace Options (p. 134)
Find Text in a Single File

1. Open the file you want to find text in. If the file is already open, choose the file's tab to make the file active.
2. On the menu bar, choose Find, Find.
3. In the find and replace bar, for Find, type the text you want to find.
4. To specify additional find options, see Find and Replace Options (p. 134).
5. If there are any matches, 0 of 0 in the Find box changes to non-zero numbers. If there are any matches, the editor goes to the first match. If there is more than one match, to go to the next match, choose the right arrow in the Find box or choose Find, Find Next on the menu bar. To go to the previous match, choose the left arrow in the Find box or choose Find, Find Previous on the menu bar.

Replace Text in a Single File

1. Open the file you want to replace text in. If the file is already open, choose the file's tab to make the file active.
2. On the menu bar, choose Find, Replace.
3. In the find and replace bar, for Find, type the text you want to find.
4. For Replace With, type the text you want to replace the text in Find with.
5. To specify additional find and replace options, see Find and Replace Options (p. 134).
6. If there are any matches, 0 of 0 in the Find box changes to non-zero numbers. If there are any matches, the editor goes to the first match. If there is more than one match, to go to the next match, choose the right arrow in the Find box or choose Find, Find Next on the menu bar. To go to the previous match, choose the left arrow in the Find box or choose Find, Find Previous on the menu bar.
7. To replace the current match with the text in Replace With and then go to the next match, choose Replace. To replace all matches with the text in Replace With, choose Replace All.

Find Text in Multiple Files

1. On the menu bar, choose Find, Find in Files.
2. In the find and replace bar, for Find, type the text you want to find.
3. To specify additional find options, see Find and Replace Options (p. 134).
4. In the box to the right of the Find button (the box with *,.*, -.*) type any set of files to include or exclude in the find. For example:
   - Blank, *, or *.*: Find all files.
   - my-file.txt: Find only the file named my-file.txt.
   - my*: Find only files with file names starting with my.
   - my*.txt: Find only files with file names starting with my and that have the file extension .txt.
   - my*.htm*: Find all files with file names starting with my and a file extension starting with .htm.
   - my*.htm, my*.html: Find all files with file names starting with my and the file extension .htm or .html.
   - ~my-file.txt: Do not search the file named my-file.txt.
   - ~my*: Do not search any files starting with my.
   - ~my*.htm*: Do not search any files with file names starting with my and a file extension starting with .htm.
• my*.htm*, -my*.html: Search all files with file names starting with my and a file extension starting with .htm. However, do not search any files with file names starting with my and a file extension of .html.

5. In the drop-down list next to the preceding box, choose one of the following to further restrict the find to only specific locations:
   • Environment: Find only files in the Environment window.
   • Project (excludes .gitignore'd): Find any file in the environment, except for files or file types listed in the .gitignore file in the environment, if a .gitignore file exists.
   • Selection: Find only files that are currently selected in the Environment window.
   
   **Note**
   To further restrict the find to only a single folder, choose a folder in the Environment window and then choose Selection. Alternatively, you can right-click the folder in the Environment window, and then choose Search In This Folder on the context menu.
   • Favorites: Find only files in the Favorites list in the Environment window.
   • Active File: Find only the active file.
   • Open Files: Find only files in the Open Files list in the Environment window.

6. Choose Find.

7. To go to a file containing matches, double-click the file name on the Search Results tab. To go to a specific match, double-click the match in the Search Results tab.

---

**Replace Text in Multiple Files**

1. On the menu bar, choose Find, Find in Files.
2. In the find and replace bar, for Find, type the text you want to find.
3. To specify additional find options, see Find and Replace Options (p. 134).
4. In the box to the right of the Find button (the box with *. * , -. *), type any set of files to include or exclude in the find. For example:
   • Blank, * , or *.*: All files.
   • my-file.txt: Only the file named my-file.txt.
   • my*: Only files with file names staring with my.
   • my*.txt: Only files with file names starting with my and that have the file extension .txt.
   • my*.htm*: All files with file names starting with my and a file extension starting with .htm.
   • my*.htm, my*.html: All files with file names starting with my and the file extension .htm or .html.
   • -my-file.txt: Do not search the file named my-file.txt.
   • -my*: Do not search any files starting with my.
   • -my*.htm*: Do not search any files with file names starting with my and a file extension starting with .htm.
   • my*.htm*, -my*.html: Search all files with file names starting with my and a file extension starting with .htm. However, do not search any files with file names starting with my and a file extension of .html.

5. In the drop-down list next to the preceding box, choose one of the following to further restrict the find to only specific locations:
   • Environment: Only files in the Environment window.
   • Project (excludes .gitignore'd): Any file in the environment, except for files or file types listed in the .gitignore file in the environment, if a .gitignore file exists.
   • Selection: /: Only files that are currently selected.
   • Favorites: Only files in the Favorites list in the Environment window.
Find and Replace Options

Choose any of the following buttons on the find and replace bar to modify find and replace operations.

- **Regular Expressions**: Find text matching the specified regular expression in Find or Find in Files. See *Writing a regular expression pattern* in the *JavaScript Regular Expressions* topic on the Mozilla Developer Network.
- **Match Case**: Find text matching the specified casing in Find or Find in Files.
- **Whole Words**: Use standard word character rules to find text in Find or Find in Files.
- **Wrap Around**: For a single file only, do not stop at the end or beginning of the file when going to the next or previous match.
- **Search Selection**: For a single file only, find only in the selection.
- **Show in Console**: For multiple files, show the Search Results tab in the Console instead of the active pane.
- **Preserve Case**: For a single file only, preserve casing as applicable when replacing text.

Previewing Files in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to preview the files in a AWS Cloud9 development environment from within the IDE.

- **Open a File for Preview (p. 135)**
Open a File for Preview

Do one of the following in the AWS Cloud9 IDE to open a file preview tab within the environment:

- In the Environment window, right-click the file you want to preview, and then choose Preview.
  
  **Note**  
  Although you can use this approach to preview any file, preview works best with files that have the following file extensions:
  
  - .htm
  - .html
  - .pdf
  - .svg
  - .xhtml
  - Any file containing content in Markdown format.

- Open a file with one of the following file extensions:
  
  - .pdf
  - .svg

- With the file you want to preview already open and active, on the menu bar, choose Preview, Preview File FILE_NAME. Or choose Tools, Preview, Preview File FILE_NAME, where FILE_NAME is the name of the file you want to preview.
  
  **Note**  
  These commands work only with the following file types:
  
  - .htm
  - .html
  - .markdown
  - .md
  - .pdf
  - .svg
  - .txt: Preview works best if the file's content is in Markdown format.
  - .xhtml: Preview works best if the file contains or references content presentation information.

  **Note**  
  The Preview Settings menu in the file preview tab is currently not functional and choosing any of its menu commands will have no effect.

Reload a File Preview

On the file preview tab, choose the Refresh button (the circular arrow).

Change the File Preview Type

On the file preview tab, choose one of the following in the preview type list:
Open a File Preview in a Separate Web Browser Tab

On the file preview tab, choose **Pop Out Into New Window**.

Switch to a Different File Preview

On the file preview tab, type the path to a different file path in the address bar. The address bar is located between the **Refresh** button and the preview type list.

Previewing Running Applications in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to preview a running application from within the IDE.

**Topics**

- Run an Application (p. 136)
- Preview a Running Application (p. 138)
- Reload an Application Preview (p. 138)
- Change the Application Preview Type (p. 138)
- Open an Application Preview in a Separate Web Browser Tab (p. 139)
- Switch to a Different Preview URL (p. 139)
- Share a Running Application over the Internet (p. 139)

Run an Application

Before you can preview your application from within the IDE, it must be running in the AWS Cloud9 development environment using HTTP over port 8080, 8081, or 8082 with the IP of 127.0.0.1, localhost, or 0.0.0.0.

**Note**

You don't have to run using HTTP over port 8080, 8081, or 8082 with the IP of 127.0.0.1, localhost, or 0.0.0.0. However, you won't be able to preview your running application from within the IDE.

If you run with the IP of 0.0.0.0, anyone can potentially access your running application. For approaches to address this issue, see the following:

- **Step 2: Set Up the Security Group for the Instance (p. 140)** in *Share a Running Application over the Internet*
To write the code to run your application on a specific port and IP, see your application's documentation.

To run your application, see Run Your Code (p. 147).

To test this behavior, for example you could add the following JavaScript code to a file with a name such as `server.js` in the root of your environment. This code runs a server using Node.js, as follows.

```javascript
var http = require('http');
var fs = require('fs');
var url = require('url');

http.createServer( function (request, response) {
    var pathname = url.parse(request.url).pathname;
    console.log("Trying to find " + pathname.substr(1) + "...");

    fs.readFile(pathname.substr(1), function (err, data) {
        if (err) {
            response.writeHead(404, {'Content-Type': 'text/html'});
            response.write("ERROR: Cannot find " + pathname.substr(1) + ".");
            console.log("ERROR: Cannot find " + pathname.substr(1) + ".");
        } else {
            console.log("Found " + pathname.substr(1) + ".");
            response.writeHead(200, {'Content-Type': 'text/html'});
            response.write(data.toString());
        }
        response.end();
    });
}).listen(8080, 'localhost'); // Or 8081 or 8082 instead of 8080. Or '127.0.0.1' instead of 'localhost'.
```

Or you could add the following Python code to a file with a name such as `server.py` in the root of your environment. This code runs a server using Python, as follows.

```python
import os
import SimpleHTTPServer
import SocketServer

ip = 'localhost' # Or '127.0.0.1' instead of 'localhost'.
port = '8080' # Or '8081' or '8082' instead of '8080'.
Handler = SimpleHTTPServer.SimpleHTTPRequestHandler
httpd = SocketServer.TCPServer((ip, int(port)), Handler)
httpd.serve_forever()
```

Next, add the following HTML code to a file with a name such as `index.html` in the root of your environment.

```html
<html>
<head>
    <title>Hello Home Page</title>
</head>
<body>
    <p style="font-family:Arial;color:blue">Hello, World!</p>
</body>
</html>
```

To see this file's HTML output on the application preview tab, run `server.js` with Node.js or `server.py` file with Python. Then follow the instructions in the next procedure to preview it. On the application preview tab, add `/index.html` to the end of the URL, and then press Enter.
Preview a Running Application

With your application already running using HTTP over port 8080, 8081, or 8082 with the IP of 127.0.0.1, localhost, or 0.0.0.0 in the environment, and with the corresponding application code file open and active in the AWS Cloud9 IDE, choose one of the following on the menu bar:

- Preview, Preview Running Application
- Tools, Preview, Preview Running Application

This opens an application preview tab within the environment, and then displays the application's output on the tab.

If the application preview tab displays an error or is blank, try following the troubleshooting steps in Application Preview Tab Displays an Error or is Blank (p. 548).

To enable others to preview the running application outside of the IDE, see Share a Running Application over the Internet (p. 139).

**Note**
If the application is not already running, you will see an error on the application preview tab. Run or restart the application, and then choose the menu bar command again.

If your application cannot run on any of the preceding ports or IPs, or if your application must run on more than one of these ports at the same time (for example, your application must run on ports 8080 and 3000 at the same time), the application preview tab might display an error or might be blank. This is because the application preview tab within the environment works only with the preceding ports and IPs, and it works with only a single port at a time.

We don't recommend sharing the URL in the application preview tab with others. (The URL displays using the format https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/, where 12a34567b8cd9012345ef67abcd890e1 is the ID that AWS Cloud9 assigns to the environment, and us-east-2 is the ID of the AWS Region for the environment.) This URL works only when the IDE for the environment is open and the application is running in the same web browser.

If you try to go to the IP of 127.0.0.1, localhost, or 0.0.0.0 by using the application preview tab in the IDE or in a separate web browser tab outside of the IDE, the default built-in behavior of the AWS Cloud9 IDE is that this will attempt to go to your local computer, instead of attempting to go to the instance or your own server that is connected to the environment.

Reload an Application Preview

On the application preview tab, choose the Refresh button (the circular arrow).

**Note**
This command does not restart the server. It just refreshes the contents of the application preview tab.

Change the Application Preview Type

On the application preview tab, choose one of the following in the preview type list:

- **Browser**: Previews the output in a web browser format.
- **Raw Content (UTF-8)**: Attempts to preview the output in Unicode Transformation Format 8-bit (UTF-8) format, if applicable.
- **Markdown**: Attempts to preview the output in Markdown format, if applicable.
Open an Application Preview in a Separate Web Browser Tab

On the application preview tab, choose **Pop Out Into New Window**.

**Note**
The application preview will not be displayed in a separate web browser tab unless the AWS Cloud9 IDE is also running in at least one other tab in the same web browser.

Switch to a Different Preview URL

On the application preview tab, type the path to a different URL in the address bar. The address bar is located between the **Refresh** button and the preview type list.

Share a Running Application over the Internet

After you preview your running application, you can make it available to others over the internet.

To do this, if an Amazon EC2 instance is connected to your environment, follow these steps. Otherwise, see your server's documentation.

**Topics**
- Step 1: Get the ID and the IP Address of the Instance (p. 139)
- Step 2: Set Up the Security Group for the Instance (p. 140)
- Step 3: Set Up the Subnet for the Instance (p. 141)
- Step 4: Change the Running Application IP (p. 142)
- Step 5: Share the Running Application URL (p. 142)

**Step 1: Get the ID and the IP Address of the Instance**

In this step, you note the instance ID and public IP address for the Amazon EC2 instance that is connected to the environment. You need the instance ID in a later step to allow incoming application requests. Then you give the public IP address to others so that they can access the running application.

1. Get the Amazon EC2 instance's ID. To get this, do one of the following:
   - In a terminal session in the AWS Cloud9 IDE for the environment, run the following command to get the Amazon EC2 instance's ID.
     
     ```bash
     ```
     
     The instance ID will look similar to this: `i-12a3b456c789d0123`. Make a note of this instance ID.
   - In the IDE for the environment, on the menu bar, choose your user icon, and then choose **Manage EC2 Instance**.
In the Amazon EC2 console that displays, make a note of the instance ID that displays in the Instance ID column. The instance ID will look similar to this: i-12a3b456c789d0123.

2. Get the Amazon EC2 instance's public IP address. To get this, do one of the following:

   • In the IDE for the environment, on the menu bar, choose Share. In the Share this environment dialog box, make a note of the public IP address in the Application box. The public IP address will look similar to this: 192.0.2.0.

   • In a terminal session in the IDE for the environment, run the following command to get the Amazon EC2 instance's public IP address:

     ```bash
     ```

     The public IP address will look similar to this: 192.0.2.0. Make a note of this public IP address.

   • In the IDE for the environment, on the menu bar, choose your user icon, and then choose Manage EC2 Instance. In the Amazon EC2 console that displays, on the Description tab, make a note of the public IP address for the IPv4 Public IP field. The public IP address will look similar to this: 192.0.2.0.

     **Note**
     The instance's public IP address might change anytime the instance restarts. To prevent this IP address from changing, one solution is to allocate an Elastic IP address and then assign that address to the running instance. For instructions, see Allocating an Elastic IP Address and Associating an Elastic IP Address with a Running Instance in the Amazon EC2 User Guide for Linux Instances. Note also that allocating an Elastic IP address might result in charges to your AWS account. For more information, see Amazon EC2 Pricing.

### Step 2: Set Up the Security Group for the Instance

In this step, you use the Amazon EC2 console to set up the Amazon EC2 security group for the instance that is connected to the environment, to allow incoming HTTP requests over port 8080, 8081, or 8082.

**Note**
You don't have to run using HTTP over port 8080, 8081, or 8082. If you are running on a different protocol or port, substitute it throughout this step. You won't be able to preview your running application from within the IDE until you switch back to running using HTTP over one of the ports and IPs as described in Preview a Running Application (p. 138).

For an additional layer of security, you can also set up a network access control list (ACL) for a subnet in a virtual private cloud (VPC) that the instance can use. For more information about security groups and network ACLs, see the following:

- Step 3: Set Up the Subnet for the Instance (p. 141)
- Security in the Amazon VPC User Guide
- Security Groups for Your VPC in the Amazon VPC User Guide
- Network ACLs in the Amazon VPC User Guide

1. In the IDE for the environment, on the menu bar, choose your user icon, and then choose Manage EC2 Instance. Then skip ahead to step 3 in this procedure.

2. If choosing Manage EC2 Instance or other steps in this procedure display errors, we recommend you sign in to the Amazon EC2 console using credentials for an IAM administrator user in your AWS account, and then complete the following instructions. If you cannot do this, check with your AWS account administrator.
a. Sign in to the AWS Management Console, if you are not already signed in, at https://console.aws.amazon.com.
b. Open the Amazon EC2 console. To do this, in the AWS navigation bar, choose Services. Then choose EC2.
c. In the AWS navigation bar, choose the AWS Region where the environment is located.
d. If the EC2 Dashboard is displayed, choose Running Instances. Otherwise, in the service navigation pane, expand Instances if it is not already expanded, and then choose Instances.
e. In the list of instances, select the instance where the Instance ID matches the instance ID you noted earlier.

3. In the Description tab for the instance, choose the security group link next to Security groups.
4. With the security group displayed, look on the Inbound tab. If a rule already exists where Type is set to Custom TCP Rule and Port Range is set to 8080, 8081, or 8082, choose Cancel, and skip ahead to Step 3: Set Up the Subnet for the Instance (p. 141). Otherwise, choose Edit.
5. In the Edit inbound rules dialog box, choose Add Rule.
6. For Type, choose Custom TCP Rule.
7. For Port Range, type 8080, 8081, or 8082.
8. For Source, choose Anywhere.

**Note**
Choosing Anywhere for Source allows incoming requests from any IP address. To restrict this to specific IP addresses, choose Custom and then type the IP address range, or choose My IP to restrict this to requests from your IP address only.
9. Choose Save.

**Step 3: Set Up the Subnet for the Instance**

In this step, you use the consoles for Amazon EC2 and Amazon Virtual Private Cloud (Amazon VPC) to set up the subnet for the Amazon EC2 instance that is connected to the environment, to also allow incoming HTTP requests over port 8080, 8081, or 8082.

**Note**
You don't have to run using HTTP over port 8080, 8081, or 8082. If you are running on a different protocol or port, substitute it throughout this step. You won't be able to preview your running application from within the IDE until you switch back to running using HTTP over the ports and IPs as described in Preview a Running Application (p. 138). This step describes how to set up a network ACL for a subnet in an Amazon VPC that the instance can use. This step is not required. However, it adds an additional layer of security when compared to just using security groups. For more information about network ACLs, see the following:

- Security in the Amazon VPC User Guide
- Network ACLs in the Amazon VPC User Guide

1. With the Amazon EC2 console already open from the previous step, in the service navigation pane, expand Instances if it is not already expanded, and then choose Instances.
2. In the list of instances, select the instance where the Instance ID matches the instance ID you noted earlier.
3. In the Description tab for the instance, note the value of Subnet ID. It should look similar to this: subnet-1fab8aEX.
4. Open the Amazon VPC console. To do this, in the AWS navigation bar, choose Services. Then choose VPC.
For this step, we recommend you sign in to the Amazon VPC console using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.

5. If the VPC Dashboard is displayed, choose Subnets. Otherwise, in the service navigation pane, choose Subnets.

6. In the list of subnets, select the subnet where the Subnet ID value matches the one you noted earlier.

7. On the Summary tab, choose the network ACL link next to Network ACL.

8. In the list of network ACLs, select the network ACL. (There is only one network ACL.)

9. Look on the Inbound Rules tab for the network ACL. If a rule already exists where Type is set to HTTP* (8080), HTTP* (8081), or HTTP* (8082), skip ahead to Step 4: Change the Running Application IP (p. 142). Otherwise, choose Edit.

10. Choose Add another rule.

11. For Rule #, type a number for the rule (for example, 200).

12. For Type, choose Custom TCP Rule.

13. For Port Range, type 8080, 8081, or 8082.

14. For Source, type the range of IP addresses to allow incoming requests from. For example, to allow incoming requests from any IP address, type 0.0.0.0/0.

15. With Allow / Deny set to ALLOW, choose Save.

**Step 4: Change the Running Application IP**

In your code, switch from using IP 127.0.0.1, localhost, or 0.0.0.0 to using the IP address or addresses you specified in the previous steps in this section. To use these new IPs, stop the application if is already running, and then run the application again.

**Note**

You won’t be able to preview your running application from within the IDE until you switch back to running using HTTP over one of the ports and IPs as described in Preview a Running Application (p. 138).

**Step 5: Share the Running Application URL**

With the application running, give to others the public IP address you noted earlier. Be sure to start the URL with the correct protocol, and add the port number if it is not the default for that protocol (for example, http://192.0.2.0:8080/index.html using HTTP over port 8080).

If the resulting web browser tab displays an error, or the tab is blank, try following the troubleshooting steps in Cannot Display Your Running Application Outside of the IDE (p. 549).

**Note**

The instance’s public IP address might change anytime the instance restarts. To prevent this IP address from changing, one solution is to allocate an Elastic IP address and then assign that address to the running instance. For instructions, see Allocating an Elastic IP Address and Associating an Elastic IP Address with a Running Instance in the Amazon EC2 User Guide for Linux Instances. Note also that allocating an Elastic IP address might result in charges to your AWS account. For more information, see Amazon EC2 Pricing.

You don’t have to run using HTTP over port 8080, 8081, or 8082. However, you won’t be able to preview your running application from within the IDE until you switch back to running using HTTP over one of the ports and IPs as described in Preview a Running Application (p. 138).

If users make requests to the preceding URL, and those requests originate from a virtual private network (VPN) that blocks traffic over the requested protocol or port, those requests might fail. Those users must use a different network that allows traffic over the requested protocol and port. For more information, see your network administrator.
We don't recommend sharing the URL in the application preview tab in the IDE with others. (The URL displays using the format https://12a34567b8cd9012345ef67abcd890e1.vfs.cloud9.us-east-2.amazonaws.com/, where 12a34567b8cd9012345ef67abcd890e1 is the ID that AWS Cloud9 assigns to the environment, and us-east-2 is the ID of the AWS Region for the environment.) This URL works only when the IDE for the environment is open and the application is running in the same web browser.

Working with File Revisions in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the File Revision History pane in the AWS Cloud9 IDE to view and manage changes to a file in an AWS Cloud9 EC2 development environment. The File Revision History pane is not available for files in an AWS Cloud9 SSH development environment.

To show the File Revision History pane for a file, open the file in the editor. Then, on the menu bar, choose File, Show File Revision History.

The File Revision History pane begins tracking a file's revision history in the IDE after you first open the file in the editor in an environment, and only for that environment. The File Revision History pane tracks a file's revisions only from the editor itself. It does not track a file's revisions made in any other way (for example by the terminal, Git, or other file revision tools).

You cannot edit a file while the File Revision History pane is displayed. To hide the pane, choose File, Show Revision History again, or choose the X (Close timeslider) in the corner of the pane.
To jump to a version of the file that is associated with a file save action, choose a **File Saved on** dot above the revision slider.

To go forward or backward one version from the currently selected version of the file on the revision slider, choose one of the step arrows (**Step revision forward** or **Step revision backward**).

To go forward automatically one version of the file at a time from the beginning to end of the revision history, choose the play button (**Playback file history**).

To make the currently selected version of the file the latest version in the revision history, choose **Revert**.

### Working with Images Files in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to view and edit image files.

- View or Edit an Image (p. 144)
- Resize an Image (p. 144)
- Crop an Image (p. 145)
- Rotate an Image (p. 145)
- Flip an Image (p. 145)
- Zoom an Image (p. 145)
- Smooth an Image (p. 146)

### View or Edit an Image

In the AWS Cloud9 IDE, open the file for the image you want to view or edit. Supported image file types include the following:

- .bmp
- .gif (view only)
- .ico (view only)
- .jpeg
- .jpg
- .png
- .tiff

### Resize an Image

1. Open the image file in the IDE.
Crop an Image

1. Open the image file in the IDE.
2. Drag the mouse pointer over the portion of the image that you want to keep.
3. To confirm the selection's dimensions, on the image editing bar, see the selection dimensions, as follows:
   - The distance in pixels from the original image's left edge to the left edge of the selection (L)
   - The distance in pixels from the original image's top edge to the top edge of the selection (T)
   - The selection's width in pixels (W)
   - The selection's height in pixels (H)
4. On the image editing bar, choose Crop.
5. To discard the crop, on the menu bar, choose Edit, Undo. To keep the new cropped image, choose File, Save.

Rotate an Image

1. Open the image file in the IDE.
2. To rotate the image counterclockwise, on the image editing bar, choose Rotate 90 Degrees Left.
3. To rotate the image clockwise, on the image editing bar, choose Rotate 90 Degrees Right.
4. To discard the rotation, on the menu bar, choose Edit, Undo. To keep the new rotated image, choose File, Save.

Flip an Image

1. Open the image file in the IDE.
2. To flip the image horizontally, on the image editing bar, choose FlipH.
3. To flip the image vertically, on the image editing bar, choose FlipV.
4. To discard the flip, on the menu bar, choose Edit, Undo. To keep the new flipped image, choose File, Save.

Zoom an Image

1. Open the image file in the IDE.
2. On the image editing bar, choose one of the available zoom factors (for example, 75%, 100%, or 200%).
Smooth an Image

1. Open the image file in the IDE.
2. On the image editing bar, select Smooth to reduce the amount of pixelation in the image. To discard the smoothing, deselect Smooth.
3. On the menu bar, choose File, Save.

Working with Builders, Runners, and Debuggers in the AWS Cloud9 Integrated Development Environment (IDE)

A builder instructs the AWS Cloud9 IDE how to build a project's files. A runner instructs the AWS Cloud9 IDE how to run files of a specific type. A runner can use a debugger to help find any problems in the source code of the files.

You can use the AWS Cloud9 IDE to build, run, and debug your code in the following ways:

• Use a builder to build your project's files. See Build Your Project's Files (p. 146).
• Use a runner to run (and optionally, to debug) your code. See Built-In Build, Run, and Debug Support (p. 146) and Run Your Code (p. 147).
• Change a built-in runner to run (and optionally, to debug) your code in a different way from how it was originally defined. See Change a Built-In Runner (p. 150).
• Use a runner to run (and optionally, to debug) your code with a custom combination of file name, command line options, debug mode, current working directory, and environment variables. See Create a Run Configuration (p. 150).
• Create your own builder or runner. See Create a Builder or Runner (p. 151).

Built-In Build, Run, and Debug Support

The AWS Cloud9 IDE provides built-in support for building, running, and debugging code for several languages. For a complete list, see Language Support (p. 120).

Built-in build support is available on the menu bar with the Run, Build System and Run, Build menu commands. To add support for a programming language or tool that isn't listed, see Create a Builder or Runner (p. 151).

Built-in run support is available with the Run button, and on the menu bar with the Run, Run With and Run, Run Configurations menu commands. To add support for a programming language or tool that isn't listed, see Create a Builder or Runner (p. 151) and Create a Run Configuration (p. 150).

Built-in debug support is available through the Debugger window. To display the Debugger window, choose the Debugger button. If the Debugger button is not visible, choose Window, Debugger on the menu bar.

Build Your Project's Files

1. Open a file that corresponds to the code you want to build.
2. On the menu bar, choose Run, Build System, and then choose the name of the builder to use, if it isn't already chosen. If the builder you want to use isn't listed, stop this procedure, complete the steps in Create a Builder or Runner (p. 151), and then return to this procedure.
3. Choose Run, Build.

Run Your Code

1. Open a file that corresponds to the code you want to run, if the file isn't already open and selected.

2. On the menu bar, choose one of the following:
   - To run the code with the closest matching built-in runner, choose Run, Run. If AWS Cloud9 cannot find one, this command is disabled.
   - To run the code with the run configuration that AWS Cloud9 last used, choose Run, Run Last.
   - To run the code with a specific runner, choose Run, Run With, and then choose the name of the runner. If the runner you want to use isn't listed, stop this procedure, complete the steps in Create a Builder or Runner (p. 151), and then return to this procedure.
   - To run the code with a specific runner with a custom combination of file name, command line options, debug mode, current working directory, and environment variables, choose Run, Run Configurations, and then choose the run configuration’s name. In the run configuration tab that is displayed, choose Runner: Auto, choose the runner you want to use, and then choose Run. If the runner you want to use isn't listed, stop this procedure, complete the steps in Create a Builder or Runner (p. 151), and then return to this procedure.

Debug Your Code

1. On the run configuration tab for your code, choose Run in Debug Mode. The bug icon turns to green on a white background. For more information, see Run Your Code (p. 147) and Create a Run Configuration (p. 150).

2. Set any breakpoints in your code you want to pause at during the run, as follows:
   a. Open each file that you want to set a breakpoint in.
   b. At each point in a file where you want to set a breakpoint, choose the blank area in the gutter to the left of the line number. A red circle appears.

      To remove a breakpoint, choose the existing breakpoint in the gutter.

      To disable a breakpoint instead of removing it, in the Debugger window, in Breakpoints, clear the box that corresponds to the breakpoint you want to disable. To enable the breakpoint again, select the box you cleared.

      To disable all breakpoints at once, in the Debugger window, choose Deactivate All Breakpoints. To enable all breakpoints again, choose Activate All Breakpoints.

      If the Debugger window isn't visible, choose the Debugger button. If the Debugger button isn't visible, on the menu bar choose Window, Debugger.

3. Set any watch expressions for which you want to get the value at the point where a run pauses, as follows:
   a. In the Debugger window, in Watch Expressions, choose Type an expression here.
   b. Type the expression you want to watch, and then press Enter.

      To change an existing watch expression, right-click the watch expression, and then choose Edit Watch Expression. Type the change, and then press Enter.

      To remove an existing watch expression, right-click the watch expression, and then choose Remove Watch Expression.

4. Run your code as described in Run Your Code (p. 147).
Whenever a run pauses, you can do the following in the Debugger window, as shown.
• Run your code to the next breakpoint (or to the next logical stopping point if there are no more breakpoints): Choose Resume.

• Skip over running statement by statement through the next method or function call: Choose Step Over.

• Run your code to the next statement and then pause again: Choose Step Into.

• Stop running statement by statement in the current method or function call: Choose Step Out.

• Disable all existing breakpoints: Choose Deactivate All Breakpoints. Re-enable all breakpoints: Choose Activate All Breakpoints.

• Don't pause whenever the code throws an exception: Keep choosing the exceptions behavior button until the tooltip reads Don't pause on exceptions (gray).
• **Pause whenever the code throws an exception**: Keep choosing the exceptions behavior button until the tooltip reads *Pause on all exceptions* (red).

• **Pause only when the code throws an uncaught exception**: Keep choosing the exceptions behavior button until the tooltip reads *Pause on uncaught exceptions* (blue).

• **Open an available script**: Choose *Available internal and external scripts*, and then choose the script.

• **View the list of current watch expressions**: See the *Watch Expressions* area.

• **View the execution path that brought the code to the current breakpoint**: See the *Call Stack* area.

• **View the list of local variables**: See the *Local Variables* area.

• **Disable individual breakpoints**: In *Breakpoints*, clear the boxes that correspond to the breakpoints you want to disable. To enable the breakpoints again, select the boxes you cleared earlier.

Whenever a run pauses, you can also pause your mouse pointer on any displayed piece of code (for example, a variable) to show any available information about it in a tooltip.

## Change a Built-In Runner

1. On the menu bar, choose **Run, Run With**, and then choose the built-in runner you want to change.
2. Stop the runner from trying to run your code by choosing, **Stop** on the run configuration tab that displays.
3. Choose **Runner: My Runner**, where *My Runner* is the name of the runner you want to change, and then choose **Edit Runner**.
4. On the **My Runner.run** tab that is displayed, change the runner’s current definition. See *Define a Builder or Runner* (p. 151).
5. Choose **File, Save As**. Save the file with the same name (*My Runner.run*) in the `my-environment/.c9/runners` directory, where *my-environment* is the name of your AWS Cloud9 development environment.

**Note**
Any changes you make to a built-in runner apply only to the environment you made those changes in. To apply your changes to a separate environment, open the other environment, and then follow the preceding steps to open, edit, and save those same changes to that built-in runner.

## Create a Run Configuration

On the menu bar, choose **Run, Run Configurations, New Run Configuration**. On the run configuration tab that is displayed, do the following:

1. In the box next to **Run and Restart**, type the name that will display on the **Run, Run Configurations** menu for this run configuration.
2. In the **Command** box, type any custom command line options you want to use.
3. To have this run configuration use the runner’s predefined debugging settings, choose **Run in Debug Mode**. The bug icon will turn to green on a white background.
4. To have this run configuration use a specific working directory, choose **CWD**, choose the directory to use, and then choose **Select**.
5. To have this run configuration use specific environment variables, choose **ENV**, and then type the name and value of each environment variable.

To use this run configuration, open the file the corresponds to the code you want to run. Choose **Run, Run Configurations** on the menu bar, and then choose this run configuration’s name. In the run
configuration tab that displays, choose **Runner: Auto**, choose the runner you want to use, and then choose **Run**.

**Note**
Any run configuration you create applies only to the environment you created that run configuration in. To add that run configuration to a separate environment, open the other environment, and then follow the preceding steps to create the same run configuration in that environment.

### Create a Builder or Runner

1. To create a builder, on the menu bar, choose **Run, Build System, New Build System**. To create a runner, on the menu bar, choose **Run, Run With, New Runner**.
2. On the builder tab (labeled **My Builder.build**) or runner tab (labeled **My Runner.run**) that is displayed, define the builder or runner. See Define a Builder or Runner (p. 151).
3. After you define the builder or runner, choose **File, Save As**. For a builder, save the file with the .build extension in the my-environment/.c9/builders directory, where my-environment is the name of your environment.
   - For a runner, save the file with the .run file extension in the my-environment/.c9/runners directory, where my-environment is the name of your environment. The file name you specify will be the name that is displayed on the **Run, Build System** menu (for a builder) or the **Run, Run With** menu (for a runner). Therefore, unless you specify a different file name, by default the display name will be **My Builder** (for a builder) or **My Runner** (for a runner).

To use this builder or runner, see Build Your Project’s Files (p. 146) or Run Your Code (p. 147).

**Note**
Any builder or runner you create applies only to the environment you created that builder or runner in. To add that run builder or runner to a separate environment, open the other environment, and then follow the preceding steps to create the same builder or runner in that environment.

### Define a Builder or Runner

This procedure assumes you have already begun to create a builder or runner by choosing **Run, Build System, New Build System** (for a builder) or **Run, Run With, New Runner** (for a runner).

On the builder or runner tab that is displayed, use JSON to define the runner or builder. Start with the following code as a template.

**For a builder, start with this code.**

```json
{
  "cmd": [],
  "info": "",
  "env": {},
  "selector": ""
}
```

**For a runner, start with this code.**

```json
{
  "cmd": [],
  "script": "",
  "working_dir": "",
  "info": "",
  "env": {},
  "selector": ""
}
```
In the preceding code:

- **cmd**: Represents a comma-separated list of strings for AWS Cloud9 to run as a single command.

  When AWS Cloud9 runs this command, each string in the list will be separated by a single space. For example, AWS Cloud9 will run `"cmd": [ "ls", "$file", "$args"]` as `ls $file $args`, where AWS Cloud9 will replace `$file` with the full path to the current file and `$args` with any arguments entered after the file name. For more information, see the list of supported variables later in this section.

- **script**: Represents a bash script (which can also be specified as an array of lines as needed for readability) that the runner executes in the terminal.

- **working_dir**: Represents the directory that the runner will run from.

- **info**: Represents any string of text you want to display to the user at the beginning of the run. This string can contain variables, for example: `Running $project_path$file_name...`, where AWS Cloud9 will replace `$project_path` with the directory path of the current file and `$file_name` with the name portion of the current file. See the list of supported variables later in this section.

- **env**: Represents any array of command line arguments for AWS Cloud9 to use, for example:

  ```json
  "env": {
    "LANG": "en_US.UTF-8",
    "SHLVL": "1"
  }
  ```

- **selector**: Represents any regular expression that you want AWS Cloud9 to use to identify the file names that apply to this runner. For example, you could specify `source.py` for Python files.

- **debugger**: Represents the name of any available debugger you want AWS Cloud9 to use that is compatible with this runner. For example, you could specify `v8` for the V8 debugger.

- **debugport**: Represents the port number you want AWS Cloud9 to use during debugging. For example, you could specify `15454` for the port number to use.

The following table shows the variables you can use.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$file_path</code></td>
<td>The directory of the current file, for example, <code>/home/ec2-user/environment</code> or <code>/home/ubuntu/environment</code>.</td>
</tr>
<tr>
<td><code>$file</code></td>
<td>The full path to the current file, for example, <code>/home/ec2-user/environment/hello.py</code> or <code>/home/ubuntu/environment/hello.py</code>.</td>
</tr>
<tr>
<td><code>$args</code></td>
<td>Any arguments entered after the file name, for example, &quot;5&quot; &quot;9&quot;.</td>
</tr>
<tr>
<td><code>$file_name</code></td>
<td>The name portion of the current file, for example, <code>hello.py</code>.</td>
</tr>
<tr>
<td><code>$file_extension</code></td>
<td>The extension of the current file, for example, <code>py</code>.</td>
</tr>
<tr>
<td><code>$file_base_name</code></td>
<td>The name of the current file without the file extension, for example, <code>hello</code>.</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$packages</td>
<td>The full path to the packages folder.</td>
</tr>
<tr>
<td>$project</td>
<td>The full path to the current project folder.</td>
</tr>
<tr>
<td>$project_path</td>
<td>The directory of the current project file, for example, /home/ec2-user/environment/ or /home/ubuntu/environment/.</td>
</tr>
<tr>
<td>$project_name</td>
<td>The name of the current project file without the file extension, for example, my-demo-environment.</td>
</tr>
<tr>
<td>$project_extension</td>
<td>The extension of the current project file.</td>
</tr>
<tr>
<td>$project_base_name</td>
<td>The name of the current project file without the extension.</td>
</tr>
<tr>
<td>$hostname</td>
<td>The hostname of the environment, for example, 192.0.2.0.</td>
</tr>
<tr>
<td>$hostname_path</td>
<td>The hostname of the environment with the relative path to the project file, for example, <a href="https://192.0.2.0/hello.js">https://192.0.2.0/hello.js</a>.</td>
</tr>
<tr>
<td>$url</td>
<td>The full URL to access the environment, for example, <a href="https://192.0.2.0">https://192.0.2.0</a>.</td>
</tr>
<tr>
<td>$port</td>
<td>The port assigned to the environment, for example, 8080.</td>
</tr>
<tr>
<td>$ip</td>
<td>The IP address to run a process against the environment, for example, 0.0.0.0.</td>
</tr>
</tbody>
</table>

As an example, the following builder file named G++.build defines a builder for GCC that runs the `g++` command with the `-o` option to compile the current file (for example, hello.cpp) into an object module. Then it links the object module into a program with the same name as the current file (for example, hello). Here the equivalent command is `g++ -o hello hello.cpp`.

```
{
    "cmd": [ "g++", "-o", "$file_base_name", "$file_name" ],
    "info": "Compiling $file_name and linking to $file_base_name...",
    "selector": "source.cpp"
}
```

As another example, the following runner file named Python.run defines a runner that uses Python to run the current file with any arguments that were provided. For example, if the current file is named hello.py and the arguments 5 and 9 were provided, the equivalent command is `python hello.py 5 9`.

```
{
    "cmd": [ "python", "$file_name", "$args" ],
    "info": "Running $file_name...",
    "selector": "source.py"
}
```
Finally, the following runner file named `Print Run Variables.run` defines a runner that simply outputs the value of each available variable and then stops.

```json
{
    "info": "file_path = $file_path, file = $file, args = $args, file_name = $file_name, file_extension = $file_extension, file_base_name = $file_base_name, packages = $packages, project = $project, project_path = $project_path, project_name = $project_name, project_extension = $project_extension, project_base_name = $project_base_name, hostname = $hostname, hostname_path = $hostname_path, url = $url, port = $port, ip = $ip"
}
```

Working with Language Projects in the AWS Cloud9 Integrated Development Environment (IDE)

The AWS Cloud9 IDE provides project productivity features for some languages in addition to those languages listed in Language Support in the AWS Cloud9 Integrated Development Environment (IDE) (p. 120). To use these features, you use the IDE to create or identify a language project (or project) based on that language. A project is a collection of related files, folders, and settings in the IDE for an AWS Cloud9 development environment.

To use the IDE to create a language project in your environment, see Create a Language Project (p. 156).

Available Project Productivity Features

The AWS Cloud9 IDE provides the following project productivity features by programming language.

<table>
<thead>
<tr>
<th>Language</th>
<th>Autocomplete</th>
<th>Gutter Icons (p. 154)</th>
<th>Quick Fixes (p. 155)</th>
<th>Find References (p. 155)</th>
<th>Go to Definition (p. 156)</th>
<th>Go to Symbol (p. 156)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TypeScript</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

**Autocomplete**

As you type in a file in the editor, a list of symbols is displayed at the insertion point for that context, if any symbols are available there.

To insert a symbol from the list at the insertion point, if the symbol isn't already chosen, choose it by using your up arrow or down arrow key, and then press Tab.

Before you press Tab, you might see a screentip that contains information about the symbol you chose, if information is available.

To close the list without inserting a symbol, press Esc.

**Gutter Icons**

Icons might appear in the gutter for the active file. These icons highlight possible issues such as warnings and errors in code before you run it.

For more information about an issue, pause your mouse pointer on the issue's icon.
Quick Fixes

In the active file in the editor, you can display information about coding errors and warnings, with possible fixes that you can automatically apply to that code. To display error or warning information and possible fixes, choose any part of the code that has a red dotted underline (for errors), or a gray dotted underline (for warnings). Or, with the cursor resting on code that has a red or gray dotted underline, press Option-Enter (for macOS), or Alt-Enter (for Linux or Windows). To apply a proposed fix, choose the fix in the list, or use the arrow keys to select the fix and then press Enter. To turn choosing quick fixes with mouse clicks on or off, choose AWS Cloud9, Preferences, User Settings, Language, Hints & Warnings, Show Available Quick Fixes on Click.

Find References

In the active file in the editor, you can display all references to the symbol at the insertion point, if the IDE has access to those references.

To do this, at the insertion point anywhere within the symbol, run the Find References command. For example:

- Right-click at the insertion point, and then choose Find References.
- On the menu bar, choose Go, Find References.
- Press Shift-F3 by default for macOS, Windows, or Linux.

If references are available, a pane opens on top of the active file, next to that symbol. The pane contains a list of the files where the symbol is referenced. The pane displays the first reference in the list. To display a different reference, choose that reference in the list.

To close the pane, choose the close (X) icon in the pane, or press Esc.

The Find References command might be disabled, or might not work as expected, under the following conditions:

- There are no references to that symbol in the active file's project.
- The IDE can't find some or all of that symbol's references in the active file's project.
- The IDE doesn't have access to one or more locations where that symbol is referenced in the active file's project.

Go to Definition

In the active file in the editor, you can go from a symbol to where that symbol is defined, if the IDE has access to that definition.

To do this, at the insertion point anywhere within the symbol, run the Jump to Definition command. For example:

- Right-click at the insertion point, and then choose Jump to Definition.
- On the menu bar, choose Go, Jump to Definition.
- Press F3 by default for macOS, Windows, or Linux.

If the definition is available, the insertion point switches to that definition, even if that definition is in a separate file.

The Jump to Definition command might be disabled, or might not work as expected, under the following conditions:
• The symbol is a primitive symbol for that language.
• The IDE can't find the definition's location in the active file's project.
• The IDE doesn't have access to the definition's location in the active file's project.

Go to Symbol

You can go to a specific symbol within a project, as follows.

1. Make one of the files in the project active by opening it in the editor. If the file is already open, choose its tab in the editor to make that file the active one.
2. Run the Go to Symbol command. For example:
   • Choose the Go window button (magnifying glass icon). In the Go to Anything box, type @, and then start typing the symbol.
   • On the menu bar, choose Go, Go To Symbol. In the Go window, start typing the symbol after @.
   • Press Command-2 or Command-Shift-O by default for macOS, or Ctrl-Shift-O by default for Windows or Linux. In the Go window, start typing the symbol after @.

   For example, to find all symbols in the project named toString, start typing @toString (or start typing toString after @, if @ is already displayed).
3. If you see the symbol you want in the Symbols list, choose it by clicking it. Or use your up arrow or down arrow key to select it, and then press Enter. The insertion point then switches to that symbol.

If the symbol that you want to go to isn't in the active file's project, this procedure might not work as expected.

Create a Language Project

Use the following procedure to create a language project that will work with supported project productivity features in the AWS Cloud9 IDE.

Note
We recommend that you use supported project productivity features on files that are part of a language project. Although you can use some supported project productivity features on a file that isn't part of a project, those features might behave with unexpected results. For example, you might use the IDE to search for references and definitions from within a file at the root level of an environment that isn't part of a project. The IDE might then search only across files at that same root level. This might result in no references or definitions found, even though those references or definitions actually exist in language projects elsewhere across the same environment.

Create a TypeScript Language Project

1. Ensure you have TypeScript installed in the environment. For more information, see Step 1: Install Required Tools (p. 453) in the TypeScript Sample for AWS Cloud9 (p. 452).
2. From a terminal session in the IDE for the environment, switch to the directory where you want to create the project. If the directory doesn't exist, create it and then switch to it. For example, the following commands create a directory named my-demo-project at the root of the environment (in ~/environment), and then switch to that directory.

```bash
mkdir ~/environment/my-demo-project
cd ~/environment/my-demo-project
```
3. At the root of the directory where you want to create the project, run the TypeScript compiler with the `--init` option.

```
tsc --init
```

If this command is successful, the TypeScript compiler creates a `tsconfig.json` file in the root of the directory for the project. You can use this file to define various project settings, such as TypeScript compiler options and specific files to include or exclude from the project.

For more information about the `tsconfig.json` file, see the following:
- `tsconfig.json Overview` on the TypeScript website.
- `tsconfig.json Schema` on the json.schemastore.org website.

---

**Working with Custom Environment Variables in the AWS Cloud9 Integrated Development Environment (IDE)**

The AWS Cloud9 IDE supports getting and setting custom environment variables. You can get and set custom environment variables in the AWS Cloud9 IDE in the following ways.

- Set Command-Level Custom Environment Variables (p. 157)
- Set Custom User Environment Variables in `~/.bash_profile` (p. 158)
- Set Local Custom Environment Variables (p. 158)
- Set Custom User Environment Variables in `~/.bashrc` (p. 158)
- Set Custom Environment Variables in the ENV List (p. 158)

**Set Command-Level Custom Environment Variables**

You can set command-level custom environment variables as you run a command in your AWS Cloud9 development environment. To test this behavior, create a file named `script.sh` with the following code:

```bash
#!/bin/bash
echo $MY_ENV_VAR
```

If you run the following command, the terminal displays `Terminal session`:

```
MY_ENV_VAR='Terminal session' sh ./script.sh
```

If you set the custom environment variable by using multiple approaches described in this topic, then when you try to get the custom environment variable's value, this setting takes priority over all of the others.
Set Custom User Environment Variables in ~/.bash_profile

You can set custom user environment variables in the ~/.bash_profile file in your environment. To test this behavior, add the following code to the ~/.bash_profile file in your environment:

```bash
export MY_ENV_VAR='~/.bash_profile file'
```

If you then run `sh ./script.sh` from the command line, the terminal displays `~/.bash_profile file`. (This assumes you created the `script.sh` file as described earlier.)

Set Local Custom Environment Variables

You can set local custom environment variables in a terminal session by running the `export` command. To test this behavior, run the following command in a terminal session:

```bash
export MY_ENV_VAR='Command line export'
```

If you then run `sh ./script.sh` from the command line, the terminal displays `Command line export`. (This assumes you created the `script.sh` file as described earlier.)

If you set the same custom environment variable with the `export` command and in your ~/.bash_profile file, then when you try to get the custom environment variable's value, the `export` command setting takes priority.

Set Custom User Environment Variables in ~/.bashrc

You can set custom user environment variables in the ~/.bashrc file in your environment. To test this behavior, add the following code to the ~/.bashrc file in your environment:

```bash
export MY_ENV_VAR='~/.bashrc file'
```

If you then run `sh ./script.sh` from the command line, the terminal displays `~/.bashrc file`. (This assumes you created the `script.sh` file as described earlier.)

If you set the same custom environment variable with the `export` command and in your ~/.bashrc file, then when you try to get the custom environment variable's value, the `export` command setting takes priority.

Set Custom Environment Variables in the ENV List

You can set custom environment variables in the ENV list on the Run tab.

To test this behavior, do the following:

1. On the menu bar, choose Run, Run Configurations, New Run Configuration.
3. Choose ENV, and then type `MY_ENV_VAR` for Name and ENV list for Value.
4. For Command, type `./script.sh`.
5. Choose the Run button, the runner tab displays ENV list. (This assumes you created the `script.sh` file as described earlier.)
If you set the same custom environment variable in your `~/.bash_profile` file, with the `export` command, in your `~/.bashrc` file, and in the `ENV` list, then when you try to get the custom environment variable’s value, the `~/.bash_profile` file setting takes first priority, followed by the `export` command setting, the `~/.bashrc` file setting, and the `ENV` list setting.

**Note**
The `ENV` list is the only approach for getting and setting custom environment variables by using code, separate from a shell script.

---

**Working with Project Settings in the AWS Cloud9 Integrated Development Environment (IDE)**

*Project settings*, which apply only to the current AWS Cloud9 development environment, include the following kinds of settings:

- Code editor behaviors, such as whether to use soft tabs and new file line ending behavior
- File types to ignore
- The types of hints and warnings to display or suppress
- Code and formatting behaviors for programming languages such as JavaScript, PHP, Python, and Go
- The types of configurations to use when running and building code

Although project settings apply to only a single environment, you can apply the project settings for one environment to any other environment.

- View or Change Project Settings (p. 159)
- Apply the Current Project Settings for an Environment to Another Environment (p. 159)
- Project Setting Changes You Can Make (p. 160)

**View or Change Project Settings**

1. On the menu bar, choose AWS Cloud9, Preferences.
2. To view the project settings for the current environment, on the Preferences tab, in the side navigation pane, choose Project Settings.
3. To change the current project settings for the environment, change the settings you want in the Project Settings pane.

See Project Setting Changes You Can Make (p. 160).

**Apply the Current Project Settings for an Environment to Another Environment**

1. In both the source and target environment, on the menu bar of the AWS Cloud9 IDE, choose AWS Cloud9, Open Your Project Settings.
2. In the source environment, copy the contents of the `project.settings` tab that is displayed.
3. In the target environment, overwrite the contents of the `project.settings` tab with the copied contents from the source environment.
4. In the target environment, save the `project.settings` tab.

**Project Setting Changes You Can Make**

These sections describe the kinds of project settings that you can change on the Preferences tab's Project Settings pane.

- EC2 Instance (p. 160)
- Code Editor (Ace) (p. 160)
- Find in Files (p. 161)
- Hints & Warnings (p. 161)
- JavaScript Support (p. 161)
- Build (p. 162)
- Run & Debug (p. 162)
- Run Configurations (p. 162)
- Code Formatters (p. 162)
- TypeScript Support (p. 164)
- PHP Support (p. 164)
- Python Support (p. 164)
- Go Support (p. 165)

**EC2 Instance**

*Stop my environment*

If the environment is an EC2 environment, after all web browser instances that are connected to the IDE for the environment are closed, the amount of time until AWS Cloud9 shuts down the Amazon EC2 instance for the environment.

**Code Editor (Ace)**

*Soft Tabs*

If selected, inserts the specified number of spaces instead of a tab character each time you press Tab.

*Autodetect Tab Size on Load*

If selected, AWS Cloud9 attempts to guess the tab size.

*New File Line Endings*

The type of line endings to use for new files.

Valid options include:
- Windows (CRLF) to end lines with a carriage return and then a line feed.
- Unix (LF) to end lines with just a line feed.

*On Save, Strip Whitespace*

If selected, AWS Cloud9 attempts to remove what it considers to be unnecessary spaces and tabs from a file each time that file is saved.
Find in Files

Ignore these Files

When finding in files, the types of files that AWS Cloud9 will ignore.

**Maximum number of files to search (in 1000)**

When finding in files, the maximum number of files, in multiples of 1,000, that AWS Cloud9 will find in the current scope.

Hints & Warnings

**Warning Level**

The minimum level of messages to enable.

Valid values include:

- **Info** to enable informational, warning, and error messages.
- **Warning** to enable just warning and error messages.
- **Error** to enable just error messages.

**Mark Missing Optional Semicolons**

If enabled, AWS Cloud9 flags in a file each time it notices a semicolon that could be used in code, but that isn’t used.

**Mark Undeclared Variables**

If enabled, AWS Cloud9 flags in a file each time it notices an undeclared variable in code.

**Mark Unused Function Arguments**

If enabled, AWS Cloud9 flags in a file each time it notices an unused argument in a function.

**Ignore Messages Matching Regex**

AWS Cloud9 will not display any messages matching the specified regular expression. For more information, see Writing a regular expression pattern in the JavaScript Regular Expressions topic on the Mozilla Developer Network.

JavaScript Support

**Customize JavaScript Warnings With .eslintrc**

If enabled, AWS Cloud9 uses an .eslintrc file to determine which JavaScript warnings to enable or disable. For more information, see Configuration File Formats on the ESLint website.

**JavaScript Library Code Completion**

The JavaScript libraries AWS Cloud9 uses to attempt to suggest or do automatic code completion.

**Format Code on Save**

If enabled, AWS Cloud9 attempts to format the code in a JavaScript file every time that file is saved.

**Use Builtin JSBeautify as Code Formatter**

If enabled, AWS Cloud9 uses its internal implementation of JSBeautify to attempt to increase the readability of code in files.
Custom Code Formatter

The command for AWS Cloud9 to attempt to run when formatting code in a JavaScript file.

Build

Builder Path in environment

The path to any custom build configurations.

Run & Debug

Runner Path in Environment

The path to any custom run configurations.

Preview URL

The URL to use to preview applications for the environment.

Run Configurations

The custom run configurations for this environment.

Remove Selected Configs

Deletes the selected run configurations.

Add New Config

Creates a new run configuration.

Set As Default

Sets the selected run configuration as the default run configuration.

Code Formatters

JSBeautify settings

Settings for increasing the readability of code in files.

Format Code on Save

If enabled, AWS Cloud9 attempts to apply JSBeautify settings whenever code files are saved.

Use JSBeautify for JavaScript

If enabled, AWS Cloud9 attempts to apply JSBeautify settings whenever JavaScript files are saved.

Preserve Empty Lines

If enabled, AWS Cloud9 does not remove empty lines in code files.

Keep Array Indentation

If enabled, AWS Cloud9 preserves the indentation of element declarations in arrays in code files.
JSLint Strict Whitespace

If enabled, AWS Cloud9 attempts to apply JSLint whitespace rules in code files. For more information, see "Whitespace" in JSLint Help.

Braces

Specifies the alignment of braces in code.

Valid values include:

- **Braces with control statement** to move each beginning and end brace to align with its related control statement, as needed.

  For example, this code:

  ```javascript
  for (var i = 0; i < 10; i++) { if (i == 5) { console.log("Halfway done.") }}
  ```

  Turns into this code when the file is saved:

  ```javascript
  for (var i = 0; i < 10; i++) {
    if (i == 5) {
      console.log("Halfway done.")
    }
  }
  ```

- **Braces on own line** to move each brace to its own line, as needed.

  For example, this code:

  ```javascript
  for (var i = 0; i < 10; i++) { if (i == 5) { console.log("Halfway done.") }}
  ```

  Turns into this code when the file is saved:

  ```javascript
  for (var i = 0; i < 10; i++) {if (i == 5)
  {
    console.log("Halfway done.")
  }
  }
  ```

- **End braces on own line** to move each end brace to its own line, as needed.

  For example, this code:

  ```javascript
  for (var i = 0; i < 10; i++) {
    if (i == 5) { console.log("Halfway done.") }
  }
  ```

  Turns into this code when the file is saved:

  ```javascript
  for (var i = 0; i < 10; i++) {
    if (i == 5) {
      console.log("Halfway done.")
    }
  }
  ```

Preserve Inline Blocks

If enabled, AWS Cloud9 does not attempt to move the beginning and ending braces for inline blocks to separate lines, if those braces are on the same line.
Space Before Conditionals

If enabled, AWS Cloud9 adds a space before each conditional declaration, as needed.

Unescape Strings

If enabled, AWS Cloud9 converts escaped strings to their unescaped equivalents. For example, converts \n to a newline character and converts \r to a carriage return character.

Indent Inner Html

If enabled, AWS Cloud9 indents <head> and <body> sections in HTML code.

TypeScript Support

Format Code on Save

If enabled, AWS Cloud9 attempts to format TypeScript code whenever TypeScript files are saved.

Custom Code Formatter

The path to any custom code formatting configuration for TypeScript code.

PHP Support

Enable PHP code Completion

If enabled, AWS Cloud9 attempts to complete PHP code.

PHP Completion Include Paths

Locations that AWS Cloud9 uses to attempt to help complete PHP code. For example, if you have custom PHP files that you want AWS Cloud9 to use for completion, and those files are somewhere in the ~/environment directory, add ~/environment to this path.

Format Code on Save

If enabled, AWS Cloud9 attempts to format PHP code whenever PHP files are saved.

Custom Code Formatter

The path to any custom code formatting configuration for PHP code.

Python Support

Enable Python code completion

If enabled, AWS Cloud9 attempts to complete Python code. To set the paths for AWS Cloud9 to use to complete Python code, use the PYTHONPATH setting.

Python Version

Specifies the version of Python to use.

Pylint command-line options

Options for AWS Cloud9 to use for Pylint with Python code. For more information, see the Pylint User Manual on the Pylint website.

PYTHONPATH

The paths to Python libraries and packages for AWS Cloud9 to use. For example, if you have custom Python libraries and packages in the ~/environment directory, add ~/environment to this path.
Format Code on Save

If enabled, AWS Cloud9 attempts to format Python code whenever Python files are saved.

Custom Code Formatter

The path to any custom code formatting configuration for Python code.

Go Support

Enable Go code completion

If enabled, AWS Cloud9 attempts to complete Go code.

Format Code on Save

If enabled, AWS Cloud9 attempts to format Go code whenever Go files are saved.

Custom Code Formatter

The path to any custom code formatting configuration for Go code.

Working with User Settings in the AWS Cloud9 IDE

User settings are settings that apply across each AWS Cloud9 development environment associated with your AWS Identity and Access Management (IAM) user. They include the following kinds of settings:

- General user interface behaviors, such as enabling animations and marking changed tabs
- File system navigation behaviors
- File find and search behaviors
- Color schemes for terminal sessions and output
- Additional code editor behaviors, such as font sizes, code folding, full line selection, scrolling animations, and font sizes

As you change your user settings, AWS Cloud9 pushes those changes to the cloud and associates them with your IAM user. AWS Cloud9 also continually scans the cloud for changes to user settings associated with your IAM user, and applies those settings to your current environment. This behavior enables you to experience the same look and feel no matter what AWS Cloud9 environment you're working in.

Note

To store and retrieve your IDE settings, AWS Cloud9 uses the internal APIs GetUserSettings and UpdateUserSettings.

You can share your user settings with other users, as follows:

- View or Change Your User Settings (p. 165)
- Share Your User Settings with Another User (p. 166)
- User Setting Changes You Can Make (p. 166)

View or Change Your User Settings

1. On the menu bar, choose AWS Cloud9, Preferences.
2. To view your user settings across each of your environments, on the Preferences tab, in the side navigation pane, choose User Settings.
3. In the User Settings pane, change your user settings across each of your environments.
4. To apply your changes to any other of your environments, simply open that environment. If that environment is already open, refresh the web browser tab for that environment.

For more information, see User Setting Changes You Can Make (p. 166).

Share Your User Settings with Another User

1. In both the source and target environment, on the menu bar of the AWS Cloud9 IDE, choose AWS Cloud9, Open Your User Settings.
2. In the source environment, copy the contents of the user.settings tab that is displayed.
3. In the target environment, overwrite the contents of the user.settings tab with the copied contents from the source environment.
4. In the target environment, save the user.settings tab.

User Setting Changes You Can Make

These sections describe the kinds of user settings you can change in the User Settings pane on the Preferences tab:

- General (p. 166)
- User Interface (p. 167)
- Collaboration (p. 167)
- Tree and Go Panel (p. 167)
- Find in Files (p. 168)
- Meta Data (p. 168)
- Watchers (p. 168)
- Terminal (p. 168)
- Output (p. 169)
- Code Editor (Ace) (p. 169)
- Input (p. 171)
- Hints & Warnings (p. 171)
- Run & Debug (p. 172)
- Preview (p. 172)
- Build (p. 172)

General

Reset to Factory Settings

If you choose the Reset to Default button, AWS Cloud9 resets all of your user settings to the AWS Cloud9 default user settings. To confirm, choose Reset settings.

Warning
You can't undo this action.

Warn Before Exiting

Whenever you attempt to close the IDE, AWS Cloud9 asks you to confirm that you want to exit.
User Interface

Enable UI Animations

AWS Cloud9 uses animations in the IDE.

Use an Asterisk (*) to Mark Changed Tabs

AWS Cloud9 adds an asterisk (*) to tabs that have changes, but for which the contents have not yet been saved.

Display Title of Active Tab as Browser Title

AWS Cloud9 changes the title of the associated web browser tab to the title of the active tab (for example, Untitled1, hello.js, Terminal, Preferences, and so on).

Automatically Close Empty Panes

Whenever you reload an environment, AWS Cloud9 automatically closes any panes it considers are empty.

Environment Files Icon and Selection Style

The icon AWS Cloud9 uses for environment files, and the file selection behaviors AWS Cloud9 uses.

Valid values include:

- Default – AWS Cloud9 uses default icons and default file selection behaviors.
- Alternative – AWS Cloud9 uses alternative icons and alternative file selection behaviors.

Collaboration

Show Notification Bubbles

AWS Cloud9 displays notifications if the environment is a shared environment and multiple users are actively collaborating in that shared environment.

Disable collaboration security warning

When a read/write member is added to an environment, AWS Cloud9 does not display the security warning dialog box.

Show Authorship Info

AWS Cloud9 underlines text entered by other environment members with related highlights in the gutter.

Tree and Go Panel

Scope Go to Anything to Favorites

Go to File in the Go window displays results scoped only to Favorites in the Environment window.

Enable Preview on Go to Anything

Go to File in the Go window displays matching file contents as you type.

Enable Preview on Tree Selection

AWS Cloud9 displays the chosen file with a single mouse click instead of a double mouse click.

Hidden File Pattern

The types of files for AWS Cloud9 to treat as hidden.
Reveal Active File in Project Tree

AWS Cloud9 highlights the active file in the Environment window.

Download Files As

The behavior for AWS Cloud9 to use when downloading files.

Valid values include:

- **auto** – AWS Cloud9 downloads files without modification.
- **tar.gz** – AWS Cloud9 downloads files as compressed TAR files.
- **auto** – AWS Cloud9 downloads files as .zip files.

Find in Files

Search In This Path When 'Project' Is Selected

On the find in files bar, when Project is selected for the search scope, the path to search in.

Show Full Path in Results

Displays the full path to each matching file in the Search Results tab.

Clear Results Before Each Search

Clears the Search Results tab of the results of any previous searches before the current search begins.

Scroll Down as Search Results Come In

Scrolls the Search Results tab to the bottom of the list of results as search results are identified.

Open Files when Navigating Results with (Up and Down)

As the up and down arrow keys are pressed in the Search Results tab within the list of results, opens each matching file.

Meta Data

Maximum of Undo Stack Items in Meta Data

The maximum number of items that AWS Cloud9 keeps in its list of actions that can be undone.

Watchers

Auto-Merge Files When a Conflict Occurs

AWS Cloud9 attempts to automatically merge files whenever a merge conflict happens.

Terminal

Text Color

The color of text in Terminal tabs.

Background Color

The background color in Terminal tabs.
Selection Color

The color of selected text in Terminal tabs.

Font Family

The text font style in Terminal tabs.

Font Size

The size of text in Terminal tabs.

Antialiased Fonts

AWS Cloud9 attempts to smooth the display of text in Terminal tabs.

Blinking Cursor

AWS Cloud9 continuously blinks the cursor in Terminal tabs.

Scrollback

The number of lines that you can scroll up or back through in Terminal tabs.

Use AWS Cloud9 as the Default Editor

Uses AWS Cloud9 as the default text editor.

Output

Text Color

The color of text in tabs that display output.

Background Color

The background color of text in tabs that display output.

Selection Color

The color of selected text in tabs that display output.

Warn Before Closing Unnamed Configuration

AWS Cloud9 prompts you to save any unsaved configuration tab before it is closed.

Preserve log between runs

AWS Cloud9 keeps a log of all attempted runs.

Code Editor (Ace)

Auto-pair Brackets, Quotes, etc.

AWS Cloud9 attempts to add a matching closing character for each related starting character that is typed in editor tabs, such as for brackets, quotation marks, and braces.

Wrap Selection with Brackets, Quote, etc.

AWS Cloud9 attempts to insert a matching closing character at the end of text in editor tabs after the text is selected and a related started character is typed, such as for brackets, quotation marks, and braces.

Code Folding

AWS Cloud9 attempts to show, expand, hide, or collapse sections of code in editor tabs according to related code syntax rules.
Fade Fold Widgets

AWS Cloud9 displays code folding controls in the gutter whenever you pause the mouse over those controls in editor tabs.

Full Line Selection

AWS Cloud9 selects an entire line that is triple-clicked in editor tabs.

Highlight Active Line

AWS Cloud9 highlights the entire active line in editor tabs.

Highlight Gutter Line

AWS Cloud9 highlights the location in the gutter next to the active line in editor tabs.

Show Invisible Characters

AWS Cloud9 displays what it considers to be invisible characters in editor tabs, for example, carriage returns and line feeds, spaces, and tabs.

Show Gutter

AWS Cloud9 displays the gutter.

Show Line Numbers

The behavior for displaying line numbers in the gutter.

Valid values include:
- **Normal** – Display line numbers.
- **Relative** – Display line numbers relative to the active line.
- **None** – Hide line numbers.

Show Indent Guides

AWS Cloud9 displays guides to more easily visualize indented text in editor tabs.

Highlight Selected Word

AWS Cloud9 selects an entire word that is double-clicked in an editor tab.

Scroll Past the End of the Document

The behavior for allowing the user to scroll past the end of the current file in editor tabs.

Valid values include:
- **Off** – Do not allow any scrolling past the end of the current file.
- **Half Editor Height** – Allow scrolling past the end of the current file to up to half the editor's screen height.
- **Full Editor Height** – Allow scrolling past the end of the current file to up to the editor's full screen height.

Animate Scrolling

AWS Cloud9 applies animation behaviors during scrolling actions in editor tabs.

Font Family

The style of font to use in editor tabs.

Font Size

The size of the font to use in editor tabs.
Antialiased Fonts
AWS Cloud9 attempts to smooth the display of text in editor tabs.

Show Print Margin
Displays a vertical line in editor tabs after the specified character location.

Mouse Scroll Speed
The relative speed of mouse scrolling in editor tabs. Larger values result in faster scrolling.

Cursor Style
The style and behavior of the cursor in editor tabs.
Valid values include:
- **Ace** – Display the cursor as a vertical bar that is relatively wider than **Slim**.
- **Slim** – Display the cursor as a relatively slim vertical bar.
- **Smooth** – Display the cursor as a vertical bar that is relatively wider than **Slim** and that blinks more smoothly than **Slim**.
- **Smooth and Slim** – Display the cursor as a relatively slim vertical bar that blinks more smoothly than **Slim**.
- **Wide** – Display the cursor as a relatively wide vertical bar.

Merge Undo Deltas
- **Always** – Allow merge conflicts to be reverted.
- **Never** – Never allow merge conflicts to be reverted.
- **Timed** – Allow merge conflicts to be reverted after a specified time period.

Enable Wrapping For New Documents
AWS Cloud9 wraps code in new files.

Input

Complete As You Type
AWS Cloud9 attempts to display possible text completions as you type.

Complete On Enter
AWS Cloud9 attempts to display possible text completions after you press **Enter**.

Highlight Variable Under Cursor
AWS Cloud9 highlights all references in code to the selected variable.

Use Cmd-Click for Jump to Definition
AWS Cloud9 goes to any original definition for code that is clicked while pressing and holding **Command** for Mac or **Ctrl** for Windows.

Hints & Warnings

Enable Hints and Warnings
AWS Cloud9 displays applicable hint and warning messages.
Ignore Messages Matching Regex

AWS Cloud9 does not display any messages matching the specified regular expression. For more information, see Writing a regular expression pattern in the JavaScript Regular Expressions topic on the Mozilla Developer Network.

Run & Debug

Save All Unsaved Tabs Before Running

Before running the associated code, AWS Cloud9 attempts to save all unsaved files with open tabs.

Preview

Preview Running Apps

AWS Cloud9 attempts to display a preview of the output for the code in the active tab whenever the Preview button is chosen.

Default Previewer

The format AWS Cloud9 uses to preview code output.

Valid values include:

- Raw – Attempt to display code output in a plain format.
- Browser – Attempt to display code output in a format that is preferred for web browsers.

When Saving Reload Previewer

The behavior AWS Cloud9 uses for previewing code output whenever a code file is saved.

Valid values include:

- Only on Ctrl-Enter – Attempt to preview code output whenever Ctrl+Enter is pressed for the current code tab.
- Always – Attempt to preview code output whenever a code file is saved.

Build

Automatically Build Supported Files

AWS Cloud9 attempts to automatically build the current code if a build action is triggered and the code is in a supported format.

Working with AWS Project and User Settings in the AWS Cloud9 Integrated Development Environment (IDE)

AWS service settings, located in the AWS Settings pane of the Preferences tab, include the following kinds of settings:

- Which AWS Region to use for the AWS Resources window
• Whether to use AWS managed temporary credentials
• Whether to display the AWS Serverless Application Model (AWS SAM) template editor in plain text or visual mode

To view or change these settings, choose AWS Cloud9, Preferences in the menu bar of an IDE for an environment.

In the following lists, project-level settings apply only to the current AWS Cloud9 development environment, while user-level settings apply across each environment associated with your IAM user. For more information, see Apply the Current Project Settings for an Environment to Another Environment (p. 159) and Share Your User Settings with Another User (p. 166).

• Project-Level Settings (p. 173)
• User-Level Settings (p. 173)

Project-Level Settings

AWS Region

Which AWS Region to use for the Lambda section of the AWS Resources window.

AWS managed temporary credentials

If turned on, uses AWS managed temporary credentials when calling AWS services from the AWS CLI, the aws-shell, or AWS SDK code from an environment. For more information, see AWS Managed Temporary Credentials (p. 536).

User-Level Settings

Use AWS SAM visual editor

If turned on, displays the AWS Serverless Application Model (AWS SAM) template editor in visual mode when using the Lambda section of the AWS Resources window. If turned off, displays the editor in text mode.

Working with Keybindings in the AWS Cloud9 Integrated Development Environment (IDE)

Keybindings define your shortcut key combinations. Keybindings apply across each AWS Cloud9 development environment associated with your IAM user. As you make changes to your keybindings, AWS Cloud9 pushes those changes to the cloud, and associates them with your IAM user. AWS Cloud9 also continually scans the cloud for changes to keybindings associated with your IAM user, and applies those changes to your current environment.

You can share your keybindings with other users.

• View or Change Your Keybindings (p. 174)
• Share Your Keybindings with Another User (p. 174)
• Change Your Keyboard Mode (p. 174)
• Change Your Operating System Keybindings (p. 175)
View or Change Your Keybindings

1. On the menu bar, choose AWS Cloud9, Preferences.
2. To view your keybindings across each environment of yours, on the Preferences tab, in the side navigation pane, choose Keybindings.
3. To change your keybindings across each environment of yours, in the Keybindings pane, change the settings you want.
4. To apply your changes to any environment, simply open that environment. If that environment is already open, refresh the web browser tab for that environment.

For more information, see the following:
- MacOS Default Keybindings Reference (p. 178)
- MacOS Vim Keybindings Reference (p. 189)
- MacOS Emacs Keybindings Reference (p. 199)
- MacOS Sublime Keybindings Reference (p. 210)
- Windows / Linux Default Keybindings Reference (p. 221)
- Windows / Linux Vim Keybindings Reference (p. 232)
- Windows / Linux Emacs Keybindings Reference (p. 242)
- Windows / Linux Sublime Keybindings Reference (p. 253)

Share Your Keybindings with Another User

1. In both the source and target environment, on the menu bar of the AWS Cloud9 IDE, choose AWS Cloud9, Open Your Keymap.
2. In the source environment, copy the contents of the keybindings.settings tab that is displayed.
3. In the target environment, overwrite the contents of the keybindings.settings tab with the copied contents from the source environment.
4. In the target environment, save the keybindings.settings tab.

Change Your Keyboard Mode

You can change the keyboard mode that the AWS Cloud9 IDE uses for interacting with text in the editor across each environment associated with your IAM user.

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, in the side navigation pane, choose Keybindings.
3. For Keyboard Mode, choose one of these keyboard modes:
   - Default to use a set of default keybindings.
   - Vim to use Vim mode. For more information, see the Vim help files website.
   - Emacs to use Emacs mode. For more information, see The Emacs Editor on the GNU Operating System website.
   - Sublime to use Sublime mode. For more information, see the Sublime Text Documentation website.
Change Your Operating System Keybindings

You can change the set of operating system keybindings the AWS Cloud9 IDE recognizes across each environment associated with your IAM user.

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, in the side navigation pane, choose Keybindings.
3. For Operating System, choose one of these operating systems:
   - Auto for the AWS Cloud9 IDE to attempt to detect which set of operating system keybindings to use.
   - MacOS for the AWS Cloud9 IDE to use the keybindings listed in Mac format.
   - Windows / Linux for the AWS Cloud9 IDE to use the keybindings listed in Windows and Linux formats.

Change Specific Keybindings

You can change individual keybindings across each environment associated with your IAM user.

To change one keybinding at a time

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, in the side navigation pane, choose Keybindings.
3. In the list of keybindings, double-click the keybinding in the Keystroke column you want to change.
4. Use the keyboard to specify the replacement key combination, and then press Enter.

   **Note**
   To completely remove the current key combination, press Backspace for Windows or Linux, or Delete for Mac.

To change multiple keybindings at once

1. On the menu bar, choose AWS Cloud9, Open Your Keymap.
2. In the keybindings.settings file, define each keybinding to be changed, for example:

   ```json
   [
   {
     "command": "addfavorite",
     "keys": {
       "win": ["Ctrl-Alt-F"],
       "mac": ["Ctrl-Option-F"]
     }
   },
   {
     "command": "copyFilePath",
     "keys": {
       "win": ["Ctrl-Shift-F"],
       "mac": ["Alt-Shift-F"]
     }
   }
   ]
   ``

   In the example, addFavorite and copyFilePath are the names of keybindings in the Keystroke column in the Keybindings pane on the Preferences tab. The keybindings you want are win and mac for Windows or Linux and Mac, respectively.
Remove All of Your Custom Keybindings

You can remove all custom keybindings and restore all keybindings to their default values, across each environment associated with your IAM user.

Warning
You cannot undo this action.

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, in the side navigation pane, choose Keybindings.
3. Choose Reset to Defaults.

Working with Themes in the AWS Cloud9 Integrated Development Environment (IDE)

A theme defines your overall IDE colors. This applies across each AWS Cloud9 development environment associated with your IAM user. As you make changes to your theme, AWS Cloud9 pushes those changes to the cloud, and associates them with your IAM user. AWS Cloud9 also continually scans the cloud for changes to the theme associated with your IAM user, and applies those changes to your current environment.

You can share any custom theme overrides you define with other users.

- View or Change Your Theme (p. 176)
- Overall Theme Settings You Can Change (p. 176)
- Theme Overrides You Can Define with Code (p. 177)
- Share Your Theme Overrides with Another User (p. 177)

View or Change Your Theme

1. On the menu bar, choose AWS Cloud9, Preferences.
2. To view your theme across each environment of yours, on the Preferences tab, in the side navigation pane, choose Themes.
3. To change your theme across each environment of yours, in the Themes pane, change the settings you want. To change portions of your theme by using code, choose the your stylesheet link.
4. To apply your changes to any environment of yours, simply open that environment. If that environment is already open, refresh the web browser tab for that environment.

Overall Theme Settings You Can Change

You can change the following kinds of overall theme settings on the Preferences tab in the Themes pane.

Flat Theme
Applies the built-in flat theme across the AWS Cloud9 IDE.
Classic Theme

Applies the selected built-in classic theme across the AWS Cloud9 IDE.

Syntax Theme

Applies the selected theme to code files across the AWS Cloud9 IDE.

Theme Overrides You Can Define with Code

You can override portions of the overall theme in the AWS Cloud9 IDE. These overrides will persist even if you change the overall theme itself in the AWS Cloud9 IDE.

For example, let's say you want to change the background color of the titles on open tabs to yellow, regardless of the related setting for the current overall theme that is currently applied to the AWS Cloud9 IDE.

First, use your web browser's developer tools to determine the CSS class for the portion of the theme you want to change. For example, do the following for Google Chrome.

1. Choose Customize and control Google Chrome, More tools, Developer tools.
2. In the Developer tools pane, choose Select an element in the page to inspect it.
3. Pause your mouse over the portion of the IDE you want to change. In this example, pause your mouse over the title of an open tab.
4. Note the CSS class name. In this example, the CSS class name for the title of an open tab is sessiontab_title.

Next, add a corresponding CSS class selector to your styles.css file.

1. On the menu bar, choose AWS Cloud9, Preferences. In the side navigation pane, choose Themes. Then choose the your stylesheet link.
2. In the styles.css file, add the CSS class selector. In this example, you use the .sessiontab_title selector to set background-color to yellow.

```
.sessiontab_title {
  background-color: yellow;
}
```

Finally, save the styles.css file, and note the change to the theme. In this example, the background color of the titles of open tabs changes to yellow. Even if you change the overall theme in the AWS Cloud9 IDE, the CSS overrides in your styles.css file persist.

**Note**

To revert this theme override, remove the preceding code from the styles.css file, and then save the file again.

Share Your Theme Overrides with Another User

1. In both the source and target environment, on the menu bar of the AWS Cloud9 IDE, choose AWS Cloud9, Open Your Stylesheet.
2. In the source environment, copy the contents of the styles.css tab that is displayed.
3. In the target environment, overwrite the contents of the styles.css tab with the copied contents from the source environment.
4. In the target environment, save the styles.css tab.
Working with Initialization Scripts in the AWS Cloud9 Integrated Development Environment (IDE)

Warning
Adding code to an initialization script in the IDE is currently an experimental feature and is not fully supported. If you add code to an initialization script, you do so at your own risk. AWS Cloud9 reserves the right to change this functionality at any time.

An initialization script defines custom initialization code to run in your IDE after all plugins are loaded. This applies across each AWS Cloud9 development environment associated with your IAM user. As you make changes to your initialization script, AWS Cloud9 pushes those changes to the cloud and associates them with your IAM user. AWS Cloud9 also continually scans the cloud for changes to the initialization script associated with your IAM user, and applies those changes to your current environment.

You can share your initialization script with other users.

- View or Change Your Initialization Script (p. 178)
- Share Your Initialization Script with Another User (p. 178)

View or Change Your Initialization Script

1. To view your initialization script, on the menu bar, choose AWS Cloud9, Open Your Init Script.
2. To change your initialization script, on the init.js tab, use code to change your initialization script's behavior.
3. To apply your changes to any other environment, simply open the environment you want to apply the changes to. If that environment is already open, refresh the web browser tab for that environment.

Share Your Initialization Script with Another User

1. In both the source and target environment, on the menu bar of the AWS Cloud9 IDE, choose AWS Cloud9, Open Your Init Script.
2. In the source environment, copy the contents of the init.js tab that is displayed.
3. In the target environment, overwrite the contents of the init.js tab with the copied contents from the source environment.
4. In the target environment, save the init.js tab.

MacOS Default Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of default keyboard mode keybindings for MacOS operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the **Preferences** tab, choose **Keybindings**.
3. For **Keyboard Mode**, choose **Default**.
4. For **Operating System**, choose **MacOS**.

See also Working with Keybindings (p. 173).

- General (p. 179)
- Tabs (p. 181)
- Panels (p. 182)
- Code Editor (p. 183)
- emmet (p. 188)
- Terminal (p. 188)
- Run and Debug (p. 188)

### General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Command-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Control-Space</td>
<td>Option-Space</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Control-Shift-Space</td>
<td>Option-Shift-Space</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Command-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Command-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Command-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Control-Option-G</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Command-G</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Command-Shift-G</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>---------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Open the <strong>Environment</strong> window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Command-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the <strong>go to line</strong> box</td>
<td>Command-L</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the <strong>Lambda</strong> section of the <strong>AWS Resources</strong> window, attempts to upload the function to Lambda as a remote function</td>
<td>Command-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Control-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Command-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a <strong>Terminal</strong> tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Command-Option-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Command-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Command-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Command-Shift-Z</td>
<td>redo</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Command-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Option-Command-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Option-Command-F</td>
<td>replace</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Command-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Command-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Control-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Option-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>--------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Command-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Command-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Shift-Command-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Command-Option-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Command-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

**Tabs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Option-Control-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Option-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Command-Control-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Option-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Control-Command-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Control-Command-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Control-Command-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Control-Command-Up</td>
<td>gottopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Command-[</td>
<td>gototableft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Command-]</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Command-Option-Shift-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Command-Option-Shift-Left</td>
<td>movetablet</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Command-Option-Shift-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very top, create a split tab there</td>
<td>Command-Option-Shift-Up</td>
<td>movetabup</td>
</tr>
</tbody>
</table>
## Description | Keybinding | Command
---|---|---
Go to the next pane | Option-Esc | nextpane
Go to the next tab | Option-Tab | nexttab
Go to the previous pane | Option-Shift-Esc | previouspane
Go to the previous tab | Option-Shift-Tab | previoustab
Go back to the last tab | Esc | refocusTab
Open the last tab again | Option-Shift-T | reopenLastTab
Show the current tab in the file tree | Command-Shift-L | revealtab
Go to the tenth tab | Command-0 | tab0
Go to the first tab | Command-1 | tab1
Go to the second tab | Command-2 | tab2
Go to the third tab | Command-3 | tab3
Go to the fourth tab | Command-4 | tab4
Go to the fifth tab | Command-5 | tab5
Go to the sixth tab | Command-6 | tab6
Go to the seventh tab | Command-7 | tab7
Go to the eighth tab | Command-8 | tab8
Go to the ninth tab | Command | tab9

### Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
</table>
Show the Go window in **Go to Anything** mode | Command-E|Command-P | gotoanything |
Show the Go window in **Go to Command** mode | Command- | gotocommand |
Show the Go window in **Go to File** mode. | Command-O | gotofile |
Show the Go window in **Go to Symbol** mode. | Command-Shift-O | gotosymbol |
Show the **Outline** window | Command-Shift-E | outline |
Show the **Console** window if hidden, or hide if shown | Control-Esc | toggleconsole |
<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the <strong>Environment</strong> window if hidden, or hide if shown</td>
<td>Command-U</td>
<td>toggletree</td>
</tr>
</tbody>
</table>

## Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already added, add another cursor above that one</td>
<td>Control-Option-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second cursor is already added, move the second cursor up one line</td>
<td>Control-Option-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already added, add another cursor below that one</td>
<td>Control-Option-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second cursor is already added, move the second cursor down one line</td>
<td>Control-Option-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned</td>
<td>Control-Option-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Control-Backspace</td>
<td>Backspace</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Control-]</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Control-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in the IDE</td>
<td>Command-\textbackslash-Z</td>
<td>Command-\textbackslash-Shift-Z</td>
</tr>
<tr>
<td>Center the selection</td>
<td>Control-L</td>
<td>centerselection</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
<td>Command-Option-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Command-Option-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>Control-Delete</td>
</tr>
<tr>
<td></td>
<td>Shift-Delete</td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied contents</td>
<td>Command-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>immediately after the selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Command-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to next matching symbol in the selection</td>
<td>Control-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code, or if a folded unit is selected, unfold it</td>
<td>Command-Option-L</td>
<td>Command-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements</td>
<td>Control-Command-Option-0</td>
<td>foldall</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection</td>
<td>Command-Option-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>scope</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>Control-N</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>Control-P</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Command-End</td>
<td>Command-Down</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>Control-B</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Command-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Command-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>F4</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>Control-V</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Shift-F4</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>Control-F</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Command-Home</td>
<td>Command-Up</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Option-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Option-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------</td>
<td>-----------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Control-P</td>
<td>jumpptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Command-+</td>
<td>Command-=</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move the selection down one line</td>
<td>Option-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move the selection up one line</td>
<td>Option-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or turn off if on</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Option-Page Down</td>
<td>pagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Option-Page Up</td>
<td>pageup</td>
</tr>
<tr>
<td>Remove the current line</td>
<td>Command-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Control-K</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Command-Backspace</td>
<td>removetolinestart</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Option-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Option-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Command-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Command-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>Control-Shift-N</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>Control-Shift-B</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Control-Option-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Control-Option-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Control-Option-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Control-Option-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Control-G</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Control-Shift-G</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selectright</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Command-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Command-Shift-Right</td>
<td>selectttolineend</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Command-Shift-Left</td>
<td>selecttolinestart</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Control-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Command-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>---------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Option-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Option-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Command-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Command--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Command-Option-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Control-Option-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Control-O</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Command-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Command-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Option-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Command-Option-E</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Control-W</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Control-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Control-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Control-T</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Command-Option-Shift-L</td>
<td>unfold</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Command-Shift-F1</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>
### emmet

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Command-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Control-Option-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Command-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Command-,</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Control-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>

### Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Option-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Option-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

### Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Command-B</td>
<td>build</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>Command-\</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Option-F5</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>Command-;</td>
</tr>
</tbody>
</table>
MacOS Vim Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Vim keyboard mode keybindings for MacOS operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose **AWS Cloud9, Preferences**.
2. On the **Preferences** tab, choose **Keybindings**.
3. For **Keyboard Mode**, choose **Vim**.
4. For **Operating System**, choose **MacOS**.

See also **Working with Keybindings (p. 173)**.

- **General (p. 189)**
- **Tabs (p. 191)**
- **Panels (p. 193)**
- **Code Editor (p. 193)**
- **emmet (p. 198)**
- **Terminal (p. 199)**
- **Run and Debug (p. 199)**

### General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Command-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Control-Space</td>
<td>Option-Space</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>-------------------------------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Control-Shift-Space</td>
<td>completeoverwrite</td>
</tr>
<tr>
<td></td>
<td>Option-Shift-Space</td>
<td></td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Command-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Command-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Command-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Control-Option-G</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Command-G</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Command-Shift-G</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Command-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Command-L</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
<td>Command-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Control-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Command-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a Terminal tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Command-Option-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>---------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Paste the clipboard’s current contents at the cursor</td>
<td>Command-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Command-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Command-Shift-Z</td>
<td>Command-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Command-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Option-Command-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Option-Command-F</td>
<td>replace</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Command-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Command-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Control-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Option-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Command-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Command-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Shift-Command-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Show the Process List dialog box</td>
<td>Command-Option-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Command-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

**Tabs**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Option-Control-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Option-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Command-Control-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Option-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Control-Command-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Control-Command-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Control-Command-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Control-Command-Up</td>
<td>gottopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Command-[</td>
<td>gototableft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Command-]</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Command-Option-Shift-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Command-Option-Shift-Left</td>
<td>movetableft</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Command-Option-Shift-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very top, create a split tab there</td>
<td>Command-Option-Shift-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>Go to the next pane</td>
<td>Option-Esc</td>
<td>nextpane</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Option-Tab</td>
<td>nexttab</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Option-Shift-Esc</td>
<td>previouspane</td>
</tr>
<tr>
<td>Go to the previous tab</td>
<td>Option-Shift-Tab</td>
<td>previousstab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Option-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Command-Shift-L</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Command-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Command-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Command-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Command-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Command-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Command-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Command-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Command-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Command-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Command</td>
<td>tab9</td>
</tr>
</tbody>
</table>

## Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in Go to Anything mode</td>
<td>Command-E</td>
<td>Command-P</td>
</tr>
<tr>
<td>Show the Go window in Go to Command mode</td>
<td>Command-.</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in Go to File mode.</td>
<td>Command-O</td>
<td>gotofile</td>
</tr>
<tr>
<td>Show the Go window in Go to Symbol mode.</td>
<td>Command-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the Outline window</td>
<td>Command-Shift-E</td>
<td>outline</td>
</tr>
<tr>
<td>Show the Console window if hidden, or hide if shown</td>
<td>Control-Esc</td>
<td>toggleconsole</td>
</tr>
<tr>
<td>Show the Environment window if hidden, or hide if shown</td>
<td>Command-U</td>
<td>toggl etree</td>
</tr>
</tbody>
</table>

## Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already added, add another cursor above that one</td>
<td>Control-Option-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second cursor is already added, move the second cursor up one line</td>
<td>Control-Option-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already added, add another cursor below that one</td>
<td>Control-Option-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second cursor is already added, move the second cursor down one line</td>
<td>Control-Option-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned</td>
<td>Control-Option-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Control-Backspace</td>
<td>Shift-Backspace</td>
</tr>
<tr>
<td>Indent selection one tab</td>
<td>Control-]</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent selection one tab</td>
<td>Control-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in the IDE</td>
<td>Command-Z</td>
<td>Command-Shift-Z</td>
</tr>
<tr>
<td>Center the selection</td>
<td>Control-L</td>
<td>centerselection</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
<td>Command-Option-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Command-Option-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>Control-Delete</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied contents immediately after the selection</td>
<td>Command-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Command-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to the next matching symbol in selection</td>
<td>Control-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code, or if a folded unit is selected, unfold it</td>
<td>Command-Option-L</td>
<td>Command-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements</td>
<td>Control-Command-Option-0</td>
<td>foldall</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Command-Option-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>Control-N</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>Control-P</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Command-End</td>
<td>Command-Down</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>---------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>Control-B</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Command-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Command-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>F4</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>Control-V</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Shift-F4</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>Control-F</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Command-Home</td>
<td>Command-Up</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Option-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Option-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Control-P</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Command-+</td>
<td>Command=-</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move selection down one line</td>
<td>Option-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move selection up one line</td>
<td>Option-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or turn off if on</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Option-Page Down</td>
<td>pagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Option-Page Up</td>
<td>pageup</td>
</tr>
<tr>
<td>Remove the current line</td>
<td>Command-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Control-K</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Command-Backspace</td>
<td>removetolinstart</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>--------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Option-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Option-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Command-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Command-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>selectdown</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>selectleft</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Control-Option-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Control-Option-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Control-Option-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Control-Option-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Control-G</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Control-Shift-G</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selecttright</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Command-Shift-End</td>
<td>Command-Shift-Down</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Command-Shift-Right</td>
<td>Shift-End</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Command-Shift-Left</td>
<td>Command-Shift-A</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Control-Shift-P</td>
<td></td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Command-Shift-Home</td>
<td>Command-Shift-Up</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>Control-Shift-P</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Option-Shift-Left</td>
<td></td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Option-Shift-Right</td>
<td></td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Command-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Command--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Command-Option-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Control-Option-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Control-O</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Command-Shift-/</td>
<td></td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Command-/</td>
<td></td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
</tbody>
</table>
### emmet

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Option-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Command-Option-E</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Control-W</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Control-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Control-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Control-T</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Command-Option-Shift-L</td>
<td>unfold</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Command-Option-Shift-0</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>

---

**emmet**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Command-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Control-Option-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Command-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Command-,</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Control-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>
Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Option-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Option-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Command-B</td>
<td>build</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>Command-\</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Option-F5</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>Command-;</td>
</tr>
<tr>
<td>Step out of the current function scope</td>
<td>Shift-F11</td>
<td>Command-Shift-'</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>Command-'</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Control-Shift-C</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

MacOS Emacs Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Emacs keyboard mode keybindings for MacOS operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, choose Keybindings.
3. For Keyboard Mode, choose Emacs.
4. For Operating System, choose MacOS.

See also Working with Keybindings (p. 173).
### General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Command-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Control-Space</td>
<td>Option-Space</td>
</tr>
<tr>
<td>Complete code, and then overwrite</td>
<td>Control-Shift-Space</td>
<td>Option-Shift-Space</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Command-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Command-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Command-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Control-Option-G</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Command-G</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Command-Shift-G</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the <strong>Environment</strong> window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Command-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Show the <em>go to line</em> box</td>
<td>Command-L</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if shown</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
<td>Command-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Control-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Command-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a Terminal tab, then switch to the parent folder of the selected file in the list of files</td>
<td>Command-Option-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Command-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Command-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Command-Shift-Z</td>
<td>Command-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Command-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Option-Command-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Option-Command-F</td>
<td>replace</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Command-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Command-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Control-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Option-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Command-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Command-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Shift-Command-F</td>
<td>searchinfiles</td>
</tr>
</tbody>
</table>
### Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Process List dialog box</td>
<td>Command-Option-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Command-Z</td>
<td>undo</td>
</tr>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Option-Control-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Option-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Command-Control-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Option-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Control-Command-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Control-Command-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Control-Command-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Control-Command-Up</td>
<td>gotottopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Command-[</td>
<td>gototableft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Command-]</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very</td>
<td>Command-Option-Shift-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>bottom, create a split tab there</td>
<td></td>
<td>movetableft</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left,</td>
<td>Command-Option-Shift-Left</td>
<td>movetabright</td>
</tr>
<tr>
<td>create a split tab there</td>
<td></td>
<td>movetabup</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right,</td>
<td>Command-Option-Shift-Right</td>
<td></td>
</tr>
<tr>
<td>create a split tab there</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very</td>
<td>Command-Option-Shift-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>top, create a split tab there</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go to the next pane</td>
<td>Option-Esc</td>
<td>nextpane</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Option-Tab</td>
<td>nexttab</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Option-Shift-Esc</td>
<td>previouspane</td>
</tr>
<tr>
<td>Go to the previous tab</td>
<td>Option-Shift-Tab</td>
<td>previoustab</td>
</tr>
</tbody>
</table>
### Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Option-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Command-Shift-L</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Command-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Command-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Command-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Command-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Command-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Command-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Command-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Command-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Command-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Command</td>
<td>tab9</td>
</tr>
</tbody>
</table>

### Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in Go to Anything mode</td>
<td>Command-E</td>
<td>Command-P</td>
</tr>
<tr>
<td>Show the Go window in Go to Command mode</td>
<td>Command-</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in Go to File mode.</td>
<td>Command-O</td>
<td>gotofile</td>
</tr>
<tr>
<td>Show the Go window in Go to Symbol mode.</td>
<td>Command-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the Outline window</td>
<td>Command-Shift-E</td>
<td>outline</td>
</tr>
<tr>
<td>Show the Console window if hidden, or hide if shown</td>
<td>Control-Esc</td>
<td>toggleconsole</td>
</tr>
<tr>
<td>Show the Environment window if hidden, or hide if shown</td>
<td>Command-U</td>
<td>toggletree</td>
</tr>
</tbody>
</table>
## Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already</td>
<td>Control-Option-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>added, add another cursor above that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second</td>
<td>Control-Option-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added, move the second cursor up one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already</td>
<td>Control-Option-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>added, add another cursor below that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second</td>
<td>Control-Option-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added, move the second cursor down one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their</td>
<td>Control-Option-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>lines, if they are misaligned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Control-Backspace</td>
<td>backspace</td>
</tr>
<tr>
<td></td>
<td>Shift-Backspace</td>
<td>Backspace</td>
</tr>
<tr>
<td>Indent selection one tab</td>
<td>Control-]</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent selection one tab</td>
<td>Control-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in</td>
<td>Command-Z</td>
<td>Command-Shift-Z</td>
</tr>
<tr>
<td>the IDE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Center the selection</td>
<td>Control-L</td>
<td>centerselection</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
<td>Command-Option-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Command-Option-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>Control-Delete</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied</td>
<td>Command-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>contents immediately after the selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Command-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to the next matching symbol in the selection</td>
<td>Control-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
<td>Command-Option-L</td>
<td>Command-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements</td>
<td>Control-Command-Option-0</td>
<td>foldall</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Command-Option-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>Control-N</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>Control-P</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Command-End</td>
<td>Command-Down</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>Control-B</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Command-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Command-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>F4</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>Control-V</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Shift-F4</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>Control-F</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Command-Home</td>
<td>Command-Up</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Option-Left</td>
<td>towordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Option-Right</td>
<td>towordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Control-P</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Command-+</td>
<td>Command-=</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move the selection down one line</td>
<td>Option-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move the selection up one line</td>
<td>Option-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Option-Page Down</td>
<td>pagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Option-Page Up</td>
<td>pageup</td>
</tr>
<tr>
<td>Remove the current line</td>
<td>Command-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Control-K</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Command-Backspace</td>
<td>removetolinestart</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Option-Backspace</td>
<td>Control-Option-Backspace</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Option-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Command-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Command-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>Control-Shift-N</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>Control-Shift-B</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Control-Option-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Keybinding</strong></td>
<td><strong>Command</strong></td>
</tr>
<tr>
<td>-----------------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Control-Option-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Control-Option-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Control-Option-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Control-G</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Control-Shift-G</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selectright</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Command-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td></td>
<td>Command-Shift-Down</td>
<td></td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Command-Shift-Right</td>
<td>selecttolineend</td>
</tr>
<tr>
<td></td>
<td>Shift-End</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Control-Shift-E</td>
<td></td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Command-Shift-Left</td>
<td>selecttolinestart</td>
</tr>
<tr>
<td></td>
<td>Control-Shift-A</td>
<td></td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Control-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Command-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td></td>
<td>Command-Shift-Up</td>
<td></td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td></td>
<td>Control-Shift-Up</td>
<td></td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Option-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Option-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Command-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Command--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Command-Option-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Control-Option-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Control-O</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Command-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Command-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Option-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Command-Option-E</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Control-W</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change selection to all lowercase</td>
<td>Control-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change selection to all uppercase</td>
<td>Control-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose selection</td>
<td>Control-T</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Command-Option-Shift-L</td>
<td>unfold</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Command-Shift-F1</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>
**emmet**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Command-Y</td>
<td><code>emmet_evaluate_math_expression</code></td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Control-Option-E</td>
<td><code>emmet_expand_abbreviation</code></td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td><code>emmet_expand_abbreviation_with_tab</code></td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Command-</td>
<td><code>emmet_select_next_item</code></td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Command-,</td>
<td><code>emmet_select_previous_item</code></td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Control-A</td>
<td><code>emmet_wrap_with_abbreviation</code></td>
</tr>
</tbody>
</table>

**Terminal**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Option-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Option-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

**Run and Debug**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Command-B</td>
<td>build</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>Command-\</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Option-F5</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>Command-;</td>
</tr>
</tbody>
</table>
Description | Keybinding | Command
--- | --- | ---
Step out of the current function scope | Shift-F11 | command-shift-' | stepout
Step over the current expression on the stack | F10 | command-' | stepover
Stop running or debugging the current application | Shift-F5 | stop
Stop building the current file | control-shift-C | stopbuild

### MacOS Sublime Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Sublime keyboard mode keybindings for MacOS operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, choose Keybindings.
3. For Keyboard Mode, choose Sublime.
4. For Operating System, choose MacOS.

See also Working with Keybindings (p. 173).

- General (p. 210)
- Tabs (p. 213)
- Panels (p. 215)
- Code Editor (p. 215)
- emmet (p. 220)
- Terminal (p. 221)
- Run and Debug (p. 221)

### General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>command-shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>control-space</td>
<td>complete</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-------------------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Control-Shift-Space</td>
<td>completeoverwrite</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Command-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Command-X</td>
<td>cut</td>
</tr>
<tr>
<td>Delete from the cursor to start of the line</td>
<td>Command-K Command-Backspace</td>
<td>delete_to_hard_bol</td>
</tr>
<tr>
<td>Delete from the cursor to end of the line</td>
<td>Command-K Command-K</td>
<td>delete_to_hard_eol</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Command-F</td>
<td>find</td>
</tr>
<tr>
<td>Highlight all matches for the selection</td>
<td>Control-Command-G</td>
<td>find_all_under</td>
</tr>
<tr>
<td>Highlight next match for the selection</td>
<td>Option-Command-G</td>
<td>find_under</td>
</tr>
<tr>
<td>Highlight around the cursor and all matches for the highlight</td>
<td>Command-D</td>
<td>find_under_expanded</td>
</tr>
<tr>
<td>Highlight around the cursor and outline all matches for the highlight</td>
<td>Command-K Command-D</td>
<td>find_under_expanded_skip</td>
</tr>
<tr>
<td>Highlight the previous match for the selection</td>
<td>Shift-Option-Command-G</td>
<td>find_under_previous</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Control-Option-G</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Command-G</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Shift-Command-G</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>---------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Control-Option-F</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Control-G</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F12</td>
<td>Command-Option-Down</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the <strong>Lambda</strong> section of the <strong>AWS Resources</strong> window, attempts to upload the function to Lambda as a remote function</td>
<td>Command-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Go to the end of the current word</td>
<td>Option-Right</td>
<td>moveToWordEndRight</td>
</tr>
<tr>
<td>Go to the start of the current word</td>
<td>Option-Left</td>
<td>moveToWordStartLeft</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Control-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Command-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a <strong>Terminal</strong> tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Command-Option-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Command-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Command-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Command-Shift-Z</td>
<td>Command-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Command-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Option-Command-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Command-Option-F</td>
<td>replace</td>
</tr>
<tr>
<td>Replace all find expression matches with replace with expression in the find and replace bar</td>
<td>Control-Option-Enter</td>
<td>replaceall</td>
</tr>
</tbody>
</table>
## Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replace next find expression match with replace with expression in the find and replace bar</td>
<td>Command-Option-E</td>
<td>replacenext</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Command-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Command-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Control-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Option-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Command-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Command-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Command-Shift-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Include from the cursor to the end of the word in the selection</td>
<td>Option-Shift-Right</td>
<td>selectToWordEndRight</td>
</tr>
<tr>
<td>Include from the cursor to the start of the word in the selection</td>
<td>Option-Shift-Left</td>
<td>selectToWordStartLeft</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Command-Option-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Command-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

## Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Option-Control-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Option-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Command-Control-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Option-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Control-Command-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Control-Command-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Control-Command-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Control-Command-Up</td>
<td>gotopaneup</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Command-Shift-[</td>
<td>Command-Option-Left</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Command-Shift-}</td>
<td>Command-Option-Right</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Command-Option-Shift-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Command-Option-Shift-Left</td>
<td>movetableft</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Command-Option-Shift-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very top, create a split tab there</td>
<td>Command-Option-Shift-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Control-Tab</td>
<td>nexttab</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Option-Shift-Esc</td>
<td>previouspane</td>
</tr>
<tr>
<td>Go to the previous tab</td>
<td>Control-Shift-Tab</td>
<td>previousstab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Command-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Command-E</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Command-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Command-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Command-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Command-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Command-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Command-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Command-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Command-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Command-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Command</td>
<td>tab9</td>
</tr>
</tbody>
</table>
## Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in <strong>Go to Anything</strong> mode.</td>
<td>Command-E</td>
<td>Command-P</td>
</tr>
<tr>
<td>Show the Go window in <strong>Go to Command</strong> mode.</td>
<td>Command-.</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in <strong>Go to File</strong> mode.</td>
<td>Command-O</td>
<td>gotofile</td>
</tr>
<tr>
<td>Show the Go window in <strong>Go to Symbol</strong> mode.</td>
<td>Command-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the <strong>Outline</strong> window</td>
<td>Command-Shift-R</td>
<td>outline</td>
</tr>
<tr>
<td>Show the <strong>Console</strong> window if hidden, or hide if shown</td>
<td>Control-`</td>
<td>toggleconsole</td>
</tr>
<tr>
<td>Show the <strong>Environment</strong> window if hidden, or hide if shown</td>
<td>Command-K Command-B</td>
<td>toggletree</td>
</tr>
</tbody>
</table>

## Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already added, add another cursor above that one</td>
<td>Control-Shift-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second cursor is already added, move the second cursor up one line</td>
<td>Control-Option-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already added, add another cursor below that one</td>
<td>Control-Shift-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second cursor is already added, move the second cursor down one line</td>
<td>Control-Option-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned</td>
<td>Control-Option-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>------------------------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Control-Backspace</td>
<td>Backspace</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shift-Backspace</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Control-]</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Control-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in the IDE</td>
<td>Command-Z</td>
<td>Command-Shift-Z</td>
</tr>
<tr>
<td>Center the selection</td>
<td>Command-K Command-C</td>
<td>Control-L</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
<td>Command-Option-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Command-Option-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>Control-Delete</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied contents immediately after the selection</td>
<td>Command-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Command-L</td>
<td>expandoLine</td>
</tr>
<tr>
<td>Include up to the next matching symbol in the selection</td>
<td>Control-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
<td>Command-Option-L</td>
<td>Command-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements</td>
<td>Control-Command-Option-0</td>
<td>foldall</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Command-K Command-1</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>Control-N</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>Control-P</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Command-End</td>
<td>Command-Down</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>Control-B</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Command-Right</td>
<td>End</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>--------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Command-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>Control-F6</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>Control-V</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Control-Shift-F6</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>Control-F</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Command-Home</td>
<td>Command-Up</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Option-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Option-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Combine selected lines into a single line</td>
<td>Command-J</td>
<td>joinlines</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Control-M</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Command-=</td>
<td>Command++</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Option-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move selection down one line</td>
<td>Control-Command-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move selection up one line</td>
<td>Control-Command-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Option-Page Down</td>
<td>pagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Option-Page Up</td>
<td>pageup</td>
</tr>
<tr>
<td>Delete the contents of the current line</td>
<td>Control-Shift-K</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Control-K</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Command-Backspace</td>
<td>removetolinenstart</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Option-Backspace</td>
<td>Control-Option-Backspace</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Option-Delete</td>
<td></td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Control-Shift-Q</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Command-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>Control-Shift-N</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>Control-Shift-B</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td></td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to cursor</td>
<td>Shift-Home</td>
<td></td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Control-Option-Right</td>
<td></td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Control-Option-Left</td>
<td></td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Control-Option-Shift-Right</td>
<td></td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Control-Option-Shift-Left</td>
<td></td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Control-G</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Control-Shift-G</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td></td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td></td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Command-Shift-End</td>
<td>Command-Shift-Down</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Command-Shift-Right</td>
<td>Shift-End</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Command-Shift-Left</td>
<td>Control-Shift-A</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Control-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Command-Shift-Home</td>
<td>Command-Shift-Up</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>Control-Shift-P</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Option-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Option-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Command-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Command--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>F5</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Command-Shift-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Control-O</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Command-Option-</td>
<td></td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Command-</td>
<td></td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>Command-Option-[</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Option-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Control-Q</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Control-W</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Command-K Command-L</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Command-K Command-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Control-T</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Command-Option-]</td>
<td>unfold</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Command-K Command-0</td>
<td>Command-K Command-J</td>
</tr>
</tbody>
</table>

**emmet**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Command-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Control-Option-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Command-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Command-.</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Control-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>
## Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Option-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Option-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

## Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>F7</td>
<td>Command-B</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>Command-\</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Command-Shift-B</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>Command-;</td>
</tr>
<tr>
<td>Step out of the current function scope</td>
<td>Shift-F11</td>
<td>Command-Shift-'</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>Command-'</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Control-Break</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

## Windows / Linux Default Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of default keyboard mode keybindings for Windows / Linux operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, **AWS Cloud9, Preferences**.
2. On the **Preferences** tab, choose **Keybindings**.
3. For **Keyboard Mode**, choose **Default**.
4. For **Operating System**, choose **Windows / Linux**.
See also Working with Keybindings (p. 173).

- General (p. 222)
- Tabs (p. 224)
- Panels (p. 225)
- Code Editor (p. 226)
- emmet (p. 231)
- Terminal (p. 231)
- Run and Debug (p. 231)

## General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Ctrl-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Ctrl-Space</td>
<td>Alt-Space</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Ctrl-Shift-Space</td>
<td>Alt-Shift-Space</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Ctrl-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Ctrl-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Ctrl-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Ctrl-Alt-K</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Ctrl-K</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Ctrl-Shift-K</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Ctrl-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Ctrl-G</td>
<td>gotolineline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
<td>Ctrl-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Alt-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Ctrl-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a Terminal tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Alt-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Ctrl-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Ctrl-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Ctrl-Shift-Z</td>
<td>Ctrl-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Ctrl-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Ctrl-Alt-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Alt-Shift-F</td>
<td>Ctrl-H</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Ctrl-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Ctrl-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Ctrl-Shift-Q</td>
<td>reverttossaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Alt-Shift-Q</td>
<td>reverttossavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Ctrl-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Ctrl-Shift-S</td>
<td>saveas</td>
</tr>
</tbody>
</table>
## Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Ctrl-Shift-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Ctrl-Alt-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Ctrl-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

<p>| Close all open tabs in the current pane, except the current tab          | Ctrl-Alt-W     | closeallbutme         |
| Close all open tabs in all panes                                        | Alt-Shift-W    | closealltabs          |
| Close the current pane                                                  | Ctrl-W         | closepane             |
| Close the current tab                                                   | Alt-W          | closetab              |
| Go one pane down                                                         | Ctrl-Meta-Down | gotopanedown          |
| Go one pane left                                                        | Ctrl-Meta-Left | gotopaneleft          |
| Go one pane right                                                       | Ctrl-Meta-Right| gotopaneright         |
| Go one pane up                                                          | Ctrl-Meta-Up   | gotopaneup            |
| Go one tab left                                                         | Ctrl-[         | gototableft           |
| Go one tab right                                                        | Ctrl-]         | gototabright          |
| Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there | Ctrl-Meta-Down | movetabdown           |
| Move the current tab left, or if the tab is already at the far left, create a split tab there | Ctrl-Meta-Left | movetableft           |
| Move the current tab right, or if the tab is already at the far right, create a split tab there | Ctrl-Meta-Right | movetabright          |
| Move the current tab up one pane, or if the tab is already at the very top, create a split tab there | Ctrl-Meta-Up | movetabup             |
| Go to the next pane                                                     | Ctrl-<code>         | nextpane              | | Go to the next tab                                                      | Ctrl-Tab | Alt-</code> | nexttab |
| Go to the previous pane                                                 | Ctrl-Shift-`   | previouspane          |</p>
<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to the previous tab</td>
<td>Ctrl-Shift-Tab</td>
<td>previoustab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Alt-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Ctrl-Shift-L</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Ctrl-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Ctrl-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Ctrl-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Ctrl-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Ctrl-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Ctrl-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Ctrl-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Ctrl-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Ctrl-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Ctrl-9</td>
<td>tab9</td>
</tr>
</tbody>
</table>

Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in Go to Anything mode</td>
<td>Ctrl-E</td>
<td>gotoanything</td>
</tr>
<tr>
<td>Show the Go window in Go to Command mode</td>
<td>Ctrl- .</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in Go to File mode.</td>
<td>Ctrl-O</td>
<td>gotofile</td>
</tr>
<tr>
<td>Show the Go window in Go to Symbol mode.</td>
<td>Ctrl-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the Outline window</td>
<td>Ctrl-Shift-E</td>
<td>outline</td>
</tr>
<tr>
<td>Show the Console window if hidden, or hide if shown</td>
<td>F6</td>
<td>toggleconsole</td>
</tr>
<tr>
<td>Show the Environment window if hidden, or hide if shown</td>
<td>Ctrl-I</td>
<td>toggletree</td>
</tr>
</tbody>
</table>
## Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already</td>
<td>Ctrl-Alt-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>added, add another cursor above that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line</td>
<td>Ctrl-Alt-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>above the active cursor, or if a second cursor is already added, move the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>second cursor up one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already</td>
<td>Ctrl-Alt-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>added, add another cursor below that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line</td>
<td>Ctrl-Alt-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>below the active cursor, or if a second cursor is already added, move the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>second cursor down one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their</td>
<td>Ctrl-Alt-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>lines, if they are misaligned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Shift-Backspace</td>
<td>backspace</td>
</tr>
<tr>
<td></td>
<td>Backspace</td>
<td></td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Ctrl-]</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Ctrl-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in</td>
<td>Ctrl-Z</td>
<td>Ctrl-Shift-Z</td>
</tr>
<tr>
<td>the IDE</td>
<td>Ctrl-Y</td>
<td></td>
</tr>
<tr>
<td>Center the selection</td>
<td>Ctrl-L</td>
<td>centerselection</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line</td>
<td>Alt-Shift-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>down</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Alt-Shift-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Cut the selection, or if there is no selection, delete one space</td>
<td>Shift-Delete</td>
<td>cut_or_delete</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>del</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied</td>
<td>Ctrl-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>----------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>contents immediately after the selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Ctrl-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to the next matching symbol in the selection</td>
<td>Ctrl-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
<td>Alt-L</td>
<td>Ctrl-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements</td>
<td>Ctrl-Command-Option-0</td>
<td>foldall</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Alt-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>golinedown</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>golineup</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Ctrl-End</td>
<td>gotoend</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>gotoleft</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Alt-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Alt-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>Alt-E</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>gotopagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Alt-Shift-E</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>gotoright</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Ctrl-Home</td>
<td>gotostart</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Ctrl-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Ctrl-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Ctrl-P</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Ctrl-+</td>
<td>Ctrl-=</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move the selection down one line</td>
<td>Alt-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move the selection up one line</td>
<td>Alt-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Option-Page Down</td>
<td>pagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Option-Page Up</td>
<td>pageup</td>
</tr>
<tr>
<td>Delete the contents of the current line</td>
<td>Ctrl-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Alt-Delete</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Alt-Backspace</td>
<td>removetolinelast</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Ctrl-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Ctrl-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Ctrl-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Scroll the current file down by one line</td>
<td>Ctrl-Down</td>
<td>scrolldown</td>
</tr>
<tr>
<td>Scroll the current file up by one line</td>
<td>Ctrl-Up</td>
<td>scrollup</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Ctrl-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>selectdown</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>selectleft</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-----------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Ctrl-Alt-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Ctrl-Alt-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Ctrl-Alt-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Ctrl-Alt-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Alt-K</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Alt-Shift-K</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selectright</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Ctrl-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Alt-Shift-Right</td>
<td>selecttolineend</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Alt-Shift-Left</td>
<td>selecttolinesstart</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Ctrl-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Ctrl-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td><strong>Keybinding</strong></td>
<td><strong>Command</strong></td>
</tr>
<tr>
<td>----------------</td>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Ctrl-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Ctrl-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Ctrl-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Ctrl--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Ctrl-Alt-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Ctrl-Alt-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Ctrl-O</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Ctrl-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Ctrl-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Alt-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Ctrl-Alt-E</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Ctrl-Q</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Ctrl-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Ctrl-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Alt-X</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Alt-Shift-L</td>
<td>unfold</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Alt-Shift-0</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>
## emmet

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Ctrl-Y</td>
<td><code>emmet_evaluate_math_expression</code></td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Ctrl-Alt-E</td>
<td><code>emmet_expand_abbreviation</code></td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td><code>emmet_expand_abbreviation_with_tab</code></td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Ctrl-.</td>
<td><code>emmet_select_next_item</code></td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Ctrl-,</td>
<td><code>emmet_select_previous_item</code></td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Ctrl-A</td>
<td><code>emmet_wrap_with_abbreviation</code></td>
</tr>
</tbody>
</table>

## Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Alt-T</td>
<td><code>openterminal</code></td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Alt-S</td>
<td><code>switchterminal</code></td>
</tr>
</tbody>
</table>

## Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Ctrl-B</td>
<td><code>build</code></td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td><code>resume</code></td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Alt-F5</td>
<td><code>run</code></td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td><code>runlast</code></td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td><code>stepinto</code></td>
</tr>
</tbody>
</table>
Windows / Linux Vim Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Vim keyboard mode keybindings for Windows / Linux operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, choose Keybindings.
3. For Keyboard Mode, choose Vim.
4. For Operating System, choose Windows / Linux.

See also Working with Keybindings (p. 173).

- General (p. 232)
- Tabs (p. 234)
- Panels (p. 236)
- Code Editor (p. 236)
- emmet (p. 241)
- Terminal (p. 242)
- Run and Debug (p. 242)

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step out of the current function scope</td>
<td>Shift-F11</td>
<td>stepout</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>stepover</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Ctrl-Shift-C</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Ctrl-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Ctrl-Space</td>
<td>Alt-Space</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>--------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Ctrl-Shift-Space</td>
<td>completeoverwrite</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Ctrl-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Ctrl-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Ctrl-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Ctrl-Alt-K</td>
<td>findall</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Ctrl-K</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Ctrl-Shift-K</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Ctrl-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Ctrl-G</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
<td>Ctrl-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Alt-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Ctrl-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a Terminal tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Alt-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Ctrl-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Ctrl-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Ctrl-Shift-Z</td>
<td>Ctrl-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Ctrl-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Ctrl-Alt-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Alt-Shift-F</td>
<td>Ctrl-H</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Ctrl-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Ctrl-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Ctrl-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Alt-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Ctrl-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Ctrl-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Ctrl-Shift-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Ctrl-Alt-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Ctrl-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

### Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Ctrl-Alt-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Alt-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Ctrl-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Alt-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Ctrl-Meta-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------</td>
<td>------------</td>
<td>---------</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Ctrl-Meta-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Ctrl-Meta-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Ctrl-Meta-Up</td>
<td>gotopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Ctrl-[]</td>
<td>gototableft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Ctrl-[</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Ctrl-Meta-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Ctrl-Meta-Left</td>
<td>movetableft</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Ctrl-Meta-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very top, create a split tab there</td>
<td>Ctrl-Meta-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>Go to the next pane</td>
<td>Ctrl-`</td>
<td>nextpane</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Ctrl-Tab</td>
<td>nexttab</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Ctrl-Shift-`</td>
<td>previouspane</td>
</tr>
<tr>
<td>Go to the previous tab</td>
<td>Ctrl-Shift-Tab</td>
<td>previoustab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Alt-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Ctrl-Shift-L</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Ctrl-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Ctrl-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Ctrl-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Ctrl-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Ctrl-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Ctrl-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Ctrl-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Ctrl-7</td>
<td>tab7</td>
</tr>
</tbody>
</table>
### Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to the eighth tab</td>
<td>Ctrl-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Ctrl-9</td>
<td>tab9</td>
</tr>
</tbody>
</table>

### Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already</td>
<td>Ctrl-Alt-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>added, add another cursor above that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second</td>
<td>Ctrl-Alt-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added, move the second cursor up one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is</td>
<td>Ctrl-Alt-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>already added, add another cursor below that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second</td>
<td>Ctrl-Alt-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>if a second cursor is already added, add another cursor below that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>-----------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>move the second cursor down one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move all cursors to the same space as the active</td>
<td>Ctrl-Alt-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>cursor on each of their lines, if they are</td>
<td></td>
<td></td>
</tr>
<tr>
<td>misaligned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Shift-Backspace</td>
<td>backspace</td>
</tr>
<tr>
<td></td>
<td>Backspace</td>
<td></td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Ctrl-[</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Ctrl-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from</td>
<td>Ctrl-Z</td>
<td>Ctrl-Shift-Z</td>
</tr>
<tr>
<td>the editor to somewhere else in the IDE</td>
<td>Ctrl-Y</td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the</td>
<td>Alt-Shift-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>copied contents one line down</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the</td>
<td>Alt-Shift-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>copied contents one line up</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cut the selection. If there is no selection,</td>
<td>Shift-Delete</td>
<td>cut_or_delete</td>
</tr>
<tr>
<td>delete one space</td>
<td>Delete</td>
<td>del</td>
</tr>
<tr>
<td>Delete one space</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste</td>
<td>Ctrl-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>the copied contents immediately after the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include the current line's contents in the</td>
<td>Ctrl-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include up to the next matching symbol in the</td>
<td>Ctrl-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is</td>
<td>Alt-L</td>
<td>Ctrl-F1</td>
</tr>
<tr>
<td>selected, unfold it</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except</td>
<td>Alt-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>for the current selection scope</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>golinedown</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>getlineup</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Ctrl-End</td>
<td>gotoend</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>gotoleft</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Alt-Right</td>
<td>End</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Alt-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>Alt-E</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>gotopagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Alt-Shift-E</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>gotoright</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Ctrl-Home</td>
<td>gotostart</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Ctrl-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Ctrl-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Ctrl-P</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Ctrl-+</td>
<td>Ctrl-=</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move the selection down one line</td>
<td>Alt-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move the selection up one line</td>
<td>Alt-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Delete the contents of the current line</td>
<td>Ctrl-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Alt-Delete</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Alt-Backspace</td>
<td>removetolinetart</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Ctrl-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Ctrl-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Ctrl-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Scroll the current file down by one line</td>
<td>Ctrl-Down</td>
<td>scrolldown</td>
</tr>
<tr>
<td>Scroll the current file up by one line</td>
<td>Ctrl-Up</td>
<td>scrollup</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Ctrl-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>selectdown</td>
</tr>
<tr>
<td>Include the next space to the left in the selection</td>
<td>Shift-Left</td>
<td>selectleft</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Ctrl-Alt-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Ctrl-Alt-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Ctrl-Alt-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Ctrl-Alt-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Alt-K</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Alt-Shift-K</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selectright</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-----------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Ctrl-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Alt-Shift-Right</td>
<td>selecttolineend</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Alt-Shift-Left</td>
<td>selecttolinestart</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Ctrl-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Ctrl-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Ctrl-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Ctrl-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Ctrl-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Ctrl--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Ctrl-Alt-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Ctrl-Alt-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Ctrl-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Ctrl-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Alt-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
</tbody>
</table>
### Description

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Ctrl-Alt-E</td>
<td>togglerrecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Ctrl-Q</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Ctrl-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Ctrl-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Alt-X</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Alt-Shift-L</td>
<td>Ctrl-Shift-F1</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Alt-Shift-0</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>

### emmet

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Ctrl-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Ctrl-Alt-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Ctrl-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Ctrl-</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Ctrl-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>
Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Alt-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the</td>
<td>Alt-S</td>
<td>switchterminal</td>
</tr>
<tr>
<td>Terminal tab</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Ctrl-B</td>
<td>build</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>resume</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Alt-F5</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>stepinto</td>
</tr>
<tr>
<td>Step out of the current function scope</td>
<td>Shift-F11</td>
<td>stepout</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>stepover</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Ctrl-Shift-C</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

Windows / Linux Emacs Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Emacs keyboard mode keybindings for Windows / Linux operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose **AWS Cloud9, Preferences**.
2. On the **Preferences** tab, choose **Keybindings**.
3. For **Keyboard Mode**, choose **Emacs**.
4. For **Operating System**, choose **Windows / Linux**.
See also Working with Keybindings (p. 173).

- General (p. 243)
- Tabs (p. 245)
- Panels (p. 246)
- Code Editor (p. 247)
- emmet (p. 252)
- Terminal (p. 252)
- Run and Debug (p. 252)

## General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Ctrl-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Ctrl-Space</td>
<td>Alt-Space</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Ctrl-Shift-Space</td>
<td>Alt-Shift-Space</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Ctrl-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Ctrl-X</td>
<td>cut</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Ctrl-F</td>
<td>find</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Ctrl-Alt-K</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>Ctrl-K</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Ctrl-Shift-K</td>
<td>findprevious</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>--------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Ctrl-Shift-B</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Ctrl-G</td>
<td>gotoline</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F3</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
<td>Ctrl-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Alt-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Ctrl-</td>
<td></td>
</tr>
<tr>
<td>Open a Terminal tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Alt-L</td>
<td>opentermhere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Ctrl-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Ctrl-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Ctrl-Shift-Z</td>
<td>Ctrl-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Ctrl-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Ctrl-Alt-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Alt-Shift-F</td>
<td>Ctrl-H</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Ctrl-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Ctrl-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Ctrl-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Alt-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Ctrl-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Ctrl-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------------------</td>
<td>----------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Ctrl-Shift-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Ctrl-Alt-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Ctrl-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

## Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Ctrl-Alt-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Alt-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Ctrl-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Alt-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Ctrl-Meta-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Ctrl-Meta-Left</td>
<td>gotopaneleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Ctrl-Meta-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Ctrl-Meta-Up</td>
<td>gottopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Ctrl-[</td>
<td>gototableft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Ctrl-]</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Ctrl-Meta-Down</td>
<td>movetabdown</td>
</tr>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Ctrl-Meta-Left</td>
<td>movetableft</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Ctrl-Meta-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at the very top, create a split tab there</td>
<td>Ctrl-Meta-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>Go to the next pane</td>
<td>Ctrl-`</td>
<td>nextpane</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Ctrl-Tab</td>
<td>Alt-`</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Ctrl-Shift-`</td>
<td>previouspane</td>
</tr>
</tbody>
</table>
## Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Go to the previous tab</td>
<td>Ctrl-Shift-Tab</td>
<td>previousstab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>refocusTab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Alt-Shift-T</td>
<td>reopenLastTab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Ctrl-Shift-L</td>
<td>revealtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Ctrl-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Ctrl-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Ctrl-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Ctrl-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Ctrl-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Ctrl-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Ctrl-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Ctrl-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Ctrl-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Ctrl-9</td>
<td>tab9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in Go to Anything mode</td>
<td>Ctrl-E</td>
<td>gotoanything</td>
</tr>
<tr>
<td>Show the Go window in Go to Command mode</td>
<td>Ctrl-.</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in Go to File mode.</td>
<td>Ctrl-O</td>
<td>gotofile</td>
</tr>
<tr>
<td>Show the Go window in Go to Symbol mode.</td>
<td>Ctrl-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the Outline window</td>
<td>Ctrl-Shift-E</td>
<td>outline</td>
</tr>
<tr>
<td>Show the Console window if hidden, or hide if shown</td>
<td>F6</td>
<td>toggleconsole</td>
</tr>
<tr>
<td>Show the Environment window if hidden, or hide if shown</td>
<td>Ctrl-I</td>
<td>toggletree</td>
</tr>
</tbody>
</table>

---

246
# Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already</td>
<td>Ctrl-Alt-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>added, add another cursor above that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second</td>
<td>Ctrl-Alt-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added, move the second cursor up one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already</td>
<td>Ctrl-Alt-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>added, add another cursor below that one</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second</td>
<td>Ctrl-Alt-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>cursor is already added, move the second cursor down one line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their</td>
<td>Ctrl-Alt-A</td>
<td>align Cursors</td>
</tr>
<tr>
<td>lines, if they are misaligned</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Shift-Backspace</td>
<td>backspace</td>
</tr>
<tr>
<td>Backspace</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Ctrl-[</td>
<td>blockindent</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Ctrl-[</td>
<td>blockoutdent</td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else</td>
<td>Ctrl-Z</td>
<td>Ctrl-Shift-Z</td>
</tr>
<tr>
<td>in the IDE</td>
<td>Ctrl-Y</td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line</td>
<td>Alt-Shift-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>down</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Alt-Shift-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Cut the selection, or if there is no selection, delete one space</td>
<td>Shift-Delete</td>
<td>cut_or_delete</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>del</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied</td>
<td>Ctrl-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>contents immediately after the selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Ctrl-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to the next matching symbol in selection</td>
<td>Ctrl-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
<td>Alt-L</td>
<td>Ctrl-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Alt-0</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>golinedown</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>getlineup</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Ctrl-End</td>
<td>gotoend</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>gotoleft</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Alt-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Alt-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>Alt-E</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>gotopagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Alt-Shift-E</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>gotoright</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Ctrl-Home</td>
<td>gotostart</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Ctrl-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Ctrl-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Ctrl-P</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Ctrl-++</td>
<td>Ctrl-=</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Ctrl-Shift-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>--------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Move selection down one line</td>
<td>Alt-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move selection up one line</td>
<td>Alt-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Delete the contents of the current line</td>
<td>Ctrl-D</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Alt-Delete</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Alt-Backspace</td>
<td>removetolinestart</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Ctrl-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Ctrl-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Ctrl-Shift-E</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Scroll the current file down by one line</td>
<td>Ctrl-Down</td>
<td>scrolldown</td>
</tr>
<tr>
<td>Scroll the current file up by one line</td>
<td>Ctrl-Up</td>
<td>scrollup</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Ctrl-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>selectdown</td>
</tr>
<tr>
<td>Include the next space left in the selection</td>
<td>Shift-Left</td>
<td>selectleft</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Ctrl-Alt-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Ctrl-Alt-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Ctrl-Alt-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Ctrl-Alt-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Alt-K</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Alt-Shift-K</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpageup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selectright</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Ctrl-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Alt-Shift-Right</td>
<td>selecttolineend</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Alt-Shift-Left</td>
<td>selecttolinestart</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Ctrl-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Ctrl-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Ctrl-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Ctrl-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Ctrl-</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>----------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Ctrl--</td>
<td>smallerfont</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>Ctrl-Alt-S</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Ctrl-Alt-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
<td>Ctrl-0</td>
<td>splitline</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Ctrl-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Ctrl-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>F2</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Alt-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Ctrl-Alt-E</td>
<td>togglerecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Ctrl-Q</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Ctrl-Shift-U</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Ctrl-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Alt-X</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Alt-Shift-L</td>
<td>Ctrl-Shift-F1</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Alt-Shift-0</td>
<td>unfoldall</td>
</tr>
</tbody>
</table>
emmet

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Ctrl-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Ctrl-Alt-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Ctrl-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Ctrl-,</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Ctrl-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>

Terminal

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Alt-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Alt-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

Run and Debug

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>Ctrl-B</td>
<td>build</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>resume</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Alt-F5</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>stepinto</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------</td>
<td>----------------</td>
<td>--------------------</td>
</tr>
<tr>
<td>Step out of the current function scope</td>
<td>Ctrl-F11</td>
<td>stepout</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>stepover</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Ctrl-Shift-C</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

Windows / Linux Sublime Keybindings Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of Sublime keyboard mode keybindings for Windows / Linux operating systems in the AWS Cloud9 IDE.

For more information, in the AWS Cloud9 IDE:

1. On the menu bar, choose AWS Cloud9, Preferences.
2. On the Preferences tab, choose Keybindings.
3. For Keyboard Mode, choose Sublime.
4. For Operating System, choose Windows / Linux.

See also Working with Keybindings (p. 173).

• General (p. 253)
• Tabs (p. 256)
• Panels (p. 257)
• Code Editor (p. 258)
• emmet (p. 263)
• Terminal (p. 263)
• Run and Debug (p. 263)

General

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add the selection as a watch expression</td>
<td>Ctrl-Shift-C</td>
<td>addwatchfromselection</td>
</tr>
<tr>
<td>Remove the cut selection from the clipboard</td>
<td>Esc</td>
<td>clearcut</td>
</tr>
<tr>
<td>Show the code completion context menu</td>
<td>Ctrl-Space</td>
<td>complete</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Code complete, and then overwrite</td>
<td>Ctrl-Shift-Space</td>
<td>completeoverwrite</td>
</tr>
<tr>
<td>Copy the selection to the clipboard</td>
<td>Ctrl-C</td>
<td>copy</td>
</tr>
<tr>
<td>Cut the selection to the clipboard</td>
<td>Ctrl-X</td>
<td>cut</td>
</tr>
<tr>
<td>Delete from the cursor to the start of the line</td>
<td>Ctrl-Shift-Backspace</td>
<td>delete_to_hard_bol</td>
</tr>
<tr>
<td>Delete from the cursor to the end of line</td>
<td>Ctrl-Shift-Delete</td>
<td>delete_to_hard_eol</td>
</tr>
<tr>
<td>Expand code, where applicable</td>
<td>Tab</td>
<td>expandSnippet</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document</td>
<td>Ctrl-F</td>
<td>find</td>
</tr>
<tr>
<td>Highlight all matches for the selection</td>
<td>Alt-F3</td>
<td>find_all_under</td>
</tr>
<tr>
<td>Highlight next match for the selection</td>
<td>Ctrl-F3</td>
<td>find_under</td>
</tr>
<tr>
<td>Highlight around cursor and all matches for highlight</td>
<td>Ctrl-D</td>
<td>find_under_expand</td>
</tr>
<tr>
<td>Highlight around cursor and outline all matches for highlight</td>
<td>Ctrl-K Ctrl-D</td>
<td>find_under_expand_skip</td>
</tr>
<tr>
<td>Highlight previous match for selection</td>
<td>Ctrl-Shift-F3</td>
<td>find_under_prev</td>
</tr>
<tr>
<td>Select all find matches in the current document</td>
<td>Ctrl-Alt-K</td>
<td>findAll</td>
</tr>
<tr>
<td>Go to the next match in the current document for the find query you entered last</td>
<td>F3</td>
<td>findnext</td>
</tr>
<tr>
<td>Go to the previous match in the current document for the find query you entered last</td>
<td>Shift-F3</td>
<td>find_previous</td>
</tr>
<tr>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
<td>Shift-F3</td>
<td>findReferences</td>
</tr>
<tr>
<td>Open the Environment window, and then make the list of files active</td>
<td>Shift-Esc</td>
<td>focusTree</td>
</tr>
<tr>
<td>Reformat the selected JavaScript code</td>
<td>Ctrl-Alt-F</td>
<td>formatcode</td>
</tr>
<tr>
<td>Show the go to line box</td>
<td>Ctrl-G</td>
<td>gotoLine</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>Hide the find and replace bar, if it is showing</td>
<td>Esc</td>
<td>hidesearchreplace</td>
</tr>
<tr>
<td>Go to the definition of the variable or function at the cursor</td>
<td>F12</td>
<td>jumptodef</td>
</tr>
<tr>
<td>If a local Lambda function is selected in the <strong>Lambda</strong> section of the <strong>AWS Resources</strong> window, attempts to upload the function to Lambda as a remote function</td>
<td>Ctrl-Shift-U</td>
<td>lambdaUploadFunction</td>
</tr>
<tr>
<td>Go to the end of the current word</td>
<td>Ctrl-Right</td>
<td>moveToWordEndRight</td>
</tr>
<tr>
<td>Go to the start of the current word</td>
<td>Ctrl-Left</td>
<td>moveToWordStartLeft</td>
</tr>
<tr>
<td>Create a new file</td>
<td>Alt-N</td>
<td>newfile</td>
</tr>
<tr>
<td>Show the <strong>Preferences</strong> tab</td>
<td>Ctrl-,</td>
<td>openpreferences</td>
</tr>
<tr>
<td>Open a <strong>Terminal</strong> tab, and then switch to the parent folder of the selected file in the list of files</td>
<td>Alt-L</td>
<td>opentermthere</td>
</tr>
<tr>
<td>Paste the clipboard's current contents at the cursor</td>
<td>Ctrl-V</td>
<td>paste</td>
</tr>
<tr>
<td>Show suggestions for fixing errors</td>
<td>Ctrl-F3</td>
<td>quickfix</td>
</tr>
<tr>
<td>Redo the last action</td>
<td>Ctrl-Shift-Z</td>
<td>Ctrl-Y</td>
</tr>
<tr>
<td>Refresh the preview pane</td>
<td>Ctrl-Enter</td>
<td>reloadpreview</td>
</tr>
<tr>
<td>Start a rename refactor for the selection</td>
<td>Ctrl-Alt-R</td>
<td>renameVar</td>
</tr>
<tr>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
<td>Ctrl-H</td>
<td>replace</td>
</tr>
<tr>
<td>Replace all find expression matches with replace with expression in the find and replace bar</td>
<td>Ctrl-Alt-Enter</td>
<td>replaceall</td>
</tr>
<tr>
<td>Replace next find expression match with replace with expression in the find and replace bar</td>
<td>Ctrl-Shift-H</td>
<td>replacenext</td>
</tr>
<tr>
<td>Rerun your initialization script</td>
<td>Ctrl-Enter</td>
<td>rerunInitScript</td>
</tr>
<tr>
<td>Restart the environment</td>
<td>Ctrl-R</td>
<td>restartc9</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Reset the current file to its last saved version</td>
<td>Ctrl-Shift-Q</td>
<td>reverttosaved</td>
</tr>
<tr>
<td>Reset each open file to its saved version</td>
<td>Alt-Shift-Q</td>
<td>reverttosavedall</td>
</tr>
<tr>
<td>Save the current file to disk</td>
<td>Ctrl-S</td>
<td>save</td>
</tr>
<tr>
<td>Save the current file to disk with a different file name</td>
<td>Ctrl-Shift-S</td>
<td>saveas</td>
</tr>
<tr>
<td>Show the find and replace bar for multiple files</td>
<td>Ctrl-Shift-F</td>
<td>searchinfiles</td>
</tr>
<tr>
<td>Include from the cursor to the end of the word in the selection</td>
<td>Ctrl-Shift-Right</td>
<td>selectToWordEndRight</td>
</tr>
<tr>
<td>Include from the cursor to the start of the word in the selection</td>
<td>Ctrl-Shift-Left</td>
<td>selectToWordStartLeft</td>
</tr>
<tr>
<td>Show the <strong>Process List</strong> dialog box</td>
<td>Ctrl-Alt-P</td>
<td>showprocesslist</td>
</tr>
<tr>
<td>Undo the last action</td>
<td>Ctrl-Z</td>
<td>undo</td>
</tr>
</tbody>
</table>

### Tabs

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close all open tabs in the current pane, except the current tab</td>
<td>Ctrl-Alt-W</td>
<td>closeallbutme</td>
</tr>
<tr>
<td>Close all open tabs in all panes</td>
<td>Alt-Shift-W</td>
<td>closealltabs</td>
</tr>
<tr>
<td>Close the current pane</td>
<td>Ctrl-W</td>
<td>closepane</td>
</tr>
<tr>
<td>Close the current tab</td>
<td>Alt-W</td>
<td>closetab</td>
</tr>
<tr>
<td>Go one pane down</td>
<td>Ctrl-Meta-Down</td>
<td>gotopanedown</td>
</tr>
<tr>
<td>Go one pane left</td>
<td>Ctrl-Meta-Left</td>
<td>gotopanelleft</td>
</tr>
<tr>
<td>Go one pane right</td>
<td>Ctrl-Meta-Right</td>
<td>gotopaneright</td>
</tr>
<tr>
<td>Go one pane up</td>
<td>Ctrl-Meta-Up</td>
<td>gototopaneup</td>
</tr>
<tr>
<td>Go one tab left</td>
<td>Ctrl-Page Up</td>
<td>gototableleft</td>
</tr>
<tr>
<td>Go one tab right</td>
<td>Ctrl-Page Down</td>
<td>gototabright</td>
</tr>
<tr>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
<td>Ctrl-Meta-Down</td>
<td>movetabdown</td>
</tr>
</tbody>
</table>
### Panels

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
<td>Ctrl-Meta-Left</td>
<td>moveleft</td>
</tr>
<tr>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
<td>Ctrl-Meta-Right</td>
<td>movetabright</td>
</tr>
<tr>
<td>Move the current tab up one pane, or if the tab is already at very top, create a split tab there</td>
<td>Ctrl-Meta-Up</td>
<td>movetabup</td>
</tr>
<tr>
<td>Go to the next tab</td>
<td>Ctrl-Tab</td>
<td>nexttab</td>
</tr>
<tr>
<td>Go to the previous pane</td>
<td>Ctrl-Shift-`</td>
<td>previouspane</td>
</tr>
<tr>
<td>Go to the previous tab</td>
<td>Ctrl-Shift-Tab</td>
<td>previoustab</td>
</tr>
<tr>
<td>Go back to the last tab</td>
<td>Esc</td>
<td>reftab</td>
</tr>
<tr>
<td>Open the last tab again</td>
<td>Ctrl-Shift-T</td>
<td>reopenlasttab</td>
</tr>
<tr>
<td>Show the current tab in the file tree</td>
<td>Ctrl-E</td>
<td>rewtab</td>
</tr>
<tr>
<td>Go to the tenth tab</td>
<td>Ctrl-0</td>
<td>tab0</td>
</tr>
<tr>
<td>Go to the first tab</td>
<td>Ctrl-1</td>
<td>tab1</td>
</tr>
<tr>
<td>Go to the second tab</td>
<td>Ctrl-2</td>
<td>tab2</td>
</tr>
<tr>
<td>Go to the third tab</td>
<td>Ctrl-3</td>
<td>tab3</td>
</tr>
<tr>
<td>Go to the fourth tab</td>
<td>Ctrl-4</td>
<td>tab4</td>
</tr>
<tr>
<td>Go to the fifth tab</td>
<td>Ctrl-5</td>
<td>tab5</td>
</tr>
<tr>
<td>Go to the sixth tab</td>
<td>Ctrl-6</td>
<td>tab6</td>
</tr>
<tr>
<td>Go to the seventh tab</td>
<td>Ctrl-7</td>
<td>tab7</td>
</tr>
<tr>
<td>Go to the eighth tab</td>
<td>Ctrl-8</td>
<td>tab8</td>
</tr>
<tr>
<td>Go to the ninth tab</td>
<td>Ctrl-9</td>
<td>tab9</td>
</tr>
</tbody>
</table>

### Description:

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the Go window in Go to Anything mode</td>
<td>Ctrl-E</td>
<td>Ctrl-P</td>
</tr>
<tr>
<td>Show the Go window in Go to Command mode</td>
<td>Ctrl-</td>
<td>gotocommand</td>
</tr>
<tr>
<td>Show the Go window in Go to File mode.</td>
<td>Ctrl-0</td>
<td>gotofile</td>
</tr>
</tbody>
</table>
### Code Editor

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show the <strong>Go</strong> window in <strong>Go to Symbol</strong> mode.</td>
<td>Ctrl-Shift-O</td>
<td>gotosymbol</td>
</tr>
<tr>
<td>Show the <strong>Outline</strong> window</td>
<td>Ctrl-R</td>
<td>Ctrl-Shift-R</td>
</tr>
<tr>
<td>Show the <strong>Console</strong> window if hidden, or hide if shown</td>
<td>Ctrl-`</td>
<td></td>
</tr>
<tr>
<td>Show the <strong>Environment</strong> window if hidden, or hide if shown</td>
<td>Ctrl-K Ctrl-B</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add a cursor one line above the active cursor, or if a cursor is already added, add another cursor above that one</td>
<td>Ctrl-Alt-Up</td>
<td>addCursorAbove</td>
</tr>
<tr>
<td>Add a second cursor one line above the active cursor, or if a second cursor is already added, move the second cursor up one line</td>
<td>Ctrl-Alt-Shift-Up</td>
<td>addCursorAboveSkipCurrent</td>
</tr>
<tr>
<td>Add a cursor one line below the active cursor, or if a cursor is already added, add another cursor below that one</td>
<td>Ctrl-Alt-Down</td>
<td>addCursorBelow</td>
</tr>
<tr>
<td>Add a second cursor one line below the active cursor, or if a second cursor is already added, move the second cursor down one line</td>
<td>Ctrl-Alt-Shift-Down</td>
<td>addCursorBelowSkipCurrent</td>
</tr>
<tr>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned</td>
<td>Ctrl-Alt-A</td>
<td>alignCursors</td>
</tr>
<tr>
<td>Backspace one space</td>
<td>Shift-Backspace</td>
<td>Backspace</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Ctrl-]</td>
<td></td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Ctrl-[</td>
<td></td>
</tr>
<tr>
<td>Control whether focus can be switched from the editor to somewhere else in the IDE</td>
<td>Ctrl-Z</td>
<td>Ctrl-Shift-Z</td>
</tr>
<tr>
<td>Center the selection</td>
<td>Ctrl-K Ctrl-C</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
<td>Alt-Shift-Down</td>
<td>copylinesdown</td>
</tr>
<tr>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
<td>Alt-Shift-Up</td>
<td>copylinesup</td>
</tr>
<tr>
<td>Cut the selection, or if there is no selection, delete one space</td>
<td>Shift-Delete</td>
<td>cut_or_delete</td>
</tr>
<tr>
<td>Delete one space</td>
<td>Delete</td>
<td>del</td>
</tr>
<tr>
<td>Copy the contents of the selection, and paste the copied contents immediately after the selection</td>
<td>Ctrl-Shift-D</td>
<td>duplicateSelection</td>
</tr>
<tr>
<td>Include the current line's contents in the selection</td>
<td>Ctrl-Shift-L</td>
<td>expandtoline</td>
</tr>
<tr>
<td>Include up to the next matching symbol in the selection</td>
<td>Ctrl-Shift-M</td>
<td>expandToMatching</td>
</tr>
<tr>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
<td>Alt-L</td>
<td>Ctrl-F1</td>
</tr>
<tr>
<td>Fold all possibly foldable elements, except for the current selection scope</td>
<td>Ctrl-K Ctrl-1</td>
<td>foldOther</td>
</tr>
<tr>
<td>Go down one line</td>
<td>Down</td>
<td>golinedown</td>
</tr>
<tr>
<td>Go up one line</td>
<td>Up</td>
<td>getlineup</td>
</tr>
<tr>
<td>Go to the end of the file</td>
<td>Ctrl-End</td>
<td>gotoend</td>
</tr>
<tr>
<td>Go left one space</td>
<td>Left</td>
<td>gotoleft</td>
</tr>
<tr>
<td>Go to the end of the current line</td>
<td>Alt-Right</td>
<td>End</td>
</tr>
<tr>
<td>Go to the start of the current line</td>
<td>Alt-Left</td>
<td>Home</td>
</tr>
<tr>
<td>Go to the next error</td>
<td>Ctrl-F6</td>
<td>goToNextError</td>
</tr>
<tr>
<td>Go down one page</td>
<td>Page Down</td>
<td>gotopagedown</td>
</tr>
<tr>
<td>Go up one page</td>
<td>Page Up</td>
<td>gotopageup</td>
</tr>
<tr>
<td>Go to the previous error</td>
<td>Ctrl-Shift-F6</td>
<td>goToPreviousError</td>
</tr>
<tr>
<td>Go right one space</td>
<td>Right</td>
<td>goright</td>
</tr>
<tr>
<td>Go to the start of the file</td>
<td>Ctrl-Home</td>
<td>gotostart</td>
</tr>
<tr>
<td>Go one word to the left</td>
<td>Ctrl-Left</td>
<td>gotowordleft</td>
</tr>
<tr>
<td>Go one word to the right</td>
<td>Ctrl-Right</td>
<td>gotowordright</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>---------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Indent the selection one tab</td>
<td>Tab</td>
<td>indent</td>
</tr>
<tr>
<td>Include from the cursor to the start of the word in the selection</td>
<td>Ctrl-J</td>
<td>joinlines</td>
</tr>
<tr>
<td>Go to the matching symbol in the current scope</td>
<td>Ctrl-M</td>
<td>jumptomatching</td>
</tr>
<tr>
<td>Increase the font size</td>
<td>Ctrl--</td>
<td>Ctrl-=</td>
</tr>
<tr>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
<td>Alt-Down</td>
<td>modifyNumberDown</td>
</tr>
<tr>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
<td>Alt-Up</td>
<td>modifyNumberUp</td>
</tr>
<tr>
<td>Move the selection down one line</td>
<td>Ctrl-Shift-Down</td>
<td>movelinesdown</td>
</tr>
<tr>
<td>Move the selection up one line</td>
<td>Ctrl-Shift-Up</td>
<td>movelinesup</td>
</tr>
<tr>
<td>Outdent the selection one tab</td>
<td>Shift-Tab</td>
<td>outdent</td>
</tr>
<tr>
<td>Turn on overwrite mode, or if on, turn off</td>
<td>Insert</td>
<td>overwrite</td>
</tr>
<tr>
<td>Delete the contents of the current line</td>
<td>Ctrl-Shift-K</td>
<td>removeline</td>
</tr>
<tr>
<td>Delete from the cursor to the end of the current line</td>
<td>Alt-Delete</td>
<td>removetolineend</td>
</tr>
<tr>
<td>Delete from the beginning of the current line up to the cursor</td>
<td>Alt-Backspace</td>
<td>removetolinenstart</td>
</tr>
<tr>
<td>Delete the word to the left of the cursor</td>
<td>Ctrl-Backspace</td>
<td>removewordleft</td>
</tr>
<tr>
<td>Delete the word to the right of the cursor</td>
<td>Ctrl-Delete</td>
<td>removewordright</td>
</tr>
<tr>
<td>Replay previously recorded keystrokes</td>
<td>Ctrl-Shift-Q</td>
<td>replaymacro</td>
</tr>
<tr>
<td>Scroll the current file down by one line</td>
<td>Ctrl-Down</td>
<td>scrolldown</td>
</tr>
<tr>
<td>Scroll the current file up by one line</td>
<td>Ctrl-Up</td>
<td>scrollup</td>
</tr>
<tr>
<td>Select all selectable content</td>
<td>Ctrl-A</td>
<td>selectall</td>
</tr>
<tr>
<td>Include the next line down in the selection</td>
<td>Shift-Down</td>
<td>selectdown</td>
</tr>
<tr>
<td>Include the next space left in the selection</td>
<td>Shift-Left</td>
<td>selectleft</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>----------------------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
<td>Shift-End</td>
<td>selectlineend</td>
</tr>
<tr>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
<td>Shift-Home</td>
<td>selectlinestart</td>
</tr>
<tr>
<td>Include more matching selections that are after the selection</td>
<td>Ctrl-Alt-Right</td>
<td>selectMoreAfter</td>
</tr>
<tr>
<td>Include more matching selections that are before the selection</td>
<td>Ctrl-Alt-Left</td>
<td>selectMoreBefore</td>
</tr>
<tr>
<td>Include the next matching selection that is after the selection</td>
<td>Ctrl-Alt-Shift-Right</td>
<td>selectNextAfter</td>
</tr>
<tr>
<td>Include the next matching selection that is before the selection</td>
<td>Ctrl-Alt-Shift-Left</td>
<td>selectNextBefore</td>
</tr>
<tr>
<td>Select or find the next matching selection</td>
<td>Alt-K</td>
<td>selectOrFindNext</td>
</tr>
<tr>
<td>Select or find the previous matching selection</td>
<td>Alt-Shift-K</td>
<td>selectOrFindPrevious</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current page in the selection</td>
<td>Shift-Page Down</td>
<td>selectpagedown</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
<td>Shift-Page Up</td>
<td>selectpagetup</td>
</tr>
<tr>
<td>Include the next space to the right of the cursor in the selection</td>
<td>Shift-Right</td>
<td>selecttright</td>
</tr>
<tr>
<td>Include from the cursor down to the end of the current file in the selection</td>
<td>Ctrl-Shift-End</td>
<td>selecttoend</td>
</tr>
<tr>
<td>Include from the cursor to the end of the current line in the selection</td>
<td>Alt-Shift-Right</td>
<td>selecttolineend</td>
</tr>
<tr>
<td>Include from the beginning of the current line to the cursor in the selection</td>
<td>Alt-Shift-Left</td>
<td>selecttolinestart</td>
</tr>
<tr>
<td>Include from the cursor to the next matching symbol in the current scope</td>
<td>Ctrl-Shift-P</td>
<td>selecttomatching</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>---------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
<td>Ctrl-Shift-Home</td>
<td>selecttostart</td>
</tr>
<tr>
<td>Include the next line up in the selection</td>
<td>Shift-Up</td>
<td>selectup</td>
</tr>
<tr>
<td>Include the next word to the left of the cursor in the selection</td>
<td>Ctrl-Shift-Left</td>
<td>selectwordleft</td>
</tr>
<tr>
<td>Include the next word to the right of the cursor in the selection</td>
<td>Ctrl-Shift-Right</td>
<td>selectwordright</td>
</tr>
<tr>
<td>Show the Preferences tab</td>
<td>Ctrl-,</td>
<td>showSettingsMenu</td>
</tr>
<tr>
<td>Clear all previous selections</td>
<td>Esc</td>
<td>singleSelection</td>
</tr>
<tr>
<td>Decrease the font size</td>
<td>Ctrl--</td>
<td>Ctrl-Shift-=</td>
</tr>
<tr>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
<td>F9</td>
<td>sortlines</td>
</tr>
<tr>
<td>Add a cursor at the end of the current line</td>
<td>Ctrl-Shift-L</td>
<td>splitIntoLines</td>
</tr>
<tr>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
<td>Ctrl-Shift-/</td>
<td>toggleBlockComment</td>
</tr>
<tr>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
<td>Ctrl-/</td>
<td>togglecomment</td>
</tr>
<tr>
<td>Fold code, or remove code folding if it is there</td>
<td>Ctrl-Shift-[</td>
<td>toggleFoldWidget</td>
</tr>
<tr>
<td>Fold parent code, or remove folding if it is there</td>
<td>Alt-F2</td>
<td>toggleParentFoldWidget</td>
</tr>
<tr>
<td>Start keystroke recording, or stop if it is already recording</td>
<td>Ctrl-Q</td>
<td>togglererecording</td>
</tr>
<tr>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
<td>Ctrl-Q</td>
<td>toggleWordWrap</td>
</tr>
<tr>
<td>Change the selection to all lowercase</td>
<td>Ctrl-K Ctrl-L</td>
<td>tolowercase</td>
</tr>
<tr>
<td>Change the selection to all uppercase</td>
<td>Ctrl-K Ctrl-U</td>
<td>touppercase</td>
</tr>
<tr>
<td>Transpose the selection</td>
<td>Alt-X</td>
<td>transposeletters</td>
</tr>
<tr>
<td>Unfold the selected code</td>
<td>Ctrl-Shift-]</td>
<td>unfold</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------</td>
<td>---------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Unfold code folding for the entire file</td>
<td>Ctrl-K Ctrl-0</td>
<td>unfoldall</td>
</tr>
<tr>
<td></td>
<td>Ctrl-K Ctrl-J</td>
<td></td>
</tr>
</tbody>
</table>

**emmet**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
<td>Shift-Ctrl-Y</td>
<td>emmet_evaluate_math_expression</td>
</tr>
<tr>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
<td>Ctrl-Alt-E</td>
<td>emmet_expand_abbreviation</td>
</tr>
<tr>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
<td>Tab</td>
<td>emmet_expand_abbreviation_with_tab</td>
</tr>
<tr>
<td>Go to the next editable code part</td>
<td>Shift-Ctrl-.</td>
<td>emmet_select_next_item</td>
</tr>
<tr>
<td>Go to the previous editable code part</td>
<td>Shift-Ctrl-,</td>
<td>emmet_select_previous_item</td>
</tr>
<tr>
<td>Expand an abbreviation, and then place the current selection within the last element of the generated snippet</td>
<td>Shift-Ctrl-A</td>
<td>emmet_wrap_with_abbreviation</td>
</tr>
</tbody>
</table>

**Terminal**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open a new Terminal tab</td>
<td>Alt-T</td>
<td>openterminal</td>
</tr>
<tr>
<td>Switch between the editor and the Terminal tab</td>
<td>Alt-S</td>
<td>switchterminal</td>
</tr>
</tbody>
</table>

**Run and Debug**

<table>
<thead>
<tr>
<th>Description</th>
<th>Keybinding</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build the current file</td>
<td>F7</td>
<td>Ctrl-B</td>
</tr>
<tr>
<td>Resume the current paused process</td>
<td>F8</td>
<td>resume</td>
</tr>
<tr>
<td>Description</td>
<td>Keybinding</td>
<td>Command</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Run or debug the current application</td>
<td>Ctrl-Shift-B</td>
<td>run</td>
</tr>
<tr>
<td>Run or debug the last run file</td>
<td>F5</td>
<td>runlast</td>
</tr>
<tr>
<td>Step into the function that is next on the stack</td>
<td>F11</td>
<td>stepinto</td>
</tr>
<tr>
<td>Step out of the current function scope</td>
<td>Shift-F11</td>
<td>stepout</td>
</tr>
<tr>
<td>Step over the current expression on the stack</td>
<td>F10</td>
<td>stepover</td>
</tr>
<tr>
<td>Stop running or debugging the current application</td>
<td>Shift-F5</td>
<td>stop</td>
</tr>
<tr>
<td>Stop building the current file</td>
<td>Ctrl-Break</td>
<td>stopbuild</td>
</tr>
</tbody>
</table>

Commands Reference for the AWS Cloud9 Integrated Development Environment (IDE)

Following is a list of default commands in the AWS Cloud9 IDE.

To run a command in the AWS Cloud9 IDE:

1. Choose the Go button (magnifying glass) to display the Go window. If the Go button is not visible, choose Window, Go on the menu bar.
2. In the Go to Anything box, type a dot (.) followed by the name of the command.
3. Do one of the following in the list of matching commands:
   - Choose the command to run.
   - Use your up and down arrow keys to choose a command, and then press Enter to run the chosen command.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addCursorAbove</td>
<td>Add a cursor one line above the active cursor, or if a cursor is already added, add another cursor above that one</td>
</tr>
<tr>
<td>addCursorAboveSkipCurrent</td>
<td>Add a second cursor one line above the active cursor, or if a second cursor is already added, move the second cursor up one line</td>
</tr>
<tr>
<td>addCursorBelow</td>
<td>Add a cursor one line below the active cursor, or if a cursor is already added, add another cursor below that one</td>
</tr>
<tr>
<td>addCursorBelowSkipCurrent</td>
<td>Add a second cursor one line below the active cursor, or if a second cursor is already added, move the second cursor down one line</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>addFavorite</td>
<td>Add the selected file or folder to the <em>Favorites</em> list in the <em>Environment</em> window</td>
</tr>
<tr>
<td>addWatchFromSelection</td>
<td>Add the selection as a watch expression</td>
</tr>
<tr>
<td>alignCursors</td>
<td>Move all cursors to the same space as the active cursor on each of their lines, if they are misaligned</td>
</tr>
<tr>
<td>aws-panel</td>
<td>Show the <em>AWS Resources</em> window</td>
</tr>
<tr>
<td>backspace</td>
<td>Backspace one space</td>
</tr>
<tr>
<td>blockindent</td>
<td>Indent the selection one tab</td>
</tr>
<tr>
<td>blockoutdent</td>
<td>Outdent the selection one tab</td>
</tr>
<tr>
<td>build</td>
<td>Build the current file</td>
</tr>
<tr>
<td>cancelBrowserAction</td>
<td>Cancel various built-in web browser key bindings that can be annoying if triggered accidentally</td>
</tr>
<tr>
<td>cancelBrowserUndoInAce</td>
<td>Control whether focus can be switched from the editor to somewhere else in the IDE</td>
</tr>
<tr>
<td>centerselection</td>
<td>Center the selection</td>
</tr>
<tr>
<td>clearcut</td>
<td>Remove the cut selection from the clipboard</td>
</tr>
<tr>
<td>clearterm</td>
<td>Clear the buffer in the <em>Terminal</em> pane</td>
</tr>
<tr>
<td>clonetab</td>
<td>Create a copy of the current tab in a new tab</td>
</tr>
<tr>
<td>closeallbutme</td>
<td>Close all open tabs in the current pane, except the current tab</td>
</tr>
<tr>
<td>closealltabs</td>
<td>Close all open tabs in all panes</td>
</tr>
<tr>
<td>closealltotheleft</td>
<td>Close all tabs to the left of the current tab</td>
</tr>
<tr>
<td>closealltotheright</td>
<td>Close all tabs to the right of the current tab</td>
</tr>
<tr>
<td>closepane</td>
<td>Close the current pane</td>
</tr>
<tr>
<td>closetab</td>
<td>Close the current tab</td>
</tr>
<tr>
<td>complete</td>
<td>Show the code completion context menu</td>
</tr>
<tr>
<td>completeoverwrite</td>
<td>Code complete, and then overwrite</td>
</tr>
<tr>
<td>convertIndentation</td>
<td>Convert between tabs and spaces in the editor</td>
</tr>
<tr>
<td>copy</td>
<td>Copy the selection to the clipboard</td>
</tr>
<tr>
<td>copyFilePath</td>
<td>Copy the full path of the current file to the clipboard</td>
</tr>
<tr>
<td>copylinesdown</td>
<td>Copy the contents of the line, and paste the copied contents one line down</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>copylinesup</td>
<td>Copy the contents of the line, and paste the copied contents one line up</td>
</tr>
<tr>
<td>cut</td>
<td>Cut the selection to the clipboard</td>
</tr>
<tr>
<td>cut_or_delete</td>
<td>Cut the selection to the clipboard, or delete to the right if the selection is empty</td>
</tr>
<tr>
<td>del</td>
<td>Delete one space</td>
</tr>
<tr>
<td>detectIndentation</td>
<td>Detect the indentation type (spaces or tabs) and length, based on the document's contents</td>
</tr>
<tr>
<td>duplicateSelection</td>
<td>Copy the contents of the selection, and paste the copied contents immediately after the selection</td>
</tr>
<tr>
<td>emmet_decrement_number_by_01</td>
<td>Decrease the selected number by 0.1, if it is a number</td>
</tr>
<tr>
<td>emmet_decrement_number_by_1</td>
<td>Decrease the selected number by 1, if is a number</td>
</tr>
<tr>
<td>emmet_decrement_number_by_10</td>
<td>Decrease the selected number by 10, if is a number</td>
</tr>
<tr>
<td>emmet_evaluate_math_expression</td>
<td>Evaluate a simple math expression (such as 2*4 or 10/2), and output its result</td>
</tr>
<tr>
<td>emmet_expand_abbreviation</td>
<td>Expand CSS-like abbreviations into HTML, XML, or CSS code, depending on the current file's syntax</td>
</tr>
<tr>
<td>emmet_expand_abbreviation_with_tab</td>
<td>Traverse expanded CSS-like abbreviations, by tab stop</td>
</tr>
<tr>
<td>emmet_increment_number_by_01</td>
<td>Increase the selected number by 0.1, if it is a number</td>
</tr>
<tr>
<td>emmet_increment_number_by_1</td>
<td>Increase the selected number by 1, if it is a number</td>
</tr>
<tr>
<td>emmet_increment_number_by_10</td>
<td>Increase the selected number by 10, if it is a number</td>
</tr>
<tr>
<td>emmet_match_pair_inward</td>
<td>Shrink the selection to the next inner set of matching tags</td>
</tr>
<tr>
<td>emmet_match_pair_outward</td>
<td>Expand the selection to include the next outer set of matching tags</td>
</tr>
<tr>
<td>emmet_matching_pair</td>
<td>Go between the opening and closing tag, if the selection is a tag</td>
</tr>
<tr>
<td>emmet_next_edit_point</td>
<td>Go to the next tag, empty attribute, or newline with indentation</td>
</tr>
<tr>
<td>emmet_prev_edit_point</td>
<td>Go to the previous tag, empty attribute, or newline with indentation</td>
</tr>
<tr>
<td>emmet_reflect_css_value</td>
<td>Copy the selected CSS property into all matching variations, if the selection is a CSS property</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>emmet_remove_tag</td>
<td>Delete the selected tag, if the selection is a tag</td>
</tr>
<tr>
<td>emmet_select_next_item</td>
<td>Go to the next editable code part</td>
</tr>
<tr>
<td>emmet_select_previous_item</td>
<td>Go to the previous editable code part</td>
</tr>
<tr>
<td>emmet_split_join_tag</td>
<td>If the selection is an empty tag, replace it with an opening and closing tag pair; if the tag has an opening and closing tag pair, replace it with an empty tag</td>
</tr>
<tr>
<td>emmet_toggle_comment</td>
<td>Add comment characters to the current line, or remove them if they are there</td>
</tr>
<tr>
<td>emmet_wrap_with_abbreviation</td>
<td>Expand an abbreviation, and then place the selection within the last element of the generated snippet</td>
</tr>
<tr>
<td>expandSnippet</td>
<td>Expand code, where applicable</td>
</tr>
<tr>
<td>expandToLine</td>
<td>Include the current line's contents in the selection</td>
</tr>
<tr>
<td>expandToMatching</td>
<td>Include up to the next matching symbol in the selection</td>
</tr>
<tr>
<td>find</td>
<td>Show the find and replace bar for the current document</td>
</tr>
<tr>
<td>findAll</td>
<td>Select all find matches in the current document</td>
</tr>
<tr>
<td>findnext</td>
<td>Go to the next match in the current document for the find query you entered last</td>
</tr>
<tr>
<td>findprevious</td>
<td>Go to the previous match in the current document for the find query you entered last</td>
</tr>
<tr>
<td>findReferences</td>
<td>Display all known references to the symbol at the insertion point in the active file in the editor</td>
</tr>
<tr>
<td>focusTree</td>
<td>Open the Environment window, and then make the list of files active</td>
</tr>
<tr>
<td>fold</td>
<td>Fold the selected code; if a folded unit is selected, unfold it</td>
</tr>
<tr>
<td>foldall</td>
<td>Fold all possibly foldable elements</td>
</tr>
<tr>
<td>foldOther</td>
<td>Fold all possibly foldable elements, except for the selection scope</td>
</tr>
<tr>
<td>forceToggleTimeslider</td>
<td>Show the File Revision History pane, or hide if shown</td>
</tr>
<tr>
<td>formatcode</td>
<td>Reformat the selected JavaScript code</td>
</tr>
<tr>
<td>formatprefs</td>
<td>Open the Project Settings section of the Preferences tab to programming language settings</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>foursplit</td>
<td>Display a four-pane layout</td>
</tr>
<tr>
<td>gethelp</td>
<td>Display the AWS Discussion Forum for AWS Cloud9</td>
</tr>
<tr>
<td>gitcloneterminal</td>
<td>Run the <code>git clone</code> command in a new terminal session</td>
</tr>
<tr>
<td>golinedown</td>
<td>Go down one line</td>
</tr>
<tr>
<td>golineup</td>
<td>Go up one line</td>
</tr>
<tr>
<td>gotoanything</td>
<td>Show the Go window in <strong>Go to Anything</strong> mode</td>
</tr>
<tr>
<td>gotocommand</td>
<td>Show the Go window in <strong>Go to Command</strong> mode</td>
</tr>
<tr>
<td>gotoend</td>
<td>Go to the end of the file</td>
</tr>
<tr>
<td>gotofile</td>
<td>Show the Go window in <strong>Go to File</strong> mode.</td>
</tr>
<tr>
<td>gotoleft</td>
<td>Go left one space</td>
</tr>
<tr>
<td>gotoline</td>
<td>Show the go to line box</td>
</tr>
<tr>
<td>gotolineend</td>
<td>Go to the end of the current line</td>
</tr>
<tr>
<td>gotolinetart</td>
<td>Go to the start of the current line</td>
</tr>
<tr>
<td>goToNextError</td>
<td>Go to the next error</td>
</tr>
<tr>
<td>gotopagedown</td>
<td>Go down one page</td>
</tr>
<tr>
<td>gotopageup</td>
<td>Go up one page</td>
</tr>
<tr>
<td>gotopanedown</td>
<td>Go one pane down</td>
</tr>
<tr>
<td>gotopaneleft</td>
<td>Go one pane left</td>
</tr>
<tr>
<td>gotopaneright</td>
<td>Go one pane right</td>
</tr>
<tr>
<td>gotopaneup</td>
<td>Go one pane up</td>
</tr>
<tr>
<td>goToPreviousError</td>
<td>Go to the previous error</td>
</tr>
<tr>
<td>gotoright</td>
<td>Go right one space</td>
</tr>
<tr>
<td>gotostart</td>
<td>Go to the start of the file</td>
</tr>
<tr>
<td>gotosymbol</td>
<td>Show the Go window in <strong>Go to Symbol</strong> mode</td>
</tr>
<tr>
<td>gototableleft</td>
<td>Go one tab left</td>
</tr>
<tr>
<td>gototabright</td>
<td>Go one tab right</td>
</tr>
<tr>
<td>gotowordleft</td>
<td>Go one word to the left</td>
</tr>
<tr>
<td>gotowordright</td>
<td>Go one word to the right</td>
</tr>
<tr>
<td>hideGotoLine</td>
<td>Hide the go to line box, if it is showing</td>
</tr>
<tr>
<td>hidesearchreplace</td>
<td>Hide the find and replace bar, if it is showing</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>hsplit</td>
<td>Split the current pane into two columns, and then move the current tab to the new column</td>
</tr>
<tr>
<td>indent</td>
<td>Indent the selection one tab</td>
</tr>
<tr>
<td>insertstring</td>
<td>Insert a string of text while typing or pasting</td>
</tr>
<tr>
<td>inserttext</td>
<td>Insert text while typing or pasting</td>
</tr>
<tr>
<td>invertSelection</td>
<td>Select everything other than the selection</td>
</tr>
<tr>
<td>joinlines</td>
<td>Remove all line breaks from the current selection</td>
</tr>
<tr>
<td>jumptodef</td>
<td>Go to the definition of the variable or function at the cursor</td>
</tr>
<tr>
<td>jumptomatching</td>
<td>Go to the matching symbol in the current scope</td>
</tr>
<tr>
<td>lambdaConvertFunction</td>
<td>Show the Convert to SAM dialog box</td>
</tr>
<tr>
<td>lambdaCreateFunction</td>
<td>Show the Create serverless application dialog box</td>
</tr>
<tr>
<td>lambdaImportFunction</td>
<td>If a remote AWS Lambda function is selected in the Lambda section of the AWS Resources window, attempts to import the function into the IDE as a local function</td>
</tr>
<tr>
<td>lambdaLinkToCFStack</td>
<td>Show the Link application to CloudFormation stack dialog box</td>
</tr>
<tr>
<td>lambdaRefreshFunctionsList</td>
<td>Refreshes the Lambda section of the AWS Resources window if shown</td>
</tr>
<tr>
<td>lambdaUploadFunction</td>
<td>If a local Lambda function is selected in the Lambda section of the AWS Resources window, attempts to upload the function to Lambda as a remote function</td>
</tr>
<tr>
<td>largerfont</td>
<td>Increase the font size</td>
</tr>
<tr>
<td>maximizeconsole</td>
<td>Expand the console to cover the entire IDE</td>
</tr>
<tr>
<td>modifyNumberDown</td>
<td>Decrease the number to the left of the cursor by 1, if it is a number</td>
</tr>
<tr>
<td>modifyNumberUp</td>
<td>Increase the number to the left of the cursor by 1, if it is a number</td>
</tr>
<tr>
<td>movelinesdown</td>
<td>Move selection down one line</td>
</tr>
<tr>
<td>movelinesup</td>
<td>Move selection up one line</td>
</tr>
<tr>
<td>movetabdown</td>
<td>Move the current tab down one pane, or if the tab is already at the very bottom, create a split tab there</td>
</tr>
<tr>
<td>movetableft</td>
<td>Move the current tab left, or if the tab is already at the far left, create a split tab there</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>movetabright</td>
<td>Move the current tab right, or if the tab is already at the far right, create a split tab there</td>
</tr>
<tr>
<td>movetabup</td>
<td>Move the current tab up one pane, or if the tab is already at very top, create a split tab there</td>
</tr>
<tr>
<td>newEnvironment</td>
<td>Show the Create new environment wizard in the AWS Cloud9 console</td>
</tr>
<tr>
<td>newfile</td>
<td>Create a new file</td>
</tr>
<tr>
<td>newfolder</td>
<td>Create a new folder relative to the selection in the Environment window</td>
</tr>
<tr>
<td>nextpane</td>
<td>Go to the next pane</td>
</tr>
<tr>
<td>nexttab</td>
<td>Go to the next tab</td>
</tr>
<tr>
<td>nosplit</td>
<td>Combine all split panes into a single pane</td>
</tr>
<tr>
<td>opencoverageview</td>
<td>Show the Code Coverage tab</td>
</tr>
<tr>
<td>openpreferences</td>
<td>Show the Preferences tab</td>
</tr>
<tr>
<td>opentermhere</td>
<td>Open a Terminal tab, and then switch to the parent folder of the selected file in the list of files</td>
</tr>
<tr>
<td>openterminal</td>
<td>Open a new Terminal tab</td>
</tr>
<tr>
<td>outdent</td>
<td>Outdent the selection one tab</td>
</tr>
<tr>
<td>outline</td>
<td>Show the Outline window</td>
</tr>
<tr>
<td>overwrite</td>
<td>Turn on overwrite mode, or if on, turn off</td>
</tr>
<tr>
<td>pagedown</td>
<td>Go down one page</td>
</tr>
<tr>
<td>pageup</td>
<td>Go up one page</td>
</tr>
<tr>
<td>passKeysToBrowser</td>
<td>Enable keys to be handled by the web browser</td>
</tr>
<tr>
<td>paste</td>
<td>Paste the clipboard's current contents at the cursor</td>
</tr>
<tr>
<td>preview</td>
<td>Show the preview pane</td>
</tr>
<tr>
<td>previouspane</td>
<td>Go to the previous pane</td>
</tr>
<tr>
<td>previousstab</td>
<td>Go to the previous tab</td>
</tr>
<tr>
<td>quickfix</td>
<td>Show suggestions for fixing errors</td>
</tr>
<tr>
<td>redo</td>
<td>Redo the last action</td>
</tr>
<tr>
<td>refocusTab</td>
<td>Go back to the last tab</td>
</tr>
<tr>
<td>reloadpreview</td>
<td>Refresh the preview pane</td>
</tr>
<tr>
<td>removefavorite</td>
<td>Delete the item from the Favorites list, if the selection is a favorite</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>removeline</td>
<td>Delete the contents of the current line</td>
</tr>
<tr>
<td>removetolineend</td>
<td>Delete from the cursor to the end of the current line</td>
</tr>
<tr>
<td>removetolinestart</td>
<td>Delete from the beginning of the current line up to the cursor</td>
</tr>
<tr>
<td>removewordleft</td>
<td>Delete the word to the left of the cursor</td>
</tr>
<tr>
<td>removewordright</td>
<td>Delete the word to the right of the cursor</td>
</tr>
<tr>
<td>renameVar</td>
<td>Start a rename refactor for the selection</td>
</tr>
<tr>
<td>reopenLastTab</td>
<td>Open the last tab again</td>
</tr>
<tr>
<td>replace</td>
<td>Show the find and replace bar for the current document, with focus on the replace with expression</td>
</tr>
<tr>
<td>replaceall</td>
<td>Replace all matches for Find with Replace With in the find and replace bar for the current document</td>
</tr>
<tr>
<td>replacenext</td>
<td>Replace the next match for Find with Replace With in the find and replace bar for the current document</td>
</tr>
<tr>
<td>replaceprevious</td>
<td>Replace the previous match for Find with Replace With in the find and replace bar for the current document</td>
</tr>
<tr>
<td>replaymacro</td>
<td>Replay previously recorded keystrokes</td>
</tr>
<tr>
<td>rerunInitScript</td>
<td>Rerun your initialization script</td>
</tr>
<tr>
<td>restartc9</td>
<td>Restart the environment</td>
</tr>
<tr>
<td>resume</td>
<td>Resume the current paused process</td>
</tr>
<tr>
<td>revealtab</td>
<td>Show the current tab in the file tree</td>
</tr>
<tr>
<td>reverttosaved</td>
<td>Reset the current file to its last saved version</td>
</tr>
<tr>
<td>reverttosavedall</td>
<td>Reset each open file to its saved version</td>
</tr>
<tr>
<td>run</td>
<td>Run or debug the current application</td>
</tr>
<tr>
<td>runlast</td>
<td>Run or debug the last run file</td>
</tr>
<tr>
<td>save</td>
<td>Save the current file to disk</td>
</tr>
<tr>
<td>saveall</td>
<td>Save all unsaved files to disk</td>
</tr>
<tr>
<td>saveas</td>
<td>Save the current file to disk with a different file name</td>
</tr>
<tr>
<td>savePaneLayout</td>
<td>Save the current pane layout in the Window, Saved Layouts menu</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>savePaneLayoutAndCloseTabs</td>
<td>Save the current pane layout in the <strong>Window, Saved Layouts</strong> menu, and then close all open tabs</td>
</tr>
<tr>
<td>scrolldown</td>
<td>Scroll down in the current document</td>
</tr>
<tr>
<td>scrollup</td>
<td>Scroll up in the current document</td>
</tr>
<tr>
<td>searchinfiles</td>
<td>Show the find and replace bar for multiple files</td>
</tr>
<tr>
<td>selectall</td>
<td>Select all selectable content</td>
</tr>
<tr>
<td>selectdown</td>
<td>Include the next line down in the selection</td>
</tr>
<tr>
<td>selectleft</td>
<td>Include the next space to the left in the selection</td>
</tr>
<tr>
<td>selectlineend</td>
<td>Include the rest of the current line in the selection, starting from the cursor</td>
</tr>
<tr>
<td>selectlinestart</td>
<td>Include the beginning of the current line in the selection, up to the cursor</td>
</tr>
<tr>
<td>selectMoreAfter</td>
<td>Include more matching selections that are after the selection</td>
</tr>
<tr>
<td>selectMoreBefore</td>
<td>Include more matching selections that are before the selection</td>
</tr>
<tr>
<td>selectNextAfter</td>
<td>Include the next matching selection that is after the selection</td>
</tr>
<tr>
<td>selectNextBefore</td>
<td>Include the next matching selection that is before the selection</td>
</tr>
<tr>
<td>selectOrFindNext</td>
<td>Select or find the next matching selection</td>
</tr>
<tr>
<td>selectOrFindPrevious</td>
<td>Select or find the previous matching selection</td>
</tr>
<tr>
<td>selectpagedown</td>
<td>Include from the cursor down to the end of the current page in the selection</td>
</tr>
<tr>
<td>selectpageup</td>
<td>Include from the cursor up to the beginning of the current page in the selection</td>
</tr>
<tr>
<td>selectright</td>
<td>Include the next space to the right of the cursor in the selection</td>
</tr>
<tr>
<td>selecttoend</td>
<td>Include from the cursor down to the end of the current file in the selection</td>
</tr>
<tr>
<td>selectttolineend</td>
<td>Include from the cursor to the end of the current line in the selection</td>
</tr>
<tr>
<td>selectttolinenstart</td>
<td>Include from the beginning of the current line to the cursor in the selection</td>
</tr>
<tr>
<td>selecttomatching</td>
<td>Include from the cursor to the next matching symbol in the current scope</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>selecttostart</td>
<td>Include from the cursor up to the beginning of the current file in the selection</td>
</tr>
<tr>
<td>selectup</td>
<td>Include the next line up in the selection</td>
</tr>
<tr>
<td>selectVar</td>
<td>Select all instances of the variable, if the selection is a variable</td>
</tr>
<tr>
<td>selectwordleft</td>
<td>Include the next word to the left of the cursor in the selection</td>
</tr>
<tr>
<td>selectwordright</td>
<td>Include the next word to the right of the cursor in the selection</td>
</tr>
<tr>
<td>setIndentation</td>
<td>Set the indentation type (spaces or tabs) and length</td>
</tr>
<tr>
<td>sharedialog</td>
<td>Show the Share this environment dialog box</td>
</tr>
<tr>
<td>showimmediate</td>
<td>Show the Immediate tab</td>
</tr>
<tr>
<td>showinstaller</td>
<td>Show the AWS Cloud9 Installer dialog box</td>
</tr>
<tr>
<td>showoutput</td>
<td>Show the Output tab</td>
</tr>
<tr>
<td>showprocesslist</td>
<td>Show the Process List dialog box</td>
</tr>
<tr>
<td>showSettingsMenu</td>
<td>Show the Preferences tab</td>
</tr>
<tr>
<td>singleSelection</td>
<td>Clear all previous selections</td>
</tr>
<tr>
<td>smallerfont</td>
<td>Decrease the font size</td>
</tr>
<tr>
<td>sortlines</td>
<td>If multiple lines are selected, rearrange them into a sorted order</td>
</tr>
<tr>
<td>splitIntoLines</td>
<td>Add a cursor at the end of the current line</td>
</tr>
<tr>
<td>splitline</td>
<td>Move the contents of the cursor to the end of the line, to its own line</td>
</tr>
<tr>
<td>stepinto</td>
<td>Step into the function that is next on the stack</td>
</tr>
<tr>
<td>stepout</td>
<td>Step out of the current function scope</td>
</tr>
<tr>
<td>stepover</td>
<td>Step over the current expression on the stack</td>
</tr>
<tr>
<td>stop</td>
<td>Stop running or debugging the current application</td>
</tr>
<tr>
<td>stopbuild</td>
<td>Stop building the current file</td>
</tr>
<tr>
<td>stripws</td>
<td>Remove whitespace from the end of each line</td>
</tr>
<tr>
<td>suspend</td>
<td>Suspend running or debugging the current application</td>
</tr>
<tr>
<td>switchterminal</td>
<td>Switch between the editor and the Terminal tab</td>
</tr>
<tr>
<td>syntax</td>
<td>Set the syntax type</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tab0</td>
<td>Go to the tenth tab</td>
</tr>
<tr>
<td>tab1</td>
<td>Go to the first tab</td>
</tr>
<tr>
<td>tab2</td>
<td>Go to the second tab</td>
</tr>
<tr>
<td>tab3</td>
<td>Go to the third tab</td>
</tr>
<tr>
<td>tab4</td>
<td>Go to the fourth tab</td>
</tr>
<tr>
<td>tab5</td>
<td>Go to the fifth tab</td>
</tr>
<tr>
<td>tab6</td>
<td>Go to the sixth tab</td>
</tr>
<tr>
<td>tab7</td>
<td>Go to the seventh tab</td>
</tr>
<tr>
<td>tab8</td>
<td>Go to the eighth tab</td>
</tr>
<tr>
<td>tab9</td>
<td>Go to the ninth tab</td>
</tr>
<tr>
<td>term_detach</td>
<td>Detach clients attached to the Terminal pane</td>
</tr>
<tr>
<td>term_help</td>
<td>Show help for the Terminal pane</td>
</tr>
<tr>
<td>term_restart</td>
<td>Restart the Terminal pane</td>
</tr>
<tr>
<td>threeleft</td>
<td>Create a three-pane layout with two panes on the left and one pane on the right</td>
</tr>
<tr>
<td>threeright</td>
<td>Create a three-pane layout with two panes on the right and one pane on the left</td>
</tr>
<tr>
<td>toggle_term_status</td>
<td>Show Terminal pane status, or hide if shown</td>
</tr>
<tr>
<td>toggleBlockComment</td>
<td>Surround the selection with block comment characters, or remove them if they are there</td>
</tr>
<tr>
<td>toggleButtons</td>
<td>Show tabs, or hide if shown</td>
</tr>
<tr>
<td>togglecomment</td>
<td>Add line comment characters at the start of each selected line, or remove them if they are there</td>
</tr>
<tr>
<td>toggleconsole</td>
<td>Show the Console window if hidden, or hide if shown</td>
</tr>
<tr>
<td>toggleddebugger</td>
<td>Show the Debugger window, or hide if shown</td>
</tr>
<tr>
<td>toggleFoldWidget</td>
<td>Fold code, or remove code folding if it is there</td>
</tr>
<tr>
<td>toggleMenubar</td>
<td>Show the menu bar, or hide if shown</td>
</tr>
<tr>
<td>toggleOpenfiles</td>
<td>Show the Open Files list in the Environment window, or hide if shown</td>
</tr>
<tr>
<td>toggleParentFoldWidget</td>
<td>Fold parent code, or remove folding if it is there</td>
</tr>
<tr>
<td>togglerecording</td>
<td>Start keystroke recording, or stop if it is already recording</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>toggletree</td>
<td>Show the <code>Environment</code> window if hidden, or hide if shown</td>
</tr>
<tr>
<td>toggleWordWrap</td>
<td>Wrap words, or stop wrapping words if they are already wrapping</td>
</tr>
<tr>
<td>tolowercase</td>
<td>Change the selection to all lowercase</td>
</tr>
<tr>
<td>touppercase</td>
<td>Change the selection to all uppercase</td>
</tr>
<tr>
<td>transposeletters</td>
<td>Transpose the selection</td>
</tr>
<tr>
<td>trimTrailingSpace</td>
<td>Trim whitespace at the ends of lines</td>
</tr>
<tr>
<td>twohsplit</td>
<td>Create a two-pane layout, with panes side by side</td>
</tr>
<tr>
<td>twovsplit</td>
<td>Create a two-pane layout, with panes top and bottom</td>
</tr>
<tr>
<td>undo</td>
<td>Undo the last action</td>
</tr>
<tr>
<td>unfold</td>
<td>Unfold selected code</td>
</tr>
<tr>
<td>unfoldall</td>
<td>Unfold code folding for the entire file</td>
</tr>
<tr>
<td>uploadLocalFiles</td>
<td>Show the <code>Upload Files</code> dialog box</td>
</tr>
<tr>
<td>vsplit</td>
<td>Split the current pane into two panes, top and bottom, and move the current tab to the top pane</td>
</tr>
</tbody>
</table>
Working with Other AWS Services

When using AWS Cloud9, you can work closely with Amazon Lightsail, AWS CodeStar, AWS Lambda, and AWS CodePipeline. The following topics describe how to do this.

Topics

- Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)
- Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 284)
- Working with AWS CodePipeline in the AWS Cloud9 Integrated Development Environment (IDE) (p. 322)

Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to work with code on Amazon Lightsail instances preconfigured with popular applications and frameworks such as WordPress, LAMP (Linux, Apache, MySQL, and PHP), Node.js, Nginx, Drupal, and Joomla, as well as Linux distributions such as Amazon Linux, Ubuntu, Debian, FreeBSD, and openSUSE.

Lightsail is the easiest way to get started with AWS for developers, small businesses, students, and other users who need a simple virtual private server (VPS) solution. Lightsail provides developers compute, storage, and networking capacity and capabilities to deploy and manage websites and web applications in the cloud. Lightsail includes everything you need to launch your project quickly—a virtual machine, SSD-based storage, data transfer, DNS management, and a static IP—for a low, predictable monthly price. For more information, see Amazon Lightsail Features.

In this topic, you create and set up a Linux-based Lightsail instance that is compatible with AWS Cloud9. You then create and connect an AWS Cloud9 SSH development environment to the Lightsail instance.

Note
Completing these procedures might result in charges to your AWS account. These include possible charges for services such as Lightsail. For more information, see Amazon Lightsail Pricing.

To create and set up a more advanced solution that includes a toolchain with the AWS Cloud9 IDE, source control, build, deployment, virtual servers or serverless resources, and more, skip the rest of this topic, and see Working with AWS CodeStar Projects (p. 283) instead.

To use the AWS Cloud9 IDE to work with an Amazon EC2 instance running Amazon Linux or Ubuntu Server that contains no sample code, skip the rest of this topic, and see Getting Started: Basic Tutorials (p. 35) instead.

- Step 1: Create a Linux-Based Lightsail Instance (p. 277)
Step 1: Create a Linux-Based Lightsail Instance

In this step, you use the Lightsail console to create an Amazon EC2 instance running an app in a Linux-based distribution. This instance automatically includes:

- A public and private IP address. (You can create a static public IP later.)
- Access to the instance using SSH over port 22, HTTP over port 80, and HTTPS over port 443. (You can change this.)
- A block storage disk. (You can attach additional disks later.)
- Built-in system reporting.

The Lightsail console enables you to back up, reboot, stop, or delete the instance later.


   We recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot sign in as an IAM administrator user, check with your AWS account administrator.

2. If prompted, choose the language to use in the console, and then choose **Save**.

3. If prompted, choose **Let's get started**.

4. On the home page, with the **Instances** tab already selected, choose **Create instance**.

5. For **Instance location**, be sure the location matches an AWS Region where you want to create the instance and where AWS Cloud9 is available. For more information, see AWS Cloud9 in the *Amazon Web Services General Reference*. To change the AWS Region, Availability Zone, or both, choose **Change AWS Region and Availability Zone**, and then follow the onscreen instructions.

6. For **Pick your instance image**, with Linux/Unix already chosen for **Select a platform**, and Apps + OS already chosen for **Select a blueprint**, choose a blueprint.
Step 2: Set up the Instance to Use It with AWS Cloud9

In this step, you connect to the running instance and then set it up so that AWS Cloud9 can use it later.

**Note**

The following instructions assume you chose Apps + OS in the previous step. If you chose OS Only and a distribution other than Ubuntu instead, you might need to adapt the following instructions accordingly.

1. With the Lightsail console still open from the previous step, on the Instances tab, in the card for the instance, choose the instance's name.
Step 2: Set up the Instance to Use It with AWS Cloud9

2. On the Connect tab, for Connect using your own SSH client, note the Public IP and User name values, as you will need them later.

3. Choose Connect using SSH.

4. Be sure that the instance has the latest system updates. To do this, in the terminal session that appears, run the command `sudo apt update`.

5. Check to see if Python is installed, and if it is, check to be sure the version is 2.7. To check the version, run the command `python --version`, and note the version number that appears. If no version number appears, or if the version is not 2.7, install Python 2.7 on the instance by running the command `sudo apt install -y python-minimal`.

6. Check to see if Node.js is installed, and if it is, check that the version is 0.6.16 or later. To check the version, run the command `node --version`, and note the version number that appears. If no version number appears, or the version is not 0.6.16 or later, we recommend you use Node Version Manager (nvm) to install Node.js on the instance.
To do this, run the following commands one at a time, in the following order, to update the instance, install Node Version Manager (nvm) on the instance, activate nvm on the instance, and then install the latest version of Node.js on the instance.

```
sudo apt update
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
. ~/.bashrc
nvm install node
```

7. Run the command `which node`, and note the value that appears. You will need it later.

   **Note**
   
   If the output of the command `which node` is something like `/usr/sbin/node`, AWS Cloud9 won't be able to find Node.js in that path. Instead, use nvm to install Node.js, as described in the previous step in this procedure. Then run the command `which node` again and note the new value that appears.

8. Download and run the AWS Cloud9 Installer (p. 482) on the instance.

---

### Step 3: Create and Connect to an AWS Cloud9 SSH Development Environment

In this step, you use the AWS Cloud9 console and the instance's terminal to create an SSH environment and then connect the environment to the running instance.

1. With the terminal session still open from the previous step, sign in to the AWS Cloud9 console, as follows:
   
   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to [https://console.aws.amazon.com/cloud9/](https://console.aws.amazon.com/cloud9/).
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
   - If you're using an AWS Educate Starter Account, see Step 2: Sign in to the AWS Cloud9 Console (p. 29) in Individual Student Signup.
   - If you're a student in a classroom, see your instructor for sign-in instructions.

   **Note**
   
   For this step, you will work with two different AWS services at the same time. If you signed in to the Lightsail console as an IAM administrator user, but you want a different entity to own the new SSH environment, we suggest opening a different web browser and signing in to the AWS Cloud9 console as that entity.

2. In the AWS Cloud9 console, choose the AWS Region that matches the one you created the instance in.
3. If a welcome page is displayed, for **New AWS Cloud9 environment**, choose **Create environment**. Otherwise, choose **Create environment**.

4. On the **Name environment** page, for **Name**, type a name for your environment.
5. To add a description to your environment, type it in **Description**.
6. Choose **Next step**.
7. On the **Configure settings** page, for **Environment type**, choose **Connect and run in remote server (SSH)**.
8. For **User**, type the **User name** value you noted earlier.
9. For **Host**, type the **Public IP** value you noted earlier.
10. For **Port**, leave the default value of **22**.
11. Expand **Advanced settings**.
12. For **Environment path**, type the path that AWS Cloud9 will start from after login, which is `~` (the root of the user's home directory).
13. For **Node.js binary path**, type the value of the command `which node` you noted earlier.
14. Leave **SSH jump host** blank.
Step 4: Use the AWS Cloud9 IDE to Change the Code on the Instance

15 Store the public SSH key that AWS Cloud9 creates for this environment in your system clipboard. To do this, choose **Copy key to clipboard**.

**Note**
To see the public SSH key value that was copied, expand **View public SSH key**.

16 Save the public SSH key value you just copied to the instance. To do this, use vi, a popular text editor, which is already installed on the instance:

a. In the terminal session for the instance, run the command `vi ~/.ssh/authorized_keys`.

b. In the vi editor that appears, go to the end of the file, and switch to insert mode. To do this, press G then A. (-- INSERT -- appears at the bottom of the vi editor.)

c. Add two carriage returns to the end of the file by pressing `Enter` twice.

d. Paste the contents of your system clipboard, which contains the public SSH key value you just copied, to the terminal session clipboard. To do this, in the bottom corner of the terminal session window, choose the clipboard button, then paste the contents of your system clipboard into the box.

e. Paste the contents of the terminal session clipboard into the vi editor. To do this, at the insertion point in the vi editor, press `Ctrl + Shift + V`.

f. Save the file. To do this, press `Esc` to enter command mode. (-- INSERT -- disappears from the bottom of the vi editor.) Type `:wq` (to write the file and then quit the vi editor), and then press `Enter`.

17 Back in the AWS Cloud9 console, choose **Next step**.

18 On the **Review choices** page, choose **Create environment**. Wait while AWS Cloud9 creates your environment and then displays the AWS Cloud9 IDE for the environment. This can take several minutes.

After AWS Cloud9 creates your environment, it displays the AWS Cloud9 IDE for the environment.

If AWS Cloud9 doesn’t display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see **Cannot Open an Environment (p. 544)** in **Troubleshooting**.

**Step 4: Use the AWS Cloud9 IDE to Change the Code on the Instance**

Now that the IDE appears for the new environment, you can use the terminal session in the IDE instead of the Lightsail terminal session. The IDE provides a rich code editing experience with support for several programming languages and runtime debuggers, as well as color themes, shortcut keybindings, programming language-specific syntax coloring and code formatting, and more.

To learn how to use the IDE, see **Tour the AWS Cloud9 IDE (p. 100)**.
To learn how to change the code on your instance, we recommend the following resources.

- **All** Getting the application password for your 'powered by Bitnami' Lightsail image on the Lightsail website
- **Drupal**: Bitnami Drupal For AWS Cloud on the Bitnami website, and Tutorials and site recipes on the Drupal website
- **GitLab CE**: Bitnami GitLab CE for AWS Cloud on the Bitnami website, and GitLab Documentation on the GitLab website
- **Joomla**: Bitnami Joomla! For AWS Cloud on the Bitnami website, and Getting Started with Joomla! on the Joomla! website
- **LAMP Stack**: Bitnami LAMP for AWS Cloud on the Bitnami website
- **Magento**: Bitnami Magento For AWS Cloud on the Bitnami website, and the Magento User Guide on the Magento website
- **MEAN**: Bitnami MEAN For AWS Cloud on the Bitnami website
- **Nginx**: Bitnami Nginx For AWS Cloud on the Bitnami website, and the NGINX Wiki on the NGINX website
- **Node.js**: Bitnami Node.Js For AWS Cloud on the Bitnami website, and the Getting Started Guide on the Node.js website
- **Plesk Hosting Stack on Ubuntu**: Set up and configure Plesk on Lightsail on the Lightsail website
- **Redmine**: Bitnami Redmine For AWS Cloud on the Bitnami website, and Getting Started on the Redmine website
- **WordPress**: Getting started using WordPress from your Amazon Lightsail instance on the Lightsail website, and Bitnami WordPress For AWS Cloud on the Bitnami website

### Working with AWS CodeStar Projects in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to work with code in AWS CodeStar projects.

AWS CodeStar is a cloud-based service for creating, managing, and working with software development projects on AWS. You can quickly develop, build, and deploy applications on AWS with an AWS CodeStar project. An AWS CodeStar project creates and integrates AWS services for your project development toolchain. Depending on your choice of AWS CodeStar project template, that toolchain might include source control, build, deployment, virtual servers or serverless resources, and more. For more information, see the AWS CodeStar User Guide.

**Note**
Completing these procedures might result in charges to your AWS account. These include possible charges for services such as Amazon EC2, AWS CodeStar, and AWS services supported by AWS CodeStar. For more information, see Amazon EC2 Pricing, AWS CodeStar Pricing, and Cloud Services Pricing.

To use the AWS Cloud9 IDE to work with a newly-launched Amazon EC2 instance preconfigured with a popular app or framework such as WordPress, MySQL, PHP, Node.js, Nginx, Drupal, or Joomla, or a Linux distribution such as Ubuntu, Debian, FreeBSD, or openSUSE, you can use Amazon Lightsail along with AWS Cloud9. To do this, skip the rest of this topic, and see Working with Amazon Lightsail Instances (p. 276) instead.

To use the AWS Cloud9 IDE to work with a newly-launched Amazon EC2 instance running Amazon Linux that contains no sample code, skip the rest of this topic, and see Getting Started: Basic Tutorials (p. 35) instead.

- **Step 1**: Prepare to Work with AWS CodeStar Projects (p. 284)
- **Step 2**: Create a Project in AWS CodeStar (p. 284)
Step 1: Prepare to Work with AWS CodeStar Projects

In this step, you create an AWS CodeStar service role and an Amazon EC2 key pair, so that you can begin creating and working with AWS CodeStar projects.

If you have used AWS CodeStar before, skip ahead to Step 2: Create a Project in AWS CodeStar (p. 284).

For this step, follow the instructions in Setting Up AWS CodeStar in the AWS CodeStar User Guide. Do not create a new AWS account, IAM user, or IAM group as part of those instructions. Use the ones you created or identified in Team Setup for AWS Cloud9 (p. 8). When you finish following those instructions, return to this topic.

Step 2: Create a Project in AWS CodeStar

In this step, you create a project in AWS CodeStar.

If you already have a project in AWS CodeStar you want to use, skip ahead to Step 3: Create an AWS Cloud9 Development Environment and Connect It to the Project (p. 284).

For this step, follow the instructions in Create a Project in AWS CodeStar in the AWS CodeStar User Guide. In the AWS CodeStar create project wizard, when you get to the Set up tools page or Connect to your source repository page, choose Skip, and then return to this topic.

Step 3: Create an AWS Cloud9 Development Environment and Connect It to the Project

In this step, you create an AWS Cloud9 development environment in the AWS CodeStar or AWS Cloud9 consoles. You then connect the new environment to an AWS CodeStar project.

For this step, follow one of the following sets of instructions, depending on the AWS Cloud9 development environment type you want to use and the type of repository where the AWS CodeStar project stores its code.

<table>
<thead>
<tr>
<th>Environment type</th>
<th>Repository type</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2 environment</td>
<td>CodeCommit</td>
<td>Create an AWS Cloud9 Environment for a Project in the AWS CodeStar User Guide</td>
</tr>
<tr>
<td>SSH environment</td>
<td>CodeCommit</td>
<td>AWS CodeCommit Sample (p. 356)</td>
</tr>
<tr>
<td>EC2 or SSH environment</td>
<td>GitHub</td>
<td>Use GitHub with AWS Cloud9 in the AWS CodeStar User Guide</td>
</tr>
</tbody>
</table>

Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to work with AWS Lambda functions and their related Amazon API Gateway APIs in an AWS Cloud9 development environment. For example, you can:
Prepare to Work with Lambda Functions

Before you can work with Lambda functions in the AWS Cloud9 IDE, you must complete the following steps:

**Topics**
- Step 1: Set Up Your IAM Group with Required Access Permissions (p. 286)
- Step 2: Set Up Your Environment with Your AWS Access Credentials (p. 287)
- Step 3: Create an Execution Role for Your Lambda Functions (p. 287)
- Step 4: Set Your Environment to the Correct AWS Region (p. 288)
- Step 5: Open the Lambda Section of the AWS Resources Window (p. 288)
Step 1: Set Up Your IAM Group with Required Access Permissions

If your AWS access credentials are associated with an IAM administrator user in your AWS account, and you want to use that user to work with Lambda functions, skip ahead to Step 2: Set Up Your Environment with Your AWS Access Credentials (p. 287).

Otherwise, complete the following instructions to use the IAM console to attach the AWS managed policies named AWSLambdaFullAccess, AmazonAPIGatewayAdministrator, AmazonAPIGatewayInvokeFullAccess, and an additional inline policy, to an IAM group to which your user belongs.

1. Sign in to the AWS Management Console, if you're not already signed in.
   For this step, we recommend you sign in using credentials for an IAM administrator in your AWS account. If you can't do this, check with your AWS account administrator.
2. Open the IAM console. To do this, in the console's navigation bar, choose Services. Then choose IAM.
3. Choose Groups.
4. Choose the group's name.
5. On the Permissions tab, for Managed Policies, choose Attach Policy.
6. In the list of policy names, choose the boxes next to AWSLambdaFullAccess, AmazonAPIGatewayAdministrator, and AmazonAPIGatewayInvokeFullAccess. (If you don't see any of these policy names in the list, type the policy name in the Search box to display it.)
7. Choose Attach Policy.
9. Open the AWS CloudFormation console. To do this, in the console's navigation bar, choose Services. Then choose CloudFormation.
10. Choose Create Stack.
11. On the Select Template page, for Choose a template, choose Upload a template to Amazon S3.
   Choose Browse, and then choose the AWS CloudFormation template file that you just extracted.
12. Choose Next.
13. On the Specify Details page, for Stack name, type a name for the stack (for example AWSCloud9LambdaAccessStack. If you type a different name, replace it throughout this procedure).
14. For Parameters, for GroupName, type the name of the existing group in your AWS account you want to attach the access policy to.
15. Choose Next.
16. On the Options page, choose Next. (Do not change any of the default settings on the Options page.)
17. On the Review page, choose I acknowledge that AWS CloudFormation might create IAM resources.
18. Choose Create.

Wait until the AWSCloud9LambdaAccessStack stack shows CREATE_COMPLETE. This might take a few moments. Please be patient.

Note
The access policy that AWS CloudFormation attaches to the group is named AWSCloud9LambdaGroupAccess and has the following definition, where ACCOUNT_ID is your AWS account ID.

```json
{
  "Version": "2012-10-17",
```
"Statement": [
{
  "Action": [
    "cloudformation:CreateChangeSet",
    "cloudformation:CreateStack",
    "cloudformation:DescribeChangeSet",
    "cloudformation:DescribeStackEvents",
    "cloudformation:DescribeStacks",
    "cloudformation:ExecuteChangeSet",
    "cloudformation:ListStackResources",
    "cloudformation:UpdateStack",
    "iam:AttachRolePolicy",
    "iam:DetachRolePolicy",
    "iam:GetRole",
    "iam:GetUser",
    "iam:PassRole"
  ],
  "Resource": "*",
  "Effect": "Allow"
},
{
  "Action": [
    "iam:CreateRole",
    "iam:DeleteRole"
  ],
  "Resource": "arn:aws:iam::ACCOUNT_ID:role/cloud9-***",
  "Effect": "Allow"
}
]}

---

**Step 2: Set Up Your Environment with Your AWS Access Credentials**

The AWS Cloud9 IDE uses the AWS Command Line Interface (AWS CLI) in your AWS Cloud9 development environment to interact with Lambda and other supporting AWS services. Therefore, the AWS CLI in your environment needs access to your AWS access credentials.

Do one of the following to set up the AWS CLI in your environment:

- If you have an EC2 environment, AWS managed temporary credentials are already set up in your environment for the AWS CLI to use, and you can skip ahead to Step 3: Create an Execution Role for Your Lambda Functions (p. 287). AWS managed temporary credentials have permission to interact with most AWS services from your environment (provided the AWS entity that is using the environment also has those permissions). For more information, see AWS managed temporary credentials (p. 536).

- If you have an EC2 environment but AWS managed temporary credentials don't meet your needs, you can attach an IAM instance profile to the Amazon EC2 instance that connects to your environment. Or you can store your permanent AWS access credentials within the environment. For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71) or Create and Store Permanent Access Credentials in an Environment (p. 75).

- If you have an SSH environment, you can store your permanent AWS access credentials within the environment. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

---

**Step 3: Create an Execution Role for Your Lambda Functions**

If you want your Lambda functions to do things using AWS resources, you must specify an IAM role (execution role) that contains the necessary access permissions for your functions to use.
When you create a Lambda function, AWS Cloud9 can create an execution role for you. This execution role contains the permissions as described in Basic Lambda Permissions in the AWS Lambda Developer Guide.

If this execution role doesn't meet your needs, you must create an execution role on your own before you create your Lambda function. For more information, see the following:

- AWS Lambda Permissions Model in the AWS Lambda Developer Guide
- Creating a Role to Delegate Permissions to an AWS Service in the IAM User Guide

### Step 4: Set Your Environment to the Correct AWS Region

You must set your AWS Cloud9 development environment to use the AWS Region where you want to create new Lambda functions in your AWS account, or where you want to import existing Lambda functions from your AWS account into your AWS Cloud9 development environment.

To do this:

1. In the AWS Cloud9 IDE, on the menu bar, choose AWS Cloud9, Preferences.
2. In the navigation pane of the Preferences tab, choose AWS Settings.
3. For AWS Region, select the AWS Region you want to use.

### Step 5: Open the Lambda Section of the AWS Resources Window

Now you’re ready to begin using the AWS Cloud9 IDE to work with Lambda functions. To do this, expand the Lambda section of the AWS Resources window, if it isn’t already expanded.
If the AWS Resources window isn’t visible, choose the AWS Resources button.

If you don’t see the AWS Resources button, choose Window, AWS Resources on the menu bar to show it.

Create a Lambda Function With the Create Serverless Application Wizard

You can use the AWS Cloud9 IDE to create a new Lambda function. If you already have a Lambda function in your AWS account for the AWS Region you set earlier, skip ahead to Import a Lambda Function (p. 295).

Note
This procedure describes how to use the Create serverless application wizard to create a single Lambda function based on function blueprints that are owned by AWS. To create multiple Lambda functions at the same time, Lambda functions along with supporting components at the same time, or Lambda functions that are owned by entities other than AWS, skip ahead to Create and Deploy Lambda Functions with the AWS Serverless Application Repository (p. 293).

1. In the Lambda section of the AWS Resources window, choose where you want to create the function:
   - To create a single function by itself, choose the Local Functions heading.
   - To create a function and then add it to an existing group of one or more functions and related AWS resources (referred to as a serverless application), in the Local Functions list, choose the serverless application for the group (represented by the Lambda icon inside of a folder).

2. Do one of the following:
   - Choose Create a new Lambda function (the button with the Lambda icon).
   - Right-click the Local Functions heading or the serverless application folder you chose earlier, and then choose Create Here.

3. In the Create serverless application dialog box, specify the following settings for the function:
   - Function Name: A name for the function.
   - Application Name: The name of the new serverless application to be associated with the new function.

      Important
      Both of these names must contain only alphanumeric characters and hyphens. Each name must start with an alphabetic character, and each name must not exceed 128 characters.

4. Choose Next.
5. Choose the function blueprint you want to start with. (Currently, only Node.js and Python function blueprints are available.)

To show blueprints for a specific runtime, for Select Runtime, choose the runtime. For example, to use the hello-world function blueprint for Node.js 6.10, choose Node.js 6.10 for Select Runtime, and then choose the hello-world blueprint for Select Blueprint.

6. Choose Next.

7. Do one of the following:
   - To skip having an AWS service automatically trigger this function, leave Function Trigger set to none, choose Next, and then skip ahead to step 9 in this procedure.
   - To have an AWS resource in your account automatically trigger your function, for Function Trigger, select the name of the AWS service that will contain the resource. (Currently, only API Gateway is available.)

8. If you chose API Gateway for Function Trigger, specify the following for Trigger Settings:
   - For Resource Path, type the URL portion of the API to use to invoke the function. For example, type / to specify the resource root.
   - For Security, choose the security mechanism for the API endpoint:
     - AWS_IAM: Require that callers provide IAM access credentials to be authenticated. See Control Access to API Gateway with IAM Permissions in the API Gateway Developer Guide.
     - NONE: Enable open access.
     - NONE_KEY: Require that callers provide an API key to be authenticated. See Set Up API Keys Using the API Gateway Console in the API Gateway Developer Guide.

9. Choose Next.

10. For Memory (MB), choose the amount of memory, in megabytes, that this function will use.

11. Do one of the following:
   - To have AWS Cloud9 create a new, basic IAM role (execution role) for this function to use, for Role, choose Automatically generate role. Then choose Next.
   - To have Lambda use an existing IAM role (execution role) in your AWS account, for Role, choose Choose an existing role. For Existing Role, choose the name of the role, and then choose Next.

12. Choose Next.


Compare your results to the following:
In the **Lambda** section of the **AWS Resources** window, AWS Cloud9 does the following:

- If you chose to create a single function by itself:
  1. AWS Cloud9 creates a serverless application with the name that you specified earlier. Then it adds a serverless application (represented by a Lambda icon inside of a folder) to the **Local Functions** list. Then it adds the Lambda function (represented by a Lambda icon by itself), to this serverless application.
  2. AWS Cloud9 creates a remote version of the function in Lambda and adds it to the **Remote Functions** list. AWS Cloud9 gives the remote version a different name. For example, if you named the serverless application `myDemoServerlessApplication` and the function `myDemoFunction`, the remote version name of your function would be `cloud9-myDemoServerlessApplication-myDemoFunction-RANDOM_ID`, where `RANDOM_ID` is a randomly determined ID.
  3. If you chose to have API Gateway automatically trigger the function, AWS Cloud9 creates an API in API Gateway with a name that corresponds to the function. For example, if you named the function
myDemoFunction, the API name would be cloud9-myDemoFunction. AWS Cloud9 uses the value you specified in Resource Path to map the function to the API using the ANY method.

- If you chose to create a single function and then add it to an existing serverless application:
  1. AWS Cloud9 adds the Lambda function (represented by a Lambda icon by itself), to the existing serverless application (represented by a Lambda icon inside of a folder).
  2. AWS Cloud9 creates a remote version of the function in Lambda and adds it to the Remote Functions list. AWS Cloud9 gives the remote version a different name. For example, if you named the function myDemoFunction and added it to a serverless application named myDemoServerlessApplication, the remote version name would be cloud9-myDemoServerlessApplication-myDemoFunction-RANDOM_ID, where RANDOM_ID is a randomly determined ID.
  3. If you chose to have API Gateway automatically trigger your function, AWS Cloud9 creates an API in API Gateway with a name that corresponds to the related serverless application, if it doesn’t already exist. For example, if the serverless application is named myDemoServerlessApplication, the API name would be cloud9-myDemoServerlessApplication. AWS Cloud9 uses the value you specified in Resource Path to map the function to the API using the ANY method.

In the Environment window, AWS Cloud9 does the following:

- If you chose to create a single function by itself, AWS Cloud9 creates a folder with the same name as the serverless application and puts this folder in the root of the AWS Cloud9 development environment. AWS Cloud9 then adds the following files to the folder:
  - .application.json: A hidden file used by AWS Cloud9 that contains JSON-formatted settings specific to the serverless application.
  - .gitignore: A hidden file that contains a list of files Git ignores, if you want to use Git to manage your source code for this function.
  - template.yaml: An AWS SAM template file that contains information about the Lambda function and any other related supported AWS resources. Whenever you update the local version of your function and then upload it to Lambda, AWS Cloud9 calls AWS SAM to use this template file to do the upload. For more information, see Using the AWS Serverless Application Model (AWS SAM) in the AWS Lambda Developer Guide.

  **Note**
  You can edit this file to create additional supporting AWS resources for your function. For more information, see the AWS Serverless Application Model (AWS SAM) repository on GitHub.

  - A subfolder with the same name as the function, containing a code file representing the function logic.
  - If the function uses Python, additional subfolders and files are added to the preceding subfolder to enable Python debugging:
    - .debug: A subfolder that contains Python modules and files for debugging purposes.
    - venv: A standard Python virtualenv folder. This includes a module named ikpdb, which AWS Cloud9 uses to debug Python applications.
    - requirements.txt: A standard file for installing Python modules.
    - AWS Cloud9 also adds a CodeUri property to the template.yaml file and sets this property to reference the .debug/ folder.

  If you chose to create a single function and then add it to an existing serverless application, AWS Cloud9 does the following to the folder that represents the serverless application:

  - Updates the template.yaml file previously described to include information about the Lambda function and any other related supported AWS resources.
  - A subfolder with the same name as the function, containing a code file representing the function logic.
• If the function uses Python, additional subfolders and files are added to the preceding subfolder to enable Python debugging:
  • .debug: A subfolder that contains Python modules and files for debugging purposes.
  • venv: A standard Python virtualenv folder. This includes a module named ikpdb, which AWS Cloud9 uses to debug Python applications.
  • __init__.py: A standard Python package initialization file.
  • requirements.txt: A standard file for installing Python modules.
  • AWS Cloud9 also adds a CodeUri property to the template.yaml file and sets this property to reference the .debug/ folder.

The .application.json and .gitignore files (and the .debug folder for Python) are hidden. To show hidden files or hide them if they're shown, in the Environment window, choose the gear icon, and then choose Show Hidden Files.

To invoke the function, see Invoke a Lambda Function (p. 297). If the function has a related API in API Gateway, to invoke the API, see Invoke an API Gateway API for a Related Lambda Function (p. 303).

Create and Deploy Lambda Functions with the AWS Serverless Application Repository

You can use the AWS Cloud9 IDE and the AWS Serverless Application Repository to create multiple Lambda functions at the same time, Lambda functions along with supporting components at the same time, or Lambda functions that are owned by entities other than AWS. If you already have Lambda functions in your AWS account for the AWS Region you set earlier, skip ahead to Import a Lambda Function (p. 295).

1. In a separate web browser tab, open the AWS Serverless Application Repository.
2. Find the serverless application you want to create, and then choose the title of the serverless application that you want inside of its card. (If the card isn't visible, begin typing information about the serverless application that you want in the Search applications by name, description, or labels box to show it.)
3. On the Application details page that appears, if a URL for a Git-based repository is displayed, copy that URL (for example, https://github.com/USER_NAME/REPOSITORY_NAME).

   Note
   If a URL isn't displayed, try choosing the Deploy button on the Application details page, and then look for a Source code URL value.
4. Back in the AWS Cloud9 IDE, open a terminal, if one isn't already open. (To open a terminal, on the menu bar, choose Window, New Terminal.)
5. In the terminal, change to the directory in your environment where you want to copy the serverless application's starter files (for example, by running the command `cd ~/environment`).

6. Run the command `git clone`, followed by the Git URL you copied earlier (for example, `git clone https://github.com/USER_NAME/REPOSITORY_NAME`). The IDE then adds the serverless application's functions to the Lambda section of the AWS Resources window.

   **Note**
   Running the `git clone` command with some of the URLs in the Application details pages or Source code URL values might not work as expected or might produce unexpected results. Alternatively, you can manually download the files you want from the desired repository to your local workstation. Then manually upload those files to the IDE by running File, Upload Local Files on the menu bar.

   When you clone the GitHub repository, the IDE uses the AWS Serverless Application Model (AWS SAM) template file in the repository to determine which of the serverless application's functions to display in the Lambda section of the AWS Resources window. The AWS SAM template file must follow the AWS Serverless Application Model (AWS SAM) file format. If the repository doesn't contain an AWS SAM template file, or if the file doesn't follow the AWS SAM file format, the IDE won't display those functions. You also won't be able to run, debug, or deploy those functions or any of their associated API Gateway resources from the Lambda section of the AWS Resources window.

7. You might need to complete some setup before you can run, debug, or deploy the serverless application from the IDE as expected. For setup instructions, see the Application details page that you opened earlier. Or look for any setup instructions within the serverless application's files that you cloned to your IDE.

To invoke the functions, see Invoke a Lambda Function (p. 297). If the functions have related APIs in API Gateway, to invoke the APIs, see Invoke an API Gateway API for a Related Lambda Function (p. 303). When you invoke a function or API this way for the first time, AWS Cloud9 adds a hidden `.application.json` file to the serverless application's component files. This file is used by AWS Cloud9 and contains JSON-formatted settings that are specific to the serverless application.

If the serverless application requires parameters to be specified during deployment, you can deploy it from the IDE only by using the terminal. To see if parameters are required, on the Application details page you opened earlier, choose the Deploy button, and then see the Configure application parameters card for any parameters. If there are any parameters, deploy the serverless application from the terminal in the IDE by running the AWS CloudFormation deploy command, for example:

```
aws cloudformation deploy --template-file TEMPLATE_FILE_PATH --parameter-overrides "PARAMETER_KEY_1=PARAMETER_VALUE_1" "PARAMETER_KEY_N=PARAMETER_VALUE_N" --region REGION_ID
```

In the preceding command:

- `TEMPLATE_FILE_PATH` represents the path to the AWS SAM template file.
- `PARAMETER_KEY_1` represents the name of the first parameter.
- `PARAMETER_VALUE_1` represents the value of the first parameter.
- `PARAMETER_KEY_N` represents the name of an additional parameter, and so on.
- `PARAMETER_VALUE_N` represents the value of an additional parameter, and so on.
- `REGION_ID` represents the ID of the AWS Region where you want to deploy the serverless application (for example, `us-east-2`).
- Additional options might need to be specified, depending on the serverless application's requirements. For more information, see the Application details page that you opened earlier, or look for any setup instructions within the serverless application's files that you cloned to your IDE.
If you try to use the Lambda section of the AWS Resources window to deploy a serverless application that requires parameters, a message is displayed that required parameters are missing, and the serverless application is not deployed.

**Import a Lambda Function**

If you have an existing Lambda function in your AWS account but not in your AWS Cloud9 development environment, you must import it before you can work with it in your environment.

**Note**

If the Lambda function is part of an existing AWS CodeStar project, and the environment was created from within the project in the AWS CodeStar console, the function is already imported, so you do not need to import it again.

To confirm this behavior, look in the Local Functions list in the Lambda section of the AWS Resources window for a serverless application (represented by a Lambda icon inside of a folder) with the same name as the AWS CodeStar project, containing a Lambda function (represented by a Lambda icon by itself) with the function's base name. Look also in the Remote Functions list for a Lambda function with a name in the format `awscodestar-PROJECT_NAME-lambda-BASE_FUNCTION_NAME-RANDOM_ID`, where `PROJECT_NAME` is the AWS CodeStar project name, `BASE_FUNCTION_NAME` is the function's base name, and `RANDOM_ID` is a randomly determined ID.

Since the function is already imported, we do not recommend that you import the remote version of the function in the AWS CodeStar project. Otherwise, you will have two versions of the same function code in your Environment window but with different folder names, which could be confusing.

To import a Lambda function, do the following:

1. In the Environment window, choose where you want to import the function.
2. In the Lambda section of the AWS Resources window, choose the function's name in the Remote Functions list.

   **Note**

   If you don't see your function in the Remote Functions list, choose the Refresh functions List button (the button with the circular arrow icon).

3. Do one of the following:
   - Double-click the function you just chose.
   - On the menu bar in the Lambda section, choose the Import the selected Lambda function button (the button with the arrow that faces down).
   - Right-click the function you just chose, and then choose Import.
**Note**
You cannot import a Lambda function into a folder that represents either a serverless application or a Lambda function. If you try to do this, AWS Cloud9 will display a message that it will import the Lambda function into the environment’s root location instead. To let AWS Cloud9 do this, choose **Import**. Otherwise, choose **Cancel**, choose a different folder to import the function (or create a new empty folder to import the function into), and then restart this procedure from the beginning.

4. When prompted to finish importing the function, choose **OK**.

AWS Cloud9 imports your function into a new folder in the root of your environment. (AWS Cloud9 also adds the function to the **Local Functions** list in the **Lambda** section of the **AWS Resources** window.) This folder has the same name as the function. AWS Cloud9 adds the following files to this folder:

- **.application.json**: A hidden file used by AWS Cloud9 that contains JSON-formatted settings specific to the function.
- **.gitignore**: A hidden file that contains a list of files Git ignores, if you want to use Git to manage your source code for this function.
- **template.yaml**: A file for AWS Cloud9 internal use.

**Note**
Although the **template.yaml** file is expressed in AWS SAM format, it isn't used by AWS SAM. Therefore, you cannot edit this file to create additional supporting AWS resources for your function. Do not modify this file.

- One or more code files containing the function logic.

The **.application.json** and **.gitignore** files are hidden. To display or hide hidden files, in the **Environment** window, choose the gear icon, and then choose **Show Hidden Files**.

To invoke the function, see **Invoke a Lambda Function (p. 297)**.
Invoke a Lambda Function

To invoke an existing Lambda function, you must first import the remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. To do this, see Import a Lambda Function (p. 295).

1. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn't already expanded.
2. Expand the serverless application folder that contains the function that you want to invoke.
3. Choose the function that you want to invoke, right-click it, and then choose Run.
4. Do one of the following:
   • To run the local version of the function within your environment, choose Run Local.
   • To run the remote version of the function within Lambda, choose Run Remote.

5. In the Test payload pane of the invoke tab that is displayed, confirm any custom input data you want your function to use when you test it. For information about the input data format, see Step 2.2: Invoke the Lambda Function Manually and Verify Results, Logs, and Metrics in the AWS Lambda Developer Guide.

6. In the invoke tab, choose the Run button.
Note
After you run the function for the first time, a `lambda-payloads.json` file is added to the function’s related serverless application folder in the Environment window. This file contains the contents of the custom input data.

If you invoke an existing Lambda function and then try to invoke the same function code for its related API in API Gateway with the same custom input data, you might get an error or the code might not run as expected. For more information, see Response Differences When Invoking a Lambda Function from API Gateway (p. 305).

The invoke tab contains two panes:

- The Test payload pane displays any custom input data that was supplied for the function.
- The Execution results pane displays any output from the function and some information from the related Amazon CloudWatch Logs for the function.

Compare your results to the following:
Invoke a Lambda Function that Uses Environment Variables

To invoke a Lambda function locally or remotely that uses environment variables, do one of the following.

Invoke a Lambda function locally with predefined environment variables and their values

Add the predefined environment variable definitions to the function's template.yml file if they are not already there, and then run the Run Local command for that function from the Lambda section of the AWS Resources window.

For example, given the following Node.js function definition in a file named index.js:

```javascript
exports.handler = (event, context, callback) => {
    var result = {
        MY_ENV_VAR_1: process.env.MY_ENV_VAR_1,
        MY_ENV_VAR_2: process.env.MY_ENV_VAR_2
    };

    const response = {
        statusCode: 200,
        headers: { "Content-type": "application/json" },
        body: result
    };

    callback(null, response);
};
```

And given the following addition to the function's template.yml file one directory above index.js:
Invoke a Lambda Function that Uses Environment Variables

Resources:
MyEnvVarsFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    Environment:
      Variables:
        MY_ENV_VAR_1: "This is my environment variable 1 value from the template.yaml file."
        MY_ENV_VAR_2: "This is my environment variable 2 value from the template.yaml file."

And running the Run Local command for that function from the Lambda section of the AWS Resources window, the following is output:

```
{
  "statusCode": 200,
  "headers": {
    "Content-type": "application/json"
  },
  "body": {
    "MY_ENV_VAR_1": "This is my environment variable 1 value from the template.yaml file."
    "MY_ENV_VAR_2": "This is my environment variable 2 value from the template.yaml file."
  }
}
```

Invoke a Lambda function locally with environment variables and their values that the caller provides dynamically at run time

You cannot use the Run Local command for such a function from the Lambda section of the AWS Resources window.

Instead, add the predefined environment variable definitions to the function's template.yaml file if they are not already there. Then use the terminal in the IDE to run the AWS SAM CLI with the invoke command along with the --env-vars argument and a separate JSON file that contains the environment variable definitions and their values.

For example, given the following Node.js function named MyEnvVarsFunction with its definition in a file named index.js:

```javascript
exports.handler = (event, context, callback) => {

  var result = {
    MY_ENV_VAR_1: process.env.MY_ENV_VAR_1,
    MY_ENV_VAR_2: process.env.MY_ENV_VAR_2
  };

  const response = {
    statusCode: 200,
    headers: { "Content-type": "application/json" },
    body: result
  };

  callback(null, response);
};
```

And given the following addition to the function's template.yaml file one directory above index.js:
Invoke a Lambda Function that Uses Environment Variables

Resources:
MyEnvVarsFunction:
  Type: 'AWS::Serverless::Function'
  Properties:
    ... Environment:
      Variables:
        MY_ENV_VAR_1: "This is my environment variable 1 value from the template.yaml file."
        MY_ENV_VAR_2: "This is my environment variable 2 value from the template.yaml file."

And given the following file named MyEnvVars.json in the same directory as index.js:

```json
{
  "MyEnvVarsFunction": {
    "MY_ENV_VAR_1": "This is my environment variable 1 value from the JSON file.",
    "MY_ENV_VAR_2": "This is my environment variable 2 value from the JSON file."
  }
}
```

And running the following AWS SAM CLI command from the same directory as index.js:

```
sam local invoke --event lambda-payloads.json --template ../template.yaml --env-vars MyEnvVars.json
```

The following is output:

```json
{
  "statusCode": 200,
  "headers": {
    "Content-type": "application/json"
  },
  "body": {
    "MY_ENV_VAR_1": "This is my environment variable 1 value from the JSON file.",
    "MY_ENV_VAR_2": "This is my environment variable 2 value from the JSON file."
  }
}
```

### Invoke a Lambda function remotely that already has predefined environment variable and their values

Run the Run Remote command for that function from the Lambda section of the AWS Resources window.

To view or change predefined environment variable values for an existing remote function before you invoke it remotely, see AWS Lambda Environment Variables in the AWS Lambda Developer Guide.

### Invoke a Lambda function remotely with environment variables and their values that the caller provides dynamically at run time

You cannot use the Run Remote command for such a function from the Lambda section of the AWS Resources window.

Instead, you can use the terminal in the IDE to run the AWS CLI with the lambda command's `update-function-configuration` action and the `--environment` argument with the environment variable definitions and their values. Then run the lambda command's `invoke` action.
For example, given the following Node.js function named `MyEnvVarsFunction` with the following definition and with predefined environment variables named `MY_ENV_VAR_1` and `MY_ENV_VAR_2`:

```javascript
exports.handler = (event, context, callback) => {
    var result = {
        MY_ENV_VAR_1: process.env.MY_ENV_VAR_1,
        MY_ENV_VAR_2: process.env.MY_ENV_VAR_2
    }

    const response = {
        statusCode: 200,
        headers: { "Content-type": "application/json" },
        body: result
    }

    callback(null, response);
}
```

And running the following AWS CLI commands, one at a time in the following order:

```bash
aws lambda update-function-configuration --function-name MyEnvVarsFunction --environment "{"Variables":{"MY_ENV_VAR_1":"This is my environment variable 1 value from the AWS CLI.","MY_ENV_VAR_2":"This is my environment variable 2 value from the AWS CLI."}}"
aws lambda invoke --function-name MyEnvVarsFunction results.json
```

The following is output to a file named `results.json`:

```json
{
    "statusCode": 200,
    "headers": {
        "Content-type": "application/json"
    },
    "body": {
        "MY_ENV_VAR_1": "This is my environment variable 1 value from the AWS CLI.",
        "MY_ENV_VAR_2": "This is my environment variable 2 value from the AWS CLI."
    }
}
```

For more details, see the discussion of `update-function-configuration` and `invoke` in Create the Lambda function and Test It in the AWS Lambda Developer Guide.

## Working with Lambda Functions that Use Versions, Aliases, or Layers

The Lambda section of the AWS Resources window currently does not provide any features for working with versions, aliases, or layers for Lambda functions. Instead, you can use the terminal in the IDE to run the AWS CLI and AWS SAM CLI with the corresponding commands, actions, and arguments. For details, see the following.

- Introduction to AWS Lambda Versioning in the AWS Lambda Developer Guide
- Introduction to AWS Lambda Aliases in the AWS Lambda Developer Guide
- Managing Versioning Using the AWS Management Console, the AWS CLI, or Lambda API Operations in the AWS Lambda Developer Guide
To invoke an API in API Gateway that is related to an existing Lambda function, you must first import the remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. To do this, see Import a Lambda Function (p. 295).

Note
You cannot debug the remote version of the API Gateway API in your environment. You can only invoke it. To debug the local version, see Debug the Local Version of a Lambda Function or Its Related API Gateway API (p. 309).

1. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn't already expanded.
2. Expand the serverless application folder that contains the function whose API you want to invoke.
3. Choose the function, right-click it, and then choose Run.
4. Do one of the following:
   - To run the local version of the API within your environment, choose Run API Gateway Local.
   - To run the remote version of the function within Lambda, choose Run API Gateway Remote.

Note
If nothing appears to happen, an invoke tab might already be open for the function. If so, choose API Gateway (local) or API Gateway (remote) in the open invoke tab.

5. In the Test payload pane of the invoke tab that is displayed, confirm the Function, Path, Method, Query String, and Body you want the API to use when you test it.

Note
Some APIs might not support settings such as Body. For more information, consult the owner of the API.

6. On the invoke tab, choose the Run button.
Invoke an API Gateway API for a Related Lambda Function

Note
If the API isn't connected to the function, a message appears that says an API Gateway trigger can't be found in the function's AWS SAM file. To use this AWS SAM file to connect an API in API Gateway to the function, see the AWS Serverless Application Model (AWS SAM) repository on GitHub.

If you invoke an API in API Gateway and then try to invoke the same code for its related function in Lambda with the same custom input data, you might get an error or the code might not run as expected. For more information, see Response Differences When Invoking a Lambda Function from API Gateway (p. 305).

The invoke tab contains two panes:

- The Test payload pane displays settings and any custom input data that was supplied for the API.
- The Execution results pane displays information such as the body, headers, and logs of the API response.

Compare your results to the following:
Response Differences When Invoking a Lambda Function from API Gateway

When you invoke a Lambda function from an API in API Gateway and then try to parse the response, you might get an error or the code might not run as expected. This is because Lambda and API Gateway use slightly different response formats. Specifically, API Gateway wraps its response in a parent `body` object. To address this issue, you can add code to a function that checks to see if a parent `body` exists in the response. If it does, you can then extract the data from that `body` object.

For example, given the following Node.js function code:

```javascript
'use strict';
exports.handler = function(event, context, callback) {
    if (event.body) {
        event = JSON.parse(event.body);
    }
    const message = "Your favorite fruit is " + event.fruit + ". " + "Your favorite vegetable is " + event.vegetable + ".";
    const response = {
        statusCode: 200,
        headers: { "Content-type": "application/json" },
        body: JSON.stringify( { "message": message } )
    };
    callback(null, response);
};
```

And given the following equivalent Python function code:

```python
import json

def lambda_handler(event, context):
    if 'body' in event:
        event = json.loads(event["body"])
```
message = ("Your favorite fruit is " + event["fruit"] + ", " +
"Your favorite vegetable is " + event["vegetable"] + ".")

response = {
    "statusCode": "200",
    "headers": { "Content-type": "application/json" },
    "body": json.dumps({"message": message})
}

return response

To invoke the preceding code, you use the following input payload (for Lambda) or input body (for API Gateway):

```
{
    "fruit": "apple",
    "vegetable": "carrot"
}
```

Which returns the following response for Lambda:

```
{
    "statusCode": 200,
    "headers": {
        "Content-type": "application/json"
    },
    "body": "{"message":"Your favorite fruit is apple. Your favorite vegetable is carrot."}"
}
```

And returns the following response for API Gateway (assuming a Path of / and a Method of POST):

```
{
    "message": "Your favorite fruit is apple. Your favorite vegetable is carrot."
}
```

If you do not include the if (event.body) check for Node.js or the if 'body' in event check for Python, then calling this function from API Gateway might return an error or the API might not run as expected.

### Add Dependent Code to a Lambda Function

For Node.js, we support using Node Package Manager (npm) to add dependent packages to Lambda functions in your environment. For Python, we support pip. For general information about npm and pip, see the npm and pip websites.

To depend on an npm package from a Node.js Lambda function, use for example the Node.js require statement. Then use npm to install the related npm package in the environment within the same directory as the function code. When you deploy the Lambda function as described in Upload Code for a Lambda Function (p. 312), AWS Cloud9 deploys both the function and its related packages to Lambda.

To demonstrate, the following example Node.js Lambda function code depends on the lodash package to sort the specified JSON input payload.

```
'use strict';

/*
Assume the following payload is input:
```
The expected response is:

```json
{
  "statusCode": 200,
  "headers": {
    "Content-type": "application/json"
  },
  "body": {
    "result": [
      {
        "firstName": "Arnav",
        "lastName": "Desai"
      },
      {
        "firstName": "Jane",
        "lastName": "Doe"
      },
      {
        "firstName": "Shirley",
        "lastName": "Rodriguez"
      }
    ]
  }
}
```

To install the lodash package in the environment, use a terminal session in the IDE to change to the directory containing the function code. Then run the following two commands, in the following order. The first command creates and configure a package.json file in that directory to make sure when you deploy the function to Lambda, the lodash package is also deployed. The second command installs the lodash package in the same directory in the environment as the function code and then updates the package.json file in that directory accordingly.

```
npm init
```
npm install lodash --save

For help with the npm init command and the package.json file, see Working with package.json on the npm website.

From the IDE, invoke the local version of the Lambda function, as described in Invoke a Lambda Function (p. 297). Deploy the function as described in Upload Code for a Lambda Function (p. 312), and then invoke the remote version of the function. The local and remote versions of the function should work as expected.

To depend on a pip package from a Python Lambda function, use for example the Python import statement. Then use pip to install the related pip package in the environment one directory above the directory that contains the function code. When you deploy the Lambda function as described in Upload Code for a Lambda Function (p. 312), AWS Cloud9 deploys both the function and its related packages to Lambda.

To demonstrate, the following example Python Lambda function code depends on the requests package to make an HTTP request and then return information about the related HTTP response.

```python
import requests
import json

def lambda_handler(event, context):
    result = requests.get(event["url"]) 

    response = {
        "statusCode": "200",
        "headers": { "Content-type": "application/json" },
        "body": json.dumps( { "statusCode": result.status_code,
                            "date": result.headers["Date"],
                            "lastModified": result.headers["Last-Modified"] } )
    }

    return response
```

To install the requests package in the environment, use a terminal session in the IDE to change to the directory containing the function code. Then run the following command. This command installs the requests package in the directory in the environment that is one directory above the function code.

To install the requests package in the environment, use a terminal session in the IDE to change to the directory containing the function code. Then run the following command. This command installs the requests package in the directory in the environment that is one directory above the function code.
pip install requests --target ../

From the IDE, invoke the local version of the Lambda function, as described in Invoke a Lambda Function (p. 297). Deploy the function as described in Upload Code for a Lambda Function (p. 312), and then invoke the remote version of the function. The local and remote versions of the function should work as expected.

For a Python Lambda function, to depend on code in a separate Python code file that is in the same directory as the function, use the `from` and `import` statements. When you deploy the Lambda function as described in Upload Code for a Lambda Function (p. 312), AWS Cloud9 deploys to Lambda both the function and the separate Python code files in the same directory as the function.

To demonstrate, take for example the following directory structure in the AWS Cloud9 IDE for a Python Lambda function:

```
myDemoServerlessApplication
  `- myDemoFunction
      |- lambda-payloads.json
      |- lambda_function.py
  `-
```

If the `myClasses.py` file contains the definition of a class named `MyClass1`, for example:

```
class MyClass1:
    # Class definition...
```

To reference the `MyClass1` class from the `lambda_function.py` file, add the following statement to the file:

```
from myDemoFunction.myClasses import MyClass1
```

**Debug the Local Version of a Lambda Function or Its Related API Gateway API**

You can debug local Lambda function code or its related API Gateway API in your environment using common debugging aids such as breakpoints, stepping through code, and setting watch expressions.

**Note**

You cannot debug the remote version of a Lambda function or its related API Gateway API in your environment. You can only invoke it.

To debug the local version of an existing Lambda function or its related API Gateway API, you must first import the remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. See Import a Lambda Function (p. 295).

**Important**

If you import the remote version of a Python function into your environment, you must choose one of the following options before you can debug it:

**Option 1:** If the Python function doesn't use venv, use `pip` to install IKPdb into the same directory as the function's `template.yaml` file.

Use a terminal session in the IDE to change to the directory containing the function's `template.yaml` file. Then run one of the following commands. This command installs the Python debugger IKPdb in the same directory as the function's `template.yaml` file:
pip install ikpdb --target .  # For a function that uses Python 2.7.
pip-3.6 install ikp3db --target . # For a function that uses Python 3.6.

Option 2: If the Python function uses venv, use pip in venv to install IKPdb into the function's venv directory, and then add the CodeUri property to the function's template.yaml file.

1. Use a terminal session in the IDE to change to the directory containing the function's template.yaml file. From that folder, run one of the following commands. This command uses pip in the function's venv/bin directory to install the Python debugger IKPdb into the function's venv/lib/pythonMAJOR.MINOR/dist-packages directory:

venv/bin/pip install ikpdb       # For a function that uses Python 2.7.
venv/bin/pip3.6 install ikp3db   # For a function that uses Python 3.6.

2. In the Environment window, open the function's template.yaml file for editing. In the Properties section for the function, add the CodeUri property, set its value to .debug/, and then save the file. For example:

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: 'AWS::Serverless-2016-10-31'
Description: An AWS Serverless Specification template describing your function.
Resources:
  myDemoFunction:
    Type: 'AWS::Serverless::Function'
    Properties:
      CodeUri: .debug/
      # ...
```

1. In the Environment window, open the file that contains the Lambda function's code you want to debug.
2. Set any breakpoints and watch expressions for your code. See Debug Your Code (p. 147).
3. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn't already expanded.
4. Expand the serverless application folder that contains the function you want to debug.
5. Choose the function to debug, right-click it, and then choose Run, Run Local or Run, Run API Gateway Local.

**Note**
If nothing appears to happen, an invoke tab might already be open for the function. If so, go to the open invoke tab and choose Lambda (local) or API Gateway (local).

6. For a Lambda function, in the Test payload pane of the invoke tab that is displayed, confirm any custom input data you want your function to use when you test it. For information about the input data format, see Step 2.2: Invoke the Lambda Function Manually and Verify Results, Logs, and Metrics in the AWS Lambda Developer Guide.
7. For an API Gateway API, in the Test payload pane of the invoke tab that is displayed, confirm the Path, Method, Query String, and Body you want the API to use when you test it.

**Note**
Some APIs might not support settings such as Body. For more information, consult the owner of the API.
8. Next to the Run button, choose Run in Debug Mode (the bug icon).
9. Choose the Run button.
10. Decide what to do whenever function execution pauses at a breakpoint. See Debug Your Code (p. 147).
To use the AWS Cloud9 IDE to change the code in a function, you must first import the related remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. To do this, see Import a Lambda Function (p. 295). Then do the following:

1. In the **Lambda** section of the **AWS Resources** window, expand the **Local Functions** list, if it isn’t already expanded.

2. Expand the serverless application folder that contains the function whose code you want to change.

**Change Code in a Lambda Function**

Compare your results to the following:
3. Right-click the function, and then choose **Edit Function**.

![Image showing the Edit Function option]({image_url})

4. Make the changes you want to the code, and then save the file.

To upload the local version of the changed function code to the related remote version in Lambda, see [Upload Code for a Lambda Function (p. 312)].

**Upload Code for a Lambda Function**

To upload the local version of a Lambda function in your environment to the related remote version of the function in Lambda, follow one of these procedures, depending on how the function was created.

<table>
<thead>
<tr>
<th>How was the function created?</th>
<th>Follow this procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>By using the <strong>Create serverless application</strong> wizard in the IDE.</td>
<td>[Upload Code for a Lambda Function Created By the Create Serverless Application Wizard (p. 313)]</td>
</tr>
<tr>
<td>As part of an AWS CodeStar project.</td>
<td>[Upload Code for a Lambda Function That is Part of an AWS CodeStar Project (p. 313)]</td>
</tr>
<tr>
<td>By using the AWS Serverless Application Repository, and the serverless application requires parameters to be specified during deployment.</td>
<td>[Upload Code for a Lambda Function That is Part of an AWS Serverless Application Repository Project with Parameters (p. 314)]</td>
</tr>
<tr>
<td>Any other way.</td>
<td>[Upload Code for a Lambda Function By Using a Specific AWS CloudFormation Stack, Function Name, or Both (p. 314)]</td>
</tr>
</tbody>
</table>
Upload Code for a Lambda Function Created By the Create Serverless Application Wizard

After you Create a Lambda Function With the Create Serverless Application Wizard (p. 289) in your environment, you can upload the local version of that function to the related remote version of the function in Lambda as follows.

1. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn’t already expanded.
2. Expand the serverless application folder that contains the function you want to upload.
3. Do one of the following:
   - Right-click the serverless application folder that you just chose, and then choose Deploy.
   - Right-click the function you want to upload, and then choose Deploy.
   - Choose the function you want to upload, and then choose Deploy the selected Lambda function (the button with the arrow that faces up).

Upload Code for a Lambda Function That is Part of an AWS CodeStar Project

As you are Working with AWS CodeStar Projects in the AWS Cloud9 Integrated Development Environment (IDE) (p. 283), you can upload the local version of that function in your environment to the related remote version of the function in Lambda as follows.

Use a terminal session in the IDE to run the git push command to push committed code changes to the repository for the AWS CodeStar project. This instructs AWS CodeStar to upload the local version of the Lambda function in your environment to the related remote version of the function in Lambda.

If you do not follow this procedure, the error "Parameters: [ProjectId] must have values" will display, and the function will not deploy.
Upload Code for a Lambda Function That is Part of an AWS Serverless Application Repository Project with Parameters

If you Create and Deploy Lambda Functions with the AWS Serverless Application Repository (p. 293), and that project requires you to specify parameters during the upload, see coverage of the AWS CloudFormation deploy command in the section called “Create and Deploy Lambda Functions with the AWS Serverless Application Repository” (p. 293) for upload instructions.

If you do not follow that procedure, an error will display that required parameters are missing, and the code will not upload.

Upload Code for a Lambda Function By Using a Specific AWS CloudFormation Stack, Function Name, or Both

To begin the upload process, AWS Cloud9 instructs AWS CloudFormation to create a stack with a specific name. AWS CloudFormation uses the information in this stack to complete the upload of the local version of the Lambda function in your environment to a specific function in Lambda. By default, the name of this stack and the name of the Lambda function is one of the following, which you can change if needed.

- If a hidden .application.json file exists in the same folder as the local version of the Lambda function, and the file contains a StackName value, the stack's name is the same as the StackName value, and the Lambda function name is cloud9-APPLICATION_NAME-FUNCTION_NAME.
- If there is no hidden .application.json file in the same folder as the local version of the Lambda function, or if the application.json file exists but has no StackName value, the stack's name is cloud9-FOLDER_NAME, and the Lambda function name is cloud9-APPLICATION_NAME-FUNCTION_NAME.

In the preceding stack and function names, FOLDER_NAME is the name of the related folder in the Environment window, while APPLICATION_NAME and FUNCTION_NAME are the related values as displayed in the Lambda section of the AWS Resources window.

If you do not want to change the name of the default AWS CloudFormation stack or the default function name in Lambda, then skip ahead to the upload procedure at the end of this section.

To use or create a non-default AWS CloudFormation stack in your AWS account in the same AWS Region as displayed in the Lambda section of the AWS Resources window, do one of the following.

- If you want to use an existing AWS CloudFormation stack to upload the code, then in the Lambda section of the AWS Resources window, right-click the serverless application folder that contains the Lambda function you want to upload the code to, choose Link to CloudFormation Stack, and follow the on-screen instructions to choose the existing stack to use.

  Note
  You can only choose from existing stacks that are in the following AWS CloudFormation states.
  - CREATE_COMPLETE
  - CREATE_IN_PROGRESS
  - REVIEW_IN_PROGRESS
  - ROLLBACK_COMPLETE
  - ROLLBACK_IN_PROGRESS
  - UPDATE_COMPLETE
  - UPDATE_COMPLETE_CLEANUP_IN_PROGRESS
  - UPDATE_IN_PROGRESS
If you want to create a new AWS CloudFormation stack with a name that you specify, then do one of the following:

- If a hidden .application.json file exists in the same folder as the local version of the Lambda function, and the file contains a StackName value, change the StackName value in the .application.json file to the name of the AWS CloudFormation stack you want to use, and then save the .application.json file.
- If a hidden .application.json file exists in the same folder as the local version of the Lambda function, but the file does not contain a StackName value, add a StackName value to the beginning of the .application.json file with the name of the new AWS CloudFormation stack you want to use, and then save the .application.json file. For example, for a stack name of MyDemoStack, add the following value to the beginning of the file. (Do not type the ellipses. They are shown only to help you add the value to the correct location in the file.)

```json
{
    "StackName": "MyDemoStack",
    ...
}
```

- If a hidden .application.json file does not exist in the same folder as the local version of the Lambda function, then create a new .application.json file in that folder, add a StackName value to the .application.json file with the name of the new AWS CloudFormation stack you want to use, and then save the .application.json file. For example, for a stack name of MyDemoStack, add the following value to the file.

```json
{
    "StackName": "MyDemoStack"
}
```

To upload the code to a non-default Lambda function in your AWS account in the same AWS Region as displayed in the Lambda section of the AWS Resources window, add the function name as a FunctionName value to the Properties section of the Lambda function resource in the related AWS SAM template file (for example, template.yaml). For example, for a Lambda function resource named MyDemoFunction, add a FunctionName value of MyDemoFunction to upload the code to a Lambda function named MyDemoFunction instead of cloud9-APPLICATION_NAME-FUNCTION_NAME. (Do not type the ellipses. They are shown only to help you add the value to the correct location in the file.)

```yaml
...
Resources:
  MyDemoFunction:
    Type: AWS::Serverless::Function
    Properties:
      FunctionName: MyDemoFunction
      ...
```

When you are ready to upload the code, do the following.

1. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn't already expanded.
2. Expand the serverless application folder that contains the function you want to upload.
3. Do one of the following:
   - Right-click the serverless application folder that you just chose, and then choose Deploy.
AWS Cloud9 User Guide
Convert a Lambda Function to a Serverless Application

- Right-click the function you want to upload, and then choose **Deploy**.
- Choose the function you want to upload, and then choose **Deploy the selected Lambda function** (the button with the arrow that faces up).

**Convert a Lambda Function to a Serverless Application**

If the local version of an existing Lambda function in your AWS Cloud9 development environment isn't already part of a serverless application, you can use the AWS Cloud9 IDE to convert that function into a serverless application. You can then use the AWS SAM template file for that serverless application to create additional supporting AWS resources for your function. For more information, see the AWS Serverless Application Model (AWS SAM) repository on GitHub.

To convert the local version of an existing Lambda function into a serverless application, you must first import the remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. See **Import a Lambda Function** (p. 295).

1. In the **Lambda** section of the **AWS Resources** window, expand the **Local Functions** list, if it isn't already expanded.
2. Right-click the function you want to convert, and then choose **Convert to SAM**.
AWS Cloud9 User Guide
Convert a Lambda Function to a Serverless Application

AWS Cloud9 does the following:

- In the function's folder in the Environment window, the DeploymentMethod setting in the .application.json file changes from lambda to cloudformation. This means that now AWS Cloud9 will instruct AWS SAM to use AWS CloudFormation whenever you use the IDE to upload the function's code as part of the serverless application. (lambda means that AWS Cloud9 will instruct Lambda to deploy the function instead.) To upload the function code, see the section called “Upload Code for a Lambda Function” (p. 312).

- In the Lambda section of the AWS Resources window, in the Local Functions list, AWS Cloud9 adds the existing Lambda function to a new serverless application (represented by a Lambda icon inside of a folder). The serverless application has the same name as the function.

When you upload the function's code as described in Upload Code for a Lambda Function (p. 312), because the function upload method is no longer Lambda but now AWS SAM using AWS CloudFormation, AWS Cloud9 creates a new remote version of the function in Lambda and adds it to the Remote Functions list. AWS Cloud9 gives the remote version a name that is different from the original Lambda function. For example, if the serverless application and the function are both named myDemoFunction, the remote version name of your function would be cloud9-myDemoFunction-myDemoFunction-RANDOM_ID, where RANDOM_ID is a randomly determined ID.

**Important**
After you do the conversion, if you then use the IDE to make any changes to the function code and then upload that code to Lambda, only the remote version of the new function (for example, cloud9-myDemoFunction-myDemoFunction-RANDOM_ID) will contain the change. The remote version of the original function (for example, myDemoFunction) will not change. If you change your mind and want to enable the IDE to go back to uploading your code changes to the remote version of the original function (for example, myDemoFunction), do the following:

1. In the function's folder in the Environment window, change the DeploymentMethod setting in the .application.json file from cloudformation back to lambda, and then save the file. This removes the serverless application folder from the Local Functions list and causes AWS Cloud9 to go back to instructing Lambda to deploy the function.

2. Upload the function code as described in Upload Code for a Lambda Function (p. 312). Now, only the remote version of the original function (for example, myDemoFunction) will contain the change. The remote version of the new function (for example, cloud9-myDemoFunction-myDemoFunction-RANDOM_ID) will not change.

3. Because AWS Cloud9 will no longer upload code changes to the remote version of the new function (for example, cloud9-myDemoFunction-myDemoFunction-RANDOM_ID), if you want you can use the Lambda console to delete the new function (for example, cloud9-myDemoFunction-myDemoFunction-RANDOM_ID).
Update Configuration Settings for a Lambda Function

You can use the AWS Cloud9 IDE to change function settings such as the description, handler identifier, amount of memory the function will use, and existing execution role the function will use.

To change configuration settings, you must first import the related remote version of the function into your AWS Cloud9 development environment, if the function isn't already there. To do this, see Import a Lambda Function (p. 295). Then do the following.

1. In the Lambda section of the AWS Resources window, expand the Local Functions list, if it isn't already expanded.

2. Expand the serverless application folder that contains the function whose setting you want to change.

3. Right-click the function, and then choose Edit Config.

4. Make changes to the configuration settings, and then save the file.

   **Note**
   By default, configuration settings are displayed in plain text. To change this behavior to display configuration settings in a visual editor by default, choose AWS Cloud9, Preferences on the menu bar. Choose AWS Settings, and then turn on Use AWS SAM visual editor. To use the visual editor, close the function's template.yaml file, and then right-click the function and choose Edit Config again. To switch back to using plain text by default, turn off the Use AWS SAM visual editor setting. To temporarily edit plain text, choose View with text editor (Ace) in the visual editor, and then choose View, Editors, Ace on the menu bar.

5. Do one of the following:
   - On the configuration settings tab, in the simplified settings view, choose the Upload Settings to Lambda button.
   - Follow the instructions in Upload Code for a Lambda Function (p. 312).
Delete a Lambda Function

You can delete the local version of a Lambda function from your environment, delete the remote version of the function from Lambda, or both, as follows.

Topics
- Delete the Local Version of a Lambda Function (p. 319)
- Delete the Remote Version of the Lambda Function (p. 319)

Delete the Local Version of a Lambda Function

Use the **Environment** window in the AWS Cloud9 IDE to delete the directory that contains the function. (You cannot use the **Lambda** section of the **AWS Resources** window in the AWS Cloud9 IDE to delete the local version of a Lambda function.)

**Warning**
If you accidentally delete the local version of a Lambda function, the only way to add it back to your environment is to import the remote version of the function into your environment, if the remote version still exists. For instructions, see Import a Lambda Function (p. 295).

Delete the Remote Version of the Lambda Function

To delete the remote version of the Lambda function, use the Lambda console, the AWS CloudFormation console, or code. (You cannot use the **Lambda** section of the **AWS Resources** window in the AWS Cloud9 IDE to delete the remote version of a Lambda function.)

To determine which approach to use to delete the remote version of a Lambda function, in the AWS Cloud9 IDE, open the `.application.json` file that is in the same directory as the local version of the Lambda function. If the `DeploymentMethod` value is set to `lambda`, you use Lambda to delete
Delete a Lambda Function

the function. If the DeploymentMethod value is set to cloudformation, you should use AWS CloudFormation to delete the function.

Note
If the DeploymentMethod value is set to cloudformation in the .application.json file, we do not recommend using Lambda to delete the function. If you use Lambda instead of AWS CloudFormation to delete the function in this case, then you might leave some associated AWS resources still remaining. Those remaining resources could result in ongoing charges to your AWS account.

Topics
• Use Lambda to Delete the Remote Version of the Function (p. 320)
• Use AWS CloudFormation to Delete the Remote Version of the Function (p. 321)

Use Lambda to Delete the Remote Version of the Function

Use the Lambda console or code to delete the function that has the same name as the function in the Lambda section of the AWS Resources window or the PhysicalId value in the .application.json file, as follows.

Warning
When you delete the remote version of a function, it is permanently deleted from Lambda. If you accidentally delete the remote version of a function and need to recover it, you can upload the local version of the function to Lambda, if the local version still exists. For instructions, see Upload Code for a Lambda Function (p. 312).

• To delete the function by using the Lambda console, do the following.
  2. On the AWS navigation bar, if the AWS Region that contains the Lambda function is not displayed, then choose it.
  3. If the list of functions is not displayed, then choose Functions in the service navigation pane.
  4. Do one of the following.
     • Choose the radio button next to the name of the function that you want to delete. Then choose Actions, Delete. Confirm the deletion by choosing Delete.
     • Choose the name of the function that you want to delete. Then choose Actions, Delete Function. Confirm the deletion by choosing Delete.

• To delete the function by using code, call the Lambda delete function operation, as follows.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>delete-function</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>DeleteFunctionRequest</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DeleteFunction, DeleteFunctionRequest, DeleteFunctionWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DeleteFunctionRequest, DeleteFunctionResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>deleteFunction</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DeleteFunctionRequest, DeleteFunctionResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>deleteFunction</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>delete_function</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>delete_function</td>
</tr>
</tbody>
</table>
Use AWS CloudFormation to Delete the Remote Version of the Function

Use the AWS CloudFormation console or code to delete the stack that has the same name as the StackName value in the .application.json file, as follows.

**Warning**
When you delete a stack, AWS CloudFormation deletes all AWS resources that are associated with that stack. This includes not only Lambda functions but could also include other related resources such as APIs in Amazon API Gateway. If you accidentally delete the remote version of a function and need to recover it, you can upload the local version of the function from the AWS Cloud9 IDE to Lambda, if the local version still exists. For instructions, see Upload Code for a Lambda Function (p. 312). All of the stack's other resources are permanently deleted and cannot be recovered.

- To delete the stack by using the AWS CloudFormation console, do the following.
  2. On the AWS navigation bar, if the AWS Region that contains the stack is not displayed, then choose it.
  3. In the list of stacks, do one of the following.
     - Select the check box next to the name of the stack that you want to delete. Then choose **Actions, Delete Stack**. Confirm the deletion by choosing **Yes, Delete**.
     - Choose the name of the stack that you want to delete. Then choose **Other Actions, Delete Stack**. Confirm the deletion by choosing **Yes, Delete**.

- To delete the stack by using code, call the AWS CloudFormation delete stack operation, as follows.

<table>
<thead>
<tr>
<th>AWS CLI</th>
<th>delete-stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS SDK for C++</td>
<td>DeleteStackRequest</td>
</tr>
<tr>
<td>AWS SDK for Go</td>
<td>DeleteStack, DeleteStackRequest, DeleteStackWithContext</td>
</tr>
<tr>
<td>AWS SDK for Java</td>
<td>DeleteStackRequest, DeleteStackResult</td>
</tr>
<tr>
<td>AWS SDK for JavaScript</td>
<td>deleteStack</td>
</tr>
<tr>
<td>AWS SDK for .NET</td>
<td>DeleteStackRequest, DeleteStackResponse</td>
</tr>
<tr>
<td>AWS SDK for PHP</td>
<td>deleteStack</td>
</tr>
<tr>
<td>AWS SDK for Python (Boto)</td>
<td>delete_stack</td>
</tr>
<tr>
<td>AWS SDK for Ruby</td>
<td>delete_stack</td>
</tr>
<tr>
<td>AWS Tools for Windows PowerShell</td>
<td>Remove-CFNStack</td>
</tr>
<tr>
<td>AWS CloudFormation API</td>
<td>DeleteStack</td>
</tr>
</tbody>
</table>
Working with AWS CodePipeline in the AWS Cloud9 Integrated Development Environment (IDE)

You can use the AWS Cloud9 IDE to work with source code in repositories that are compatible with AWS CodePipeline.

CodePipeline is a continuous delivery service you can use to model, visualize, and automate the steps required to release your software and ongoing changes you make to it. You can use CodePipeline to quickly model and configure the different stages of a software release process. For more information, see the AWS CodePipeline User Guide.

Note
Completing these procedures might result in charges to your AWS account. These include possible charges for services such as Amazon EC2, CodePipeline, Amazon S3, and AWS services supported by CodePipeline. For more information, see Amazon EC2 Pricing, AWS CodePipeline Pricing, Amazon S3 Pricing, and Cloud Services Pricing.

AWS CodeStar provides additional features along with pipelines, such as project templates, dashboards, and teams. To use AWS CodeStar instead of CodePipeline, skip the rest of this topic, and see Working with AWS CodeStar Projects (p. 283) instead.

• Step 1: Create or Identify Your Source Code Repository (p. 322)
• Step 2: Create an AWS Cloud9 Development Environment, Connect It to the Code Repository, and Upload Your Code (p. 323)
• Step 3: Prepare to Work with AWS CodePipeline (p. 324)
• Step 4: Create a Pipeline in AWS CodePipeline (p. 324)

Step 1: Create or Identify Your Source Code Repository

In this step, you create or identify a source code repository that is compatible with CodePipeline.

Later in this topic, you upload your software's source code to that repository. CodePipeline will build, test, and deploy the uploaded source code in that repository by using related pipelines that you also create.

Your source code repository must be one of the following repository types that CodePipeline supports:

• **AWS CodeCommit.** If you already have a repository in CodeCommit that you want to use, skip ahead to Step 2: Create an AWS Cloud9 Development Environment, Connect It to the Code Repository, and Upload Your Code (p. 323). Otherwise, to use CodeCommit, follow these instructions in the AWS CodeCommit Sample in this order, and then return to this topic:
  • Step 1: Set Up Your IAM Group with Required Access Permissions (p. 356)
  • Step 2: Create a Repository in AWS CodeCommit (p. 357)

• **Amazon S3.** If you already have a bucket in Amazon S3 that you want to use, skip ahead to Step 2: Create an AWS Cloud9 Development Environment, Connect It to the Code Repository, and Upload Your Code (p. 323). Otherwise, to use Amazon S3, follow these instructions in the Amazon Simple Storage Service Getting Started Guide in this order, and then return to this topic:
  • Sign Up for Amazon S3
  • Create a Bucket

• **GitHub.** If you already have a repository in GitHub that you want to use, skip ahead to Step 2: Create an AWS Cloud9 Development Environment, Connect It to the Code Repository, and Upload Your
Step 2: Create an AWS Cloud9 Development Environment, Connect It to the Code Repository, and Upload Your Code

In this step, you create an AWS Cloud9 development environment in the AWS Cloud9 console. You then connect the environment to the repository that CodePipeline will use. Finally, you use the AWS Cloud9 IDE for the environment to upload your source code to the repository.

To create the environment, follow the instructions in Creating an Environment (p. 55), and then return to this topic. (If you already have an environment, you can use it. You don't need to create a new one.)

To connect the environment to the repository, and then upload your source code to the repository if it isn't already there, use one of the following sets of instructions. The set you choose depends on the type of repository that stores the source code.

<table>
<thead>
<tr>
<th>Repository type</th>
<th>Instructions</th>
</tr>
</thead>
</table>
| CodeCommit      | Follow these instructions in the AWS CodeCommit Sample:  
|                 | • Step 3: Connect Your Environment to the Remote Repository (p. 358)  
|                 | • Step 4: Clone the Remote Repository into Your Environment (p. 359)  
|                 | • Step 5: Add Files to the Repository (p. 360), substituting your own source code for this step |
| Amazon S3        | • Install and configure the AWS CLI or aws-shell in the environment, as described in the AWS CLI and aws-shell Sample (p. 352).  
|                 | • To upload your source code to the bucket, use the AWS CLI or the aws-shell in the environment to run the aws s3 cp command. (For the aws-shell, you can remove aws from the command.) |
| GitHub           | Follow these instructions in the GitHub Sample:  
|                 | • Step 3: Install Git in Your Environment (p. 362)  
|                 | • Step 4: Clone the Remote Repository into Your Environment (p. 363)  
|                 | • Step 5: Add Files to the Repository (p. 363), substituting your own source code for this step |

After you connect the environment to the repository, whenever you push source code changes from the AWS Cloud9 IDE to the repository, CodePipeline automatically sends those changes through related pipelines to be built, tested, and deployed. You create a related pipeline later in this topic.
Step 3: Prepare to Work with AWS CodePipeline

In this step, you attach a specific AWS managed policy to the IAM group you created or identified in Team Setup (p. 8). This enables the group’s users to begin creating and working with pipelines in CodePipeline.

If you have used CodePipeline before, skip ahead to Step 4: Create a Pipeline in AWS CodePipeline (p. 324).

For this step, follow these instructions in Step 3: Use an IAM Managed Policy to Assign AWS CodePipeline Permissions to the IAM User in the AWS CodePipeline User Guide, and then return to this topic.

Step 4: Create a Pipeline in AWS CodePipeline

In this step, you create a pipeline in CodePipeline that uses the repository you created or identified earlier in this topic.

For this step, follow the instructions in Create a Pipeline in AWS CodePipeline in the AWS CodePipeline User Guide.

After you create the pipeline, CodePipeline sends the current version of the source code in the repository through the pipeline to be built, tested, and deployed. Then, whenever you push source code changes from the AWS Cloud9 IDE to the repository, CodePipeline automatically sends those changes through the pipeline to be built, tested, and deployed.

To view the pipeline, follow the instructions in View Pipeline Details and History in AWS CodePipeline in the AWS CodePipeline User Guide.
Tutorials and Samples for AWS Cloud9

Are you new to AWS Cloud9? Take a tour of the IDE in Getting Started: Basic Tutorials (p. 35).

Experiment with these tutorials to increase your knowledge and confidence using AWS Cloud9 with various programming languages and AWS services.

Topics
- AWS Lambda Tutorial for AWS Cloud9 (p. 325)
- Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342)
- AWS Command Line Interface and aws-shell Sample for AWS Cloud9 (p. 352)
- AWS CodeCommit Sample for AWS Cloud9 (p. 356)
- GitHub Sample for AWS Cloud9 (p. 361)
- Amazon DynamoDB Sample for AWS Cloud9 (p. 366)
- AWS CDK Sample for AWS Cloud9 (p. 377)
- LAMP Sample for AWS Cloud9 (p. 383)
- WordPress Sample for AWS Cloud9 (p. 390)
- Java Sample for AWS Cloud9 (p. 400)
- C++ Sample for AWS Cloud9 (p. 410)
- Python Tutorial for AWS Cloud9 (p. 418)
- .NET Core Sample for AWS Cloud9 (p. 423)
- Node.js Sample for AWS Cloud9 (p. 430)
- PHP Sample for AWS Cloud9 (p. 435)
- Tutorial: Ruby in AWS Cloud9 (p. 440)
- Go Sample for AWS Cloud9 (p. 446)
- TypeScript Sample for AWS Cloud9 (p. 452)
- Docker Sample for AWS Cloud9 (p. 458)
- Related Samples (p. 465)

AWS Lambda Tutorial for AWS Cloud9

In this tutorial, you use the AWS Cloud9 IDE to create a function in AWS Lambda and an accompanying API in Amazon API Gateway. After you create the function and its API, you run and debug them locally. Then you run the function and API in Lambda and API Gateway in production.

Note
Completing this tutorial might result in charges to your AWS account. These include possible charges for Amazon EC2, Lambda, and API Gateway. For more information, see Amazon EC2 Pricing, AWS Lambda Pricing, and Amazon API Gateway Pricing.

Prerequisites
To successfully complete this tutorial, you must first complete the steps in Setting Up AWS Cloud9 (p. 6).
Steps

- Step 1: Create and Open the Environment (p. 326)
- Step 2: Create the Lambda Function and API (p. 328)
- Step 3: Add Code to the Function (p. 331)
- Step 4: Run or Debug the Function Locally (p. 333)
- Step 5: Run or Debug the API Locally (p. 336)
- Step 6: Run the Function in Production (p. 338)
- Step 7: Run the API in Production (p. 338)
- Step 8: Change the Function and Deploy the Change (p. 339)
- Step 9: Clean Up (p. 340)
- Related Information (p. 341)

Step 1: Create and Open the Environment

(First step of AWS Lambda Tutorial for AWS Cloud9 (p. 325))

In this step, you use the AWS Cloud9 console to create and then open an AWS Cloud9 development environment.

**Note**

If you already have an environment, open it, and then skip ahead to Step 2: Create the Lambda Function and API (p. 328).

In AWS Cloud9, a *development environment* (or just *environment*) is a place where you store your development project's files and where you run the tools to develop your applications. In this tutorial, you create a special kind of environment called an *EC2 environment*. For this kind of environment, AWS Cloud9 launches and manages a new Amazon EC2 instance running Amazon Linux or Ubuntu Server, creates the environment, and then connects the environment to the newly-launched instance. When you open the environment, AWS Cloud9 displays the AWS Cloud9 IDE that enables you to work with the files and tools in that environment.

1. Sign in to the AWS Cloud9 console as follows:

   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to https://console.aws.amazon.com/cloud9/.
   - If your organization uses AWS Single Sign-On (AWS SSO), see your AWS account administrator for sign-in instructions.
   - If you're using an AWS Educate Starter Account, see Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29) in *Individual Student Signup*.
   - If you're a student in a classroom, see your instructor for sign-in instructions.

2. After you sign in to the AWS Cloud9 console, in the top navigation bar, choose an AWS Region to create the environment in. For a list of available AWS Regions, see AWS Cloud9 in the AWS General Reference.
3. Choose the large **Create environment** button in one of the locations shown below.

   If you have no AWS Cloud9 environments yet, the button is shown on a welcome page.

   ![Create environment button on welcome page](image)

   If you already have AWS Cloud9 environments, the button is shown as follows.

   ![Create environment button on existing environment](image)

4. On the **Name environment** page, for **Name**, enter a name for your environment. For this tutorial, use `my-lambda-environment`.

5. For **Description**, enter something about your environment. For this tutorial, use `This environment is for the AWS Cloud9 tutorial for Lambda`.

6. Choose **Next step**.

7. On the **Configure settings** page, for **Environment type**, choose **Create a new instance for environment (EC2)**.

   **Warning**
   Choosing **Create a new instance for environment (EC2)** might result in possible charges to your AWS account for Amazon EC2.

8. For **Instance type**, leave the default choice. This choice has relatively low RAM and vCPUs, which is sufficient for this tutorial.
Warning
Choosing instance types with more RAM and vCPUs might result in additional charges to your AWS account for Amazon EC2.

9. For Platform, choose the type of Amazon EC2 instance that you want: Amazon Linux or Ubuntu. AWS Cloud9 creates the instance and then connects the environment to it.

10. Choose a value for Cost-saving setting. When all web browser instances that are connected to the IDE for the environment are closed, AWS Cloud9 waits this amount of time and then shuts down the Amazon EC2 instance for the environment.

Warning
Choosing a longer time period might result in more charges to your AWS account.

11. Add up to 50 tags by supplying a Key and a Value for each tag. The tags will be attached to the AWS Cloud9 environment as resource tags, and are propagated to the following underlying resources: the AWS CloudFormation stack, the Amazon EC2 instance, and Amazon EC2 security groups. You can find information about tags in Control Access Using AWS Resource Tags in the IAM User Guide. Also see the advanced information (p. 516) about tags.

Warning
If you update these tags after you create them, the changes are NOT automatically propagated to the underlying resources. For more information, see Propagating Tag Updates to Underlying Resources (p. 516) in the advanced information about tags (p. 516).

12. Choose Next step.

13. On the Review page, choose Create environment. Wait while AWS Cloud9 creates your environment. This can take several minutes.

Note
If account creation fails, a banner is displayed at the top of the console page. Additionally, the card for the environment, if it exists, indicates that environment creation failed.

After AWS Cloud9 creates your environment, it displays the AWS Cloud9 IDE for the environment.

If AWS Cloud9 doesn't display the IDE after at least five minutes, there might be a problem with your web browser, your AWS access permissions, the instance, or the associated virtual private cloud (VPC). For possible fixes, see Cannot Open an Environment (p. 544) in Troubleshooting.

Next Step

Step 2: Create the Lambda Function and API (p. 328)

Step 2: Create the Lambda Function and API

(Previous step: Step 1: Create and Open the Environment (p. 326))

In this step, you use the AWS Cloud9 IDE to create the Lambda function and its accompanying API at the same time. AWS Cloud9 stores the new function on the Amazon EC2 instance and deploys a copy of the function to Lambda. AWS Cloud9 also stores the new accompanying API on the instance and deploys a copy of the API to API Gateway.

The Lambda function returns information about the day or time you specify, for example, the day, month, and year, or the current hour, minute, and second.

Currently, you can use the IDE to automatically create functions that use only Node.js or Python. This function uses Node.js.

1. Make sure you have completed the prerequisites for work with Lambda functions in the AWS Cloud9 IDE for the environment. This includes checking to see whether you must:
• Set up your IAM group with required access permissions.
• Set up your environment with AWS access credentials.
• Create an execution rule for your Lambda functions.

For more information, see Prepare to Work with Lambda Functions (p. 285).

2. With the IDE open from the previous step, set the AWS Region that you want to create the function in. To do this, on the menu bar, choose AWS Cloud9, Preferences.

3. In the navigation pane of the Preferences tab, choose AWS Settings.

4. For AWS Region, select the AWS Region you want to create the function in.

5. On the edge of the IDE, choose AWS Resources.

6. Expand the Lambda, if it isn't already expanded. On the toolbar, choose the Create a new Lambda function icon.

7. In the Create serverless application dialog box, for Function name, type a name for the function (for example, myDateTimeFunction).

8. For Application name, type a name for the function's containing serverless application (for example, MyDateTimeApplication).

9. Choose Next.

10. For Select runtime, choose Node.js 12.x.

11. For Select blueprint, choose empty-nodejs. This creates some starter code that you work with in the next step.
12. Choose Next.

13. For Function trigger, choose API Gateway. This creates an API in API Gateway that you use to run and debug the function in a later step.

14. For Resource Path, type / (a forward slash). The Resource Path defines a portion of the URL that you use to run and debug the API in a later step. For more information, see Set up API Methods in API Gateway in the API Gateway Developer Guide.

15. For Security, choose NONE, and then choose Next. This specifies that you don't need any special credentials to run and debug this API.

16. For Memory (MB), leave the default value of 128 MB, which is sufficient for this tutorial.

17. For Role, leave the default value of Automatically generate role, and then choose Next. This specifies the access permissions that the function needs to work properly.

18. Review your choices, and then choose Finish.

AWS Cloud9 creates the function and its related API on the instance. Then it deploys a copy of the function and API to Lambda and API Gateway. The serverless application and function are displayed in the Local Functions and Remote Functions lists in the Lambda pane of the AWS Resources window. The serverless application's and function's component files are displayed in the Environment window. The editor opens the function's code file, index.js.

Note
In addition to the function's code file, AWS Cloud9 creates the following helper files.

- .application.json: A hidden file that contains JSON-formatted settings that are specific to the serverless application. AWS Cloud9 uses these settings for its internal use. Do not edit this file.
- .gitignore: A hidden file that contains a list of files Git ignores, if you want to use Git to manage your source code for this function.
- template.yaml: An AWS Serverless Application Model (AWS SAM) template file that contains information about the Lambda function, the related API in API Gateway, and any other related, supported AWS resources. Whenever you update the local version of your function and then upload it to Lambda, AWS Cloud9 calls AWS SAM to use this template file to do the upload. For more information, see Using the AWS Serverless Application Model (AWS SAM) in the AWS Lambda Developer Guide.

The .application.json and .gitignore files are hidden. To show hidden files or hide them if they're shown, in the Environment window, choose the gear icon. Then choose Show Hidden Files.
Next Step

Step 3: Add Code to the Function (p. 331)

Step 3: Add Code to the Function

(Previous step: Step 2: Create the Lambda Function and API (p. 328))

In this step, you replace the starter code for the generated function with specific code that returns information about the day or time you specify.

With the `index.js` file already open in the editor, completely replace the file’s contents with the following code, and then save the file.

```javascript
'use strict';
exports.handler = function(event, context, callback) {
    if (event.body) {
        event = JSON.parse(event.body);
    }

    var sc; // Status code
    var result = ""; // Response payload
    switch(event.option) {
        case "date":
            switch(event.period) {
                case "yesterday":
                    result = setDateResult("yesterday");
                    sc = 200;
                    break;
                case "today":
                    result = setDateResult();
                    sc = 200;
                    break;
                case "tomorrow":
                    result = setDateResult("tomorrow");
                    sc = 200;
                    break;
                default:
                    result = {
                        "error": "Must specify 'yesterday', 'today', or 'tomorrow'."
                    };
                    sc = 400;
```
This function takes an incoming payload with an option value of date or time. If date is specified, you must also specify a period value of yesterday, today, or tomorrow. The function then returns the
corresponding month, day, and year. If, however, an option value of time is specified, the function returns the current hour, minute, and second.

Next Step

Step 4: Run or Debug the Function Locally (p. 333)

Step 4: Run or Debug the Function Locally

(Previous step: Step 3: Add Code to the Function (p. 331))

Step 4.1: Run the Function Locally

In this step, you use the IDE to run the function on the instance. Currently, you can use the IDE to run functions that use only Node.js or Python.

1. In the Lambda pane of the AWS Resources window, expand the MyDateTimeApplication Lambda folder, and then right-click the myDateTimeFunction Lambda icon.

2. Choose Run, Run Local.

3. In the Payload pane of the run tab, replace {} with the following, which sends this data as input to the function for processing.

```json
{
  "option": "date",
  "period": "today"
}
```


5. The Response pane displays the following.

```json
{
  "statusCode": 200,
  "body": "Today is...

333"
"headers": {
  "Content-type": "application/json"
},
"body": "{"month":4,"day":12,"year":2018}"

**Note**
If the **Response** pane displays null, be sure to save the **index.js**, and then choose Run again.

Compare your results to the following.

6. Run the function several times with the following payloads to see what the **Response** pane displays.

   ```json
   {  
      "option": "date",
      "period": "yesterday"
   }
   ```

   ```json
   {  
      "option": "date",
      "period": "tomorrow"
   }
   ```

   ```json
   {  
      "option": "time"
   }
   ```

   Displays an error. Must also specify a 'period' of 'yesterday', 'today', or 'tomorrow'.

   ```json
   {  
      "option": "date"
   }
   ```

   Displays an error. Must specify an 'option' of 'date' or 'time'.

   ```json
   {  
      "option": "dates"
   }
   ```
Step 4.2: Debug the Function Locally

In this step, you use the IDE to debug the function on the instance. Currently, you can use the IDE to debug functions that use only Node.js or Python. Also, you can use the IDE to debug functions locally only. You cannot use the IDE to debug functions in Lambda itself.

1. In the `index.js` file, create a breakpoint for the debugger. To do this, in the editor, next to the line of code `callback(null, response)`, click the gutter just to the left of line 62. A red circle is displayed, representing the breakpoint.

2. On the right edge of the IDE, choose Debugger.

3. Add four expressions for the debugger to watch. To do this, in the Watch Expressions area, for Type an expression here, type `event.option`, and then press Enter. Do this three more times, typing `event.period`, `sc`, and `response.body`.

![Watch Expressions](image)

4. On the run tab that AWS Cloud9 opened in Step 4.1: Run the Function Locally (p. 333), choose the icon that looks like a bug. (It switches from grey to green.)

![Run](image)

5. In the Payload pane on the run tab, be sure the following payload is still visible.

```json
{
    "option": "date",
    "period": "today"
}
```


Code execution pauses at the breakpoint. The current values of `event.option`, `event.period`, `sc`, and `response.body` are displayed in the Watch Expressions area of the Debugger window.

You can also see these values in the code by hovering your mouse over `option` in line 12 in the code, `period` in line 14, `sc` in line 21, and `response` in line 56. For `response` in line 56, expand Object {}, and then see the body value.
Compare your results to the following.

7. In the Debugger window, choose the blue Resume arrow to finish running the code. (It looks like a triangular play button.)

Next Step

Step 5: Run or Debug the API Locally (p. 336)

Step 5: Run or Debug the API Locally

(Previous step: Step 4: Run or Debug the Function Locally (p. 333))

Step 5.1: Run the API Locally

In this step, you use the IDE to have API Gateway run the Lambda function on the instance.

1. In the Lambda pane of the AWS Resources window, right-click the myDateTimeFunction Lambda icon, and then choose Run, Run APIGateway Local.
   
   Note
   You can also do this by choosing the Lambda (local) list on the run tab from the previous step, and then choosing API Gateway (local).

2. On the run tab, for Path, type / (a forward slash).

3. For Method, choose POST.

4. For Body, replace {} with the following, which sends this data as input to the API for processing.

   ```
   {
   "option": "date",
   "period": "today"
   }
   ```

5. If the bug icon is green, choose it to turn it off. (It switches back to grey.)


7. The Response pane displays the following response.

   ```
   {
   "month": 4,
   "day": 12,
   "year": 2018
   }
   ```

8. Run the function several times with the following payloads to see what the Response pane displays.
Step 5: Run or Debug the API Locally

Step 5.2: Debug the API Locally

In this step, you use the IDE to have API Gateway debug the Lambda function on the instance.

1. Ensure that the `index.js` file still has a breakpoint set on the line of code `callback(null, response).`
2. Ensure that the Watch Expressions area of the Debugger window is still watching `event.option`, `event.period`, `sc`, and `response.body`.
3. On the run tab from the previous step, choose the icon that looks like a bug. (It switches from grey to green.)
4. In the Body pane on the run tab, ensure the following is still visible.
5. Choose Run.

   Code execution pauses at the breakpoint, and the current values of event.option, event.period, sc, and response.body are displayed in the Watch Expressions area of the Debugger window.

   You can also see these values in the code by hovering your mouse over option, period, sc, and response in the code. For response, expand Object {}, and then see the body value.

6. In the Debugger window, choose the blue Resume arrow to finish running the code.

Next Step

Step 6: Run the Function in Production (p. 338)

Step 6: Run the Function in Production

(Previous step: Step 5: Run or Debug the API Locally (p. 336))

In this step, you use the IDE to run the function in Lambda itself.

AWS Cloud9 deployed the function to Lambda during Step 2: Create the Lambda Function and API (p. 328). However, AWS Cloud9 deployed the function before you made the changes in Step 3: Add Code to the Function (p. 331). That original function was very basic, taking no payload and returning no response. So, you must first deploy your changes to Lambda, and then you can run the deployed function there.

1. In the Lambda pane of the AWS Resources window, expand Local Functions, expand the MyDateTimeApplication Lambda folder, right-click the myDateTimeFunction Lambda icon, and then choose Deploy.

2. After the deployment finishes, right-click the myDateTimeFunction Lambda icon, and then choose Run, Run Remote.

   **Note**  
   You can also do this by choosing the API Gateway (local) list on the run tab from the previous step, and then choosing Lambda (remote).

3. In the Payload pane on the run tab, be sure one of the payloads from Step 4.1: Run the Function Locally (p. 333) is visible. If it isn’t there, add it.

4. Choose Run, and see the results in the Response area. These should be the same results as in Step 4.1: Run the Function Locally (p. 333).

Next Step

Step 7: Run the API in Production (p. 338)

Step 7: Run the API in Production

(Previous step: Step 6: Run the Function in Production (p. 338))
In this step, you use the IDE to run the API in API Gateway itself.

**Note**
AWS Cloud9 deployed the API to API Gateway during Step 2: Create the Lambda Function and API (p. 328). Because you haven't made any changes to the API since AWS Cloud9 first deployed it, you don't need to deploy the API again before you can run it in API Gateway itself.

1. In the Lambda pane of the AWS Resources window, right-click the myDateTimeFunction Lambda icon, and then choose Run, Run APIGateway Remote.

   **Note**
   You can also do this by choosing the Lambda (remote) list on the run tab from the previous step, and then choosing API Gateway (remote).

2. Ensure Path is still set to /, Method is still set to POST, and Body is still set to one of the bodies in Step 5.1: Run the API Locally (p. 336). If any of these aren't set correctly, set them.

3. Choose Run, and see the results in the Response area, which should be the same results as in Step 5.1: Run the API Locally (p. 336).

**Next Step**

**Step 8: Change the Function and Deploy the Change**

(Previous step: Step 7: Run the API in Production (p. 338))

**Step 8.1: Change the Function Locally**

In this step, you use the IDE to make a small change to the Lambda function. You then test the changed function locally and the original function in production to see the differences.

1. In the index.js file, on line 41, add a time value to the result variable, as follows.

   ```javascript
   result = {
   "time": d.toTimeString(),
   "hour": h,
   "minute": mi,
   "second": s
   };
   ```

2. On line 86, add a date value to the result variable, as follows.

   ```javascript
   result = {
   "date": d.toDateString(),
   "month": mo,
   "day": da,
   "year": y
   };
   ```

3. Save your changes to the index.js file.

4. Run the changed function locally by following the instructions in Step 4.1: Run the Function Locally (p. 333). Notice that the time or date value is now displayed in the response.

5. Run the original function in production by following the instructions in Step 6: Run the Function in Production (p. 338). Notice that the time or date value doesn't display in the response yet. This is because you have not deployed the changed function into production. You do this in the next step.
Step 8.2: Deploy the Changed Function into Production

In this step, you deploy the changed function again to Lambda itself. You then test the changed function in production to confirm the deployment.

1. In the Lambda pane of the AWS Resources window, right-click the myDateTimeFunction Lambda icon, and then choose Deploy.

2. After the deployment succeeds, run the changed function in production by following the instructions in Step 6: Run the Function in Production (p. 338). Notice that the time or date value now appears in the response.

Remember, whenever you make a change to the local function and you want to deploy those changes to Lambda (and API Gateway, if an accompanying API exists), be sure to follow this step.

Next Step

Step 9: Clean Up (p. 340)

Step 9: Clean Up

(Previous step: Step 8: Change the Function and Deploy the Change (p. 339))

To prevent ongoing charges to your AWS account that are related to this tutorial after you're finished, you can delete the function from Lambda, the API from API Gateway, and the environment from AWS Cloud9.

Step 9.1: Delete the Function and the API from Lambda and API Gateway

For AWS Cloud9 to create the function and its associated API, behind the scenes AWS Cloud9 uses the AWS Serverless Application Model (AWS SAM) to create a stack in AWS CloudFormation. This stack then creates the function and its associated API. In this procedure, you use the IDE to have AWS CloudFormation delete the stack, which also deletes the function and the API. (You could use the Lambda and API Gateway consoles instead of AWS CloudFormation to delete the function and its associated API. However, that approach takes longer and still leaves the stack in AWS CloudFormation when it’s no longer needed.)

**Warning**
Deleting a stack cannot be undone. When you delete this stack, the associated function and its API are deleted from Lambda and API Gateway and cannot be recovered.

1. From the IDE, use the AWS Command Line Interface (AWS CLI) in the terminal to run the AWS CloudFormation delete-stack command, specifying the name and the AWS Region ID for the stack. (To display the terminal, on the menu bar, choose Window, New Terminal). This stack’s name follows the format cloud9-APPLICATION_NAME, so you would specify cloud9-MyDateTimeApplication for this tutorial. To get the AWS Region ID (represented in the following command as us-east-2), see the corner of the Lambda pane in the AWS Resources window.

   ```bash
   aws cloudformation delete-stack --stack-name cloud9-MyDateTimeApplication --region us-east-2
   ```

   If the command ran successfully, no output and no error message are displayed.
Note
If you use an IAM user to run this command for this tutorial, instead of an AWS account root user or an IAM administrator, the IAM user must have the following additional AWS access permissions.

- cloudformation:ListStacks
- cloudformation:DeleteStack

If you cannot add these permissions to the IAM user, see your organization's AWS account administrator.

2. To verify that the stack is deleted, use the AWS CLI to run the AWS CloudFormation describe-stacks command. If the function is deleted, a message is displayed that the stack doesn't exist. (You might need to run this command multiple times until that message is displayed.)

```bash
aws cloudformation describe-stacks --query 'Stacks[0].StackStatus' --output text --stack-name cloud9-MyDateTimeApplication --region us-east-2
```

3. If you no longer want to keep the local function in the IDE, delete the ~/environment/MyDateTimeApplication folder (for example, by running the command `rm -rf ~/environment/MyDateTimeApplication`).

   **Note**
   In the IDE, ~/environment is the same as specifying the root directory in the Environment window.

Step 9.2: Delete the Environment from AWS Cloud9

**Warning**
Deleting an environment cannot be undone. Also, when you delete an EC2 environment, AWS Cloud9 also terminates the Amazon EC2 instance that it previously launched and connected to the environment. Once terminated in Amazon EC2, the instance cannot be reactivated or recovered.

1. From the IDE, open the dashboard in the AWS Cloud9 console. To do this, on the menu bar, choose AWS Cloud9, Go To Your Dashboard.

2. Do one of the following:
   - Choose the title inside of the my-demo-lambda-environment card, and then choose Delete.
   - Select the my-demo-lambda-environment card, and then choose Delete.

3. In the Delete dialog box, type Delete, and then choose Delete.

Next Step

Related Information (p. 341)

**Related Information**

The following is some additional information for the AWS Lambda tutorial (p. 325).

Explore any or all of the following topics to continue getting familiar with AWS Cloud9.
## Advanced AWS Lambda Tutorial for AWS Cloud9

In this tutorial, you use the AWS Cloud9 IDE to create a function in AWS Lambda and an accompanying API in Amazon API Gateway. After you create the function and its API, you run and debug them locally. Then you run the function and API in Lambda and API Gateway in production. The function and its API call Amazon Simple Notification Service (Amazon SNS) to send messages to an email address that you specify.

**Note**
Completing this tutorial might result in charges to your AWS account. These include possible charges for Amazon EC2, Lambda, API Gateway, and Amazon SNS. For more information, see [Amazon EC2 Pricing](https://aws.amazon.com/ec2/pricing/), [AWS Lambda Pricing](https://aws.amazon.com/lambda/pricing/), [Amazon API Gateway Pricing](https://aws.amazon.com/api-gateway/pricing/), and [Amazon SNS Pricing](https://aws.amazon.com/sns/pricing/).

---

<table>
<thead>
<tr>
<th>Task</th>
<th>See this topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more about how to use AWS Cloud9 with Lambda</td>
<td>Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342) and Working with AWS Lambda Functions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 284)</td>
</tr>
<tr>
<td>Learn more about the AWS Cloud9 IDE</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your environment with you, in real time and with text chat support</td>
<td>Working with Shared Environments in AWS Cloud9 (p. 80)</td>
</tr>
<tr>
<td>Create SSH environments (environments that use cloud compute instances or servers that you create, instead of an Amazon EC2 instances that AWS Cloud9 creates for you).</td>
<td>Creating an Environment in AWS Cloud9 (p. 55) and AWS Cloud9 SSH Development Environment host requirements (p. 479)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with Amazon Lightsail</td>
<td>Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with the AWS CLI, the aws-shell, AWS CodeCommit, the AWS Cloud Development Kit (AWS CDK), GitHub, or Amazon DynamoDB, as well as Node.js, Python, or other programming languages</td>
<td>Tutorials and Samples (p. 325)</td>
</tr>
<tr>
<td>Work with code for intelligent robotics applications in AWS RoboMaker.</td>
<td>Developing with AWS Cloud9 in the AWS RoboMaker Developer Guide</td>
</tr>
</tbody>
</table>

---

To get help with AWS Cloud9 from the community, see the [AWS Cloud9 Discussion Forum](https). (When you enter this forum, AWS might require you to sign in.)

To get help with AWS Cloud9 directly from AWS, see the support options on the [AWS Support](https) page.
Prerequisites

Before you start this tutorial, we recommend that you first complete the companion AWS Lambda Tutorial for AWS Cloud9 (p. 325). This tutorial builds on the prerequisites and concepts that are presented there.

Note
If you don’t want to complete that entire tutorial first, you must at least complete the following steps in that tutorial or else create the equivalent AWS resources.

- Prerequisites (p. 325): This includes deciding which user in an AWS account or organization you’ll use to complete this tutorial.
- Step 1: Create and Open the Environment (p. 326): This includes creating an AWS Cloud9 EC2 development environment and opening the AWS Cloud9 IDE for that environment.

Step 1: Create the Lambda Function and API

In this step, you use the AWS Cloud9 IDE to create the Lambda function and its accompanying API at the same time. AWS Cloud9 stores the new function on the Amazon EC2 instance and deploys a copy of the function to Lambda. AWS Cloud9 also stores the new accompanying API on the instance and deploys a copy of the API to API Gateway.

The function uses Amazon Simple Notification Service (Amazon SNS) to send messages to your email address. In a later step, you create the Amazon SNS resources that this function needs.

Currently, you can use the IDE to create functions that use only Node.js or Python. This function uses Node.js.

1. With the IDE displayed for the environment, on the menu bar, choose AWS Cloud9, Preferences.
2. In the navigation pane of the Preferences tab, choose AWS Settings.
3. For AWS Region, select the AWS Region you want to create the function in.
4. On the edge of the IDE, choose **AWS Resources**.
5. Expand the **Lambda** pane, if it's not already expanded. On the toolbar, choose **Create a new Lambda function**.

6. In the **Create serverless application** dialog box, for **Function name**, type a name for the function (for example, mySNSFunction).
7. For **Application name**, type a name for the function's containing serverless application (for example, MySNSApplication).
8. Choose **Next**.
9. For **Select runtime**, choose **Node.js 12.x**.
10. For **Select blueprint**, choose **sns-send-message**. (You might need to scroll through the list of blueprints to see it.)
11. Choose **Next**.
12. For **Function trigger**, choose **API Gateway**.
13. For **Resource Path**, type /.
14. For **Security**, choose **NONE**, and then choose **Next**.
15. For **Memory (MB)**, leave the default value of **128 MB**, which is sufficient for this tutorial.
16. For **Role**, leave the default value of **Automatically generate role**, and then choose **Next**. (You change this role in the next step.)
17. Review your choices, and then choose **Finish**.
Step 2: Set up Amazon SNS

In this tutorial, the Lambda function uses Amazon SNS to send messages to your email address. In this step, you run an AWS CloudFormation stack that quickly creates a topic in Amazon SNS and then subscribes your email address to the topic. The stack also creates an execution role in IAM to allow the Lambda function permission to use Amazon SNS. (You could do all of this setup in Amazon SNS and IAM manually, but AWS CloudFormation makes this setup easier and faster.) After AWS CloudFormation creates the stack, you attach the execution role to the function, and then give Amazon SNS permission to start sending messages to your email address.

1. In the terminal, change to the directory that contains the AWS CloudFormation template file named sns-create-topic-subscription.yaml.

```
cd ~/environment/MySNSApplication
```

**Note**
In the IDE, `~/environment` is the same as specifying the root directory in the Environment window.

2. Use the AWS CLI to run the following command, which creates and runs a AWS CloudFormation stack based on this template file.

```
aws cloudformation create-stack --template-body file://sns-create-topic-subscription.yaml --capabilities CAPABILITY_NAMED_IAM --parameters ParameterKey=SNSTopicName,ParameterValue=MySNSTopic ParameterKey=EmailAddress,ParameterValue=me@example.com --stack-name MySNSStack --region us-east-2
```

In the preceding command, do the following.

- Replace MySNSTopic with whatever you want to name the Amazon SNS topic to send messages to.
- Replace me@example.com with your email address for Amazon SNS to send messages to.
- Replace MySNSStack with whatever you want to name the stack.
- Replace us-east-2 with the ID of the AWS Region where you created the function (see the corner of the Lambda pane in the AWS Resources window).

**Note**
If you use an IAM user to call AWS CloudFormation for this tutorial, instead of an AWS account root user or an IAM administrator user, the IAM user must have the following additional AWS access permissions.

- cloudformation:CreateUploadBucket
- cloudformation:GetTemplateSummary
- cloudformation:ListStacks
- iam:CreateRole
- iam:PutRolePolicy
Step 3: Run the Function Locally

AWS Cloud9 User Guide

In this step, you use the IDE to run the newly created function on the instance, which sends messages to your email address. Currently, you can use the IDE to run functions that use only Node.js or Python.

1. With the IDE still displayed for the environment, in the Lambda pane of the AWS Resources window, expand Local Functions, expand the MySNSApplication Lambda folder, right-click the mySNSFunction Lambda icon, and then choose Run, Run Local.

2. In the Payload pane on the run tab, replace the pane's contents with the following, which sends the specified data to the function for processing.

```json
{
"region": "us-east-2",
"message": "You just sent an email by using Amazon SNS.",
"subject": "Hello from Amazon SNS",
"topicARN": "arn:aws:sns:us-east-2:123456789012:MySNSTopic"
}
```
In the preceding payload, do the following.

- Replace `us-east-2` with the ID of the AWS Region where the Amazon SNS topic exists.
- Replace `123456789012` with your AWS account ID.
- Replace `MySNSTopic` with the name of the Amazon SNS topic.

3. Choose **Run**.

4. If the response shows a **statusCode** of **200**, then in a few minutes, check your email for the message that was sent.

Compare your results to the following.

---

**Step 4: Debug the Function Locally**

In this step, you use the IDE to debug the function on the instance. Currently, you can use the IDE to debug functions that use only Node.js or Python. Also, you can use the IDE to debug functions locally only. You cannot use the IDE to debug functions in Lambda itself.

1. With the IDE still displayed for the environment, open the **index.js** file.

2. Create a breakpoint for the debugger. To do this, in the editor, click inside the gutter next to the line of code that starts with `sns.publish` on line 45. A red circle is displayed, representing the breakpoint.

3. On the edge of the IDE, choose **Debugger**.

4. Add four expressions for the debugger to watch. To do this, in the **Watch Expressions** area, for **Type an expression here**, type `event['subject']`, and then press Enter. Do this two more times, typing `context['memoryLimitInMB']` and `sns.endpoint.hostname`.

   **Note**
   If you completed the previous Lambda tutorial, you can delete any of those watch expressions that might still be there. To do this, simply right-click an expression, and then choose **Remove Watch Expression**.

5. On the run tab from the previous step, choose the icon that looks like a bug. (It will switch from grey to green.)

6. Choose **Run**.

   Code execution pauses at the breakpoint and displays the current values of the message's subject line, the function's memory limit in megabytes, and the Amazon SNS service's hostname.
You can also see these values by hovering your mouse over event, context, and sns in the code, followed by expanding the screen tip that is displayed.

7. In the Debugger window, choose the blue Resume button to finish running the code.

8. On the run tab, if the response shows a statusCode of 200, then in a few minutes, check your email for the message that was sent.

**Step 5: Run the API Locally**

In this step, you use the IDE to have API Gateway run the Lambda function on the instance.

1. In the Lambda pane of the AWS Resources window, right-click the mySNSFunction Lambda icon, and then choose Run, Run APIGateway Local.

   **Note**
   You can also do this by choosing the Lambda (local) list on the run tab from the previous step, and then choosing API Gateway (local).

2. On the run tab, for Path, type /.

3. For Method, choose POST.

4. For Body, replace the pane's contents with the following, which sends this data as input to the API for processing.

   ```json
   {
   "region": "us-east-2",
   "message": "You just sent an email by using Amazon SNS.",
   "subject": "Hello from Amazon SNS",
   "topicARN": "arn:aws:sns:us-east-2:123456789012:MyDemoSNSTopic"
   }
   ```

5. If the bug icon is green, choose it to turn it off. (It switches back to grey.)


7. If the response shows success, then in a few minutes, check your email for the message that was sent.

   **Note**
   If the response is Missing required key 'Message' in params instead of success, then add the following code to your index.js file, save the file, and then try repeating this step again.

   ```javascript
   var sns = new AWS.SNS({apiVersion: '2010-03-31'});
   
   // Begin adding code here.
   if (event.body) {
     event = JSON.parse(event.body);
   }
   
   // End adding code here.
   var params ={
     Message: event['message'],
     Subject: event['subject'],
     TopicArn: event['topicARN']
   };
   ```

   For more information, see Response Differences When Invoking a Lambda Function from API Gateway (p. 305).
Step 6: Debug the API Locally

In this step, you use the IDE to have API Gateway debug the Lambda function on the instance.

1. Ensure that the `index.js` file still has a breakpoint set on the line of code `sns.publish`.
2. Ensure that the **Watch Expressions** area of the **Debugger** window is still watching `event['subject']`, `context['memoryLimitInMB']`, and `sns.endpoint.hostname`.
3. On the run tab from the previous step, choose the icon that looks like a bug. (It switches from grey to green.)
4. Choose **Run**.

   Code execution pauses at the breakpoint and displays the current values of the message's subject line, the function's memory limit in megabytes, the Amazon SNS service's hostname, and the caller's AWS access key ID.

   You can also see these values by hovering your mouse over `event`, `context`, and `sns` followed by expanding the screen tip that is displayed.

5. In the **Debugger** window, choose the blue **Resume** button to finish running the code.
6. If the response shows **success**, then in a few minutes, check your email for the message that was sent.

Step 7: Deploy and Run the Changed Function in Production

In this step, you deploy the function that you changed in **Step 5: Run the API Locally** (p. 348) to Lambda and API Gateway. You then test the changes in production to confirm the deployment.

1. In the **Lambda** pane of the **AWS Resources** window, right-click the `mySNSFunction` Lambda icon, and then choose **Deploy**.
2. After the deployment succeeds, run the changed function in production. To do this, in the **Lambda** pane of the **AWS Resources** window, right-click the `mySNSFunction` Lambda icon, and then choose **Run, Run Remote**.

   **Note**
   
   You can also do this by choosing the **API Gateway (local)** list on the run tab from the previous step, and then choosing **Lambda (remote)**.

3. Ensure that the **Payload** pane on the run tab still contains the following data.

   ```json
   {
   "region": "us-east-2",
   "message": "You just sent an email by using Amazon SNS.",
   "subject": "Hello from Amazon SNS",
   "topicARN": "arn:aws:sns:us-east-2:123456789012:MyDemoSNSTopic"
   }
   ```

4. Choose **Run**.
5. If the response shows a **statusCode** of **200**, then in a few minutes, check your email for the message that was sent.
6. Run the API in production. To do this, in the **Lambda** pane of the **AWS Resources** window, right-click the `mySNSFunction` Lambda icon, and then choose **Run, Run APIGateway Remote**.

   **Note**
   
   You can also do this by choosing the **Lambda (remote)** list on the run tab, and then choosing **API Gateway (remote)**.
7. On the run tab, for **Path**, type `/.`.
8. For **Method**, choose **POST**.
9. For **Body**, be sure the following data is still displayed.

   ```json
   {
     "region": "us-east-2",
     "message": "You just sent an email by using Amazon SNS.",
     "subject": "Hello from Amazon SNS",
     "topicARN": "arn:aws:sns:us-east-2:123456789012:MyDemoSNSTopic"
   }
   ```
10. Choose **Run**.
11. If the response shows **success**, then in a few minutes, check your email for the message that was sent.

### Step 8: Clean Up

To prevent ongoing charges to your AWS account related to this tutorial, you can delete the function from Lambda, the API from API Gateway, the topic and subscription from Amazon SNS, the Lambda execution role from IAM, and the environment from AWS Cloud9.

### Step 8.1: Delete the Function and the API from Lambda and API Gateway

For AWS Cloud9 to create the function and its associated API, behind the scenes AWS Cloud9 uses the AWS Serverless Application Model (AWS SAM) to create a stack in AWS CloudFormation. This stack then creates the function and its associated API. In this procedure, you use the IDE to have AWS CloudFormation delete the stack, which also deletes the function and the API. (You could use the Lambda and API Gateway consoles instead of AWS CloudFormation to delete the function and its associated API. However, that approach takes longer and still leaves the stack in AWS CloudFormation when it's no longer needed.)

**Warning**
Deleting a stack cannot be undone. When you delete this stack, the associated function and its API are deleted from Lambda and API Gateway and cannot be recovered.

1. From the IDE, use the AWS CLI in the terminal to run the `aws cloudformation delete-stack` command, specifying the name of the stack. This stack's name follows the format `cloud9-APPLICATION_NAME`, so you would specify `cloud9-MySNSApplication` for this tutorial.

   ```bash
   aws cloudformation delete-stack --stack-name cloud9-MySNSApplication --region us-east-2
   ```

   If the command ran successfully, no output and no error message are displayed.

   **Note**
   If you use an IAM user to run this command for this tutorial, instead of an AWS account root user or an IAM administrator user, the IAM user must have the following additional AWS access permissions:
   - `cloudformation:ListStacks`
   - `cloudformation:DeleteStack`
   
   If you cannot add these permissions to the IAM user, see your organization's AWS account administrator.

2. To verify that the stack is deleted, use the AWS CLI to run the `aws cloudformation describe-stacks` command. If the function is deleted, a message is displayed that the stack doesn't exist.
Step 8: Clean Up

3. If you no longer want to keep the local function in the IDE, delete the ~/environment/MySNSApplication folder (for example, by running the command `rm -rf ~/environment/MySNSApplication`).

Step 8.2: Delete the Topic and Subscription from Amazon SNS and the Lambda Execution Role from IAM

When you delete the AWS CloudFormation stack that you created in Step 2: Set up Amazon SNS (p. 345), the Amazon SNS topic and subscription are deleted, as well as the execution role for the Lambda function.

**Warning**
Deleting a stack cannot be undone. When you delete this stack, the associated topic, subscription, and execution role are deleted from Amazon SNS and IAM and cannot be recovered.

1. With the IDE still displayed for the environment, use the AWS CLI in the terminal to run the AWS CloudFormation `delete-stack` command, specifying the name of the stack.

```bash
aws cloudformation delete-stack --stack-name MySNSStack --region us-east-2
```

**Note**
If you use an IAM user to run this command, instead of an AWS account root user or an IAM administrator user, the IAM user must have the following additional AWS access permissions.

- cloudFormation:DeleteStack
- iam:DeleteRole
- iam:DeleteRolePolicy
- sns:DeleteTopic
- sns:Unsubscribe

If you cannot add these permissions to the IAM user, see your organization's AWS account administrator.

If the command ran successfully, no output and no error message are displayed.

2. To verify that the stack is deleted, use the AWS CLI to run the following command.

```bash
aws cloudformation describe-stacks --query 'Stacks[0].StackStatus' --output text --stack-name MySNSStack --region us-east-2
```

Keep running the preceding command until the output states that the stack doesn’t exist.

Step 8.3: Delete the Environment from AWS Cloud9

**Warning**
Deleting an environment cannot be undone. Also, when you delete an EC2 environment, AWS Cloud9 also terminates the Amazon EC2 instance that it previously launched and connected to the environment. Once terminated in Amazon EC2, the instance cannot be reactivated or recovered.
1. With the IDE still displayed for the environment, open the dashboard in the AWS Cloud9 console. To do this, on the menu bar in the IDE, choose AWS Cloud9, Go To Your Dashboard.

2. Do one of the following.
   - Choose the title that matches the name of the environment, and then choose Delete.
   - Select the card that contains the name of the environment, and then choose Delete.

3. In the Delete dialog box, type Delete, and then choose Delete.

Next Steps

Explore any or all of the following topics to continue getting familiar with AWS Cloud9.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn more about how to use AWS Cloud9 with Lambda</td>
<td>Working with AWS Lambda Functions (p. 284)</td>
</tr>
<tr>
<td>Learn more about the AWS Cloud9 IDE</td>
<td>Getting Started: Basic Tutorials (p. 35) and Working with the IDE (p. 99)</td>
</tr>
<tr>
<td>Invite others to use your environment with you, in real time and with text chat support</td>
<td>Working with Shared Environments (p. 80)</td>
</tr>
<tr>
<td>Create SSH environments (environments that use cloud compute instances or servers that you create, instead of an Amazon EC2 instances that AWS Cloud9 creates for you).</td>
<td>Creating an Environment (p. 55) and SSH Environment Host Requirements (p. 479)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with Amazon Lightsail</td>
<td>Working with Amazon Lightsail Instances (p. 276)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with AWS CodePipeline</td>
<td>Working with AWS CodePipeline (p. 322)</td>
</tr>
<tr>
<td>Use AWS Cloud9 with the AWS CLI, the aws-shell, AWS CodeCommit, the AWS Cloud Development Kit (AWS CDK), GitHub, or Amazon DynamoDB, as well as Node.js, Python, or other programming languages</td>
<td>Tutorials and Samples (p. 325)</td>
</tr>
<tr>
<td>Work with code for intelligent robotics applications in AWS RoboMaker</td>
<td>Developing with AWS Cloud9 in the AWS RoboMaker Developer Guide</td>
</tr>
</tbody>
</table>

To get help with AWS Cloud9 from the community, see the AWS Cloud9 Discussion Forum. (When you enter this forum, AWS might require you to sign in.)

To get help with AWS Cloud9 directly from AWS, see the support options on the AWS Support page.

AWS Command Line Interface and aws-shell Sample for AWS Cloud9

This sample enables you to set up the AWS Command Line Interface (AWS CLI), the aws-shell, or both in an AWS Cloud9 development environment. The AWS CLI and the aws-shell are unified tools that provide a consistent interface for interacting with all parts of AWS. You can use the AWS CLI or the aws-shell...
instead of the AWS Management Console to quickly run commands to interact with AWS, and some of these commands can only be run with the AWS CLI or the aws-shell.

For more information about the AWS CLI, see the AWS Command Line Interface User Guide. For the aws-shell, see the following resources:

- aws-shell on the GitHub website
- aws-shell on the pip website

For a list of commands you can run with the AWS CLI to interact with AWS, see the AWS CLI Command Reference. You use the same commands with the aws-shell, except that you start commands without the aws prefix.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

**Prerequisites**

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

**Step 1: Install the AWS CLI, the aws-shell, or Both in Your Environment**

In this step, you use the AWS Cloud9 IDE to install the AWS CLI, the aws-shell, or both in your environment so you can run commands to interact with AWS.

If you are using an AWS Cloud9 EC2 development environment and you only want to use the AWS CLI, you can skip ahead to Step 3: Run Some Basic Commands with the AWS CLI or the aws-shell in Your Environment (p. 355). This is because the AWS CLI is already installed in an EC2 environment, and a set of AWS access credentials is already set up in the environment. For more information, see AWS managed temporary credentials (p. 536).

If you are not using an EC2 environment, do the following to install the AWS CLI:

1. With your environment open, in the IDE, check whether the AWS CLI is already installed. In the terminal, run the `aws --version` command. (To start a new terminal session, on the menu bar,
Step 2: Set up Credentials Management in Your Environment

Each time you use the AWS CLI or the aws-shell to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS CLI or the aws-shell has the appropriate permissions to make that call. If the credentials don't cover the appropriate permissions, the call will fail.
Step 3: Run Some Basic Commands with the AWS CLI or the aws-shell in Your Environment

In this step, you use the AWS CLI or the aws-shell in your environment to create a bucket in Amazon S3, list your available buckets, and then delete the bucket.

1. If you want to use the aws-shell but haven’t started it yet, start the aws-shell by running the `aws-shell` command. The `aws>` prompt is displayed.

2. Create a bucket. Run the `aws s3 mb` command with the AWS CLI or the `s3 mb` command with the aws-shell, supplying the name of the bucket to create. In this example, we use a bucket named `cloud9-123456789012-bucket`, where `123456789012` is your AWS account ID. If you use a different name, substitute it throughout this step.

   ```bash
   aws s3 mb s3://cloud9-123456789012-bucket # For the AWS CLI.
   s3 mb s3://cloud9-123456789012-bucket     # For the aws-shell.
   ```

   **Note**
   Bucket names must be unique across all of AWS, not just your AWS account. The preceding suggested bucket name can help you come up with a unique bucket name. If you get a message that contains the error `BucketAlreadyExists`, you must run the command again with a different bucket name.

3. List your available buckets. Run the `aws s3 ls` command with the AWS CLI or the `s3 ls` command with the aws-shell. A list of your available buckets is displayed.

4. Delete the bucket. Run the `aws s3 rb` command with the AWS CLI or the `s3 rb` command with the aws-shell, supplying the name of the bucket to delete.

   ```bash
   aws s3 rb s3://cloud9-123456789012-bucket # For the AWS CLI.
   s3 rb s3://cloud9-123456789012-bucket     # For the aws-shell.
   ```

To confirm whether the bucket was deleted, run the `aws s3 ls` command again with the AWS CLI or the `s3 ls` command again with the aws-shell. The name of the bucket that was deleted should no longer appear in the list.

**Note**
You don’t have to delete the bucket if you want to keep using it. For more information, see Add an Object to a Bucket in the Amazon Simple Storage Service Getting Started Guide. See also `s3` commands in the AWS CLI Command Reference. (Remember, if you don’t delete the bucket, it might result in ongoing charges to your AWS account.)

To continue experimenting with the AWS CLI, see Working with Amazon Web Services in the AWS Command Line Interface User Guide as well as the AWS CLI Command Reference. To continue experimenting with the aws-shell, see the AWS CLI Command Reference, noting that you start commands without the `aws` prefix.
Step 4: Clean Up

If you're using the aws-shell, you can stop using it by running the `.exit` or `.quit` command.

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

AWS CodeCommit Sample for AWS Cloud9

This sample enables you to set up an AWS Cloud9 development environment to interact with a remote code repository in CodeCommit. CodeCommit is a source code control service that enables you to privately store and manage Git repositories in the AWS Cloud. For more information about CodeCommit, see the AWS CodeCommit User Guide.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and CodeCommit. For more information, see Amazon EC2 Pricing and AWS CodeCommit Pricing.

- Prerequisites (p. 356)
- Step 1: Set Up Your IAM Group with Required Access Permissions (p. 356)
- Step 2: Create a Repository in AWS CodeCommit (p. 357)
- Step 3: Connect Your Environment to the Remote Repository (p. 358)
- Step 4: Clone the Remote Repository into Your Environment (p. 359)
- Step 5: Add Files to the Repository (p. 360)
- Step 6: Clean Up (p. 361)

Prerequisites

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Set Up Your IAM Group with Required Access Permissions

If your AWS credentials are associated with an IAM administrator user in your AWS account, and you want to use that user to work with CodeCommit, skip ahead to Step 2: Create a Repository in AWS CodeCommit (p. 357).

You can complete this step using the AWS Management Console (p. 357) or the AWS Command Line Interface (AWS CLI) (p. 357).
Set Up Your IAM Group with Required Access Permissions Using the Console

1. Sign in to the AWS Management Console, if you are not already signed in.

   For this step, we recommend you sign in using credentials for an IAM administrator user in your AWS
account. If you cannot do this, check with your AWS account administrator.

2. Open the IAM console. To do this, in the console's navigation bar, choose Services. Then choose IAM.

3. Choose Groups.

4. Choose the group’s name.

5. On the Permissions tab, for Managed Policies, choose Attach Policy.

6. In the list of policy names, select one of the following boxes:
   • Select AWSCodeCommitPowerUser for access to all of the functionality of CodeCommit and
     repository-related resources, except it does not allow deletion of CodeCommit repositories or create
     or delete repository-related resources in other AWS services, such as Amazon CloudWatch Events.
   • Select AWSCodeCommitFullAccess for full control over CodeCommit repositories and related
     resources in the AWS account, including the ability to delete repositories.

   (If you don’t see either of these policy names in the list, type the policy name in the Filter box to
   display it.)

7. Choose Attach Policy.

To see the list of access permissions that these AWS managed policies give to a group, see AWS Managed

Skip ahead to Step 2: Create a Repository in AWS CodeCommit (p. 357).

Set Up Your IAM Group with Required Access Permissions Using the AWS CLI

Run the IAM attach-group-policy command, specifying the group's name and the Amazon Resource
Name (ARN) of the AWS managed policy that describes the required access permissions, for example:

```
aws iam attach-group-policy --group-name MyGroup --policy-arn POLICY_ARN
```

In the preceding command, replace MyGroup with the name of the group. Replace POLICY_ARN with the
ARN of the AWS managed policy, as follows:

• arn:aws:iam::aws:policy/AWSCodeCommitPowerUser for access to all of the functionality
  of CodeCommit and repository-related resources, except it does not allow deletion of CodeCommit
  repositories or create or delete repository-related resources in other AWS services, such as Amazon
  CloudWatch Events.
• arn:aws:iam::aws:policy/AWSCodeCommitFullAccess for full control over CodeCommit
  repositories and related resources in the AWS account, including the ability to delete repositories.

To see the list of access permissions that these AWS managed policies give to a group, see AWS Managed

Step 2: Create a Repository in CodeCommit

In this step, you create a remote code repository in CodeCommit by using the CodeCommit console.
If you already have a CodeCommit repository, skip ahead to Step 3: Connect Your Environment to the Remote Repository (p. 358).

You can complete this step using the AWS Management Console (p. 358) or the AWS Command Line Interface (AWS CLI) (p. 358).

Create a Repository in CodeCommit Using the Console

1. If you are signed in to the AWS Management Console as an IAM administrator user from the previous step, and you do not want to use the IAM administrator user to create the repository, sign out of the AWS Management Console.
3. In the console's navigation bar, use the region selector to choose the AWS Region you want to create the repository in (for example, US East (Ohio)).
4. If a welcome page is displayed, choose Get started. Otherwise, choose Create repository.
5. On the Create repository page, for Repository name, type a name for your new repository, for example MyDemoCloud9Repo. If you choose a different name, substitute it throughout this sample.
6. (Optional) For Description, type something about the repository, for example This is a demonstration repository for the AWS Cloud9 sample.
7. Choose Create repository. A Connect to your repository pane is displayed. Choose Close, as you will connect to your repository in a different way later in this topic.

Skip ahead to Step 3: Connect Your Environment to the Remote Repository (p. 358).

Create a Repository in CodeCommit Using the AWS CLI

Run the AWS CodeCommit create-repository command, specifying the repository's name, an optional description, and the AWS Region to create the repository in, for example:

```
aws codecommit create-repository --repository-name MyDemoCloud9Repo --repository-description "This is a demonstration repository for the AWS Cloud9 sample." --region us-east-2
```

In the preceding command, replace us-east-2 with the ID of the AWS Region to create the repository in. For a list of supported regions, see AWS CodeCommit in the Amazon Web Services General Reference.

If you choose to use a different repository name, substitute it throughout this sample.

Step 3: Connect Your Environment to the Remote Repository

In this step, you use the AWS Cloud9 IDE to connect to the CodeCommit repository you created or identified in the previous step.

Complete one of the following sets of procedures, depending on the type of AWS Cloud9 development environment you have.

<table>
<thead>
<tr>
<th>Environment type</th>
<th>Follow these procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC2 environment</td>
<td>1. From a terminal session in the IDE, run the following 2 commands:</td>
</tr>
<tr>
<td></td>
<td><code>git config --global credential.helper '!aws codecommit credential-helper @'</code></td>
</tr>
</tbody>
</table>
### Step 4: Clone the Remote Repository into Your Environment

In this step, you use the AWS Cloud9 IDE to clone the remote repository in CodeCommit into your environment.

To clone the repository, run the `git clone` command, supplying the repository's clone URL, shown here as `CLONE_URL`.

```bash
git clone CLONE_URL
```

For an EC2 environment, you supply an HTTPS clone URL that starts with `https://`. For an SSH environment, you supply an SSH clone URL that starts with `ssh://`.

To get the repository's full clone URL, see Use the AWS CodeCommit Console to View Repository Details in the *AWS CodeCommit User Guide*.

If your repository doesn't have any files in it, a warning message is displayed, such as `You appear to have cloned an empty repository`. This is expected behavior, which you will address later.

---

<table>
<thead>
<tr>
<th>Environment type</th>
<th>Follow these procedures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>git config --global credential.UseHttpPath true</code></td>
</tr>
<tr>
<td></td>
<td>For more information, see Step 2: Configure the AWS CLI Credential Helper On Your AWS Cloud9 EC2 Development Environment in Integrate AWS Cloud9 with AWS CodeCommit in the <em>AWS CodeCommit User Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>2. Skip ahead to Step 4: Clone the Remote Repository into Your Environment (p. 359), later in this topic.</td>
</tr>
<tr>
<td>SSH environment</td>
<td>1. If Git is not already installed in the environment, use a terminal session in the IDE to install it. For more information, see Step 2: Install Git in Setup Steps for SSH Connections to AWS CodeCommit Repositories on Linux, macOS, or Unix in the <em>AWS CodeCommit User Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>2. Complete Step 3: Configure Credentials on Linux, macOS, or Unix in Setup Steps for SSH Connections to AWS CodeCommit Repositories on Linux, macOS, or Unix in the <em>AWS CodeCommit User Guide</em>.</td>
</tr>
<tr>
<td></td>
<td>When you are instructed to sign in to the AWS Management Console and open the IAM console, we recommend you sign in using credentials for an IAM administrator user in your AWS account. If you cannot do this, check with your AWS account administrator.</td>
</tr>
<tr>
<td></td>
<td>3. Skip ahead to Step 4: Clone the Remote Repository into Your Environment (p. 359), later in this topic.</td>
</tr>
</tbody>
</table>
Step 5: Add Files to the Repository

In this step, you create three simple files in the cloned repository in your environment. Then you add the files to the Git staging area in your cloned repository, commit the staged files, and push the commit to your remote repository in CodeCommit.

If the cloned repository already has files in it, you're done and can skip the rest of this sample.

To add files to the repository

2. Type the following content into the file, and then choose File, Save to save the file as bird.txt in the MyDemoCloud9Repo directory in your environment.

   bird.txt
   --------
   Birds are a group of endothermic vertebrates, characterized by feathers, toothless beaked jaws, the laying of hard-shelled eggs, a high metabolic rate, a four-chambered heart, and a lightweight but strong skeleton.

   Note
   To confirm you are saving this file in the correct directory, in the Save As dialog box, choose the MyDemoCloud9Repo folder, and be sure Folder displays /MyDemoCloud9Repo.

3. Create two more files, named insect.txt and reptile.txt, with the following content, and saving them in the same MyDemoCloud9Repo directory.

   insect.txt
   ---------
   Insects are a class of invertebrates within the arthropod phylum that have a chitinous exoskeleton, a three-part body (head, thorax, and abdomen), three pairs of jointed legs, compound eyes, and one pair of antennae.

   reptile.txt
   -----------
   Reptiles are tetrapod (four-limbed vertebrate) animals in the class Reptilia, comprising today’s turtles, crocodilians, snakes, amphisbaenians, lizards, tuatara, and their extinct relatives.

4. In the terminal, run the cd command to switch to the MyDemoCloud9Repo directory.

   cd MyDemoCloud9Repo

5. Confirm the files were successfully saved in the MyDemoCloud9Repo directory by running the git status command. All three files will be listed as untracked files.

   Untracked files:
   (use "git add <file>..." to include in what will be committed)
   
   bird.txt
   insect.txt
   reptile.txt

6. Add the files to the Git staging area by running the git add command.

   git add --all
7. Confirm the files were successfully added to the Git staging area by running the `git status` command again. All three files are now listed as changes to commit.

```
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
  new file:   bird.txt
  new file:   insect.txt
  new file:   reptile.txt
```

8. Commit the staged files by running the `git commit` command.

```
git commit -m "Added information about birds, insects, and reptiles."
```

9. Push the commit to your remote repository in CodeCommit by running the `git push` command.

```
git push -u origin master
```

10. Confirm whether the files were successfully pushed. Open the CodeCommit console, if it isn't already open, at https://console.aws.amazon.com/codecommit.

11. In the top navigation bar, near the right edge, choose the AWS Region where you created the repository (for example, **US East (Ohio)**).

12. On the Dashboard page, choose **MyDemoCloud9Repo**. The three files are displayed.

To continue experimenting with your CodeCommit repository, see Browse the Contents of Your Repository in the **AWS CodeCommit User Guide**.

If you're new to Git and you don't want to mess up your CodeCommit repository, experiment with a sample Git repository on the **Try Git** website.

**Step 6: Clean Up**

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the CodeCommit repository. For instructions, see Delete an AWS CodeCommit Repository in the **AWS CodeCommit User Guide**.

You should also delete the environment. For instructions, see Deleting an Environment (p. 95).

**GitHub Sample for AWS Cloud9**

This sample enables you to set up an AWS Cloud9 development environment to interact with a remote code repository in GitHub. For more information about GitHub, see the **GitHub** and **GitHub Help** websites.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2. For more information, see **Amazon EC2 Pricing**.

**Topics**

- Prerequisites (p. 362)
- Step 1: Create a GitHub Account (p. 362)
- Step 2: Create a GitHub Repository (p. 362)
- Step 3: Install Git in Your Environment (p. 362)
- Step 4: Clone the Remote Repository into Your Environment (p. 363)
- Step 5: Add Files to the Repository (p. 363)
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See *Creating an Environment in AWS Cloud9* (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See *Opening an Environment in AWS Cloud9* (p. 67) for details.

Start with the following step, depending on what you already have.

<table>
<thead>
<tr>
<th>Do you have a GitHub account?</th>
<th>Do you have a GitHub repository?</th>
<th>Do you have Git installed?</th>
<th>Start with this step</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>--</td>
<td>--</td>
<td>Step 1: Create a GitHub Account (p. 362)</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>--</td>
<td>Step 2: Create a GitHub Repository (p. 362)</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No (or Not Sure)</td>
<td>Step 3: Install Git in Your Environment (p. 362)</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Step 4: Clone the Remote Repository into Your Environment (p. 363)</td>
</tr>
</tbody>
</table>

**Step 1: Create a GitHub Account**

If you already have a GitHub account, skip ahead to *Step 2: Create a GitHub Repository* (p. 362).

To create a GitHub account, see *Join GitHub* on the GitHub website.

**Step 2: Create a GitHub Repository**

If you already have a GitHub repository, skip ahead to *Step 3: Install Git in Your Environment* (p. 362).

To create the repository, see *Create A Repo* on the GitHub Help website.

**Step 3: Install Git in Your Environment**

In this step, you use the AWS Cloud9 IDE to install Git in your environment so that you can clone your remote repository into the environment later.
If you already have Git installed in your environment, skip ahead to Step 4: Clone the Remote Repository into Your Environment (p. 363). To check whether you already have Git installed, run the `git --version` command as described in this step.

1. With your environment open, in the AWS Cloud9 IDE, start a new terminal session, if one isn’t already started. (To start a new terminal session, on the menu bar, choose Window, New Terminal.)
2. Check whether Git is already installed. In the terminal, run the `git --version` command. If Git is installed, the version number is displayed, for example, `git version N.N.N`. The installed version must be 1.7.9 or later. If it is, skip ahead to step 4 in this procedure to set your Git name and email address.
3. To install Git, see Git Downloads on the Git website. For example, for an EC2 environment running Amazon Linux, run these three commands in the terminal, one at a time, to install Git.

   ```bash
   sudo yum -y update      # Install the latest system updates.
   sudo yum -y install git # Install Git.
   git --version           # Confirm Git was installed.
   ```

   For an EC2 environment running Ubuntu Server, run these three commands in the terminal instead, one at a time, to install Git.

   ```bash
   sudo apt update      # Install the latest system updates.
   sudo apt install -y git # Install Git.
   git --version           # Confirm Git was installed.
   ```

4. Set your Git name and email address. In the terminal, run these two commands, one at a time, substituting your Git name and email address for `USER_NAME` and `EMAIL_ADDRESS`.

   ```bash
   git config --global user.name "USER_NAME"
   git config --global user.email EMAIL_ADDRESS
   ```

---

**Step 4: Clone the Remote Repository into Your Environment**

In this step, you use the AWS Cloud9 IDE to clone the remote repository in GitHub into your environment.

To clone the repository, see Cloning a Repository on the GitHub website.

**Note**

The rest of this sample assumes the current working directory that you clone the repository into is the environment root directory. If you clone it somewhere else, substitute that location wherever you see `YOUR_CLONED_REPO_NAME`.

---

**Step 5: Add Files to the Repository**

In this step, you create three simple files in the cloned repository in your environment. Then you add the files to the Git staging area in your cloned repository, commit the staged files, and push the commit to your remote repository in GitHub.

If the cloned repository already has files in it, skip ahead to Step 6: Keep Working with the IDE and GitHub (p. 365).

2. Type the following content into the file, and then choose File, Save to save the file as `bird.txt` in the `/YOUR_CLONED_REPO_NAME` directory in your environment.
bird.txt
--------
Birds are a group of endothermic vertebrates, characterized by feathers, toothless beaked jaws, the laying of hard-shelled eggs, a high metabolic rate, a four-chambered heart, and a lightweight but strong skeleton.

Note
To confirm you are saving this file in the correct directory, in the Save As dialog box, choose the YOUR_CLONED_REPO_NAME folder, and be sure Folder displays /YOUR_CLONED_REPO_NAME.

3. Create two more files, named insect.txt and reptile.txt, with the following content, saving them also in the same /YOUR_CLONED_REPO_NAME directory.

insect.txt
----------
Insects are a class of invertebrates within the arthropod phylum that have a chitinous exoskeleton, a three-part body (head, thorax, and abdomen), three pairs of jointed legs, compound eyes, and one pair of antennae.

reptile.txt
-----------
Reptiles are tetrapod (four-limbed vertebrate) animals in the class Reptilia, comprising today’s turtles, crocodilians, snakes, amphibiaeans, lizards, tuatara, and their extinct relatives.

4. In the terminal, run the cd command to switch to the /YOUR_CLONED_REPO_NAME directory.

cd YOUR_CLONED_REPO_NAME

5. Confirm the files were successfully saved in the /YOUR_CLONED_REPO_NAME directory by running the git status command. All three files are listed as untracked files.

Untracked files:
(use “git add <file>...” to include in what will be committed)

    bird.txt
    insect.txt
    reptile.txt

6. Add the files to the Git staging area by running the git add command.

git add --all

7. Confirm the files were successfully added to the Git staging area by running the git status command again. All three files are now listed as changes to commit.

Changes to be committed:
(use “git reset HEAD <file>...” to unstage)

    new file:  bird.txt
    new file:  insect.txt
    new file:  reptile.txt

8. Commit the staged files by running the git commit command.

git commit -m "Added information about birds, insects, and reptiles."
9. Push the commit to your remote repository in CodeCommit by running the `git push` command.

```
  git push
```

**Note**

You are prompted for your GitHub user name and password. As you continue to work with GitHub, you might be prompted again. To keep from being prompted each time you try to interact with the remote repository in the future, consider installing and configuring a Git credentials manager. For example, you can run this command in the terminal to be prompted no sooner than every 15 minutes: `git config credential.helper 'cache --timeout 900'`. Or you can run this command to never be prompted again, although Git stores your credentials in clear text in a plain file in your home directory: `git config credential.helper 'store --file ~/.git-credentials'`. For more information, see [Git Tools - Credential Storage](https://git-scm.com/docs/git-credential) on the Git website.

If you use GitHub two-factor authentication, you must enter a personal access token whenever you are prompted for a password. If you enter a password instead of a personal access token, an "invalid user name or password" message is displayed, and the operation fails. For more information, see [Creating a personal access token for the command line](https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token) on the GitHub Help website.

You will not see your password or personal access token whenever you enter it in the terminal. This is by design.

10. To confirm whether the files were successfully pushed from your local copy of the repository to the remote repository, open your repository in the GitHub console, and look for the three files you just pushed.

**Step 6: Keep Working with the IDE and GitHub**

Use the AWS Cloud9 IDE and GitHub to keep working with your code. Here are some things to try.

- Use the **Environment** window and editor tabs in the IDE to view, change, and save code. For more information, see [Step 3: Environment Window](p. 103) and [Step 4: Editor, Tabs, and Panes](p. 103) in **Tour the IDE** (p. 100).

- Use the IDE to run, debug, and build your code. For more information, see [Working with Builders, Runners, and Debuggers](p. 146) in the AWS Cloud9 Integrated Development Environment (IDE).

- Use Git in the terminal session in the IDE to continue pushing more code changes to the GitHub repository, as well as periodically pull code changes from others from the repository. For more information, see [Pushing to a Remote](p. 91) and [Fetching a remote](p. 92) on the GitHub Help website.

- Use additional Git commands as you need them. For a list of these commands, see [Git cheatsheet](https://git-scm.com/docs/git-cheatsheet) on the GitHub Help website.

- If you're new to Git and you don't want to mess up your GitHub repository, experiment with a sample Git repository on the Try Git website.

- Invite others to work on your code with you in the same environment, in real time and with text chat. For more information, see [Working with Shared Environments](p. 80) in AWS Cloud9.

**Step 7: Clean Up**

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see [Deleting an Environment](p. 95).

To delete the GitHub repository, see [Deleting a Repository](https://docs.github.com/en/github/administering-a-repository/force-deleting-a-repository) on the GitHub Help website.
Amazon DynamoDB Sample for AWS Cloud9

This sample enables you to set up an AWS Cloud9 development environment to work with Amazon DynamoDB.

DynamoDB is a fully managed NoSQL database service. You can use DynamoDB to create a database table that can store and retrieve any amount of data, and serve any level of request traffic. DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity specified and the amount of data stored, while maintaining consistent and fast performance. For more information, see Amazon DynamoDB on the AWS website.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and DynamoDB. For more information, see Amazon EC2 Pricing and Amazon DynamoDB Pricing.

For information about additional AWS database offerings, see Amazon Relational Database Service (RDS), Amazon ElastiCache, and Amazon Redshift on the AWS website. See also AWS Database Migration Service on the AWS website.

- Prerequisites (p. 366)
- Step 1: Install and Configure the AWS CLI, the aws-shell, or Both in Your Environment (p. 366)
- Step 2: Create a Table (p. 367)
- Step 3: Add an Item to the Table (p. 368)
- Step 4: Add Multiple Items to the Table (p. 369)
- Step 5: Create a Global Secondary Index (p. 372)
- Step 6: Get Items from the Table (p. 373)
- Step 7: Clean Up (p. 376)

Prerequisites

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install and Configure the AWS CLI, the aws-shell, or Both in Your Environment

In this step, you use the AWS Cloud9 IDE to install and configure the AWS CLI, the aws-shell, or both in your environment so you can run commands to interact with DynamoDB. Then you use the AWS CLI to run a basic DynamoDB command to test your installation and configuration.

1. To set up credentials management for the AWS CLI or the aws-shell and to install the AWS CLI, the aws-shell, or both in your environment, follow Steps 1 and 2 in the AWS CLI and aws-shell
Sample (p. 352), and then return to this topic. If you already installed and configured the AWS CLI, the aws-shell, or both in your environment, you don’t need to do it again.

2. Test the installation and configuration of the AWS CLI, the aws-shell, or both by running the DynamoDB `list-tables` command from a terminal session in your environment to list your existing DynamoDB tables, if there are any. To start a new terminal session, on the menu bar, choose **Windows, New Terminal**.

   ```
   aws dynamodb list-tables # For the AWS CLI.
   dynamodb list-tables     # For the aws-shell.
   ```

   **Note**
   Throughout this sample, if you’re using the aws-shell, omit `aws` from each command that starts with `aws`. To start the aws-shell, run the `aws-shell` command. To stop using the aws-shell, run the `.exit` or `.quit` command.

   If this command succeeds, it outputs a `TableNames` array containing a list of existing DynamoDB tables that you might already have. If you have no DynamoDB tables yet, the `TableNames` array will be empty.

   ```json
   {
   "TableNames": []
   }
   ```

   If you do have any DynamoDB tables, the `TableNames` array contains a list of the table names.

**Step 2: Create a Table**

In this step, you create a table in DynamoDB and specify the table’s name, layout, simple primary key, and data throughput settings.

This sample table, named `Weather`, contains information about weather forecasts for a few cities in the United States. The table holds the following types of information (in DynamoDB, each piece of information is known as an *attribute*):

- Required unique city ID (*CityID*)
- Required forecast date (*Date*)
- City name (*City*)
- State name (*State*)
- Forecast weather conditions (*Conditions*)
- Forecast temperatures (*Temperatures*)
  - Forecast high, in degrees Fahrenheit (*HighF*)
  - Forecast low, in degrees Fahrenheit (*LowF*)

To create the table, in a terminal session in the AWS Cloud9 IDE, run the DynamoDB `create-table` command.

```sh
aws dynamodb create-table \
  --table-name Weather \
  --attribute-definitions \
  AttributeName=CityID,AttributeType=N AttributeName=Date,AttributeType=S \ 
  --key-schema \
  AttributeName=CityID,KeyType=HASH AttributeName=Date,KeyType=RANGE \
```
---provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5

In this command:

- **--table-name** represents the table name (Weather in this sample). Table names must be unique within each AWS Region in your AWS account.
- **--attribute-definitions** represents the attributes that are used to uniquely identify the table items. Each of this table's items are uniquely identified by a combination of a numerical ID attribute and a Date attribute represented as an ISO-8601 formatted string.
- **--key-schema** represents the table's key schema. This table has a composite primary key of CityID and Date. This means that each of the table items must have a CityID attribute value and a Date attribute value, but no two items in the table can have both the same CityID attribute value and Date attribute value.
- **--provisioned-throughput** represents the table's read-write capacity. DynamoDB allows up to 5 strongly consistent reads per second for items up to 4 KB in size, or up to 5 eventually consistent reads per second for items up to 4 KB in size. DynamoDB also allows up to 5 writes per second for items up to 1 KB in size.

**Note**

Setting higher provisioned throughput might result in additional charges to your AWS account.

For more information about this and other DynamoDB commands, see dynamodb in the AWS CLI Command Reference.

If this command succeeds, it displays summary information about the new table that is being created. To confirm the table is successfully created, run the DynamoDB describe-table command, specifying the table's name (**--table-name**).

```bash
aws dynamodb describe-table --table-name Weather
```

When the table is successfully created, the TableStatus value changes from CREATING to ACTIVE. Do not proceed past this step until the table is successfully created.

### Step 3: Add an Item to the Table

In this step, you add an item to the table you just created.

1. Create a file named `weather-item.json` with the following content. To create a new file, on the menu bar, choose **File, New File**. To save the file, choose **File, Save**.

```json
{
    "CityID": { "N": "1" },
    "Date": { "S": "2017-04-12" },
    "City": { "S": "Seattle" },
    "State": { "S": "WA" },
    "Conditions": { "S": "Rain" },
    "Temperatures": { "M": {
        "HighF": { "N": "59" },
        "LowF": { "N": "46" }
    }
}
```

In this code, 
represents an attribute value that is a number. 
is a string attribute value. 
is a map attribute, which is a set of attribute-value pairs. You must specify an attribute's data type whenever
you work with items. For additional available attribute data types, see Data Types in the Amazon DynamoDB Developer Guide.

2. Run the DynamoDB put-item command, specifying the table's name (--table-name) and the path to the JSON-formatted item (--item).

```bash
aws dynamodb put-item \
--table-name Weather \
--item file://weather-item.json
```

If the command succeeds, it runs without error, and no confirmation message is displayed.

3. To confirm the table's current contents, run the DynamoDB scan command, specifying the table's name (--table-name).

```bash
aws dynamodb scan --table-name Weather
```

If the command succeeds, summary information about the table and the item you just added is displayed.

**Step 4: Add Multiple Items to the Table**

In this step, you add several more items to the Customers table.

1. Create a file named more-weather-items.json with the following content.

```json
{
  "Weather": [
    
    "PutRequest": {
      "Item": {
        "CityID": { "N": "1" },
        "Date": { "S": "2017-04-13" },
        "City": { "S": "Seattle" },
        "State": { "S": "WA" },
        "Conditions": { "S": "Rain" },
        "Temperatures": { "M": {
          "HighF": { "N": "52" },
          "LowF": { "N": "43" }
        }
      }
    },
    
    "PutRequest": {
      "Item": {
        "CityID": { "N": "1" },
        "Date": { "S": "2017-04-14" },
        "City": { "S": "Seattle" },
        "State": { "S": "WA" },
        "Conditions": { "S": "Rain" },
        "Temperatures": { "M": {
          "HighF": { "N": "49" },
          "LowF": { "N": "43" }
        }
      }
    },
    
    "PutRequest": {
      "Item": {
        "CityID": { "N": "1" },
        "Date": { "S": "2017-04-15" },
        "City": { "S": "Seattle" },
        "State": { "S": "WA" },
        "Conditions": { "S": "Rain" },
        "Temperatures": { "M": {
          "HighF": { "N": "47" },
          "LowF": { "N": "43" }
        }
      }
    }
  ],
}
```
"PutRequest": {  
  "Item": {  
    "CityID": { "N": "2" },  
    "Date": { "S": "2017-04-12" },  
    "City": { "S": "Portland" },  
    "State": { "S": "OR" },  
    "Conditions": { "S": "Thunderstorms" },  
    "Temperatures": { "M": {  
      "HighF": { "N": "59" },  
      "LowF": { "N": "43" }  
    }  
  }  
},  
"PutRequest": {  
  "Item": {  
    "CityID": { "N": "2" },  
    "Date": { "S": "2017-04-13" },  
    "City": { "S": "Portland" },  
    "State": { "S": "OR" },  
    "Conditions": { "S": "Rain" },  
    "Temperatures": { "M": {  
      "HighF": { "N": "51" },  
      "LowF": { "N": "41" }  
    }  
  }  
},  
"PutRequest": {  
  "Item": {  
    "CityID": { "N": "2" },  
    "Date": { "S": "2017-04-14" },  
    "City": { "S": "Portland" },  
    "State": { "S": "OR" },  
    "Conditions": { "S": "Rain Showers" },  
    "Temperatures": { "M": {  
      "HighF": { "N": "49" },  
      "LowF": { "N": "39" }  
    }  
  }  
},  
"PutRequest": {  
  "Item": {  
    "CityID": { "N": "3" },  
    "Date": { "S": "2017-04-12" },  
    "City": { "S": "Portland" },  
    "State": { "S": "ME" },  
    "Conditions": { "S": "Rain" },  
    "Temperatures": { "M": {  
      "HighF": { "N": "59" },  
      "LowF": { "N": "40" }  
    }  
  }  
},  
"PutRequest": {  
  "Item": {  
    "CityID": { "N": "3" },  
    "Date": { "S": "2017-04-13" },  
    "City": { "S": "Portland" },  
    "State": { "S": "ME" },  
    "Conditions": { "S": "Rain" },  
    "Temperatures": { "M": {  
      "HighF": { "N": "59" },  
      "LowF": { "N": "40" }  
    }  
  }  
}  
}  
}

In this code, 8 Item objects define the 8 items to add to the table, similar to the single item defined in the previous step. However, when you run the DynamoDB batch-write-item command in the next step, you must provide a JSON-formatted object that includes each Item object in a containing PutRequest object. Then you must include those PutRequest objects in a parent array that has the same name as the table.

2. Run the DynamoDB batch-write-item command, specifying the path to the JSON-formatted items to add (--request-items).

```
aws dynamodb batch-write-item \
--request-items file://more-weather-items.json
```

If the command succeeds, it displays the following message, confirming that the items were successfully added.

```
{}
```

3. To confirm the table's current contents, run the DynamoDB scan command again.

```
aws dynamodb scan --table-name Weather
```

If the command succeeds, 9 items are now displayed.
Step 5: Create a Global Secondary Index

Running the DynamoDB scan command to get information about items can be slow, especially as a table grows in size or if the type of information you want to get is complex. You can create one or more secondary indexes to speed things up and make getting information easier. In this step, you learn about two types of secondary indexes that DynamoDB supports to do just that. These are known as a local secondary index and a global secondary index. Then you create a global secondary index.

To understand these secondary index types, you first need to know about primary keys, which uniquely identify a table’s items. DynamoDB supports a simple primary key or a composite primary key. A simple primary key has a single attribute, and that attribute value must be unique for each item in the table. This attribute is also known as a partition key (or a hash attribute), which DynamoDB can use to partition items for faster access. A table can also have a composite primary key, which contains two attributes. The first attribute is the partition key, and the second is a sort key (also known as a range attribute). In a table with a composite primary key, any two items can have the same partition key value, but they cannot also have the same sort key value. The Weather table has a composite primary key.

A local secondary index has the same partition key as the table itself, but this index type can have a different sort key. A global secondary index can have a partition key and a sort key that are both different from the table itself.

For example, you can already use the primary key to access Weather items by CityID. To access Weather items by State, you could create a local secondary index that has a partition key of CityID (it must be the same as the table itself) and a sort key of State. To access Weather items by City, you could create a global secondary index that has a partition key of City and a sort key of Date.

You can create local secondary indexes only while you are creating a table. Because the Weather table already exists, you cannot add any local secondary indexes to it. However, you can add global secondary indexes. Practice adding one now.

Note
Creating secondary indexes might result in additional charges to your AWS account.

1. Create a file named weather-global-index.json with the following content.

```
[
  {
    "Create": {
      "IndexName": "weather-global-index",
      "KeySchema": [
        {
          "AttributeName": "City",
          "KeyType": "HASH"
        },
        {
          "AttributeName": "Date",
          "KeyType": "RANGE"
        }
      ],
      "Projection": {
        "ProjectionType": "INCLUDE",
        "NonKeyAttributes": [
          "State",
          "Conditions",
          "Temperatures"
        ]
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": 5,
        "WriteCapacityUnits": 5
      }
    }
  }
]
```
In this code:

- The name of the global secondary index is `weather-global-index`.
- The `City` attribute is the partition key (hash attribute), and the `Date` attribute is the sort key (range attribute).
- `Projection` defines the attributes to retrieve by default (in addition to the hash attribute and any range attribute) for every item matching a table search that uses this index. In this sample, the `State`, `Conditions`, `HighF` (part of `Temperatures`), and `LowF` (also part of `Temperatures`) attributes (as well as the `City` and `Date` attributes) are retrieved for every matching item.
- Similar to tables, a global secondary index must define its provisioned throughput settings.
- The `IndexName`, `KeySchema`, `Projection`, and `ProvisionedThroughput` settings must be contained in a `Create` object, which defines the global secondary index to create when you run the `DynamoDB update-table` command in the next step.

2. Run the DynamoDB `update-table` command.

```bash
aws dynamodb update-table \
  --table-name Weather \
  --attribute-definitions \
    AttributeName=City,AttributeType=S AttributeName=Date,AttributeType=S \
  --global-secondary-index-updates file://weather-global-index.json
```

In this command:

- `--table-name` is the name of the table to update.
- `--attribute-definitions` are the attributes to include in the index. The partition key is always listed first, and any sort key is always listed second.
- `--global-secondary-index-updates` is the path to the file that defines the global secondary index.

If this command succeeds, it displays summary information about the new global secondary index that is being created. To confirm the global secondary index is successfully created, run the DynamoDB `describe-table` command, specifying the table's name (`--table-name`).

```bash
aws dynamodb describe-table --table-name Weather
```

When the global secondary index is successfully created, the `TableStatus` value changes from `UPDATING` to `ACTIVE`, and the `IndexStatus` value changes from `CREATING` to `ACTIVE`. Do not proceed past this step until the global secondary index is successfully created. This can take several minutes.

### Step 6: Get Items from the Table

There are many ways to get items from tables. In this step, you get items by using the table's primary key, by using the table's other attributes, and by using the global secondary index.

#### To get a single item from a table based on the item's primary key value

If you know an item's primary key value, you can get the matching item by running the DynamoDB command `get-item`, `scan`, or `query`. The following are the main differences in these commands:
• **get-item** returns a set of attributes for the item with the given primary key.
• **scan** returns one or more items and item attributes by accessing every item in a table or a secondary index.
• **query** finds items based on primary key values. You can query any table or secondary index that has a composite primary key (a partition key and a sort key).

In this sample, here's how to use each of these commands to get the item that contains the CityID attribute value of 1 and the Date attribute value of 2017-04-12.

1. To run the DynamoDB **get-item** command, specify the name of the table (**--table-name**), the primary key value (**--key**), and the attribute values for the item to display (**--projection-expression**). Because Date is a reserved keyword in DynamoDB, you must also provide an alias for the Date attribute value (**--expression-attribute-names**). (State is also a reserved keyword, and so you will see an alias provided for it in later steps.)

   ```bash
   aws dynamodb get-item \
   --table-name Weather \n   --key '{ "CityID": { "N": "1" }, "Date": { "S": "2017-04-12" } }' \n   --projection-expression \n   "City, #D, Conditions, Temperatures.HighF, Temperatures.LowF" \n   --expression-attribute-names '{ "#D": "Date" }'
   ```

   In this and the other commands, to display all of the item's attributes, don't include **--projection-expression**. In this example, because you are not including **--projection-expression**, you also don't need to include **--expression-attribute-names**.

   ```bash
   aws dynamodb get-item \
   --table-name Weather \n   --key '{ "CityID": { "N": "1" }, "Date": { "S": "2017-04-12" } }'
   ```

2. To run the DynamoDB **scan** command, specify:
   - The name of the table (**--table-name**).
   - The search to run (**--filter-expression**).
   - The search criteria to use (**--expression-attribute-values**).
   - The kinds of attributes to display for the matching item (**--select**).
   - The attribute values for the item to display (**--projection-expression**).
   - If any of your attributes are using reserved keywords in DynamoDB, aliases for those attributes (**--expression-attribute-names**).

   ```bash
   aws dynamodb scan \
   --table-name Weather \n   --filter-expression "(CityID = :cityID) and (#D = :date)" \n   --expression-attribute-values \n   '{ "cityID": { "N": "1" }, "date": { "S": "2017-04-12" } }' \n   --select SPECIFIC_ATTRIBUTES \n   --projection-expression \n   "City, #D, Conditions, Temperatures.HighF, Temperatures.LowF" \n   --expression-attribute-names '{ "#D": "Date" }'
   ```

3. To run the DynamoDB **query** command, specify:
   - The name of the table (**--table-name**).
   - The search to run (**--key-condition-expression**).
   - The attribute values to use in the search (**--expression-attribute-values**).
   - The kinds of attributes to display for the matching item (**--select**).
The attribute values for the item to display (\--projection-expression).

If any of your attributes are using reserved keywords in DynamoDB, aliases for those attributes (\--expression-attribute-names).

```
aws dynamodb query
--table-name Weather
--key-condition-expression "(CityID = :cityID) and (#D = :date)"
--expression-attribute-values
'{ "cityID": { "N": "1" }, "date": { "S": "2017-04-12" } }'
--select SPECIFIC_ATTRIBUTES
--projection-expression
"City, #D, Conditions, Temperatures.HighF, Temperatures.LowF"
--expression-attribute-names '{ "#D": "Date" }'
```

Notice that the \texttt{scan} command needed to scan all 9 items to get the result, while the \texttt{query} command only needed to scan for 1 item.

### To get multiple items from a table based on the items' primary key values

If you know the items' primary key values, you can get the matching items by running the DynamoDB \texttt{batch-get-item} command. In this sample, here's how to get the items that contain the \texttt{CityID} attribute value of 3 and \texttt{Date} attribute values of 2017-04-13 or 2017-04-14.

Run the DynamoDB \texttt{batch-get-item} command, specifying the path to a file describing the items to get (\--request-items).

```
aws dynamodb batch-get-item --request-items file://batch-get-item.json
```

For this sample, the code in the \texttt{batch-get-item.json} file specifies to search the \texttt{Weather} table for items with a \texttt{CityID} of 3 and \texttt{Date} of 2017-04-13 or 2017-04-14. For each item found, the attribute values for \texttt{City}, \texttt{State}, \texttt{Date}, and \texttt{HighF} (part of \texttt{Temperatures}) are displayed, if they exist.

```
{
"Weather" : {
  "Keys": [
    {
      "CityID": { "N": "3" },
      "Date": { "S": "2017-04-13" }
    },
    {
      "CityID": { "N": "3" },
      "Date": { "S": "2017-04-14" }
    }
  ],
  "ProjectionExpression": "City, #S, #D, Temperatures.HighF",
  "ExpressionAttributeNames": { "#S": "State", "#D": "Date" }
}
```

### To get all matching items from a table

If you know something about the attributes' values in the table, you can get matching items by running the DynamoDB \texttt{scan} command. In this sample, here's how to get the dates when the \texttt{Conditions} attribute value contains \texttt{Sunny} and the \texttt{HighF} attribute value (part of \texttt{Temperatures}) is greater than 53.
Run the DynamoDB `scan` command, specifying:

- The name of the table (`--table-name`).
- The search to run (`--filter-expression`).
- The search criteria to use (`--expression-attribute-values`).
- The kinds of attributes to display for the matching item (`--select`).
- The attribute values for the item to display (`--projection-expression`).
- If any of your attributes are using reserved keywords in DynamoDB, aliases for those attributes (`--expression-attribute-names`).

```
aws dynamodb scan
--table-name Weather
--filter-expression
  "(contains (Conditions, :sun)) and (Temperatures.HighF > :h)"
--expression-attribute-values
  '{ "sun": { "S": "Sunny" }, "h": { "N": "53" } }'
--select SPECIFIC_ATTRIBUTES
--projection-expression "City, #S, #D, Conditions, Temperatures.HighF"
--expression-attribute-names '{ "#S": "State", "#D": "Date" }'
```

To get all matching items from a global secondary index

To search using a global secondary index, use the DynamoDB `query` command. In this sample, here’s how to use the `weather-global-index` secondary index to get the forecast conditions for cities named Portland for the dates of 2017-04-13 and 2017-04-14.

Run the DynamoDB `query` command, specifying:

- The name of the table (`--table-name`).
- The name of the global secondary index (`--index-name`).
- The search to run (`--key-condition-expression`).
- The attribute values to use in the search (`--expression-attribute-values`).
- The kinds of attributes to display for the matching item (`--select`).
- If any of your attributes are using reserved keywords in DynamoDB, aliases for those attributes (`--expression-attribute-names`).

```
aws dynamodb query
--table-name Weather
--index-name weather-global-index
--key-condition-expression "(City = :city) and (#D between :date1 and :date2)"
--expression-attribute-values
  '{ "city": { "S": "Portland" }, "date1": { "S": "2017-04-13" }, "date2": { "S": "2017-04-14" } }'
--select SPECIFIC_ATTRIBUTES
--projection-expression "City, #S, #D, Conditions, Temperatures.HighF"
--expression-attribute-names '{ "#S": "State", "#D": "Date" }'
```

Step 7: Clean Up

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the table. Deleting the table deletes the global secondary index as well. You should also delete your environment.
To delete the table, run the DynamoDB `delete-table` command, specifying the table's name (`--table-name`).

```bash
aws dynamodb delete-table --table-name Weather
```

If the command succeeds, information about the table is displayed, including the `TableStatus` value of `DELETING`.

To confirm the table is successfully deleted, run the DynamoDB `describe-table` command, specifying the table's name (`--table-name`).

```bash
aws dynamodb describe-table --table-name Weather
```

If the table is successfully deleted, a message containing the phrase `Requested resource not found` is displayed.

To delete your environment, see Deleting an Environment (p. 95).

---

**AWS CDK Sample for AWS Cloud9**

This sample shows you how to work with the AWS Cloud Development Kit (AWS CDK) in an AWS Cloud9 development environment. The AWS CDK is a set of software tools and libraries that developers can use to model AWS infrastructure components as code.

The AWS CDK includes the AWS Construct Library that you can use to quickly resolve many tasks on AWS. For example, you can use the `Fleet` construct to fully and securely deploy code to a fleet of hosts. You can create your own constructs to model various elements of your architectures, share them with others, or publish them to the community. For more information, see the [AWS Cloud Development Kit Developer Guide](https://docs.aws.amazon.com/cdk/latest/dev/).

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2, Amazon SNS, and Amazon SQS. For more information, see Amazon EC2 Pricing, Amazon SNS Pricing, and Amazon SQS Pricing.

**Topics**

- Prerequisites (p. 377)
- Step 1: Install Required Tools (p. 378)
- Step 2: Add Code (p. 380)
- Step 3: Run the Code (p. 382)
- Step 4: Clean Up (p. 383)

**Prerequisites**

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.
Step 1: Install Required Tools

In this step, you install all of the tools in your environment that the AWS CDK needs to run a sample that is written in the TypeScript programming language.

1. **Node Version Manager (p. 378)**, or **nvm**, which you use to install Node.js later.
2. **Node.js (p. 378)**, which is required by the sample and contains Node Package Manager, or **npm**, which you use to install TypeScript and the AWS CDK later.
3. **TypeScript (p. 379)**, which is required by this sample. (The AWS CDK also provides support for several other programming languages.)
4. The **AWS CDK (p. 379)**.

Step 1.1: Install Node Version Manager (nvm)

1. In a terminal session in the AWS Cloud9 IDE, ensure the latest security updates and bug fixes are installed. To do this, run the **yum update** (for Amazon Linux) or **apt update** command (for Ubuntu Server). (To start a new terminal session, on the menu bar, choose **Window, New Terminal**.)

   For Amazon Linux:

   ```
   sudo yum -y update
   ```

   For Ubuntu Server:

   ```
   sudo apt update
   ```

2. Confirm whether **nvm** is already installed. To do this, run the **nvm** command with the **--version** option.

   ```
   nvm --version
   ```

   If successful, the output contains the **nvm** version number, and you can skip ahead to **Step 1.2: Install Node.js (p. 378)**.

3. Download and install **nvm**. To do this, run the install script. In this example, v0.33.0 is installed, but you can check for the latest version of **nvm** here.

   ```
   curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
   ```

4. Start using **nvm**. You can either close the terminal session and then restart it, or source the `~/.bashrc` file that contains the commands to load **nvm**.

   ```
   . ~/.bashrc
   ```

Step 1.2: Install Node.js

1. Confirm whether you already have Node.js installed, and if you do, confirm that the installed version is 10.3.0 or greater. **This sample has been tested with Node.js 10.3.0.** To check, with the terminal session still open in the IDE, run the **node** command with the **--version** option.

   ```
   node --version
   ```
Step 1: Install Required Tools

If you do have Node.js installed, the output contains the version number. If the version number is v10.3.0, skip ahead to Step 1.3: Install TypeScript (p. 379).

2. Install Node.js 10.3.0 by running the `nvm` command with the `install` action.

   **Note**
   You can also run `nvm install stable` to install the long-term support (LTS) version of Node.js. AWS Cloud9 support tracks the LTS version of Node.js.

   ```bash
   nvm install v10.3.0
   ```

3. Start using Node.js 10.3.0. To do this, run the `nvm` command with the `alias` action, the version number to alias, and the version to use for that alias, as follows.

   ```bash
   nvm alias default 10.3.0
   ```

   **Note**
   The preceding command sets Node.js 10.3.0 as the default version of Node.js. Alternatively, you can run the `nvm` command along with the `use` action instead of the `alias` action (for example, `nvm use 10.3.0`). However, the `use` action causes that version of Node.js to run only while the current terminal session is running.

4. To confirm that you're using Node.js 10.3.0, run the `node --version` command again. If the correct version is installed, the output contains version v10.3.0.

Step 1.3: Install TypeScript

1. Confirm whether you already have TypeScript installed. To do this, with the terminal session still open in the IDE, run the command-line TypeScript compiler with the `--version` option.

   ```bash
   tsc --version
   ```

   If you do have TypeScript installed, the output contains the TypeScript version number. If TypeScript is installed, skip ahead to Step 1.4: Install the AWS CDK (p. 379).

2. Install TypeScript. To do this, run the `npm` command with the `install` action, the `-g` option, and the name of the TypeScript package. This installs TypeScript as a global package in the environment.

   ```bash
   npm install -g typescript
   ```

3. Confirm that TypeScript is installed. To do this, run the command line TypeScript compiler with the `--version` option.

   ```bash
   tsc --version
   ```

   If TypeScript is installed, the output contains the TypeScript version number.

Step 1.4: Install the AWS CDK

1. Confirm whether you already have the AWS CDK installed. To do this, with the terminal session still open in the IDE, run the `cdk` command with the `--version` option.

   ```bash
   cdk --version
   ```
If the AWS CDK is installed, the output contains the AWS CDK version and build numbers. Skip ahead to Step 2: Add Code (p. 380).

2. Install the AWS CDK by running the `npm` command along with the `install` action, the name of the AWS CDK package to install, and the `-g` option to install the package globally in the environment.

```
npm install -g aws-cdk
```

3. Confirm that the AWS CDK is installed and correctly referenced. To do this, run the `cdk` command with the `--version` option.

```
cdk --version
```

If successful, the AWS CDK version and build numbers are displayed.

## Step 2: Add Code

In this step, you create a sample TypeScript project that contains all of the source code you need for the AWS CDK to programmatically deploy an AWS CloudFormation stack. This stack creates an Amazon SNS topic and an Amazon SQS queue in your AWS account and then subscribes the queue to the topic.

1. With the terminal session still open in the IDE, create a directory to store the project's source code, for example a `~/environment/hello-cdk` directory in your environment. Then switch to that directory.

```
rm -rf ~/environment/hello-cdk  # Remove this directory if it already exists.
mkdir ~/environment/hello-cdk   # Create the directory.
cd ~/environment/hello-cdk     # Switch to the directory.
```

2. Set up the directory as a TypeScript language project for the AWS CDK. To do this, run the `cdk` command with the `init` action, the `sample-app` template, and the `--language` option along with the name of the programming language.

```
cdk init sample-app --language typescript
```

This creates the following files and subdirectories in the directory:

- A hidden `.git` subdirectory and a hidden `.gitignore` file, which makes the project compatible with source control tools such as Git.
- A `lib` subdirectory, which includes a `hello-cdk-stack.ts` file. This file contains the code for your AWS CDK stack. This code is described in the next step in this procedure.
- A `bin` subdirectory, which includes a `hello-cdk.ts` file. This file contains the entry point for your AWS CDK app.
- A `node_modules` subdirectory, which contains supporting code packages that the app and stack can use as needed.
- A hidden `.npmignore` file, which lists the types of subdirectories and files that `npm` doesn't need when it builds the code.
- A `cdk.json` file, which contains information to make running the `cdk` command easier.
- A `package-lock.json` file, which contains information that `npm` can use to reduce possible build and run errors.
- A `package.json` file, which contains information to make running the `npm` command easier and with possibly fewer build and run errors.
- A `README.md` file, which lists useful commands you can run with `npm` and the AWS CDK.
• A tsconfig.json file, which contains information to make running the tsc command easier and with possibly fewer build and run errors.

3. In the Environment window, open the lib/hello-cdk-stack.ts file, and browse the following code in that file.

```javascript
import sns = require('@aws-cdk/aws-sns');
import sqs = require('@aws-cdk/aws-sqs');
import cdk = require('@aws-cdk/cdk');

export class HelloCdkStack extends cdk.Stack {
  constructor(parent: cdk.App, name: string, props?: cdk.StackProps) {
    super(parent, name, props);

    const queue = new sqs.Queue(this, 'HelloCdkQueue', {
      visibilityTimeoutSec: 300
    });

    const topic = new sns.Topic(this, 'HelloCdkTopic');
    topic.subscribeQueue(queue);
  }
}
```

• The Stack, App, StackProps, Queue, and Topic classes represent an AWS CloudFormation stack and its properties, an executable program, an Amazon SQS queue, and an Amazon SNS topic, respectively.

• The HelloCdkStack class represents the AWS CloudFormation stack for this application. This stack contains the new Amazon SQS queue and Amazon SNS topic for this application.

4. In the Environment window, open the bin/hello-cdk.ts file, and browse the following code in that file.

```javascript
#!/usr/bin/env node
import cdk = require('@aws-cdk/cdk');
import { HelloCdkStack } from '../lib/hello-cdk-stack';

const app = new cdk.App();
new HelloCdkStack(app, 'HelloCdkStack');
app.run();
```

This code loads, instantiates, and then runs the HelloCdkStack class from the lib/hello-cdk-stack.ts file.

5. Use npm to run the TypeScript compiler to check for coding errors, and then enable the AWS CDK to execute the project's bin/hello-cdk.js file. To do this, from the project's root directory, run the npm command with the run action, specifying the build command value in the package.json file, as follows.

```bash
npm run build
```

The preceding command runs the TypeScript compiler, which adds supporting bin/hello-cdk.d.ts and lib/hello-cdk-stack.d.ts files. The compiler also transpiles the hello-cdk.ts and hello-cdk-stack.ts files into hello-cdk.js and hello-cdk-stack.js files.
Step 3: Run the Code

In this step, you instruct the AWS CDK to create an AWS CloudFormation stack template based on the code in the `bin/hello-cdk.js` file. You then instruct the AWS CDK to deploy the stack, which creates the Amazon SNS topic and Amazon SQS queue and then subscribes the queue to the topic. You then confirm that the topic and queue were successfully deployed by sending a message from the topic to the queue.

1. Have the AWS CDK create the AWS CloudFormation stack template. To do this, with the terminal session still open in the IDE, from the project's root directory, run the `cdk` command with the ` synth` action and the name of the stack.

```
cdk synth HelloCdkStack
```

If successful, the output displays the AWS CloudFormation stack template's **Resources** section.

2. The first time that you deploy an AWS CDK app into an environment for a specific AWS account and AWS Region combination, you must install a **bootstrap stack**. This stack includes various resources that the AWS CDK needs to complete its various operations. For example, this stack includes an Amazon S3 bucket that the AWS CDK uses to store templates and assets during its deployment processes. To install the bootstrap stack, run the `cdk` command with the `bootstrap` action.

```
cdk bootstrap
```

3. Have the AWS CDK run the AWS CloudFormation stack template to deploy the stack. To do this, from the project's root directory, run the `cdk` command with the `deploy` action and the name of the stack.

```
cdk deploy HelloCdkStack
```

If successful, the output displays that the `HelloCdkStack` stack deployed without errors.

**Note**

If the output displays a message that the stack does not define an environment and that AWS credentials could not be obtained from standard locations or no region was configured, make sure that your AWS credentials are set correctly in the IDE, and then run the `cdk deploy` command again. For more information, see Calling AWS Services from an Environment in AWS Cloud9 (p. 70).

4. To confirm that the Amazon SNS topic and Amazon SQS queue were successfully deployed, send a message to the topic, and then check the queue for the received message. To do this, you can use a tool such as the AWS Command Line Interface (AWS CLI) or the aws-shell. For more information about these tools, see the AWS Command Line Interface and aws-shell Sample for AWS Cloud9 (p. 352).

For example, to send a message to the topic, with the terminal session still open in the IDE, use the AWS CLI to run the Amazon SNS `publish` command, supplying the message's subject and body, the AWS Region for the topic, and the topic's Amazon Resource Name (ARN).

```
aws sns publish --subject "Hello from the AWS CDK" --message "This is a message from the AWS CDK." --topic-arn arn:aws:sns:us-east-2:123456789012:HelloCdkStack-HelloCdkTopic1A234567-8BCD9EFGHIJ0K
```

In the preceding command, replace `arn:aws:sns:us-east-2:123456789012:HelloCdkStack-HelloCdkTopic1A234567-8BCD9EFGHIJ0K` with the ARN that AWS CloudFormation assigns to the topic. To get the ID, you can run the Amazon SNS `list-topics` command.
Step 4: Clean Up

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the AWS CloudFormation stack. This deletes the the Amazon SNS topic and Amazon SQS queue. You should also delete the environment.

Step 4.1: Delete the Stack

With the terminal session still open in the IDE, from the project's root directory, run the `cdk` command with the `destroy` action and the stack's name.

```
cdk destroy HelloCdkStack
```

When prompted to delete the stack, type `y`, and then press Enter.

If successful, the output displays that the HelloCdkStack stack was deleted without errors.

Step 4.2: Delete the Environment

To delete the environment, see Deleting an Environment in AWS Cloud9 (p. 95).

LAMP Sample for AWS Cloud9

This sample enables you to set up and run LAMP (Linux, Apache HTTP Server, MySQL, and PHP) within an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon Elastic Compute Cloud (Amazon EC2). For more information, see Amazon EC2 Pricing.
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install the Tools

In this step, you install the following tools:

- Apache HTTP Server, a web server host.
- PHP, a scripting language that is especially suited for web development and can be embedded into HTML.
- MySQL, a database management system.

You then finish this step by starting Apache HTTP Server and then MySQL.

1. Ensure that the latest security updates and bug fixes are installed on the instance. To do this, in a terminal session in the AWS Cloud9 IDE, run the `yum update` (for Amazon Linux) or `apt update` (for Ubuntu Server) command. (To start a new terminal session, on the menu bar, choose Window, New Terminal.)

   For Amazon Linux:

   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:

   ```bash
   sudo apt -y update
   ```

2. Check whether Apache HTTP Server is already installed. To do this, run the `httpd -v` (for Amazon Linux) or `apache2 -v` (for Ubuntu Server) command.

   If successful, the output contains the Apache HTTP Server version number.

   If you see an error, then install Apache HTTP Server by running the `install` command.

   For Amazon Linux:
Step 1: Install the Tools

1. Install the tools using the following commands:

   For Amazon Linux:
   ```shell
   sudo yum install -y httpd24
   ```

   For Ubuntu Server:
   ```shell
   sudo apt install -y apache2
   ```

3. Confirm whether PHP is already installed by running the `php -v` command.
   
   If successful, the output contains the PHP version number.
   
   If you see an error, then install PHP by running the `install` command.

   For Amazon Linux:
   ```shell
   sudo yum install -y php56
   ```

   For Ubuntu Server:
   ```shell
   sudo apt install -y php libapache2-mod-php php-xml
   ```

4. Confirm whether MySQL is already installed by running the `mysql --version` command.
   
   If successful, the output contains the MySQL version number.
   
   If you see an error, then install MySQL by running the `install` command.

   For Amazon Linux:
   ```shell
   sudo yum install -y mysql-server
   ```

   For Ubuntu Server:
   ```shell
   sudo apt install -y mysql-server
   ```

5. After you install Apache HTTP Server, PHP, and MySQL, start Apache HTTP Server, and then confirm it has started, by running the following command.

   For Amazon Linux (you might need to run this command twice):
   ```shell
   sudo service httpd start && sudo service httpd status
   ```

   For Ubuntu Server (to return to the command prompt, press q):
   ```shell
   sudo service apache2 start && sudo service apache2 status
   ```

6. Start MySQL, and then confirm it has started, by running the following command.

   For Amazon Linux:
   ```shell
   sudo service mysqld start && sudo service mysqld status
   ```

   For Ubuntu Server (to return to the command prompt, press q):
Step 2: Set Up MySQL

In this step, you set up MySQL to follow MySQL security best practices. These security best practices include setting a password for root accounts, removing root accounts that are accessible from outside the local host, removing anonymous user accounts, removing the test database, and removing privileges that permit anyone to access databases with names that start with test_.

You then finish this step by practicing the starting and then exiting of the MySQL command-line client.

1. Implement MySQL security best practices for the MySQL installation by running the following command in a terminal session in the AWS Cloud9 IDE.

```
sudo mysql_secure_installation
```

2. When prompted, answer the following questions as specified.

   For Amazon Linux:

   1. **Enter current password for root (enter for none)** – Press Enter (for no password).
   2. **Set root password** – Type Y, and then press Enter.
   3. **New password** – Type a password, and then press Enter.
   4. **Re-enter new password** – Type the password again, and then press Enter. (Be sure to store the password in a secure location for later use.)
   5. **Remove anonymous users** – Type Y, and then press Enter.
   6. **Disallow root login remotely** – Type Y, and then press Enter.
   7. **Remove test database and access to it** – Type Y, and then press Enter.
   8. **Reload privilege tables now** – Type Y, and then press Enter.

   For Ubuntu Server:

   1. **Would you like to setup VALIDATE PASSWORD plugin** – Type y, and then press Enter.
   2. **There are three levels of password validation policy** – Type 0, 1, or 2, and then press Enter.
   3. **New password** – Type a password, and then press Enter.
   4. **Re-enter new password** – Type the password again, and then press Enter. (Be sure to store the password in a secure location for later use.)
   5. **Do you wish to continue with the password provided** – Type y, and then press Enter.
   6. **Remove anonymous users** – Type y, and then press Enter.
   7. **Disallow root login remotely** – Type y, and then press Enter.
   8. **Remove test database and access to it** – Type y, and then press Enter.
   9. **Reload privilege tables now** – Type y, and then press Enter.

   3. To interact directly with MySQL, start the MySQL command-line client as the root user by running the following command. When prompted, type the root user's password that you set earlier, and then press Enter. (The prompt changes to `mysql>` while you are in the MySQL command-line client.)

```
sudo mysql -uroot -p
```
4. To exit the MySQL command-line client, run the following command. (The prompt changes back to `$`.)

```
exit;
```

## Step 3: Set Up a Website

In this step, you set up the default website root for the Apache HTTP Server with recommended owners and access permissions. You then create a PHP-based webpage within that default website root.

You then enable incoming web traffic to view that webpage by setting up the security group in Amazon EC2 and network access control list (network ACL) in Amazon Virtual Private Cloud (Amazon VPC) that are associated with this EC2 environment. Each EC2 environment must be associated with both a security group in Amazon EC2 and a network ACL in Amazon VPC. However, while the default network ACL in an AWS account allows all incoming and outgoing traffic for the environment, the default security group allows only incoming traffic using SSH over port 22. For more information, see the section called “Amazon VPC Settings” (p. 467).

You then finish this step by successfully viewing the webpage from outside of the AWS Cloud9 IDE.

1. Set up the default website root for the Apache HTTP Server (`/var/www/html`) with recommended owners and access permissions. To do this, run the following six commands, one at a time in the following order, in a terminal session in the AWS Cloud9 IDE. To understand what each command does, read the information after the `#` character after each command.

   **For Amazon Linux:**
   ```bash
   sudo groupadd web-content # Create a group named web-content.
   sudo usermod -G web-content -a ec2-user # Add the user ec2-user (your default user for this environment) to the group web-content.
   sudo usermod -G web-content -a apache # Add the user apache (Apache HTTP Server) to the group web-content.
   sudo chown -R ec2-user:web-content /var/www/html # Change the owner of /var/www/html and its files to user ec2-user and group web-content.
   sudo find /var/www/html -type f -exec chmod u=rw,g=rx,o=rx {} \; # Change all file permissions within /var/www/html to user read/write, group read-only, and others read/execute.
   sudo find /var/www/html -type d -exec chmod u=rwx,g=rx,o=rx {} \; # Change /var/www/html directory permissions to user read/write/execute, group read/execute, and others read/execute.
   ```

   **For Ubuntu Server:**
   ```bash
   sudo groupadd web-content # Create a group named web-content.
   sudo usermod -G web-content -a ubuntu # Add the user ubuntu (your default user for this environment) to the group web-content.
   sudo usermod -G web-content -a www-data # Add the user www-data (Apache HTTP Server) to the group web-content.
   sudo chown -R ubuntu:web-content /var/www/html # Change the owner of /var/www/html and its files to user ubuntu and group web-content.
   ```
Step 3: Set Up a Website

2. Create a PHP-based webpage named `index.php` in the default website root folder for the Apache HTTP Server (which is `/var/www/html`) by running the following command.

For Amazon Linux:

```
%s
%s' '<?php' '  phpinfo();' '?>' >> /var/www/html/index.php
```

The preceding command for Amazon Linux also changes the file's owner to `ec2-user`, changes the file's group to `web-content`, and changes the file's permissions to read/write for the user, and read/execute for the group and others.

For Ubuntu Server:

```
%s
%s' '<?php' '  phpinfo();' '?>' >> /var/www/html/index.php
```

The preceding command for Ubuntu Server also changes the file's owner to `ubuntu`, changes the file's group to `web-content`, and changes the file's permissions to read/write for the user, and read/execute for the group and others.

If successful, the preceding commands create the `index.php` file with the following contents.

```
<?php
  phpinfo();
?>
```

3. Enable incoming web traffic over port 80 to view the new webpage by setting up the network ACL in Amazon VPC and the security group Amazon EC2 that is associated with this EC2 environment. To do this, run the following eight commands, one at a time in the following order. To understand what each command does, read the information after the `#` character for each command.

**Important**

Running the following commands enables incoming web traffic over port 80 for all EC2 environments and Amazon EC2 instances that are associated with the security group and network ACL for this environment. This might result in unexpectedly enabling incoming web traffic over port 80 for EC2 environments and Amazon EC2 instances other than this one.

**Note**

The following second through fourth commands enable the security group to allow incoming web traffic over port 80. If you have a default security group, which only allows incoming SSH traffic over port 22, then you must run the first command followed by these second through fourth commands. However, if you have a custom security group already allows incoming web traffic over port 80, you can safely skip running those commands. The following fifth through eighth commands enable the network ACL to allow incoming web traffic over port 80. If you have a default network ACL, which already allows all incoming traffic over all ports, then you can safely skip running those commands. However, if you have a custom network ACL that doesn't allow incoming web traffic over port 80, then you must run the first command followed by these fifth through eighth commands.
Step 4: Clean Up

If you want to keep using this environment but you want to disable incoming web traffic over port 80, then run the following eight commands, one at a time in the following order, to delete the corresponding incoming traffic rules that you set earlier in the security group and network ACL that are associated with the environment. To understand what each command does, read the information after the # character for each command.

Important
Running the following commands disables incoming web traffic over port 80 for all EC2 environments and Amazon EC2 instances that are associated with the security group and network ACL for this environment. This might result in unexpectedly disabling incoming web traffic over port 80 for EC2 environments and Amazon EC2 instances other than this one.

```
MY_SECURITYGROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text) # Get the ID of the security group associated with the instance, and store it temporarily.

aws ec2 authorize-security-group-ingress --group-id $MY_SECURITYGROUP_ID --protocol tcp --cidr 0.0.0.0/0 --port 80 # Add an inbound rule to the security group to allow all incoming IPv4-based traffic over port 80.
aws ec2 authorize-security-group-ingress --group-id $MY_SECURITYGROUP_ID --ip-permissions IpProtocol=tcp,Ipv6Ranges='[CidrIpv6=::/0]',FromPort=80,ToPort=80 # Add an inbound rule to the security group to allow all incoming IPv6-based traffic over port 80.

MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet associated with the instance, and store it temporarily.
MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters Name=association.subnet-id,Values=$MY_SUBNET_ID --query 'NetworkAcls[].Associations[0].NetworkAclId' --output text) # Get the ID of the network ACL associated with the subnet, and store it temporarily.

aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --protocol tcp --rule-action allow --rule-number 10000 --cidr-block 0.0.0.0/0 --port-range From=80,To=80 # Add an inbound rule to the network ACL to allow all IPv4-based traffic over port 80. Advanced users: change this suggested rule number as desired.
aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --protocol tcp --rule-action allow --rule-number 10100 --ipv6-cidr-block ::/0 --port-range From=80,To=80 # Add an inbound rule to the network ACL to allow all IPv6-based traffic over port 80. Advanced users: change this suggested rule number as desired.

4. Get the URL to the index.php file within the web server root. To do this, run the following command, and use a new web browser tab or a different web browser separate from the AWS Cloud9 IDE to go to the URL that is displayed. If successful, the webpage displays information about Apache HTTP Server, MySQL, PHP, and other related settings.

```
```

---

389
Note
The following fifth through eighth commands remove existing rules in order to block the network ACL from allowing incoming web traffic over port 80. If you have a default network ACL, which already allows all incoming traffic over all ports, then you can safely skip running those commands. However, if you have a custom network ACL with existing rules that allow incoming web traffic over port 80 and you want to delete those rules, then you must run the first command followed by these fifth through eighth commands.

```bash
MY_SECURITY_GROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text) # Get the ID of the security group associated with the instance, and store it temporarily.
aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --protocol tcp --cidr 0.0.0.0/0 --port 80 # Delete the existing inbound rule from the security group to block all incoming IPv4-based traffic over port 80.
aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --ip-permissions IpProtocol=tcp,Ipv6Ranges='[CidrIpv6=::/0]',FromPort=80,ToPort=80 # Delete the existing inbound rule from the security group to block all incoming IPv6-based traffic over port 80.
MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet associated with the instance, and store it temporarily.
MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters Name=association.subnet-id,Values=$MY_SUBNET_ID --query 'NetworkAcls[].Associations[0].NetworkAclId' --output text) # Get the ID of the network ACL associated with the subnet, and store it temporarily.
aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-number 10000 # Delete the existing inbound rule from the network ACL to block all IPv4-based traffic over port 80. Advanced users: if you originally created this rule with a different number, change this suggested rule number to match.
aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-number 10100 # Delete the existing inbound rule from the network ACL to block all IPv6-based traffic over port 80. Advanced users: if you originally created this rule with a different number, change this suggested rule number to match.
```

If you are done using this environment, you should delete it to prevent ongoing charges to your AWS account. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

WordPress Sample for AWS Cloud9

This sample enables you to run WordPress within an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon Elastic Compute Cloud (Amazon EC2). For more information, see Amazon EC2 Pricing.

Topics
- Prerequisites (p. 391)
- Step 1: Install the Required Tools (p. 391)
- Step 2: Set Up MySQL (p. 393)
Prerequisites

Before you use this sample, be sure to meet the following requirements.

• **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

• **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install the Required Tools

In this step, you install the following tools, which WordPress depends on:

• Apache HTTP Server, which hosts WordPress websites.
• PHP, which WordPress uses for scripting on the websites.
• MySQL, which WordPress uses to store and retrieve information for the websites.

You then finish this step by starting Apache HTTP Server and MySQL and then installing WordPress.

1. Ensure that the latest security updates and bug fixes are installed. To do this, in a terminal session in the AWS Cloud9 IDE, run the **yum update** for (Amazon Linux) or **apt update** for (Ubuntu Server) command. (To start a new terminal session, on the menu bar, choose **Window, New Terminal**)

   For Amazon Linux:
   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:
   ```bash
   sudo apt -y update
   ```

2. Check whether Apache HTTP Server is already installed. To do this, run the **httpd -v** (for Amazon Linux) or **apache2 -v** (for Ubuntu Server) command.

   If successful, the output contains the Apache HTTP Server version number.

   If you see an error, then install Apache HTTP Server by running the **install** command.

   For Amazon Linux:
   ```bash
   sudo yum install -y httpd24
   ```

   For Ubuntu Server:
Step 1: Install the Required Tools

3. Confirm whether PHP is already installed by running the `php -v` command.
   
   If successful, the output contains the PHP version number.
   
   If you see an error, then install PHP by running the `install` command.
   
   For Amazon Linux:
   
   ```bash
   sudo yum install -y php56
   ```
   
   For Ubuntu Server:
   
   ```bash
   sudo apt install -y php libapache2-mod-php php-xml
   ```

4. Confirm whether MySQL is already installed by running the `mysql --version` command.
   
   If successful, the output contains the MySQL version number.
   
   If you see an error, then install MySQL by running the `install` command.
   
   For Amazon Linux:
   
   ```bash
   sudo yum install -y mysql-server
   ```
   
   For Ubuntu Server:
   
   ```bash
   sudo apt install -y mysql-server
   ```

5. After you install Apache HTTP Server, PHP, and MySQL, start Apache HTTP Server, and then confirm it has started, by running the following command.
   
   For Amazon Linux (you might need to run this command twice):
   
   ```bash
   sudo service httpd start && sudo service httpd status
   ```
   
   For Ubuntu Server (to return to the command prompt, press `q`):
   
   ```bash
   sudo service apache2 start && sudo service apache2 status
   ```

6. Start MySQL, and then confirm it has started, by running the following command.
   
   For Amazon Linux:
   
   ```bash
   sudo service mysql start && sudo service mysql status
   ```
   
   For Ubuntu Server (to return to the command prompt, press `q`):
   
   ```bash
   sudo service mysql start && sudo service mysql status
   ```

7. Download WordPress by running the following command.
   
   ```bash
   wget http://wordpress.org/latest.tar.gz
   ```
8. Install WordPress by running the following command.

```
tar -xzvf latest.tar.gz
```

If you run the preceding command from the default location in your environment, you can find the WordPress installation files within a folder named `wordpress` in the Environment window in the AWS Cloud9 IDE. You can also get to these files from a terminal session by changing to the following location.

For Amazon Linux:

```
cd /home/ec2-user/environment/wordpress/
```

For Ubuntu Server:

```
cd /home/ubuntu/environment/wordpress/
```

**Step 2: Set Up MySQL**

In this step, you set up MySQL to follow MySQL security best practices. These security best practices include setting a password for root accounts, removing root accounts that are accessible from outside the local host, removing anonymous user accounts, removing the test database, and removing privileges that permit anyone to access databases with names that start with `test_`.

You then finish this step by setting up MySQL to store and retrieve information for a new WordPress website.

1. Implement MySQL security best practices for the MySQL installation by running the following command in a terminal session in the AWS Cloud9 IDE.

```
sudo mysql_secure_installation
```

2. When prompted, answer the following questions as specified.

   For Amazon Linux:

   1. **Enter current password for root (enter for none)** – Press Enter (for no password).
   2. **Set root password** – Type Y, and then press Enter.
   3. **New password** – Type a password, and then press Enter.
   4. **Re-enter new password** – Type the password again, and then press Enter. (Be sure to store the password in a secure location for later use.)
   5. **Remove anonymous users** – Type Y, and then press Enter.
   6. **Disallow root login remotely** – Type Y, and then press Enter.
   7. **Remove test database and access to it** – Type Y, and then press Enter.
   8. **Reload privilege tables now** – Type Y, and then press Enter.

   For Ubuntu Server:

   1. **Would you like to setup VALIDATE PASSWORD plugin** – Type y, and then press Enter.
   2. **There are three levels of password validation policy** – Type 0, 1, or 2, and then press Enter.
   3. **New password** – Type a password, and then press Enter.
4. **Re-enter new password** – Type the password again, and then press Enter. (Be sure to store the password in a secure location for later use.)

5. **Do you wish to continue with the password provided** – Type y, and then press Enter.

6. **Remove anonymous users** – Type y, and then press Enter.

7. **Disallow root login remotely** – Type y, and then press Enter.

8. **Remove test database and access to it** – Type y, and then press Enter.

9. **Reload privilege tables now** – Type y, and then press Enter.

3. Start the MySQL command-line client as the root user by running the following command. When prompted, type the root user's password that you set earlier, and then press Enter. (The prompt changes to `mysql>` while you are in the MySQL command-line client.)

   ```
   sudo mysql -uroot -p
   ```

4. Create a MySQL database for the WordPress site to use. To do this, run the following command, replacing `my_db_name` with a name for the new database, for example, `mysite`. (Be sure to save this database name in a secure location for later use.)

   ```
   CREATE DATABASE my_db_name;
   ```

5. Create a MySQL user for the WordPress site to use. To do this, run the following command, replacing `my_user_name` with the user's name (for example, `wordpress-user`) and replacing `my_password` with a password for the user. (Be sure to save this user name and password in a secure location for later use.)

   ```
   GRANT ALL PRIVILEGES ON *.* TO 'my_user_name'@'localhost' IDENTIFIED BY 'my_password';
   ```

6. Exit the MySQL command-line client by running the following command. (The prompt changes back to `#`.)

   ```
   exit;
   ```

---

**Step 3: Set Up the WordPress Website**

In this step, you set up a WordPress website by setting its base configuration details, such as MySQL database connection information.

You then finish this step by opening the website and specifying its display title, user name and password, and other settings.

1. Rename the `wp-config-sample.php` file to `wp-config.php` within the WordPress installation, as follows.

   a. In the **Environment** window, expand the `wordpress` folder.
   c. Choose **Rename**.
   d. Type `wp-config` to rename this file to `wp-config.php`, and then press Enter.

2. Configure the `wp-config.php` file for the WordPress website. To do this, double-click the `wp-config.php` file to open it in the editor, replace the following values, and then save and close the file.

   - Replace `database_name_here` with the name of the MySQL database that you created earlier, for example, `mysite`.
Step 4: Share the WordPress Website Over the Internet

In this step, you set up the Apache HTTP Server with recommended ports, file locations, owners, and access permissions for the WordPress website.

You then enable incoming web traffic to view that website by setting up the security group in Amazon EC2 and network access control list (network ACL) in Amazon Virtual Private Cloud (Amazon VPC) that are associated with this EC2 environment. Each EC2 environment must be associated with both a security group in Amazon EC2 and a network ACL in Amazon VPC. However, while the default network ACL
Step 4: Share the WordPress Website Over the Internet

in an AWS account allows all incoming and outgoing traffic for the environment, the default security
group allows only incoming traffic using SSH over port 22. For more information, see the section called
"Amazon VPC Settings" (p. 467).

You then finish this step by successfully viewing the website from outside of the AWS Cloud9 IDE.

1. Stop the WordPress website. To do this, on the PHP (built-in web server) runner tab in the AWS
Cloud9 IDE, choose Stop.

2. Make backup copies of key Apache HTTP Server configuration files that you'll be modifying later
in this procedure, in case you accidentally make the original files inoperable. To do this, run the
following file copy commands.

   For Amazon Linux, run the following single file copy command only:

   `sudo cp /etc/httpd/conf/httpd.conf /etc/httpd/conf/httpd.conf.bak`

   For Ubuntu Server, run the following three file copy commands, one after another in the following
   order:

   `sudo cp /etc/apache2/ports.conf /etc/apache2/ports.conf.bak`
   `sudo cp /etc/apache2/sites-enabled/000-default.conf /etc/apache2/sites-enabled/000-
default.conf.bak`
   `sudo cp /etc/apache2/apache2.conf /etc/apache2/apache2.conf.bak`

3. Bind Apache HTTP Server to port 8080, instead of the default port of 80. To do this, run the
following file search-and-replace command.

   For Amazon Linux:

   `sudo sed -i 's/Listen 80/Listen 8080/g' /etc/httpd/conf/httpd.conf`

   For Ubuntu Server:

   `sudo sed -i 's/Listen 80/Listen 8080/g' /etc/apache2/ports.conf`

4. Add or change virtual host settings to listen on port 8080, instead of the default port of 80. To do
this, run the following file append or search-and-replace command.

   For Amazon Linux, append the virtual host settings to the existing configuration file:

   `sudo echo -e "<VirtualHost *:8080>
    DocumentRoot /var/www/html
</VirtualHost>" | sudo tee -a /etc/httpd/conf/httpd.conf`

   For Ubuntu Server, search and replace existing virtual host settings in an existing configuration file:

   `sudo sed -i 's/<VirtualHost \*:80>/<VirtualHost \*:8080>/g' /etc/apache2/sites-enabled/000-default.conf`

5. Restart the Apache HTTP Server to have it use the new settings. To do this, run the following
command.

   For Amazon Linux (you might need to run this command twice):

   `sudo service httpd restart && sudo service httpd status`
Step 4: Share the WordPress Website Over the Internet

For Ubuntu Server (to return to the command prompt, press q):

```
sudo service apache2 restart & & sudo service apache2 status
```

6. View the Apache HTTP Server default information webpage from within the AWS Cloud9 IDE. To do this, on the main menu bar, choose Preview, Preview Running Application. A new window opens in the IDE and displays the Apache HTTP Server default information webpage.

7. Now switch the Apache HTTP Server to use the WordPress website's root directory by running the following file search-and-replace command.

For Amazon Linux:

```
sudo sed -i 's/<Directory "/var/www/html">/<Directory "/home/ec2-user/environment/wordpress">/g' /etc/httpd/conf/httpd.conf
```

For Ubuntu Server:

```
sudo sed -i 's/<Directory "/var/www/">/<Directory "/home/ubuntu/environment/wordpress/">/g' /etc/apache2/apache2.conf
```

8. Switch the Apache HTTP Server to specify using the document root for the WordPress website by running the following file search-and-replace command.

For Amazon Linux:

```
```

For Ubuntu Server:

```
sudo sed -i 's/DocumentRoot "/var/www/html/DocumentRoot "/home/ubuntu/environment/wordpress/g' /etc/apache2/sites-enabled/000-default.conf
```

9. Set up the website root with recommended owners and access permissions. To do this, run the following six commands, one at a time in the following order. To understand what each command does, read the information after the # character after each command.

For Amazon Linux:

```
sudo groupadd web-content # Create a group named web-content.
sudo usermod -G web-content -a ec2-user # Add the user ec2-user (your default user for this environment) to the group web-content.
sudo usermod -G web-content -a apache # Add the user apache (Apache HTTP Server) to the group web-content.
sudo chown -R ec2-user:web-content /home/ec2-user/environment/wordpress # Change the owner of /home/ec2-user/environment/wordpress and its files to user ec2-user and group web-content.
sudo find /home/ec2-user/environment/wordpress -type f -exec chmod u=rw,g=rx,o=rx {} \; # Change all file permissions within /home/ec2-user/environment/wordpress to user read/write, group read-only, and others read/execute.
```

For Ubuntu Server:

```
sudo chown -R ec2-user:web-content /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R g=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R o=rx /home/ec2-user/environment/wordpress
```

```
sudo chmod -R u=rwx /home/ec2-user/environment/wordpress
```
Step 4: Share the WordPress Website Over the Internet

For Ubuntu Server:

```
sudo groupadd web-content # Create a group named web-content.
sudo usermod -G web-content -a ubuntu # Add the user ubuntu (your default user for this environment) to the group web-content.
sudo usermod -G web-content -a www-data # Add the user www-data (Apache HTTP Server) to the group web-content.
sudo chown -R ubuntu:web-content /home/ubuntu/environment/wordpress # Change the owner of /home/ubuntu/environment/wordpress and its files to user ubuntu and group web-content.
```

```
sudo find /home/ubuntu/environment/wordpress -type f -exec chmod u=rw,g=rx,o=rx {} \; # Change all file permissions within /home/ubuntu/environment/wordpress to user read/write, group read-only, and others read/execute.
sudo find /home/ubuntu/environment/wordpress -type d -exec chmod u=rwx,g=rx,o=rx {} \; # Change /home/ubuntu/environment/wordpress directory permissions to user read/write/execute, group read/execute, and others read/execute.
```

10. Restart the Apache HTTP Server to have it use the new settings. To do this, run the following command.

For Amazon Linux (you might need to run this command twice):

```
sudo service httpd restart && sudo service httpd status
```

For Ubuntu Server (to return to the command prompt, press q):

```
sudo service apache2 restart && sudo service apache2 status
```

11. View the WordPress website from within the AWS Cloud9 IDE. To do this, on the main menu bar, choose Preview, Preview Running Application. A new window opens in the IDE and displays a Not Found page (which is expected at this point).

12. Open the WordPress website in a new tab within the same web browser as the AWS Cloud9 IDE. To do this, on the address bar in the new window, choose Pop Out Into New Window. The new tab displays the same Not Found page (which is still expected at this point).

13. In the new tab within the same web browser as the AWS Cloud9 IDE, add /index.php to the end of the existing URL, and then press Enter. The WordPress website's home page is displayed.

14. Enable incoming web traffic over port 8080 to view the new webpage by setting up the network ACL in Amazon VPC and the security group Amazon EC2 that is associated with this EC2 environment. To do this, run the following eight commands, one at a time in the following order. To understand what each command does, read the information after the # character for each command.

**Important**

Running the following commands enables incoming web traffic over port 8080 for all EC2 environments and Amazon EC2 instances that are associated with the security group and network ACL for this environment. This might result in unexpectedly enabling incoming web traffic over port 8080 for EC2 environments and Amazon EC2 instances other than this one.

**Note**

The following second through fourth commands enable the security group to allow incoming web traffic over port 8080. If you have a default security group, which only allows
incoming SSH traffic over port 22, then you must run the first command followed by these
second through fourth commands. However, if you have a custom security group already
allows incoming web traffic over port 8080, you can safely skip running those commands.
The following fifth through eighth commands enable the network ACL to allow incoming
web traffic over port 8080. If you have a default network ACL, which already allows all
incoming traffic over all ports, then you can safely skip running those commands. However,
if you have a custom network ACL that doesn’t allow incoming web traffic over port 80,
then you must run the first command followed by these fifth through eighth commands.

```bash
MY_INSTANCE_ID=$(curl http://169.254.169.254/latest/meta-data/instance-id) # Get the ID
of the instance for the environment, and store it temporarily.

MY_SECURITY_GROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query
'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text) # Get the ID
of the security group associated with the instance, and store it temporarily.

aws ec2 authorize-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --protocol
tcp --cidr 0.0.0.0/0 --port 8080 # Add an inbound rule to the security group to allow
all incoming IPv4-based traffic over port 8080.

aws ec2 authorize-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --ip-
permissions IpProtocol=tcp,Ipv6Ranges=’{{CidrIpv6=::/0}}’,FromPort=8080,ToPort=8080
# Add an inbound rule to the security group to allow all incoming IPv6-based traffic
over port 8080.

MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query
'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet
associated with the instance, and store it temporarily.

MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters Name=association.subnet-
Id,Values=$MY_SUBNET_ID --query 'NetworkAcls[].Associations[0].NetworkAclId' --output
text) # Get the ID of the network ACL associated with the subnet, and store it
temporarily.

aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --
protocol tcp --rule-action allow --rule-number 10000 --cidr-block 0.0.0.0/0 --port-ange From=8080,To=8080 # Add an inbound rule to the network ACL to allow all IPv4-
based traffic over port 8080. Advanced users: change this suggested rule number as
desired.

aws ec2 create-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --
protocol tcp --rule-action allow --rule-number 10100 --ipv6-cidr-block ::/0 --port-ange From=8080,To=8080 # Add an inbound rule to the network ACL to allow all IPv6-
based traffic over port 8080. Advanced users: change this suggested rule number as
desired.

15. Get the URL to the index.php file within the web server root. To do this, run the following
command, and use a new web browser tab or a different web browser separate from the AWS
Cloud9 IDE to go to the URL that is displayed. If successful, the webpage displays the WordPress
website home page.

```bash
http://$MY_PUBLIC_IP:8080/index.php # Get the URL to the index.php file within the web
server root.
```

**Step 5: Clean Up**

If you want to keep using this environment but you want to disable incoming web traffic over port
8080, then run the following eight commands, one at a time in the following order, to delete the
corresponding incoming traffic rules that you set earlier in the security group and network ACL that are
associated with the environment. To understand what each command does, read the information after the # character for each command.

**Important**
Running the following commands disables incoming web traffic over port 8080 for all EC2 environments and Amazon EC2 instances that are associated with the security group and network ACL for this environment. This might result in unexpectedly disabling incoming web traffic over port 8080 for EC2 environments and Amazon EC2 instances other than this one.

**Note**
The following fifth through eighth commands remove existing rules in order to block the network ACL from allowing incoming web traffic over port 8080. If you have a default network ACL, which already allows all incoming traffic over all ports, then you can safely skip running those commands. However, if you have a custom network ACL with existing rules that allow incoming web traffic over port 8080 and you want to delete those rules, then you must run the first command followed by these fifth through eighth commands.

```bash
MY_SECURITY_GROUP_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SecurityGroups[0].GroupId' --output text) # Get the ID of the security group associated with the instance, and store it temporarily.
aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --protocol tcp --cidr 0.0.0.0/0 --port 8080 # Delete the existing inbound rule from the security group to block all incoming IPv4-based traffic over port 8080.
aws ec2 revoke-security-group-ingress --group-id $MY_SECURITY_GROUP_ID --ip-permissions IpProtocol=tcp,Ipv6Ranges='[{CidrIpv6=::/0}]',FromPort=8080,ToPort=8080 # Delete the existing inbound rule from the security group to block all incoming IPv6-based traffic over port 8080.
MY_SUBNET_ID=$(aws ec2 describe-instances --instance-id $MY_INSTANCE_ID --query 'Reservations[].Instances[0].SubnetId' --output text) # Get the ID of the subnet associated with the instance, and store it temporarily.
MY_NETWORK_ACL_ID=$(aws ec2 describe-network-acls --filters Name=association.subnet-id,Values=$MY_SUBNET_ID --query 'NetworkAcls[].Associations[0].NetworkAclId' --output text) # Get the ID of the network ACL associated with the subnet, and store it temporarily.
aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-number 10000 # Delete the existing inbound rule from the network ACL to block all IPv4-based traffic over port 8080. Advanced users: if you originally created this rule with a different number, change this suggested rule number to match.
aws ec2 delete-network-acl-entry --network-acl-id $MY_NETWORK_ACL_ID --ingress --rule-number 10100 # Delete the existing inbound rule from the network ACL to block all IPv6-based traffic over port 8080. Advanced users: if you originally created this rule with a different number, change this suggested rule number to match.
```

To prevent ongoing charges to your AWS account after you’re done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

### Java Sample for AWS Cloud9

This sample enables you to run some Java code in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

### Step 1: Install Required Tools

In this step, you install a set of Java development tools in your AWS Cloud9 development environment. If you already have a set of Java development tools such as the Oracle JDK or OpenJDK installed in your environment, you can skip ahead to Step 2: Add Code (p. 402). This sample was developed with OpenJDK 8, which you can install in your environment by completing the following procedure.

1. Confirm whether OpenJDK 8 is already installed. To do this, in a terminal session in the AWS Cloud9 IDE, run the command line version of the Java runner with the \(-version\) option. (To start a new terminal session, on the menu bar, choose **Window, New Terminal**.)

```bash
java -version
```

Based on the output of the preceding command, do one of the following:

- If the output states that the **java** command isn't found, continue with step 2 in this procedure to install OpenJDK 8.
- If the output contains values starting with **Java(TM), Java Runtime Environment, Java SE, J2SE, or Java2, the OpenJDK isn't installed or isn't set as the default Java development toolset. Continue with step 2 in this procedure to install OpenJDK 8, and then switch to using OpenJDK 8.
- If the output contains values starting with **java version 1.8 and OpenJDK, skip ahead to Step 2: Add Code (p. 402). OpenJDK 8 is installed correctly for this sample.
- If the output contains a **java version** less than 1.8 and values starting with **OpenJDK, continue with step 2 in this procedure to upgrade the installed OpenJDK version to OpenJDK 8.

2. Ensure the latest security updates and bug fixes are installed. To do this, run the **yum** tool (for Amazon Linux) or the **apt** tool (for Ubuntu Server) with the **update** command.
For Amazon Linux:

```
sudo yum -y update
```

For Ubuntu Server:

```
sudo apt update
```

3. Install OpenJDK 8. To do this, run the `yum` tool (for Amazon Linux) or the `apt` tool (for Ubuntu Server) with the `install` command, specifying the OpenJDK 8 package.

For Amazon Linux:

```
sudo yum -y install java-1.8.0-openjdk-devel
```

For Ubuntu Server:

```
sudo apt install -y openjdk-8-jdk
```

For more information, see How to download and install prebuilt OpenJDK packages on the OpenJDK website.

4. Switch or upgrade the default Java development toolset to OpenJDK 8. To do this, run the `update-alternatives` command with the `--config` option. Run this command twice to switch or upgrade the command line versions of the Java runner and compiler.

```
sudo update-alternatives --config java
sudo update-alternatives --config javac
```

At each prompt, type the selection number for OpenJDK 8 (the one that contains `java-1.8`).

5. Confirm that the command line versions of the Java runner and compiler are using OpenJDK 8. To do this, run the command line versions of the Java runner and compiler with the `--version` option.

```
java --version
javac --version
```

If OpenJDK 8 is installed and set correctly, the Java runner version output contains a value starting with `openjdk version 1.8`, and the Java compiler version output starts with the value `javac 1.8`.

### Step 2: Add Code

In the AWS Cloud9 IDE, create a file with the following code, and save the file with the name `hello.java`. (To create a file, on the menu bar, choose File, New File. To save the file, choose File, Save.)

```java
public class hello {
    public static void main(String []args) {
        System.out.println("Hello, World!");
        System.out.println("The sum of 2 and 3 is 5.");
    }
}
```
int sum = Integer.parseInt(args[0]) + Integer.parseInt(args[1]);
System.out.format("The sum of %s and %s is %s.\n", args[0], args[1], Integer.toString(sum));

Step 3: Build and Run the Code

1. Use the command line version of the Java compiler to compile the hello.java file into a hello.class file. To do this, using the terminal in the AWS Cloud9 IDE, from the same directory as the hello.java file, run the Java compiler, specifying the hello.java file.

   javac hello.java

2. Use the command line version of the Java runner to run the hello.class file. To do this, from the same directory as the hello.class file, run the Java runner, specifying the name of the hello class that was declared in the hello.java file, with two integers to add (for example, 5 and 9).

   java hello 5 9

3. Compare your output.

   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.

Step 4: Set Up to Use the AWS SDK for Java

You can enhance this sample to use the AWS SDK for Java to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install Apache Maven or Gradle in your environment. Maven and Gradle are common build automation systems that can be used with Java projects. After you install Maven or Gradle, you use it to generate a new Java project. In this new project, you add a reference to the AWS SDK for Java. This AWS SDK for Java provides a convenient way to interact with AWS services such as Amazon S3, from your Java code.

Topics
- Set Up With Maven (p. 403)
- Set Up With Gradle (p. 406)

Set Up With Maven

1. Install Maven in your environment. To see whether Maven is already installed, using the terminal in the AWS Cloud9 IDE, run Maven with the -version option.

   mvn -version

   If successful, the output contains the Maven version number. If Maven is already installed, skip ahead to step 4 in this procedure to use Maven to generate a new Java project in your environment.

2. Install Maven by using the terminal to run the following commands.
For Amazon Linux, the following commands get information about the package repository where Maven is stored, and then use this information to install Maven.

```bash
sudo wget http://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo
-s -O /etc/yum.repos.d/epel-apache-maven.repo
sudo sed -i s/$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo
sudo yum install -y apache-maven
```

For more information about the preceding commands, see Extra Packages for Enterprise Linux (EPEL) on the Fedora Project Wiki website.

For Ubuntu Server, run the following command instead.

```bash
sudo apt install -y maven
```

3. Confirm the installation by running Maven with the `-version` option.

```bash
mvn -version
```

4. Use Maven to generate a new Java project. To do this, use the terminal to run the following command from the directory where you want Maven to generate the project (for example, the root directory of your environment).

```bash
mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

The preceding command creates the following directory structure for the project in your environment.

```
my-app
|-- src
|  |-- main
|     |-- java
|        |-- com
|           |-- mycompany
|               |-- app
|                   |-- App.java
|-- test
|  |-- java
|     |-- com
|        |-- mycompany
|           |-- app
|               |-- AppTest.java
|-- pom.xml
```

For more information about the preceding directory structure, see Maven Quickstart Archetype and Introduction to the Standard Directory Layout on the Apache Maven Project website.

5. Modify the Project Object Model (POM) file for the project. (A POM file defines a Maven project's settings.) To do this, from the Environment window, open the `my-app/pom.xml` file. In the editor, replace the file's current contents with the following code, and then save the `pom.xml` file.

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany.app</groupId>
</project>
```
<artifactId>my-app</artifactId>
<packaging>jar</packaging>
<version>1.0-SNAPSHOT</version>
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-assembly-plugin</artifactId>
      <version>3.0.0</version>
      <configuration>
        <descriptorRefs>
          <descriptorRef>jar-with-dependencies</descriptorRef>
        </descriptorRefs>
        <archive>
          <manifest>
            <mainClass>com.mycompany.app.App</mainClass>
          </manifest>
        </archive>
      </configuration>
      <executions>
        <execution>
          <phase>package</phase>
          <goals>
            <goal>single</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
  </plugins>
</build>
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
  </dependency>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-java-sdk</artifactId>
    <version>1.11.330</version>
  </dependency>
</dependencies>
</project>

The preceding POM file includes project settings that specify declarations such as the following:

- The `artifactId` setting of `my-app` sets the project's root directory name, and the `group-id` setting of `com.mycompany.app` sets the `com/mycompany/app` subdirectory structure and the package declaration in the `App.java` and `AppTest.java` files.

- The `artifactId` setting of `my-app`, with the `packaging` setting of `jar`, the `version` setting of `1.0-SNAPSHOT`, and the descriptorRef setting of `jar-with-dependencies` set the output JAR file's name of `my-app-1.0-SNAPSHOT-jar-with-dependencies.jar`.

- The `plugin` section declares that a single JAR, which includes all dependencies, will be built.

- The `dependency` section with the `groupId` setting of `com.amazonaws` and the `artifactId` setting of `aws-java-sdk` includes the AWS SDK for Java library files. The AWS SDK for Java version to use is declared by the `version` setting. To use a different version, replace this version number.

Skip ahead to Step 5: Set Up AWS Credentials Management in Your Environment (p. 408).
Set Up With Gradle

1. Install Gradle in your environment. To see whether Gradle is already installed, using the terminal in the AWS Cloud9 IDE, run Gradle with the `-version` option.

   ```bash
   gradle -version
   ```

   If successful, the output contains the Gradle version number. If Gradle is already installed, skip ahead to step 4 in this procedure to use Gradle to generate a new Java project in your environment.

2. Install Gradle by using the terminal to run the following commands. These commands install and run the SDKMAN! tool, and then use SDKMAN! to install the latest version of Gradle.

   ```bash
   curl -s "https://get.sdkman.io" | bash
   source "$HOME/.sdkman/bin/sdkman-init.sh"
   sdk install gradle
   ```

   For more information about the preceding commands, see Installation on the SDKMAN! website and Install with a package manager on the Gradle website.

3. Confirm the installation by running Gradle with the `-version` option.

   ```bash
   gradle -version
   ```

4. Use Gradle to generate a new Java project in your environment. To do this, use the terminal to run the following commands to create a directory for the project, and then switch to that directory.

   ```bash
   mkdir my-app
   cd my-app
   ```

5. Run the following command to have Gradle generate a new Java application project in the `my-app` directory in your environment.

   ```bash
   gradle init --type java-application
   ```

   The preceding command creates the following directory structure for the project in your environment.

   ```
   my-app
      |- .gradle
      |  `- (various supporting project folders and files)
      `- gradle
         |- (various supporting project folders and files)
         `- src
            |- main
            |  `- java
            |     `- App.java
            `- test
               `- java
                  `- AppTest.java
           `- build.gradle
           `- gradlew
           `- gradlew.bat
              `- settings.gradle
   ```

6. Modify the `AppTest.java` for the project. (If you do not do this, the project might not build or run as expected). To do this, from the Environment window, open the `my-app/src/test/java/`
AppTest.java file. In the editor, replace the file’s current contents with the following code, and then save the AppTest.java file.

```java
import org.junit.Test;
import static org.junit.Assert.*;

public class AppTest {
    @Test public void testAppExists () {
        try {
            Class.forName("com.mycompany.app.App");
        } catch (ClassNotFoundException e) {
            fail("Should have a class named App.");
        }
    }
}
```

7. Modify the build.gradle file for the project. (A build.gradle file defines a Gradle project’s settings.) To do this, from the Environment window, open the my-app/build.gradle file. In the editor, replace the file’s current contents with the following code, and then save the build.gradle file.

```groovy
apply plugin: 'java'
apply plugin: 'application'

repositories {
    jcenter()
    mavenCentral()
}

buildscript {
    repositories {
        mavenCentral()
    }
    dependencies {
        classpath "io.spring.gradle:dependency-management-plugin:1.0.3.RELEASE"
    }
}

apply plugin: "io.spring.dependency-management"

dependencyManagement {
    imports {
        mavenBom 'com.amazonaws:aws-java-sdk-bom:1.11.330'
    }
}

dependencies {
    compile 'com.amazonaws:aws-java-sdk-s3'
    testCompile group: 'junit', name: 'junit', version: '4.12'
}

run {
    if (project.hasProperty("appArgs")) {
        args Eval.me(appArgs)
    }
}

mainClassName = 'App'
```

The preceding build.gradle file includes project settings that specify declarations such as the following:
The `io.spring.dependency-management` plugin is used to import the AWS SDK for Java Maven Bill of Materials (BOM) to manage AWS SDK for Java dependencies for the project. `classpath` declares the version to use. To use a different version, replace this version number.

- `com.amazonaws:aws-java-sdk-s3` includes the Amazon S3 portion of the AWS SDK for Java library files. `mavenBom` declares the version to use. If you want to use a different version, replace this version number.

**Step 5: Set Up AWS Credentials Management in Your Environment**

Each time you use the AWS SDK for Java to call an AWS service, you must provide a set of AWS credentials with the call. These credentials determine whether the AWS SDK for Java has the appropriate permissions to make that call. If the credentials don't cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in [Calling AWS Services from an Environment in AWS Cloud9](p. 70), and then return to this topic.

For additional information, see [Set up AWS Credentials and Region for Development](p. 70) in the *AWS SDK for Java Developer Guide*.

**Step 6: Add AWS SDK Code**

In this step, you add code to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created.

From the Environment window, open the `my-app/src/main/java/com/mycompany/app/App.java` file for Maven or the `my-app/src/main/java/App.java` file for Gradle. In the editor, replace the file's current contents with the following code, and then save the `App.java` file.

```java
package com.mycompany.app;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import com.amazonaws.services.s3.model.Bucket;
import com.amazonaws.services.s3.model.CreateBucketRequest;
import java.util.List;

public class App {
    private static AmazonS3 s3;

    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.format("Usage: <the bucket name> <the AWS Region to use>\nExample: my-test-bucket us-east-2\n");
            return;
        }

        String bucket_name = args[0];
        String region = args[1];

        s3 = AmazonS3ClientBuilder.standard()
            .withCredentials(new ProfileCredentialsProvider())
            .withRegion(region)
            .build();
    }
}
```
Step 7: Build and Run the AWS SDK Code

To run the code from the previous step, run the following commands from the terminal. These commands use Maven or Gradle to create an executable JAR file for the project, and then use the Java runner to run the JAR. The JAR runs with the name of the bucket to create in Amazon S3 (for example, my-test-bucket) and the ID of the AWS Region to create the bucket in as input (for example, us-east-2).

For Maven, run the following commands.

```
cd my-app
mvn package
```

For Gradle, run the following commands.

```
cd my-app
gradle war
```
Step 8: Clean Up

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

C++ Sample for AWS Cloud9

This sample enables you to run some C++ code in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

Topics
- Prerequisites (p. 410)
- Step 1: Install Required Tools (p. 411)
- Step 2: Add Code (p. 411)
- Step 3: Run the Code (p. 412)
- Step 4: Install and Configure the AWS SDK for C++ (p. 412)
- Step 5: Add AWS SDK Code (p. 414)
- Step 6: Build and Run the AWS SDK Code (p. 417)
- Step 7: Clean Up (p. 418)

Prerequisites

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.
- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.
Step 1: Install Required Tools

In this step, you install and configure the GNU Compiler Collection (GCC), which is required to run this sample.

1. In a terminal session in the AWS Cloud9 IDE, confirm whether GCC is already installed by running the `g++ --version` command. (To start a new terminal session, on the menu bar, choose Window, New Terminal.) If successful, the output contains the GCC version number. Otherwise, an error message should be output. If GCC is installed, skip ahead to Step 2: Add Code (p. 411).

2. Run the `yum update` command (on Amazon Linux) or the `apt update` command (on Ubuntu Server) to help ensure the latest security updates and bug fixes are installed.

   For Amazon Linux:
   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:
   ```bash
   sudo apt update
   ```

3. To install GCC, run the `yum install` command (for Amazon Linux) or the `apt install` command (for Ubuntu Server).

   For Amazon Linux:
   ```bash
   sudo yum -y install gcc-c++
   ```

   For Ubuntu Server:
   ```bash
   sudo apt install -y g++
   ```

4. Confirm that GCC is now successfully installed by running the `g++ --version` command. If successful, the output contains the GCC version number.

Step 2: Add Code

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `hello.cpp`. (To create a file, on the menu bar, choose File, New File. To save the file, choose File, Save.)

```cpp
#include <iostream>
#include <stdlib.h>

int main( int argc, char *argv[] )
{
    std::cout << "Hello, World!\n";
    std::cout << "The sum of 2 and 3 is 5.\n";

    if (argc > 2) {
        std::cout << "The sum of " << argv[1] << " and " << argv[2]
                  << " is " << atoi(argv[1]) + atoi(argv[2]) << "\n";
    }

    return 0;
}
```
Step 3: Run the Code

1. Compile the `hello.cpp` source code into an object module, and then link the object module into a program named `hello`. Do this by choosing Run, Build System, G++ followed by Run, Build on the menu bar.

   **Note**
   If G++ is not available, you can create a custom builder for G++.

   1. Choose Run, Build System, New Build System on the menu bar.
   2. On the My Builder.build tab, replace the tab’s contents with this code.

   ```
   {  
     "cmd": [ "g++", "-o", "$file_base_name", "$file_name" ],  
     "info": "Compiling $file_name and linking to $file_base_name...",  
     "selector": "source.cpp"
   }
   ```

   3. Choose File, Save As on the menu bar, and then save the file as G++.build in the /.c9/ builders folder.

   4. Choose the hello.cpp tab to make it active.

   5. Choose Run, Build System, G++ followed by Run, Build.

2. In the AWS Cloud9 IDE, run the code by choosing Run, Run Configurations, New Run Configuration on the menu bar.


   **Note**
   If C++ isn't available, you can create a custom runner for C++.


   2. On the My Runner.run tab, replace the tab’s contents with this code.

   ```
   {  
     "cmd" : ["$file", "$args"],  
     "info" : "Running $project_path$file_name...",  
     "selector" : "source"
   }
   ```

   3. Choose File, Save As on the menu bar, and then save the file as C++.run in the /.c9/ runners folder.


5. Choose the Run button, and compare your output.

   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.

Step 4: Install and Configure the AWS SDK for C++

You can enhance this sample to use the AWS SDK for C++ to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install and configure the AWS SDK for C++, which provides a convenient way to interact with AWS services, such as Amazon S3, from your C++ code. Before you install the AWS SDK for C++,
you must install some dependencies. After you install the AWS SDK for C++, you must set up credentials management in your environment. The AWS SDK for C++ needs these credentials to interact with AWS services.

**Note**
The following steps require your environment to be running on an Amazon EC2 instance or your own server that has at least 4 GB of RAM.

### To install AWS SDK for C++ dependencies

From a terminal session in the AWS Cloud9 IDE, run the following command to install several packages that the AWS SDK for C++ depends on to run correctly.

For Amazon Linux:

```bash
sudo yum -y install libcurl-devel openssl-devel libuuid-devel cmake3
```

For Ubuntu Server:

```bash
sudo apt install -y libcurl4-openssl-dev libssl-dev uuid-dev zlib1g-dev libpulse-dev cmake
```

### To download and extract the AWS SDK for C++ source code

1. Run the `wget` command, specifying the location of the AWS SDK for C++ source.

   ```bash
   wget https://github.com/aws/aws-sdk-cpp/archive/master.zip
   ```

2. Run the `unzip` command, specifying the name of the .zip file you just downloaded.

   ```bash
   unzip master.zip
   ```

3. Run the `rm` command to delete the .zip file, as you no longer need it.

   ```bash
   rm master.zip
   ```

### To build the AWS SDK for C++

**Note**
This step could take up to one or more hours to complete, depending on the computing resources available to your Amazon EC2 instance or your own server and how much of the AWS SDK for C++ you choose to build.

1. Create a folder to build the AWS SDK for C++ into.

   ```bash
   mkdir sdk_build
   ```

2. Switch to the folder you just created.

   ```bash
   cd sdk_build
   ```

3. Prepare to build the AWS SDK for C++ into this folder.

   For Amazon Linux:
Step 5: Add AWS SDK Code

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You will run this code later.

1. In the AWS Cloud9 IDE, create a file with this content, and save the file with the name s3-demo.cpp at the root (/) of your environment.

```cpp
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/Bucket.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>
#include <string>

bool ListMyBuckets(Aws::S3::S3Client s3_client);
```

To set up credentials management in your environment

Each time you use the AWS SDK for C++ to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for C++ has the appropriate permissions to make that call. If the credentials don't cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Providing AWS Credentials in the AWS SDK for C++ Developer Guide.

Step 5: Add AWS SDK Code

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You will run this code later.

1. In the AWS Cloud9 IDE, create a file with this content, and save the file with the name s3-demo.cpp at the root (/) of your environment.

```cpp
#include <aws/core/Aws.h>
#include <aws/s3/S3Client.h>
#include <aws/s3/model/Bucket.h>
#include <aws/s3/model/CreateBucketConfiguration.h>
#include <aws/s3/model/CreateBucketRequest.h>
#include <aws/s3/model/DeleteBucketRequest.h>
#include <string>

bool ListMyBuckets(Aws::S3::S3Client s3_client);
```
bool CreateMyBucket(Aws::S3::S3Client s3_client, Aws::String bucket_name,
        Aws::S3::Model::BucketLocationConstraint region);
bool DeleteMyBucket(Aws::S3::S3Client s3_client, Aws::String bucket_name);
void Cleanup(Aws::SDKOptions options);

int main(int argc, char** argv) {
    if (argc < 3) {
        std::cout << "Usage: ./s3-demo <the bucket name> <the AWS Region to use>" << std::endl;
        return false;
    }

    Aws::String bucket_name = argv[1];
    Aws::Client::ClientConfiguration client_configuration;
    Aws::S3::Model::BucketLocationConstraint region;

    // Set the AWS Region to use, based on the user's AWS Region input ID.
    if (strcmp(argv[2], "ap-northeast-1") == 0) {
        client_configuration.region = Aws::Region::AP_NORTHEAST_1;
        region = Aws::S3::Model::BucketLocationConstraint::ap_northeast_1;
    } else if (strcmp(argv[2], "ap-northeast-2") == 0) {
        client_configuration.region = Aws::Region::AP_NORTHEAST_2;
        region = Aws::S3::Model::BucketLocationConstraint::ap_northeast_2;
    } else if (strcmp(argv[2], "ap-south-1") == 0) {
        client_configuration.region = Aws::Region::AP_SOUTH_1;
        region = Aws::S3::Model::BucketLocationConstraint::ap_south_1;
    } else if (strcmp(argv[2], "ap-southeast-1") == 0) {
        client_configuration.region = Aws::Region::AP_SOUTHEAST_1;
        region = Aws::S3::Model::BucketLocationConstraint::ap_southeast_1;
    } else if (strcmp(argv[2], "ap-southeast-2") == 0) {
        client_configuration.region = Aws::Region::AP_SOUTHEAST_2;
        region = Aws::S3::Model::BucketLocationConstraint::ap_southeast_2;
    } else if (strcmp(argv[2], "cn-north-1") == 0) {
        client_configuration.region = Aws::Region::CN_NORTH_1;
        region = Aws::S3::Model::BucketLocationConstraint::cn_north_1;
    } else if (strcmp(argv[2], "eu-central-1") == 0) {
        client_configuration.region = Aws::Region::EU_CENTRAL_1;
        region = Aws::S3::Model::BucketLocationConstraint::eu_central_1;
    } else if (strcmp(argv[2], "eu-west-1") == 0) {
        client_configuration.region = Aws::Region::EU_WEST_1;
        region = Aws::S3::Model::BucketLocationConstraint::eu_west_1;
    } else if (strcmp(argv[2], "sa-east-1") == 0) {
        client_configuration.region = Aws::Region::SA_EAST_1;
        region = Aws::S3::Model::BucketLocationConstraint::sa_east_1;
    } else if (strcmp(argv[2], "us-west-1") == 0) {
        client_configuration.region = Aws::Region::US_WEST_1;
        region = Aws::S3::Model::BucketLocationConstraint::us_west_1;
    } else if (strcmp(argv[2], "us-west-2") == 0) {
        client_configuration.region = Aws::Region::US_WEST_2;
        region = Aws::S3::Model::BucketLocationConstraint::us_west_2;
    } else {
        std::cout << "Unrecognized AWS Region ID " << argv[2] << " " << std::endl;
        return false;
    }

    Aws::SDKOptions options;
    Aws::InitAPI(options);
    {
        Aws::S3::S3Client s3_client(client_configuration);
        if (!ListMyBuckets(s3_client)) {
            Cleanup(options);
        }
    }
}
if (!CreateMyBucket(s3_client, bucket_name, region)) {
    Cleanup(options);
}

if (!ListMyBuckets(s3_client)) {
    Cleanup(options);
}

if (!DeleteMyBucket(s3_client, bucket_name)) {
    Cleanup(options);
}

if (!ListMyBuckets(s3_client)) {
    Cleanup(options);
}

Cleanup(options);

// List all of your available buckets.
bool ListMyBuckets(Aws::S3::S3Client s3_client) {
    auto outcome = s3_client.ListBuckets();

    if (outcome.IsSuccess()) {
        std::cout << "My buckets now are:"
        << std::endl
        << std::endl;
        Aws::Vector<Aws::S3::Model::Bucket> bucket_list =
        outcome.GetResult().GetBuckets();

        for (auto const &bucket: bucket_list) {
            std::cout << bucket.GetName() << std::endl;
        }

        std::cout << std::endl;
        return true;
    } else {
        std::cout << "ListBuckets error: "
        << outcome.GetError().GetExceptionName() << std::endl
        << outcome.GetError().GetMessage() << std::endl;

        return false;
    }
}

// Create a bucket in this AWS Region.
bool CreateMyBucket(Aws::S3::S3Client s3_client, Aws::String bucket_name,
        Aws::S3::Model::BucketLocationConstraint region) {
    std::cout << "Creating a new bucket named "
    << bucket_name
    << "..." << std::endl
    << "...

    Aws::S3::Model::CreateBucketConfiguration bucket_configuration;
    bucket_configuration.WithLocationConstraint(region);

    Aws::S3::Model::CreateBucketRequest bucket_request;
    bucket_request.WithBucket(bucket_name).WithCreateBucketConfiguration(bucket_configuration);

    auto outcome = s3_client.CreateBucket(bucket_request);

    if (outcome.IsSuccess()) {
        return true;
    } else {
        std::cout << "CreateBucket error: "
        << outcome.GetError().GetExceptionName() << std::endl
        << outcome.GetError().GetMessage() << std::endl;

        return false;
    }
}
// Delete the bucket you just created.
bool DeleteMyBucket(Aws::S3::S3Client s3_client, Aws::String bucket_name) {
    std::cout << "Deleting the bucket named "
              << bucket_name
              << "...
              " << std::endl << std::endl;

    Aws::S3::Model::DeleteBucketRequest bucket_request;
    bucket_request.WithBucket(bucket_name);

    auto outcome = s3_client.DeleteBucket(bucket_request);

    if (outcome.IsSuccess()) {
        return true;
    } else {
        std::cout << "DeleteBucket error: "
                  << outcome.GetError().GetExceptionName() << std::endl
                  << outcome.GetError().GetMessage() << std::endl;

        return false;
    }
}

void Cleanup(Aws::SDKOptions options) {
    Aws::ShutdownAPI(options);
}

2. Create a file with this content, and save the file with the name CMakeLists.txt at the root (/) of your environment. This file enables you to build your code into an executable file.

```cmake
# A minimal CMakeLists.txt file for the AWS SDK for C++.

# The minimum version of CMake that will work.
cmake_minimum_required(VERSION 2.8)

# The project name.
project(s3-demo)

# Locate the AWS SDK for C++ package.
# Requires that you build with:
#   -Daws-sdk-cpp_DIR=/path/to/sdk_build
# or export/set:
#   CMAKE_PREFIX_PATH=/path/to/sdk_build
find_package(aws-sdk-cpp)

# Link to the AWS SDK for C++ shared libraries.
add_definitions(-DUSE_IMPORT_EXPORT)

# The executable name and its source files.
add_executable(s3-demo s3-demo.cpp)

# The libraries used by your executable.
target_link_libraries(s3-demo aws-cpp-sdk-s3)
```

---

**Step 6: Build and Run the AWS SDK Code**

1. In the terminal, prepare to build your source code.
For Amazon Linux:

```shell
cmake3 -Daws-sdk-cpp_DIR=sdk_build .
```

For Ubuntu Server:

```shell
cmake -Daws-sdk-cpp_DIR=sdk_build .
```

2. Build your source code.

```shell
make
```

3. Run the code by choosing Run, Run Configurations, New Run Configuration on the menu bar.


5. For Command, type `s3-demo my-test-bucket us-east-2`, where `my-test-bucket` is the name of the bucket you want to create and then delete, and `us-east-2` is the ID of the AWS Region you want to create the bucket in. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.

   **Note**
   
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.

6. Choose the Run button, and compare your output.

   My buckets now are:
   
   Creating a new bucket named 'my-test-bucket'...
   
   My buckets now are:
   
   my-test-bucket
   
   Deleting the bucket named 'my-test-bucket'...
   
   My buckets now are:

---

**Step 7: Clean Up**

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

---

**Python Tutorial for AWS Cloud9**

This tutorial shows you how to run Python code in an AWS Cloud9 development environment.

Following this tutorial might result in charges to your AWS account. These include possible charges for services such as Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3). For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

**Topics**

- Prerequisites (p. 419)
- Step 1: Install Python 3.6 (p. 419)
- Step 2: Add Code (p. 420)
AWS Cloud9 User Guide
Prerequisites

- Step 3: Run the Code (p. 420)
- Step 4: Install and Configure the AWS SDK for Python (Boto 3) (p. 420)
- Step 5: Add AWS SDK Code (p. 421)
- Step 6: Run the AWS SDK Code (p. 422)
- Step 7: Clean Up (p. 423)

Prerequisites

Before you use this tutorial, be sure to meet the following requirements.

- **You have an AWS Cloud9 EC2 development environment**
  
  This tutorial assumes that you have an EC2 environment, and that the environment is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. See Creating an EC2 Environment (p. 56) for details.

  If you have a different type of environment or operating system, you might need to adapt this tutorial's instructions.

- **You have opened the AWS Cloud9 IDE for that environment**
  
  When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install Python 3.6

1. In a terminal session in the AWS Cloud9 IDE, confirm whether Python 3.6 is already installed by running the `python --version` command. (To start a new terminal session, on the menu bar choose Window, New Terminal.) If Python 3.6 is installed, skip ahead to Step 2: Add Code (p. 420).

2. Run the `yum update` (for Amazon Linux) or `apt update` (for Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

   For Amazon Linux:
   
   ```
   sudo yum -y update
   ```

   For Ubuntu Server:
   
   ```
   sudo apt update
   ```

3. Install Python 3.6 by running the `install` command.

   For Amazon Linux:
   
   ```
   sudo yum -y install python36
   ```

   For Ubuntu Server:
   
   ```
   sudo apt -y install python3
   ```
Step 2: Add Code

In the AWS Cloud9 IDE, create a file with the following content and save the file with the name hello.py. (To create a file, on the menu bar choose File, New File. To save the file, choose File, Save.)

```python
import sys

print('Hello, World!')
print('The sum of 2 and 3 is 5.')
sum = int(sys.argv[1]) + int(sys.argv[2])
print('The sum of {0} and {1} is {2}.'.format(sys.argv[1], sys.argv[2], sum))
```

Step 3: Run the Code

1. In the AWS Cloud9 IDE, on the menu bar choose Run, Run Configurations, New Run Configuration.
2. On the [New] - Stopped tab, enter hello.py 5 9 for Command. In the code, 5 represents sys.argv[1], and 9 represents sys.argv[2].
3. Choose Run and compare your output.
   
   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.

4. By default, AWS Cloud9 automatically selects a runner for your code. To change the runner, choose Runner, and then choose Python 2 or Python 3.
   
   Note
   You can create custom runners for specific versions of Python. For details, see Create a Builder or Runner (p. 151).

Step 4: Install and Configure the AWS SDK for Python (Boto 3)

The AWS SDK for Python (Boto 3) enables you to use Python code to interact with AWS services like Amazon S3. For example, you can use the SDK to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

Install pip

In the AWS Cloud9 IDE, confirm whether pip is already installed for the active version of Python by running the `python -m pip --version` command. If pip is installed, skip to the next section.

To install pip, run the following commands. Because sudo is in a different environment from your user, you must specify the version of Python to use if it differs from the current aliased version.

```
curl -O https://bootstrap.pypa.io/get-pip.py # Get the install script.
sudo python36 get-pip.py # Install pip for Python 3.6.
python -m pip --version # Verify pip is installed.
rm get-pip.py # Delete the install script.
```

For more information, see Installation on the pip website.
Install the AWS SDK for Python (Boto 3)

After you install pip, install the AWS SDK for Python (Boto 3) by running the `pip install` command.

```
sudo python36 -m pip install boto3  # Install boto3 for Python 3.6.
python -m pip show boto3            # Verify boto3 is installed for the current version of Python.
```

For more information, see the "Installation" section of Quickstart in the AWS SDK for Python (Boto 3).

Set Up Credentials in Your Environment

Each time you use the AWS SDK for Python (Boto 3) to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the SDK has the necessary permissions to make the call. If the credentials don't cover the necessary permissions, the call fails.

To store your credentials within the environment, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Credentials in the AWS SDK for Python (Boto 3).

Step 5: Add AWS SDK Code

Add code that uses Amazon S3 to create a bucket, list your available buckets, and optionally delete the bucket you just created.

In the AWS Cloud9 IDE, create a file with the following content and save the file with the name `s3.py`.

```python
import sys
import boto3
from botocore.exceptions import ClientError

def get_s3(region=None):
    """
    Get a Boto 3 Amazon S3 resource with a specific AWS Region or with your default AWS Region.
    """
    return boto3.resource('s3', region_name=region) if region else boto3.resource('s3')

def list_my_buckets(s3):
    print('Buckets:
	', *[b.name for b in s3.buckets.all()], sep="\n\t")

def create_and_delete_my_bucket(bucket_name, region, keep_bucket):
    s3 = get_s3(region)
    list_my_buckets(s3)
    try:
        print('Creating new bucket:', bucket_name)
        bucket = s3.create_bucket(
            Bucket=bucket_name,
            CreateBucketConfiguration={
                'LocationConstraint': region
            }
        )
    except ClientError as e:
```

```
print(e)
sys.exit('Exiting the script because bucket creation failed. ')

bucket.wait_until_exists()
list_my_buckets(s3)

if not keep_bucket:
    print('
Deleting bucket:', bucket.name)
    bucket.delete()

    bucket.wait_until_not_exists()
    list_my_buckets(s3)
else:
    print('
Keeping bucket:', bucket.name)

def main():
    import argparse

    parser = argparse.ArgumentParser()
paper.add_argument('bucket_name', help='The name of the bucket to create.')
parser.add_argument('region', help='The region in which to create your bucket.')
parser.add_argument('--keep_bucket', help='Keeps the created bucket. When not ' +
'specified, the bucket is deleted ' +
'at the end of the demo.',
    action='store_true')

    args = parser.parse_args()

    create_and_delete_my_bucket(args.bucket_name, args.region, args.keep_bucket)

if __name__ == '__main__':
    main()

Step 6: Run the AWS SDK Code

1. On the menu bar, choose Run, Run Configurations, New Run Configuration.
2. For Command, enter s3.py my-test-bucket us-west-2, where my-test-bucket is the name of the bucket to create, and us-west-2 is the ID of the AWS Region where your bucket is created. By default, your bucket is deleted before the script exits. To keep your bucket, add --keep_bucket to your command. For a list of AWS Region IDs, see Amazon Simple Storage Service Endpoints and Quotas in the AWS General Reference.

   Note
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.
3. Choose Run, and compare your output.
Step 7: Clean Up

To prevent ongoing charges to your AWS account after you're done with this tutorial, delete the AWS Cloud9 environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

.NET Core Sample for AWS Cloud9

This sample enables you to run some .NET Core code in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

Topics
- Prerequisites (p. 423)
- Step 1: Install Required Tools (p. 423)
- Step 2: Create a .NET Core Console Application Project (p. 425)
- Step 3: Add Code (p. 426)
- Step 4: Build and Run the Code (p. 426)
- Step 5: Create and Set Up a .NET Core Console Application Project That Uses the AWS SDK for .NET (p. 427)
- Step 6: Add AWS SDK Code (p. 428)
- Step 7: Build and Run the AWS SDK Code (p. 430)
- Step 8: Clean Up (p. 430)

Prerequisites

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install Required Tools

In this step, you install the .NET Core 2 SDK into your environment, which is required to run this sample.

1. Confirm whether the .NET Core 2 SDK is already installed in your environment. To do this, in a terminal session in the AWS Cloud9 IDE, run the .NET Core command line interface (CLI) with the --version option.
Step 1: Install Required Tools

1. **Install Required Tools**

   dotnet --version

   If the .NET Command Line Tools version is displayed, and the version is 2.0 or greater, skip ahead to Step 2: Create a .NET Core Console Application Project (p. 425). If the version is less than 2.0, or if an error such as `bash: dotnet: command not found` is displayed, continue on to install the .NET Core 2 SDK.

2. **For Amazon Linux**, in a terminal session in the AWS Cloud9 IDE, run the following commands to help ensure the latest security updates and bug fixes are installed, and to install a `libunwind` package that the .NET Core 2 SDK needs. (To start a new terminal session, on the menu bar, choose **Window, New Terminal**.)

   ```
   sudo yum -y update
   sudo yum -y install libunwind
   ```

   For **Ubuntu Server**, in a terminal session in the AWS Cloud9 IDE, run the following command to help ensure the latest security updates and bug fixes are installed. (To start a new terminal session, on the menu bar, choose **Window, New Terminal**.)

   ```
   sudo apt -y update
   ```

3. **Download the .NET Core 2 SDK installer script into your environment by running the following command.**

   ```
   curl -O https://dot.net/v1/dotnet-install.sh
   ```

4. **Make the installer script executable by the current user by running the following command.**

   ```
   sudo chmod u=rx dotnet-install.sh
   ```

5. **Run the installer script, which downloads and installs the .NET Core 2 SDK, by running the following command.**

   ```
   ./dotnet-install.sh -c Current
   ```

6. **Add the .NET Core 2 SDK to your PATH.** To do this, in the shell profile for the environment (for example, the `.bashrc` file), add the `$HOME/.dotnet` subdirectory to the `PATH` variable for the environment, as follows.

   a. Open the `.bashrc` file for editing by using the `vi` command.

   ```
   vi ~/.bashrc
   ```

   b. For Amazon Linux, using the down arrow or `j` key, move to the line that starts with `export PATH`. For Ubuntu Server, move to the last line of the file by typing `G`.

   c. Using the right arrow or `#` key, move to the end of that line.

   d. Switch to insert mode by pressing the `i` key. (`-- INSERT ----` will appear at the end of the display.)

   e. For Amazon Linux, add the `$HOME/.dotnet` subdirectory to the `PATH` variable by typing `:`. Be sure to include the colon character (`:`). The line should now look similar to the following.
Step 2: Create a .NET Core Console Application Project

In this step, you use .NET Core to create a project named hello. This project contains all of the files that .NET Core needs to run a simple application from the terminal in the IDE. The application's code is written in C#.

1. In the terminal, run the following commands to create a directory for the project, and then switch to that new directory.

   ```
   mkdir hello
cd hello
   ```

2. Create a .NET Core console application project. To do this, run the .NET Core CLI with the `new` command, specifying the console application project template type and the programming language to use (in this sample, C#).

   ```
   dotnet new console -lang C#
   ```

   The preceding command adds a subdirectory named `obj` with several files, and some additional standalone files, to the `hello` directory. You should note the following two key files:

   - The `hello/hello.csproj` file contains information about the console application project.
   - The `hello/Program.cs` file contains the application's code to run.

   For Ubuntu Server, press the right arrow key and then press Enter twice, followed by typing the following line by itself at the end of the file.

   ```
   export PATH=$HOME/.dotnet:$PATH
   ```

   f. Save the file. To do this, press the `Esc` key (`-- INSERT ---` will disappear from the end of the display), type `:wq` (to write to and then quit the file), and then press Enter.

   7. Load the .NET Core 2 SDK by sourcing the `.bashrc` file.

   ```
   . ~/.bashrc
   ```

   8. Confirm the .NET Core 2 SDK is loaded by running .NET Core CLI with the `--help` option.

   ```
   dotnet --help
   ```

   If successful, the .NET Core 2 SDK version number is displayed, with additional usage information.

   9. If you no longer want to keep the .NET Core 2 SDK installer script in your environment, you can delete it as follows.

   ```
   rm dotnet-install.sh
   ```
Step 3: Add Code

In this step, you add some code to the application.

From the Environment window in the AWS Cloud9 IDE, open the hello/Program.cs file.

In the editor, replace the file's current contents with the following code, and then save the Program.cs file.

```csharp
using System;
namespace hello
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello, World!");
            Console.WriteLine("The sum of 2 and 3 is 5.");
            int sum = Int32.Parse(args[0]) + Int32.Parse(args[1]);
            Console.WriteLine("The sum of {0} and {1} is {2}.", args[0], args[1], sum.ToString());
        }
    }
}
```

Step 4: Build and Run the Code

In this step, you build the project and its dependencies into a set of binary files, including a runnable application file. Then you run the application.

1. In the IDE, create a builder for .NET Core as follows.
   a. On the menu bar, choose Run, Build System, New Build System.
   b. On the My Builder.build tab, replace the tab's contents with the following code.

   ```json
   {
       "cmd" : ["dotnet", "build"],
       "info" : "Building..."
   }
   ``

   c. Choose File, Save As.
   d. For Filename, type .NET Core.build.
   e. For Folder, type /.c9/builders.
   f. Choose Save.

   2. With the contents of the Program.cs file displayed in the editor, choose Run, Build System, .NET Core. Then choose Run, Build.

   This builder adds a subdirectory named bin and adds a subdirectory named Debug to the hello/obj subdirectory. Note the following three key files.
   - The hello/bin/Debug/netcoreapp2.0/hello.dll file is the runnable application file.
   - The hello/bin/Debug/netcoreapp2.0/hello.deps.json file lists the application's dependencies.
Step 5: Create and Set Up a .NET Core Console Application Project That Uses the AWS SDK for .NET

You can enhance this sample to use the AWS SDK for .NET to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this new project, you add a reference to the AWS SDK for .NET. The AWS SDK for .NET provides a convenient way to interact with AWS services such as Amazon S3, from your .NET code. You then set up AWS credentials management in your environment. The AWS SDK for .NET needs these credentials to interact with AWS services.

To create the project

1. In the terminal, run the following commands to change to the root directory of the environment, create a directory for a project named s3, and then switch to that new directory.

```
  cd..
mkdir s3
```
Step 6: Add AWS SDK Code

In this step, you add code to interact with Amazon S3 to create a bucket, delete the bucket you just created, and then list your available buckets.

From the Environment window in the AWS Cloud9 IDE, open the s3/Program.cs file. In the editor, replace the file's current contents with the following code, and then save the Program.cs file.

```csharp
using Amazon;
using Amazon.S3;
using Amazon.S3.Model;
using Amazon.S3.Util;
using System;
using System.Threading.Tasks;

namespace s3
{
    class Program
    {
        private static RegionEndpoint bucketRegion;
        private static IAmazonS3 s3Client;

        static void Main(string[] args)
        {
            if (args.Length < 2) {
```
Step 6: Add AWS SDK Code

```csharp
Console.Write("Usage: <the bucket name> <the AWS Region to use>\n" +
    "Example: my-test-bucket us-east-2\n");
return;
}

if (args[1] == "us-east-2") {
    bucketRegion = RegionEndpoint.USEast2;
} else {
    Console.WriteLine("Cannot continue. The only supported AWS Region ID is " +
        "'us-east-2'.");
    return;
}
// Note: You could add more valid AWS Regions above as needed.

s3Client = new AmazonS3Client(bucketRegion);
var bucketName = args[0];

// Create the bucket.
try {
    if (!DoesBucketExist(bucketName))
    {
        Console.WriteLine("Cannot continue. Cannot create bucket. \n" +
            "A bucket named '{0}' already exists.", bucketName);
        return;
    } else {
        Console.WriteLine("\nCreating the bucket named '{0}'...", bucketName);
        s3Client.PutBucketAsync(bucketName).Wait();
    }
} catch (AmazonS3Exception e)
{
    Console.WriteLine("Cannot continue. {0}", e.Message);
} catch (Exception e)
{
    Console.WriteLine("Cannot continue. {0}", e.Message);
}

// Confirm that the bucket was created.
if (!DoesBucketExist(bucketName))
{
    Console.WriteLine("Created the bucket named '{0}'.", bucketName);
} else {
    Console.WriteLine("Did not create the bucket named '{0}'.", bucketName);
}

// Delete the bucket.
Console.WriteLine("\nDeleting the bucket named '{0}'...", bucketName);
s3Client.DeleteBucketAsync(bucketName).Wait();

// Confirm that the bucket was deleted.
if (DoesBucketExist(bucketName))
{
    Console.WriteLine("Did not delete the bucket named '{0}'.", bucketName);
} else {
    Console.WriteLine("Deleted the bucket named '{0}'.", bucketName);
};

// List current buckets.
Console.WriteLine("\nMy buckets now are:");
var response = s3Client.ListBucketsAsync().Result;
foreach (var bucket in response.Buckets)
{
    Console.WriteLine(bucket.BucketName);
}
```
Step 7: Build and Run the AWS SDK Code

In this step, you build the project and its dependencies into a set of binary files, including a runnable
application file. Then you run the application.

1. Build the project. To do this, with the contents of the s3/Program.cs file displayed in the editor,
on the menu bar, choose Run, Build.

2. Run the application with the name of the Amazon S3 bucket to create and the ID of the AWS Region
to create the bucket in (for example, my-test-bucket and us-east-2) as follows.

   a. With the contents of the s3/Program.cs file still displayed in the editor, choose Run, Run
      Configurations, New Run Configuration.
   b. In the [New] - Idle tab, choose Runner: Auto, and then choose .NET Core.
   c. In the Command box, type the name of the application, the name of the Amazon S3 bucket
to create, and the ID of the AWS Region to create the bucket in (for example, s3 my-test-
   bucket us-east-2).
   d. Choose Run.

      By default, this runner instructs .NET Core to run the s3.dll file in the s3/bin/Debug/
      netcoreapp2.0 directory.

      Compare your results to the following output.

      Creating a new bucket named 'my-test-bucket'...
      Created the bucket named 'my-test-bucket'.

      Deleting the bucket named 'my-test-bucket'...
      Deleted the bucket named 'my-test-bucket'.

      My buckets now are:

Step 8: Clean Up

To prevent ongoing charges to your AWS account after you’re done using this sample, you should delete
the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

Node.js Sample for AWS Cloud9

This sample enables you to run some Node.js scripts in an AWS Cloud9 development environment.
Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

Topics
- Prerequisites (p. 431)
- Step 1: Install Required Tools (p. 431)
- Step 2: Add Code (p. 432)
- Step 3: Run the Code (p. 432)
- Step 4: Install and Configure the AWS SDK for JavaScript in Node.js (p. 433)
- Step 5: Add AWS SDK Code (p. 433)
- Step 6: Run the AWS SDK Code (p. 435)
- Step 7: Clean Up (p. 435)

Prerequisites

Before you use this sample, be sure to meet the following requirements.

- You must have an existing AWS Cloud9 EC2 development environment. This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.
- You have the AWS Cloud9 IDE for the existing environment already open. When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install Required Tools

In this step, you install Node.js, which is required to run this sample.

1. In a terminal session in the AWS Cloud9 IDE, confirm whether Node.js is already installed by running the `node --version` command. (To start a new terminal session, on the menu bar, choose Window, New Terminal.) If successful, the output contains the Node.js version number. If Node.js is installed, skip ahead to Step 2: Add Code (p. 432).

2. Run the `yum update` for (Amazon Linux) or `apt update` for (Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

For Amazon Linux:

```
sudo yum -y update
```

For Ubuntu Server:

```
sudo apt update
```

3. To install Node.js, begin by running this command to download Node Version Manager (nvm). (nvm is a simple Bash shell script that is useful for installing and managing Node.js versions. For more information, see Node Version Manager on the GitHub website.)

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.33.0/install.sh | bash
```
4. To start using nvm, either close the terminal session and start it again, or source the `.bashrc` file that contains the commands to load nvm.

   `. ~/.bashrc`

5. Run this command to install the latest version of Node.js.

   `nvm install node`

---

**Step 2: Add Code**

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `hello.js`. (To create a file, on the menu bar, choose **File, New File**. To save the file, choose **File, Save**.)

```javascript
console.log('Hello, World!');

console.log('The sum of 2 and 3 is 5.');

var sum = parseInt(process.argv[2], 10) + parseInt(process.argv[3], 10);

```

---

**Step 3: Run the Code**

1. In the AWS Cloud9 IDE, on the menu bar, choose **Run, Run Configurations, New Run Configuration**.
2. On the **[New] - Idle** tab, choose **Runner: Auto**, and then choose **Node.js**.
3. For **Command**, type `hello.js 5 9`. In the code, 5 represents `process.argv[2]`, and 9 represents `process.argv[3]`. `process.argv[0]` represents the name of the runtime (node), and `process.argv[1]` represents the name of the file (`hello.js`).
4. Choose the **Run** button, and compare your output.

   ```text
   Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
   ```
Step 4: Install and Configure the AWS SDK for JavaScript in Node.js

You can enhance this sample to use the AWS SDK for JavaScript in Node.js to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install and configure the AWS SDK for JavaScript in Node.js, which provides a convenient way to interact with AWS services such as Amazon S3, from your JavaScript code. After you install the AWS SDK for JavaScript in Node.js, you must set up credentials management in your environment. The AWS SDK for JavaScript in Node.js needs these credentials to interact with AWS services.

To install the AWS SDK for JavaScript in Node.js

Use npm to run the `install` command.

```
npm install aws-sdk
```

For more information, see Installing the SDK for JavaScript in the AWS SDK for JavaScript Developer Guide.

To set up credentials management in your environment

Each time you use the AWS SDK for JavaScript in Node.js to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for JavaScript in Node.js has the appropriate permissions to make that call. If the credentials do not cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Setting Credentials in Node.js in the AWS SDK for JavaScript Developer Guide.

Step 5: Add AWS SDK Code

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You will run this code later.

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `s3.js`.

```javascript
if (process.argv.length < 4) {
    console.log('Usage: node s3.js <the bucket name> <the AWS Region to use>
        Example: node s3.js my-test-bucket us-east-2');
    process.exit(1);
}

var AWS = require('aws-sdk'); // To set the AWS credentials and region.
var async = require('async'); // To call AWS operations asynchronously.

AWS.config.update({
    region: region
});

var s3 = new AWS.S3({apiVersion: '2006-03-01'});
var bucket_name = process.argv[2];
var region = process.argv[3];
```
var create_bucket_params = {
    Bucket: bucket_name,
    CreateBucketConfiguration: {
        LocationConstraint: region
    }
};

var delete_bucket_params = {Bucket: bucket_name};

// List all of your available buckets in this AWS Region.
function listMyBuckets(callback) {
    s3.listBuckets(function(err, data) {
        if (err) {
            } else {
                console.log("My buckets now are:
");  
                for (var i = 0; i < data.Buckets.length; i++) {
                    console.log(data.Buckets[i].Name);  
                }  
                callback(err);  
        });  
    });

// Create a bucket in this AWS Region.
function createMyBucket(callback) {
    console.log('Creating a bucket named ' + bucket_name + '...
');
    s3.createBucket(create_bucket_params, function(err, data) {
        if (err) {
            console.log(err.code + ': ' + err.message);
        }  
        callback(err);  
    });

// Delete the bucket you just created.
function deleteMyBucket(callback) {
    console.log('Deleting the bucket named ' + bucket_name + '...
');
    s3.deleteBucket(delete_bucket_params, function(err, data) {
        if (err) {
            console.log(err.code + ': ' + err.message);
        }  
        callback(err);  
    });

// Call the AWS operations in the following order.
async.series([listMyBuckets, createMyBucket, listMyBuckets, deleteMyBucket, listMyBuckets]);
**Step 6: Run the AWS SDK Code**

1. Enable the code to call Amazon S3 operations asynchronously by using npm to run the `install` command.

   ```
   npm install async
   ```

2. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.


4. For Command, type `s3.js my-test-bucket us-east-2`, where `my-test-bucket` is the name of the bucket you want to create and then delete, and `us-east-2` is the ID of the AWS Region you want to create the bucket in. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.

   **Note**
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.

5. Choose the Run button, and compare your output.

   ```
   My buckets now are:
   Creating a new bucket named 'my-test-bucket'...
   My buckets now are:
   my-test-bucket
   Deleting the bucket named 'my-test-bucket'...
   My buckets now are:
   ```

**Step 7: Clean Up**

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

**PHP Sample for AWS Cloud9**

This sample enables you to run some PHP scripts in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

**Topics**

- Prerequisites (p. 436)
- Step 1: Install Required Tools (p. 436)
- Step 2: Add Code (p. 437)
- Step 3: Run the Code (p. 437)
- Step 4: Install and Configure the AWS SDK for PHP (p. 438)
- Step 5: Add AWS SDK Code (p. 439)
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See *Creating an Environment in AWS Cloud9* (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See *Opening an Environment in AWS Cloud9* (p. 67) for details.

### Step 1: Install Required Tools

In this step, you install PHP, which is required to run this sample.

**Note**
The following procedure installs PHP only. To install related tools such as an Apache web server and a MySQL database, see *Tutorial: Installing a LAMP Web Server on Amazon Linux* in the *Amazon EC2 User Guide for Linux Instances*.

1. In a terminal session in the AWS Cloud9 IDE, confirm whether PHP is already installed by running the `php --version` command. (To start a new terminal session, on the menu bar, choose *Window, New Terminal.*) If successful, the output contains the PHP version number. If PHP is installed, skip ahead to Step 2: Add Code (p. 437).

2. Run the `yum update` for (Amazon Linux) or `apt update` for (Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

   For Amazon Linux:

   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:

   ```bash
   sudo apt update
   ```

3. Install PHP by running the `install` command.

   For Amazon Linux:

   ```bash
   sudo yum -y install php56
   ```

   For Ubuntu Server:

   ```bash
   sudo apt install -y php php-xml
   ```

   For more information, see *Installation and Configuration* on the PHP website.
Step 2: Add Code

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name hello.php. (To create a file, on the menu bar, choose File, New File. To save the file, choose File, Save, type hello.php for Filename, and then choose Save.)

```php
<?php
   print('Hello, World!');

   print("\nThe sum of 2 and 3 is 5.");
   $sum = (int)$argv[1] + (int)$argv[2];
   print("\nThe sum of $argv[1] and $argv[2] is $sum.");
?>
```

**Note**
The preceding code doesn't rely on any external files. However, if you ever include or require other PHP files in your file, and you want AWS Cloud9 to use those files to do code completion as you type, turn on the Project, PHP Support, Enable PHP code completion setting in Preferences, and then add the paths to those files to the Project, PHP Support, PHP Completion Include Paths setting. (To view and change your preferences, choose AWS Cloud9, Preferences on the menu bar.)

Step 3: Run the Code

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
3. For Command, type `hello.php 5 9`. In the code, 5 represents $argv[1], and 9 represents $argv[2]. ($argv[0] represents the name of the file (hello.php).)
4. Choose the Run button, and compare your output.

Hello, World!
The sum of 2 and 3 is 5.
The sum of 5 and 9 is 14.
Step 4: Install and Configure the AWS SDK for PHP

You can enhance this sample to use the AWS SDK for PHP to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install and configure the AWS SDK for PHP, which provides a convenient way to interact with AWS services such as Amazon S3, from your PHP code. Before you can install the AWS SDK for PHP, you should install Composer. After you install the AWS SDK for PHP, you must set up credentials management in your environment. The AWS SDK for PHP needs these credentials to interact with AWS services.

To install Composer

Run the `curl` command with the silent (-s) and show error (-S) options, piping the Composer installer into a PHP archive (PHAR) file, named `composer.phar` by convention.

```
curl -sS https://getcomposer.org/installer | php
```

To install the AWS SDK for PHP

For Ubuntu Server, install additional packages that Composer needs to install the AWS SDK for PHP.

```
sudo apt install -y php-xml php-curl
```

For Amazon Linux or Ubuntu Server, use the `php` command to run the Composer installer to install the AWS SDK for PHP.

```
php composer.phar require aws/aws-sdk-php
```

This command creates several folders and files in your environment. The primary file you will use is `autoload.php`, which is in the `vendor` folder in your environment.

**Note**

After installation, Composer might suggest that you install additional dependencies. You can do this with a command such as the following, specifying the list of dependencies to install. For example, the following command instructs Composer to install the following list of dependencies.

```
php composer.phar require psr/log ext-curl doctrine/cache aws/aws-php-sns-message-validator
```

For more information, see Installation in the AWS SDK for PHP Developer Guide.

To set up credentials management in your environment

Each time you use the AWS SDK for PHP to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for PHP has the appropriate permissions to make that call. If the credentials don’t cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see the "Creating a client" section of Basic Usage in the AWS SDK for PHP Developer Guide.
**Step 5: Add AWS SDK Code**

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You will run this code later.

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `s3.php`.

```php
<?php
require './vendor/autoload.php';

if ($argc < 4) {
    exit("Usage: php s3.php <the time zone> <the bucket name> <the AWS Region to use>\n" .
    "Example: php s3.php America/Los_Angeles my-test-bucket us-east-2");
}

$timeZone = $argv[1];
$bucketName = $argv[2];
$region = $argv[3];

date_default_timezone_set($timeZone);

$s3 = new Aws\S3\S3Client(
    ['region' => $region,
    'version' => '2006-03-01'
]);

# Lists all of your available buckets in this AWS Region.
function listMyBuckets($s3) {
    print("My buckets now are:\n");

    $promise = $s3->listBucketsAsync();
    $result = $promise->wait();

    foreach ($result['Buckets'] as $bucket) {
        print("\n");
        print($bucket['Name']);
    }
}

listMyBuckets($s3);

# Create a new bucket.
print("Creating a new bucket named '$bucketName'...");

try {
    $promise = $s3->createBucketAsync(
        ['Bucket' => $bucketName,
        'CreateBucketConfiguration' => [
            'LocationConstraint' => $region
        ]
    );
    $promise->wait();
} catch (Exception $e) {
    if ($e->getCode() == 'BucketAlreadyExists') {
        exit("A bucket with the name '$bucketName' already exists. Exiting.");
    }
}

listMyBuckets($s3);
```
Step 6: Run the AWS SDK Code

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
3. For Command, type `s3.php America/Los_Angeles my-test-bucket us-east-2`, where:
   - `America/Los_Angeles` is your default time zone ID. For more IDs, see List of Supported Timezones on the PHP website.
   - `my-test-bucket` is the name of the bucket you want to create and then delete.
   
   **Note**
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.
   - `us-east-2` is the ID of the AWS Region you want to create the bucket in. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.
4. Choose the Run button, and compare your output.

```
My buckets now are:
Creating a new bucket named 'my-test-bucket'...
My buckets now are:
my-test-bucket
Deleting the bucket named 'my-test-bucket'...
My buckets now are:
```

Step 7: Clean Up

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

Tutorial: Ruby in AWS Cloud9

This tutorial shows you how to run Ruby scripts in an AWS Cloud9 development environment.

**Note**
Following this tutorial might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.
Prerequisites

Before you use this tutorial, be sure to meet the following requirements.

- **You have an AWS Cloud9 EC2 development environment**

  This tutorial assumes that you have an EC2 environment, and that the environment is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. See Creating an EC2 Environment (p. 56) for details.

  If you have a different type of environment or operating system, you might need to adapt this tutorial's instructions.

- **You have opened the AWS Cloud9 IDE for that environment**

  When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Steps

- **Step 1: Install Required Tools**

  (First step of Tutorial: Ruby in AWS Cloud9 (p. 440))

  In this step, you install Ruby, which is required to run this tutorial.

  1. In a terminal session in the AWS Cloud9 IDE, confirm whether Ruby is already installed by running the `ruby --version` command. (To start a new terminal session, on the menu bar, choose Window, New Terminal.) If successful, the output contains the Ruby version number. If Ruby is installed, skip ahead to Step 2: Add Code (p. 442).

  2. Run the `yum update` for (Amazon Linux) or `apt update` for (Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

  For Amazon Linux:

  ```bash
  sudo yum -y update
  ```

  For Ubuntu Server:

  ```bash
  sudo apt update
  ```

  3. Install Ruby by running the `install` command.

     For Amazon Linux:

     ```bash
     ```
sudo yum -y install ruby

For Ubuntu Server:

sudo apt install -y ruby

For more information, see Installing Ruby on the Ruby website.

Next Step

Step 2: Add Code

Step 2: Add Code

(Previous step: Step 1: Install Required Tools (p. 441))

1. In the AWS Cloud9 IDE, create a new file (File, New File on the menu bar).
2. Add the following code.

   ```ruby
   puts "Hello, World!"
   puts "The sum of 2 and 3 is 5."
   argv0 = ARGV[0]
   argv1 = ARGV[1]
   sum = argv0.to_i + argv1.to_i
   puts "The sum of #{argv0} and #{argv1} is #{sum}.
   ```

3. Save the file with the name hello.rb (File, Save).

Next Step

Step 3: Run the Code

Step 3: Run the Code

(Previous step: Step 2: Add Code (p. 442))

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
2. On the [New] - Idle tab, choose Runner: Auto, and then choose Ruby (item 1 in the screenshot shown below).
3. For Command (item 2 in the screenshot), type hello.rb 5 9. Given this command, ARGV[0] in the code receives a value of 5 and ARGV[1] receives a value of 9.
4. Choose the Run button (item 3 below), and compare your output to the following.

   ```text
   Hello, World!
   The sum of 2 and 3 is 5.
   The sum of 5 and 9 is 14.
   ```
Step 4: Install and Configure the AWS SDK for Ruby

(Previous step: Step 3: Run the Code (p. 442))

Note
If you don’t want to perform these enhanced procedures, be sure to at least perform Step 7: Clean Up (p. 446).

You can enhance this tutorial to use the AWS SDK for Ruby to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install and configure the AWS SDK for Ruby, which provides a convenient way to interact with AWS services such as Amazon S3 from your Ruby code.

• Before you can install the AWS SDK for Ruby, you must install RubyGems.
• After you install the AWS SDK for Ruby, you must set up credentials management in your environment. The AWS SDK for Ruby needs these credentials to interact with AWS services.

4.1 Install RubyGems

1. In a terminal session in the AWS Cloud9 IDE, confirm whether RubyGems is already installed by running the `gem --version` command. If successful, the output contains the RubyGems version number. Otherwise, an error message is displayed.

   If RubyGems is installed, skip ahead to 4.2: Install the AWS SDK for Ruby (p. 444).

2. To install RubyGems, run the `install` command as follows.

   For Amazon Linux:

   ```bash
   sudo yum -y install gem
   ```

   For Ubuntu Server:
sudo apt install -y gem

For more information, see Download RubyGems on the RubyGems website.

4.2: Install the AWS SDK for Ruby

After you install RubyGems, run the RubyGems `install` command in a terminal session as follows.

**Note**
The installation of the full aws-sdk package might run several minutes before it starts showing progress in the terminal window.

gem install aws-sdk

**Note**
If you are not using an EC2 instance that is managed by AWS Cloud9 (that is, an EC2 environment), depending on the permissions and user configuration on your instance, you might need to use `sudo` to install the SDK, as shown in the following command.

sudo gem install aws-sdk

If this is the case, use standard Unix-based practices.

For more information, see Installing the AWS SDK for Ruby in the AWS SDK for Ruby Developer Guide.

4.3: Set up credentials management in your environment

Each time you use the AWS SDK for Ruby to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for Ruby has the appropriate permissions to make that call. If the credentials don't cover the appropriate permissions, the call will fail.

If you are following this tutorial strictly, you will have allowed AWS Cloud9 to create and manage your EC2 instance. If this is the case for you, AWS Cloud9 is also managing temporary credentials for you, so you can skip this step.

If this is NOT the case for you, you need to store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Configuring the AWS SDK for Ruby in the AWS SDK for Ruby Developer Guide.

Next Step

Step 5: Add AWS SDK Code (p. 444)

Step 5: Add AWS SDK Code

(Previous step: Step 4: Install and Configure the AWS SDK for Ruby (p. 443))

In this step, you provide code to interact with Amazon S3. This code creates a bucket, lists your available buckets, and then deletes the bucket that was created. (You will run this code in the next step.)
In the AWS Cloud9 IDE, create a file with the following code and save the file with the name `s3.rb`.

```ruby
require 'aws-sdk'

if ARGV.length < 2
  puts "Usage: ruby s3.rb <the bucket name> <the AWS Region to use>
  Example: ruby s3.rb my-test-bucket us-east-2"
end

bucket_name = ARGV[0]
region = ARGV[1]
s3 = Aws::S3::Client.new(region: region)

# Lists all of your available buckets in this AWS Region.
def list_my_buckets(s3)
  resp = s3.list_buckets()
  puts "My buckets now are:
  
  \n"
  resp.buckets.each do |bucket|
    puts bucket.name
  end
end

list_my_buckets(s3)

# Create a new bucket.
begind
  puts "Creating a new bucket named '#{bucket_name}'...
  s3.create_bucket({
    bucket: bucket_name,
    create_bucket_configuration: {
      location_constraint: region
    }
  })

  s3.wait_until(:bucket_exists, {bucket: bucket_name,})
rescue Aws::S3::Errors::BucketAlreadyExists
  puts "Cannot create the bucket. A bucket with the name '#{bucket_name}' already exists. Exiting."
  exit(false)
end

list_my_buckets(s3)

# Delete the bucket you just created.
puts "Deleting the bucket named '#{bucket_name}'...
  s3.delete_bucket(bucket: bucket_name)
  s3.wait_until(:bucket_not_exists, {bucket: bucket_name,})
list_my_buckets(s3)
```

**Next Step**

**Step 6: Run the AWS SDK Code**

(Previous step: Step 5: Add AWS SDK Code)

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
Step 7: Clean Up

2. In the [New] - Idle tab, choose Runner: Auto, and then choose Ruby.

3. For Command, type `s3.rb YOUR_BUCKET_NAME THE_AWS_REGION`, where:
   - `YOUR_BUCKET_NAME` is the name of the bucket you want to create and then delete.
     Amazon S3 bucket names must be globally unique across AWS, not just within your AWS account.
   - `THE_AWS_REGION` is the ID of the AWS Region you want to create the bucket in.
     For example, for the US East (Ohio) Region, use `us-east-2`. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.

4. Choose the Run button. Your output should be similar to the following.

```
My buckets now are:
Creating a new bucket named 'my-test-bucket'...
My buckets now are:
my-test-bucket
Deleting the bucket named 'my-test-bucket'...
My buckets now are:
```

Next Step

Step 7: Clean Up (p. 446)

Step 7: Clean Up

(Previous step: Step 6: Run the AWS SDK Code (p. 445))

To prevent ongoing charges to your AWS account after you're done using this tutorial, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

(Back to Tutorial: Ruby (p. 440))

Go Sample for AWS Cloud9

This sample enables you to run some Go code in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

Topics

- Prerequisites (p. 447)
- Step 1: Install Required Tools (p. 447)
- Step 2: Add Code (p. 448)
- Step 3: Run the Code (p. 448)
- Step 4: Install and Configure the AWS SDK for Go (p. 449)
- Step 5: Add AWS SDK Code (p. 450)
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample's instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install Required Tools

In this step, you install and configure Go, which is required to run this sample.

1. In a terminal session in the AWS Cloud9 IDE, confirm whether Go is already installed by running the `go version` command. (To start a new terminal session, on the menu bar, choose Window, New Terminal.) If successful, the output should contain the Go version number. Otherwise, an error message should be output. If Go is installed, skip ahead to Step 2: Add Code (p. 448).

2. Run the `yum update` for (Amazon Linux) or `apt update` for (Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

   For Amazon Linux:
   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:
   ```bash
   sudo apt update
   ```

3. To install Go, run these commands, one at a time.

   ```bash
   wget https://storage.googleapis.com/golang/go1.9.3.linux-amd64.tar.gz # Download the Go installer.
   sudo tar -C /usr/local -xzf ./go1.9.3.linux-amd64.tar.gz              # Install Go.
   rm ./go1.9.3.linux-amd64.tar.gz                                       # Delete the installer.
   ```

   The preceding commands assume the latest stable version of Go at the time this topic was written. For more information, see Downloads on The Go Programming Language website.

4. Add the path to the Go binary to your `PATH` environment variable, like this.

   a. Open your shell profile file (for example, `.bashrc`) for editing.
   b. At the end of this line of code, type the following, so that the code now looks like this.

   ```bash
   PATH=$PATH:/usr/local/go/bin
   ```

   c. Save the file.
5. Source the ~/.bashrc file so that the terminal can now find the Go binary you just referenced.

```
. ~/.bashrc
```

6. Confirm that Go is now successfully installed and configured by running the `go version` command. If successful, the output contains the Go version number.

### Step 2: Add Code

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `hello.go`. (To create a file, on the menu bar, choose File, New File. To save the file, choose File, Save.)

```go
package main

import (
    "fmt"
    "os"
    "strconv"
)

func main() {
    fmt.Printf("Hello, World!\n")

    fmt.Printf("The sum of 2 and 3 is 5.\n")

    first, _ := strconv.Atoi(os.Args[1])
    second, _ := strconv.Atoi(os.Args[2])
    sum := first + second

    fmt.Printf("The sum of %s and %s is %s.\n", os.Args[1], os.Args[2], strconv.Itoa(sum))
}
```

### Step 3: Run the Code

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
2. On the [New] - Idle tab, choose Runner: Auto, and then choose Go.

   **Note**
   
   If Go is not available, you can create a custom runner for Go.

   2. On the My Runner.run tab, replace the tab's contents with this code.

   ```
   {
     "cmd": ["go", "run", "$file", "$args"],
     "info": "Running $project_path$file_name...",
     "selector": "source.go"
   }
   ```

   3. Choose File, Save As on the menu bar, and save the file as Go.run in the /.c9/runners folder.


   5. Choose the hello.go tab to make it active.

3. For Command, type `hello.go 5 9`. In the code, 5 represents os.Args[1], and 9 represents os.Args[2].
Step 4: Install and Configure the AWS SDK for Go

You can enhance this sample to use the AWS SDK for Go to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.

In this step, you install and configure the AWS SDK for Go, which provides a convenient way to interact with AWS services such as Amazon S3, from your Go code. Before you install the AWS SDK for Go, you must set your `GOPATH` environment variable. After you install the AWS SDK for Go and set your `GOPATH` environment variable, you must set up credentials management in your environment. The AWS SDK for Go needs these credentials to interact with AWS services.

**To set your `GOPATH` environment variable**

1. Open your `~/.bashrc` file for editing.
2. After the last line in the file, type this code.

   ```bash
   GOPATH=~/environment/go
   export GOPATH
   ```
3. Save the file.
4. Source the `~/.bashrc` file so that the terminal can now find the `GOPATH` environment variable you just referenced.
5. Confirm that the `GOPATH` environment variable is successfully set by running the `echo $GOPATH` command. If successful, `/home/ec2-user/environment/go` or `/home/ubuntu/environment/go` should be output.

### To install the AWS SDK for Go

Run the `go get` command, specifying the location of the AWS SDK for Go source.

```bash
go get -u github.com/aws/aws-sdk-go/...
```

Go installs the AWS SDK for Go source into the location specified by your `GOPATH` environment variable, which is the `go` folder in your environment.

### To set up credentials management in your environment

Each time you use the AWS SDK for Go to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for Go has the appropriate permissions to make that call. If the credentials don’t cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Specifying Credentials in the AWS SDK for Go Developer Guide.

### Step 5: Add AWS SDK Code

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You will run this code later.

In the AWS Cloud9 IDE, create a file with this content, and save the file with the name `s3.go`.

```go
package main

import (  
    "fmt"  
    "os"  
    "github.com/aws/aws-sdk-go/aws"  
    "github.com/aws/aws-sdk-go/aws/session"  
    "github.com/aws/aws-sdk-go/service/s3"
)

func main() {
    if len(os.Args) < 3 {
        fmt.Printf("Usage: go run s3.go <the bucket name> <the AWS Region to use>
        "Example: go run s3.go my-test-bucket us-east-2"
        os.Exit(1)
    }
    sess := session.Must(session.NewSessionWithOptions(session.Options{
        SharedConfigState: session.SharedConfigEnable,
    })))
    svc := s3.New(sess, &aws.Config{
        Region: aws.String(os.Args[2]),
    }
```
Step 5: Add AWS SDK Code

```go
// List all of your available buckets in this AWS Region.
func listMyBuckets(svc *s3.S3) {
    result, err := svc.ListBuckets(nil)
    if err != nil {
        exitErrorf("Unable to list buckets, %v", err)
    }
    fmt.Println("My buckets now are:
    
    for _, b := range result.Buckets {
        fmt.Printf(aws.StringValue(b.Name) + "\n")
    }
    fmt.Printf("\n")
}

// Create a bucket in this AWS Region.
func createMyBucket(svc *s3.S3, bucketName string, region string) {
    _, err := svc.CreateBucket(&s3.CreateBucketInput{
        Bucket: aws.String(bucketName),
        CreateBucketConfiguration: &s3.CreateBucketConfiguration{
            LocationConstraint: aws.String(region),
        },
    })
    if err != nil {
        exitErrorf("Unable to create bucket, %v", err)
    }

    // Wait until bucket is created before finishing
    err = svc.WaitUntilBucketExists(&s3.HeadBucketInput{
        Bucket: aws.String(bucketName),
    })
}

// Delete the bucket you just created.
func deleteMyBucket(svc *s3.S3, bucketName string) {
    _, err := svc.DeleteBucket(&s3.DeleteBucketInput{
        Bucket: aws.String(bucketName),
    })
    if err != nil {
        exitErrorf("Unable to delete bucket, %v", err)
    }

    // Wait until bucket is deleted before finishing
    err = svc.WaitUntilBucketNotExists(&s3.HeadBucketInput{
        Bucket: aws.String(bucketName),
    })
}
```
Step 6: Run the AWS SDK Code

1. In the AWS Cloud9 IDE, on the menu bar, choose Run, Run Configurations, New Run Configuration.
2. On the [New] - Idle tab, choose Runner: Auto, and then choose Go.
3. For Command, type `s3.go YOUR_BUCKET_NAME THE_AWS_REGION`, where YOUR_BUCKET_NAME is the name of the bucket you want to create and then delete, and THE_AWS_REGION is the ID of the AWS Region you want to create the bucket in. For example, for the US East (Ohio) Region, use `us-east-2`. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.
   
   **Note**
   
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.
4. Choose the Run button, and compare your output.

   My buckets now are:
   
   Creating a new bucket named 'my-test-bucket'...
   
   My buckets now are:
   
   my-test-bucket
   
   Deleting the bucket named 'my-test-bucket'...
   
   My buckets now are:

Step 7: Clean Up

To prevent ongoing charges to your AWS account after you’re done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

TypeScript Sample for AWS Cloud9

This sample shows you how to work with TypeScript in an AWS Cloud9 development environment.

Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2 and Amazon S3. For more information, see Amazon EC2 Pricing and Amazon S3 Pricing.

Topics
- Prerequisites (p. 453)
- Step 1: Install Required Tools (p. 453)
- Step 2: Add Code (p. 454)
Prerequisites

Before you use this sample, be sure to meet the following requirements.

- **You must have an existing AWS Cloud9 EC2 development environment.** This sample assumes you already have an EC2 environment that is connected to an Amazon EC2 instance running Amazon Linux or Ubuntu Server. If you have a different type of environment or operating system, you might need to adapt this sample’s instructions to set up related tools. See Creating an Environment in AWS Cloud9 (p. 55) for details.

- **You have the AWS Cloud9 IDE for the existing environment already open.** When you open an environment, AWS Cloud9 opens the IDE for that environment in your web browser. See Opening an Environment in AWS Cloud9 (p. 67) for details.

Step 1: Install Required Tools

In this step, you install TypeScript by using Node Package Manager (npm). To install npm, you use Node Version Manager (nvm). If you don't have nvm, you install it in this step first.

1. In a terminal session in the AWS Cloud9 IDE, confirm whether TypeScript is already installed by running the command line TypeScript compiler with the `--version` option. (To start a new terminal session, on the menu bar, choose Windows, New Terminal.) If successful, the output contains the TypeScript version number. If TypeScript is installed, skip ahead to Step 2: Add Code (p. 454).

   ```bash
   tsc --version
   ```

2. Confirm whether npm is already installed by running npm with the `--version` option. If successful, the output contains the npm version number. If npm is installed, skip ahead to step 10 in this procedure to use npm to install TypeScript.

   ```bash
   npm --version
   ```

3. Run the `yum update` for (Amazon Linux) or `apt update` for (Ubuntu Server) command to help ensure the latest security updates and bug fixes are installed.

   For Amazon Linux:

   ```bash
   sudo yum -y update
   ```

   For Ubuntu Server:

   ```bash
   sudo apt update
   ```

4. To install npm, begin by running the following command to download Node Version Manager (nvm). (nvm is a simple Bash shell script that's useful for installing and managing Node.js versions. For more information, see Node Version Manager on the GitHub website.)
Step 2: Add Code

1. In the AWS Cloud9 IDE, create a file named hello.ts. (To create a file, on the menu bar, choose File, New File. To save the file, choose File, Save.)

2. In a terminal in the IDE, from the same directory as the hello.ts file, run npm to install the @types/node library.

   
   ```bash
   npm install @types/node
   ```

   This adds a node_modules/@types/node folder in the same directory as the hello.ts file. This new folder contains Node.js type definitions that TypeScript needs later in this procedure for the console.log and process.argv properties that you will add to the hello.ts file.

3. Add the following code to the hello.ts file:

   ```javascript
   console.log('Hello, World!');
   console.log('The sum of 2 and 3 is 5.);
   const sum: number = parseInt(process.argv[2], 10) + parseInt(process.argv[3], 10);
   ```
Step 3: Run the Code

1. In the terminal, from the same directory as the `hello.ts` file, run the TypeScript compiler. Specify the `hello.ts` file and additional libraries to include.

   ```bash
   tsc hello.ts --lib es6
   ```

   TypeScript uses the `hello.ts` file and a set of ECMAScript 6 (ES6) library files to transpile the TypeScript code in the `hello.ts` file into equivalent JavaScript code in a file named `hello.js`.

2. In the Environment window, open the `hello.js` file.

3. On the menu bar, choose Run, Run Configurations, New Run Configuration.


5. For Command, type `hello.js 5 9`. In the code, 5 represents `process.argv[2]`, and 9 represents `process.argv[3]`. (`process.argv[0]` represents the name of the runtime (node), and `process.argv[1]` represents the name of the file (hello.js)).

6. Choose Run, and compare your output. When you're done, choose Stop.

   ```javascript
   console.log('Hello, World!');
   console.log('The sum of 2 and 3 is 5.');
   console.log('The sum of 5 and 9 is 14.');
   ```

Note

Instead of creating a new run configuration in the IDE, you can also execute this code by running the command `node hello.js 5 9` from the terminal.

Step 4: Install and Configure the AWS SDK for JavaScript in Node.js

You can enhance this sample to use the AWS SDK for JavaScript in Node.js to create an Amazon S3 bucket, list your available buckets, and then delete the bucket you just created.
In this step, you install and configure the AWS SDK for JavaScript in Node.js. The SDK provides a convenient way to interact with AWS services such as Amazon S3, from your JavaScript code. After you install the AWS SDK for JavaScript in Node.js, you must set up credentials management in your environment. The SDK needs these credentials to interact with AWS services.

**To install the AWS SDK for JavaScript in Node.js**

In a terminal session in the AWS Cloud9 IDE, from the same directory as the `hello.js` file from Step 3: Run the Code (p. 455), run `npm` to install the AWS SDK for JavaScript in Node.js.

```
npm install aws-sdk
```

This command adds several folders to the `node_modules` folder from Step 3: Run the Code (p. 455). These folders contain source code and dependencies for the AWS SDK for JavaScript in Node.js. For more information, see Installing the SDK for JavaScript in the AWS SDK for JavaScript Developer Guide.

**To set up credentials management in your environment**

Each time you use the AWS SDK for JavaScript in Node.js to call an AWS service, you must provide a set of credentials with the call. These credentials determine whether the AWS SDK for JavaScript in Node.js has the appropriate permissions to make that call. If the credentials don't cover the appropriate permissions, the call will fail.

In this step, you store your credentials within the environment. To do this, follow the instructions in Calling AWS Services from an Environment in AWS Cloud9 (p. 70), and then return to this topic.

For additional information, see Setting Credentials in Node.js in the AWS SDK for JavaScript Developer Guide.

**Step 5: Add AWS SDK Code**

In this step, you add some more code, this time to interact with Amazon S3 to create a bucket, list your available buckets, and then delete the bucket you just created. You'll run this code later.

1. In the AWS Cloud9 IDE, in the same directory as the `hello.js` file in previous steps, create a file named `s3.ts`.
2. From a terminal in the AWS Cloud9 IDE, in the same directory as the `s3.ts` file, enable the code to call Amazon S3 operations asynchronously by running `npm` twice to install the async library for TypeScript and again for JavaScript.

```
npm install @types/async # For TypeScript.
npm install async        # For JavaScript.
```

3. Add the following code to the `s3.ts` file:

```javascript
import * as async from 'async';
import * as AWS from 'aws-sdk';

if (process.argv.length < 4) {
    console.log('Usage: node s3.js <the bucket name> <the AWS Region to use>
    Example: node s3.js my-test-bucket us-east-2');
    process.exit(1);
}

const AWS = require('aws-sdk'); // To set the AWS credentials and AWS Region.
const async = require('async'); // To call AWS operations asynchronously.
```
const s3: AWS.S3 = new AWS.S3({apiVersion: '2006-03-01'});
const bucket_name: string = process.argv[2];
const region: string = process.argv[3];

AWS.config.update({
    region: region
});

const create_bucket_params: any = {
    Bucket: bucket_name,
    CreateBucketConfiguration: {
        LocationConstraint: region
    }
};

const delete_bucket_params: any = {
    Bucket: bucket_name
};

// List all of your available buckets in this AWS Region.
function listMyBuckets(callback): void {
    s3.listBuckets(function(err, data) {
        if (err) {
            console.log("My buckets now are:
            
        } else {
            console.log("My buckets now are:
            
            for (let i: number = 0; i < data.Buckets.length; i++) {
                console.log(data.Buckets[i].Name);
            }
            
            callback(err);
        });
    });
}

// Create a bucket in this AWS Region.
function createMyBucket(callback): void {
    console.log("Creating a bucket named " + bucket_name + "}
    ");
    s3.createBucket(create_bucket_params, function(err, data) {
        if (err) {
            console.log(err.code + ": " + err.message);
        }
        callback(err);
    });
}

// Delete the bucket you just created.
function deleteMyBucket(callback): void {
    console.log("Deleting the bucket named " + bucket_name + "}
    ");
    s3.deleteBucket(delete_bucket_params, function(err, data) {
        if (err) {
            console.log(err.code + ": " + err.message);
        }
        callback(err);
    });
}

// Call the AWS operations in the following order.
async.series([listMyBuckets,
    createMyBucket,
Step 6: Run the AWS SDK Code

1. In the terminal, from the same directory as the s3.ts file, run the TypeScript compiler. Specify the s3.ts file and additional libraries to include.

```
tsc s3.ts --lib es6
```

TypeScript uses the s3.ts file, the AWS SDK for JavaScript in Node.js, the async library, and a set of ECMA Script 6 (ES6) library files to transpile the TypeScript code in the s3.ts file into equivalent JavaScript code in a file named s3.js.

2. In the Environment window, open the s3.js file.

3. On the menu bar, choose Run, Run Configurations, New Run Configuration.


5. For Command, type `s3.js YOUR_BUCKET_NAME THE_AWS_REGION`, where YOUR_BUCKET_NAME is the name of the bucket you want to create and then delete, and THE_AWS_REGION is the ID of the AWS Region to create the bucket in. For example, for the US East (Ohio) Region, use `us-east-2`. For more IDs, see Amazon Simple Storage Service (Amazon S3) in the Amazon Web Services General Reference.

   **Note**
   Amazon S3 bucket names must be unique across AWS—not just your AWS account.

6. Choose Run, and compare your output. When you're done, choose Stop.

   ```
   My buckets now are:
   Creating a new bucket named 'my-test-bucket'...
   My buckets now are:
   my-test-bucket
   Deleting the bucket named 'my-test-bucket'...
   My buckets now are:
   ```

Step 7: Clean Up

To prevent ongoing charges to your AWS account after you're done using this sample, you should delete the environment. For instructions, see Deleting an Environment in AWS Cloud9 (p. 95).

Docker Sample for AWS Cloud9

This sample shows you how to connect an AWS Cloud9 SSH development environment to a running Docker container inside of an Amazon Linux instance in Amazon EC2. This enables you to use the AWS Cloud9 IDE to work with code and files inside of a Docker container and to run commands on that container. For information about Docker, see What is Docker on the Docker website.
Creating this sample might result in charges to your AWS account. These include possible charges for services such as Amazon EC2. For more information, see Amazon EC2 Pricing.

**Topics**

- Prerequisites (p. 459)
- Step 1: Install and Run Docker (p. 459)
- Step 2: Build the Image (p. 460)
- Step 3: Run the Container (p. 462)
- Step 4: Create the Environment (p. 463)
- Step 5: Run the Code (p. 464)
- Step 6: Clean Up (p. 464)

**Prerequisites**

- **You should have an Amazon EC2 instance running Amazon Linux or Ubuntu Server.** This sample assumes you already have an Amazon EC2 instance running Amazon Linux or Ubuntu Server in your AWS account. To launch an Amazon EC2 instance, see Launch a Linux Virtual Machine. In the Choose an Amazon Machine Image (AMI) page of the wizard, choose an AMI whose display name starts with Amazon Linux AMI or Ubuntu Server.

- **If the Amazon EC2 instance runs within an Amazon VPC, there are additional requirements.** See VPC Settings for AWS Cloud9 Development Environments (p. 467).

- **The Amazon EC2 instance should have at least 8 to 16 GB of free disk space available.** This sample uses Docker images that are over 3 GB in size and can use additional increments of 3 GB or more of disk space to build images. If you try to run this sample on a disk that has 8 GB of free space or less, we’ve found that the Docker image might not build or the Docker container might not run. To check the instance’s free disk space, you can run a command such as `df -h` (for “disk filesystem information in human-readable format”) on the instance. To increase an existing instance’s disk size, see Modifying a Volume in the Amazon EC2 User Guide for Linux Instances.

**Step 1: Install and Run Docker**

In this step, you check if Docker is installed on the Amazon EC2 instance, and install Docker if it isn’t already installed. After you install Docker, you run it on the instance.

1. Connect to the running Amazon EC2 instance by using an SSH client such as the `ssh` utility or PuTTY. To do this, see “Step 3: Connect to Your Instance” in Launch a Linux Virtual Machine.

2. Check if Docker is installed on the instance. To do this, run the `docker` command on the instance with the `--version` option.

   `docker --version`

   If Docker is installed, the Docker version and build number are displayed. In this case, skip ahead to step 5 later in this procedure.

3. Install Docker. To do this, run the `yum` or `apt` command with the `install` action, specifying the `docker` or `docker.io` package to install.

   For Amazon Linux:

   `sudo yum install -y docker`
Step 2: Build the Image

In this step, you use a Dockerfile to build a Docker image onto the instance. This sample uses an image that includes Node.js and a sample chat server application.

1. On the instance, create the Dockerfile. To do this, with the SSH client still connected to the instance, in the /tmp directory on the instance, create a file named Dockerfile. For example, run the touch command as follows.

```
sudo touch /tmp/Dockerfile
```

2. Add the following contents to the Dockerfile file.

```dockerfile
# Build a Docker image based on the cloud9/ws-nodejs Docker image.
FROM cloud9/ws-nodejs

# Enable the Docker container to communicate with AWS Cloud9 by
# installing SSH.
RUN apt update && apt install -y openssh-server

# Ensure that Node.js is installed.
RUN apt install -y nodejs && ln -s /usr/bin/nodejs /usr/bin/node

# Disable password authentication by turning off the
# Pluggable Authentication Module (PAM).
RUN sed -i 's/UsePAM yes/UsePAM no/g' /etc/ssh/sshd_config

# Add the AWS Cloud9 SSH public key to the Docker container.
# This assumes a file named authorized_keys containing the
# AWS Cloud9 SSH public key already exists in the same
# directory as the Dockerfile.
RUN mkdir -p /home/ubuntu/.ssh
ADD ./authorized_keys /home/ubuntu/.ssh/authorized_keys
RUN chown -R ubuntu /home/ubuntu/.ssh
chmod 700 /home/ubuntu/.ssh
chmod 600 /home/ubuntu/.ssh/authorized_keys

# Start SSH in the Docker container.
CMD /usr/sbin/sshd -D
```
AWS Cloud9 User Guide
Step 2: Build the Image

To add the preceding contents to the Dockerfile file, you could use the vi utility on the instance as follows.

a. Use vi to open the /tmp/Dockerfile file.

```bash
sudo vi /tmp/Dockerfile
```

b. Paste the preceding contents into the Dockerfile file. If you're not sure how to do this, see your SSH client's documentation.

c. Switch to command mode. To do this, press the Esc key. (-- INSERT -- disappears from the bottom of the window.)

d. Type :wq (to write to the /tmp/Dockerfile file, save the file, and then exit vi), and then press Enter.

**Note**
You can access a frequently updated list of Docker images from AWS CodeBuild. For more information, see Docker images provided by CodeBuild in the AWS CodeBuild User Guide.

3. On the instance, create a file that contains the AWS Cloud9 SSH public key for the Docker container to use. To do this, in the same directory as the Dockerfile file, create a file named authorized_keys, for example, by running the touch command.

```bash
sudo touch /tmp/authorized_keys
```

4. Add the AWS Cloud9 SSH public key to the authorized_keys file. To get the AWS Cloud9 SSH public key, do the following:


b. In the AWS navigation bar, in the AWS Region selector, choose the AWS Region where you'll want to create the AWS Cloud9 development environment later in this topic.

c. If a welcome page is displayed, for New AWS Cloud9 environment, choose Create environment. Otherwise, choose Create environment.

d. On the Name environment page, for Name, type a name for the environment. (The name doesn't matter here. You'll choose a different name later.)

e. Choose Next step.

f. For Environment type, choose Connect and run in remote server (SSH).

g. Expand View public SSH key.

h. Choose Copy key to clipboard. (This is between View public SSH key and Advanced settings.)

i. Choose Cancel.

j. Paste the contents of the clipboard into the authorized_keys file, and then save the file. For example, you can use the vi utility, as described earlier in this step.

5. Build the image by running the docker command with the build action, adding the tag cloud9-image:latest to the image and specifying the path to the Dockerfile file to use.

```bash
sudo docker build -t cloud9-image:latest /tmp
```

If successful, the last two lines of the build output display Successfully built and Successfully tagged.
To confirm that Docker successfully built the image, run the `docker` command with the `image ls` action.

```
sudo docker image ls
```

If successful, the output displays an entry where the `REPOSITORY` field is set to `cloud9-image` and the `TAG` field is set to `latest`.

6. Make a note of the Amazon EC2 instance's public IP address. You'll need it for Step 4: Create the Environment (p. 463). If you're not sure what the public IP address of the instance is, you can run the following command on the instance to get it.

```
```

## Step 3: Run the Container

In this step, you run a Docker container on the instance. This container is based on the image you built in the previous step.

1. To run the Docker container, run the `docker` command on the instance with the `run` action and the following options.

```
sudo docker run -d -it --expose 9090 -p 0.0.0.0:9090:22 --name cloud9 cloud9-image:latest
```

- `-d` runs the container in detached mode, exiting whenever the root process that is used to run the container (in this sample, the SSH client) exits.
- `-it` runs the container with an allocated pseudo-TTY and keeps STDIN open, even if the container is not attached.
- `--expose` makes the specified port (in this sample, port 9090) available from the container.
- `-p` makes the specified port available internally to the Amazon EC2 instance over the specified IP address and port. In this sample, port 9090 on the container can be accessed internally through port 22 on the Amazon EC2 instance.
- `--name` is a human-readable name for the container (in this sample, `cloud9`).
- `cloud9-image:latest` is the human-readable name of the built image to use to run the container.

To confirm that Docker is successfully running the container, run the `docker` command with the `container ls` action.

```
sudo docker container ls
```

If successful, the output displays an entry where the `IMAGE` field is set to `cloud9-image:latest` and the `NAMES` field is set to `cloud9`.

2. Log in to the running container. To do this, run the `docker` command with the `exec` action and the following options.

```
sudo docker exec -it cloud9 bash
```
Step 4: Create the Environment

In this step, you use AWS Cloud9 to create an AWS Cloud9 SSH development environment and connect it to the running Docker container. After AWS Cloud9 creates the environment, it displays the AWS Cloud9 IDE so that you can start working with the files and code in the container.

1. Sign in to the AWS Cloud9 console as follows:
   - If you're the only individual using your AWS account or you are an IAM user in a single AWS account, go to https://console.aws.amazon.com/cloud9/.
   - If your organization uses AWS Single Sign-On (SSO), see your AWS account administrator for sign-in instructions.
   - If you're using an AWS Educate Starter Account, see Step 2: Use an AWS Educate Starter Account to sign in to the AWS Cloud9 console (p. 29) in Individual Student Signup.
   - If you're a student in a classroom, see your instructor for sign-in instructions.

2. In the AWS navigation bar, in the AWS Region selector, choose the AWS Region where you want to create the SSH environment.

3. If a welcome page is displayed, for New AWS Cloud9 environment, choose Create environment. Otherwise, choose Create environment.

4. On the Name environment page, for Name, type a name for the environment.

5. To add a description to the environment, type it in Description.

6. Choose Next step.

7. For Environment type, choose Connect and run in remote server (SSH).

8. For User, type ubuntu.

9. For Host, type the public IP address of the Amazon EC2 instance, which you noted earlier.
10. For **Port**, type 9090.
11. Expand **Advanced settings**.
12. For **Environment path**, type the path to the directory on the running container that you want AWS Cloud9 to start from after it logs in.
13. For **Node.js binary path**, type the path to the directory on the running container that contains the Node.js binary, which you noted earlier.
14. Choose **Next step**.
15. Choose **Create environment**.
16. When the **AWS Cloud9 Installer** dialog box appears, choose **Next**.
17. In the list of components to be installed, clear the **c9.ide.lambda.docker** check box, and then choose **Next**. This is because AWS Cloud9 cannot run Docker inside of Docker.
18. When the **AWS Cloud9 Installer** dialog box displays **Installation Completed**, choose **Next**, and then choose **Finish**. The AWS Cloud9 IDE appears for the running container, and you can start working with the container's files and code.

**Note**
If the container stops running, you can no longer use the IDE to access the container until you start running the container again. To do this, go back to Step 3: Run the Container (p. 462).

**Step 5: Run the Code**
In this step, you use the AWS Cloud9 IDE to run a sample application inside the running Docker container.

1. With the AWS Cloud9 IDE displayed for the running container, start the sample chat server. To do this, in the **Environment** window, right-click the sample workspace/server.js file, and then choose **Run**.
2. Preview the sample application. To do this, in the **Environment** window, open the the workspace/client/index.html file. Then, on the menu bar, choose **Tools, Preview, Preview Running Application**.
3. On the application preview tab, for **Your Name**, type your name. For **Message**, type a message. Then choose **Send**. The chat server adds your name and message to the list.

**Step 6: Clean Up**
In this step, you delete the environment and remove AWS Cloud9 and Docker support files from the Amazon EC2 instance. Also, to prevent ongoing charges to your AWS account after you're done using this sample, you should terminate the Amazon EC2 instance that is running Docker.

**Step 6.1: Delete the Environment**
To delete the environment, see Deleting an Environment in AWS Cloud9 (p. 95).

**Step 6.2: Remove AWS Cloud9 Support Files from the Container**
After you delete the environment, some AWS Cloud9 support files still remain in the container. If you want to keep using the container but no longer need these support files, delete the .c9 folder from the directory on the container that you specified AWS Cloud9 to start from after it logs in. For example, if the directory is ~, run the `rm` command with the `-r` option as follows.

```
sudo rm -r ~/.c9
```
Step 6.3: Remove Docker Support Files from the Instance

If you no longer want to keep the Docker container, the Docker image, and Docker on the Amazon EC2 instance, but you want to keep the instance, you can remove these Docker support files as follows.

1. Remove the Docker container from the instance. To do this, run the `docker` command on the instance with the `stop` and `rm` stop actions and the human-readable name of the container.

   ```bash
   sudo docker stop cloud9
   sudo docker rm cloud9
   ```

2. Remove the Docker image from the instance. To do this, run the `docker` command on the instance with the `image rm` action and the image's tag.

   ```bash
   sudo docker image rm cloud9-image:latest
   ```

3. Remove any additional Docker support files that might still exit. To do this, run the `docker` command on the instance with the `system prune` action.

   ```bash
   sudo docker system prune -a
   ```

4. Uninstall Docker. To do this, run the `yum` command on the instance with the `remove` action, specifying the `docker` package to uninstall.

   For Amazon Linux:

   ```bash
   sudo yum -y remove docker
   ```

   For Ubuntu Server:

   ```bash
   sudo apt -y remove docker
   ```

   You can also remove the Dockerfile and authorized_keys files you created earlier. For example, run the `rm` command on the instance.

   ```bash
   sudo rm /tmp/Dockerfile
   sudo rm /tmp/authorized_keys
   ```

Step 6.4: Terminate the Instance

To terminate the Amazon EC2 instance, see Terminate Your Instance in the Amazon EC2 User Guide for Linux Instances.

Related Samples

- **Getting Started with AWS RoboMaker** in the AWS RoboMaker Developer Guide. This sample uses AWS Cloud9 to modify, build, and bundle a sample robot application.
Advanced Topics for AWS Cloud9

These topics contain the following types of information:

- Information that is used for advanced configuration and decision making.
- Information that is related to a particular task and can provide a better understand but is not critical to accomplishing that task.

Topics

- EC2 Environments versus SSH Environments in AWS Cloud9 (p. 466)
- VPC Settings for AWS Cloud9 Development Environments (p. 467)
- AWS Cloud9 SSH Development Environment host requirements (p. 479)
- Using the AWS Cloud9 Installer (p. 481)
- Inbound SSH IP address ranges for AWS Cloud9 (p. 483)
- Amazon Machine Image (AMI) Contents for an AWS Cloud9 EC2 Development Environment (p. 485)
- Access Permissions Reference for AWS Cloud9 (p. 486)
- Using Service-Linked Roles for AWS Cloud9 (p. 504)
- Logging AWS Cloud9 API Calls with AWS CloudTrail (p. 507)
- Tags (p. 516)

EC2 Environments versus SSH Environments in AWS Cloud9

As discussed in the introduction for environments and computing resources (p. 2) and when working with environments (p. 55), your AWS Cloud9 environments can be set up as EC2 environments or SSH environments.

The following table provides information about the similarities and differences between EC2 environments and SSH environments.

<table>
<thead>
<tr>
<th>EC2 environments</th>
<th>SSH environments</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Cloud9 creates an associated Amazon EC2 instance and manages that instance's lifecycle (for example, start, stop, and terminate).</td>
<td>You use an existing cloud compute instance or your own server. You manage that instance's or server's lifecycle.</td>
</tr>
<tr>
<td>The instance runs on Amazon Linux or Ubuntu Server.</td>
<td>You can use any cloud compute instance that runs Linux, or your own server running Linux.</td>
</tr>
<tr>
<td>AWS Cloud9 automatically sets up the instance to start working with AWS Cloud9.</td>
<td>You must manually configure the instance or your own server to work with AWS Cloud9.</td>
</tr>
<tr>
<td>AWS Cloud9 automatically sets up the AWS Command Line Interface (AWS CLI) on the instance for you to start using.</td>
<td>If you want to use the AWS CLI on the instance or your own server, you must set it up yourself.</td>
</tr>
<tr>
<td>The instance has access to hundreds of useful packages, with some common packages already installed and configured, such as Git, Docker, Node.js, and Python.</td>
<td>You might need to download, install, and configure additional packages to complete common tasks.</td>
</tr>
</tbody>
</table>
VPC Settings for AWS Cloud9 Development Environments

Every AWS Cloud9 development environment associated with an Amazon Virtual Private Cloud (Amazon VPC) must meet specific VPC requirements. These environments include EC2 environments, as well as SSH environments associated with AWS cloud compute instances (for example Amazon EC2 and Amazon Lightsail instances) that run within a VPC.

Topics
- Amazon VPC Requirements for AWS Cloud9 (p. 467)
- Create an Amazon VPC for AWS Cloud9 (p. 477)
- Create a Subnet for AWS Cloud9 (p. 478)

Amazon VPC Requirements for AWS Cloud9

The Amazon VPC that AWS Cloud9 uses requires the following settings. If you’re already familiar with these requirements and just want to quickly create a compatible VPC, skip ahead to Create an Amazon VPC for AWS Cloud9 (p. 477).

Use the following checklist to confirm that the VPC meets all of the following requirements.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>How to confirm</th>
<th>Additional resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>The VPC can be in the same AWS account and AWS Region as the AWS Cloud9 development environment. —OR— The VPC can be a shared VPC in a different AWS account than the environment. (However, the VPC must be in the same AWS Region as the environment).</td>
<td>View a List of VPCs For an AWS Region (p. 468)</td>
<td>• Create an Amazon VPC for AWS Cloud9 (p. 477) • Working with Shared VPCs in the Amazon VPC User Guide</td>
</tr>
<tr>
<td>The VPC must have a public subnet. (A subnet is public if its traffic is routed to an internet gateway.)</td>
<td>• View a List of Subnets for a VPC (p. 469) • Confirm Whether a Subnet is Public (p. 469)</td>
<td>• Create a Subnet for AWS Cloud9 (p. 478) • View or Change Settings For an Internet Gateway (p. 470) • Create an Internet Gateway (p. 470)</td>
</tr>
</tbody>
</table>
### Criteria | How to confirm | Additional resources
--- | --- | ---
The subnet must have a route table with a minimum set of routes. | • Confirm Whether a Subnet Has a Route Table (p. 470)<br>• View or Change Settings For a Route Table (p. 472)<br>• View or Change Settings For a Route Table (p. 472) | • Attach an Internet Gateway to a VPC (p. 470)<br>• Create a Route Table (p. 471)<br>• Attach a Route Table to a Subnet (p. 471)

The associated security groups for the VPC (or for the AWS cloud compute instance, depending on your architecture) must allow a minimum set of inbound and outbound traffic. | • View a List of Security Groups for a VPC (p. 472)<br>• View a List of Security Groups For an AWS Cloud Compute Instance (p. 472)<br>• View or Change Settings For a Security Group in a VPC (p. 473)<br>• View a List of Security Groups For an AWS Cloud Compute Instance (p. 472)<br>• Minimum Inbound and Outbound Traffic Settings for AWS Cloud9 (p. 473) | • Create a Security Group in a VPC (p. 475)

For an additional layer of security, if the VPC has a network ACL, the network ACL must allow a minimum set of inbound and outbound traffic. | • Confirm Whether a VPC Has at Least One Network ACL (p. 475)<br>• View a List of Network ACLs For a VPC (p. 476)<br>• View or Change Settings For a Network ACL (p. 476)<br>• Minimum Inbound and Outbound Traffic Settings for AWS Cloud9 (p. 473) | Create a Network ACL (p. 476)

**Note**
For the following procedures, if you use the Amazon VPC or Amazon EC2 consoles, we recommend you sign in to the AWS Management Console and open the Amazon VPC console (https://console.aws.amazon.com/vpc) or Amazon EC2 console (https://console.aws.amazon.com/ec2) using credentials for an IAM administrator user in your AWS account. If you use the AWS CLI or the aws-shell, we recommend you configure the AWS CLI or the aws-shell with the credentials for an IAM administrator user in your AWS account. If you can't do this, check with your AWS account administrator.

**View a List of VPCs For an AWS Region**

To use the Amazon VPC console, in the AWS navigation bar, choose the AWS Region that AWS Cloud9 will create the environment in. Then choose **Your VPCs** in the navigation pane.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-vpcs` command, for example as follows.
In the preceding command, replace `us-east-2` with the AWS Region that AWS Cloud9 will create the environment in. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

The output contains the list of VPC IDs.

### View a List of Subnets for a VPC

To use the Amazon VPC console, choose **Your VPCs** in the navigation pane. Note the VPC’s ID in the **VPC ID** column. Then choose **Subnets** in the navigation pane, and look for subnets that contain that ID in the **VPC** column.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-subnets` command, for example as follows.

```bash
aws ec2 describe-subnets --output table --query 'Subnets[*].[SubnetId,VpcId]' --region us-east-2
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the subnets. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

In the output, look for subnets that match the VPC’s ID.

### Confirm Whether a Subnet is Public

To use the Amazon VPC console, choose **Subnets** in the navigation pane. Select the box next to the subnet you want AWS Cloud9 to use. On the **Route Table** tab, if there is an entry in the **Target** column that starts with `igw-`, the subnet is public.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-route-tables` command, for example as follows.

```bash
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Routes[*].{GatewayIds:GatewayId}' --region us-east-2 --filters Name=association.subnet-id,Values=subnet-12a3456b
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the subnet, and replace `subnet-12a3456b` with the subnet ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

In the output, if there is at least one result that starts with `igw-`, the subnet is public.

In the output, if there are no results, then the route table might be associated with the VPC instead of the subnet. To confirm this, run the Amazon EC2 `describe-route-tables` command for the subnet's related VPC instead of the subnet itself, for example as follows.

```bash
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Routes[*].{GatewayIds:GatewayId}' --region us-east-1 --filters Name=vpc-id,Values=vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`. 

469
In the output, if there is at least one result that starts with `igw-`, the VPC contains an internet gateway.

**View or Change Settings For an Internet Gateway**

To use the Amazon VPC console, choose **Internet Gateways** in the navigation pane. Select the box next to the internet gateway. To see the settings, look at each of the tabs. To change a setting on a tab, choose **Edit** if applicable, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell to see the settings, run the Amazon EC2 `describe-internet-gateways` command, for example as follows.

```
aws ec2 describe-internet-gateways --output table --region us-east-2 --internet-gateway-id igw-1234ab5c
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the internet gateway, and replace `igw-1234ab5c` with the internet gateway ID. To run the preceding command with the aws-shell, omit `aws`.

**Create an Internet Gateway**

To use the Amazon VPC console, choose **Internet Gateways** in the navigation pane. Choose **Create internet gateway**, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `create-internet-gateway` command, for example as follows.

```
aws ec2 create-internet-gateway --output text --query 'InternetGateway.InternetGatewayId' --region us-east-2
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the new internet gateway. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

The output contains the ID of the new internet gateway.

**Attach an Internet Gateway to a VPC**

To use the Amazon VPC console, choose **Internet Gateways** in the navigation pane. Select the box next to the internet gateway. Choose **Actions, Attach to VPC** if available, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `attach-internet-gateway` command, for example as follows.

```
aws ec2 attach-internet-gateway --region us-east-2 --internet-gateway-id igw-a1b2cdef --vpc-id vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the internet gateway, replace `igw-a1b2cdef` with the internet gateway ID, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command with the aws-shell, omit `aws`.

**Confirm Whether a Subnet Has a Route Table**

To use the Amazon VPC console, choose **Subnets** in the navigation pane. Select the box next to the VPC's public subnet that you want AWS Cloud9 to use. On the **Route Table** tab, if there is a value for **Route Table**, the public subnet has a route table.
To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-route-tables` command, for example as follows.

```bash
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Associations[*].{RouteTableIds:RouteTableId}' --region us-east-2 --filters Name=association.subnet-id,Values=subnet-12a3456b
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the public subnet, and replace `subnet-12a3456b` with the public subnet ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

If there are values in the output, the public subnet has at least one route table.

In the output, if there are no results, then the route table might be associated with the VPC instead of the subnet. To confirm this, run the Amazon EC2 `describe-route-tables` command for the subnet's related VPC instead of the subnet itself, for example as follows.

```bash
aws ec2 describe-route-tables --output table --query 'RouteTables[*].Associations[*].{RouteTableIds:RouteTableId}' --region us-east-2 --filters Name=vpc-id,Values=vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

In the output, if there is at least one result, the VPC has at least one route table.

## Attach a Route Table to a Subnet

To use the Amazon VPC console, choose **Route Tables** in the navigation pane. Select the box next to the route table that you want to attach. On the **Subnet Associations** tab, choose **Edit**, select the box next to the subnet you want to attach it to, and then choose **Save**.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `associate-route-table` command, for example as follows.

```bash
aws ec2 associate-route-table --region us-east-2 --subnet-id subnet-12a3456b --route-table-id rtb-ab12cde3
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the route table, replace `subnet-12a3456b` with the subnet ID, and replace `rtb-ab12cde3` with the route table ID. To run the preceding command with the aws-shell, omit `aws`.

## Create a Route Table

To use the Amazon VPC console, choose **Route Tables** in the navigation pane. Choose **Create Route Table**, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `create-route-table` command, for example as follows.

```bash
aws ec2 create-route-table --output text --query 'RouteTable.RouteTableId' --region us-east-2 --vpc-id vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the new route table, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.  

471
The output contains the ID of the new route table.

**View or Change Settings For a Route Table**

To use the Amazon VPC console, choose **Route Tables** in the navigation pane. Select the box next to the route table. To see the settings, look at each of the tabs. To change a setting on a tab, choose **Edit**, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell to see the settings, run the Amazon EC2 `describe-route-tables` command, for example as follows.

```bash
aws ec2 describe-route-tables --output table --region us-east-2 --route-table-ids rtb-\ab12cde3
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the route table, and replace `rtb-ab12cde3` with the route table ID. To run the preceding command with the aws-shell, omit `aws`.

**Minimum Suggested Route Table Settings for AWS Cloud9**

<table>
<thead>
<tr>
<th>Destination</th>
<th>Target</th>
<th>Status</th>
<th>Propagated</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIDR-BLOCK</td>
<td>local</td>
<td>Active</td>
<td>No</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>igw-INTERNET-GATEWAY-ID</td>
<td>Active</td>
<td>No</td>
</tr>
</tbody>
</table>

In these settings, **CIDR-BLOCK** is the subnet's CIDR block, and **igw-INTERNET-GATEWAY-ID** is the ID of a compatible internet gateway.

**View a List of Security Groups for a VPC**

To use the Amazon VPC console, choose **Security Groups** in the navigation pane. In the **Search Security Groups** box, type the VPC's ID or name, and then press **Enter**. Security groups for that VPC appear in the list of search results.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-security-groups` command, for example as follows.

```bash
aws ec2 describe-security-groups --output table --query 'SecurityGroups[*].GroupId' --region us-east-2 --filters Name=vpc-id,Values=vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

The output contains the list of security group IDs for that VPC.

**View a List of Security Groups For an AWS Cloud Compute Instance**

To use the Amazon EC2 console, expand **Instances** in the navigation pane, and then choose **Instances**. In the list of instances, select the box next to the instance. Security groups for that instance appear in the **Description** tab next to **Security groups**.
To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-security-groups` command, for example as follows.

```
aws ec2 describe-instances --output table --query 'Reservations[*].Instances[*].NetworkInterfaces[*].Groups[*].GroupId' --region us-east-2 --instance-ids i-12a3c456d7890123
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the instance, and replace `i-12a3c456d7890123` with the instance ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

The output contains the list of security group IDs for that instance.

**View or Change Settings For a Security Group in a VPC**

To use the Amazon VPC console, choose **Security Groups** in the navigation pane. Select the box next to the security group. To see the settings, look at each of the tabs. To change a setting on a tab, choose **Edit** if applicable, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell to see the settings, run the Amazon EC2 `describe-security-groups` command, for example as follows.

```
aws ec2 describe-security-groups --output table --region us-east-2 --group-ids sg-12a3b456
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the instance, and replace `sg-12a3b456` with the security group ID. To run the preceding command with the aws-shell, omit `aws`.

**View or Change Settings For a Security Group For an AWS Cloud Compute Instance**

To use the Amazon EC2 console, expand **Instances** in the navigation pane, and then choose **Instances**. In the list of instances, select the box next to the instance. In the **Description** tab, for **Security groups**, choose the security group. Look at each of the tabs. To change a setting on a tab, choose **Edit** if applicable, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell to see the settings, run the Amazon EC2 `describe-security-groups` command, for example as follows.

```
aws ec2 describe-security-groups --output table --region us-east-2 --group-ids sg-12a3b456
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the instance, and replace `sg-12a3b456` with the security group ID. To run the preceding command with the aws-shell, omit `aws`.

**Minimum Inbound and Outbound Traffic Settings for AWS Cloud9**

- **Inbound**: All IP addresses using SSH over port 22. However, you can restrict these IP addresses to only those that AWS Cloud9 uses. For more information, see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).
Note
For EC2 environments created on or after July 31 2018, AWS Cloud9 uses security groups to automatically restrict inbound IP addresses using SSH over port 22 to only those addresses that AWS Cloud9 uses. For more information, see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).

- **Inbound (network ACLs only):** For EC2 environments, and for SSH environments associated with Amazon EC2 instances running Amazon Linux or Ubuntu Server, all IP addresses using TCP over ports 32768-61000. For more information, and for port ranges for other Amazon EC2 instance types, see Ephemeral Ports in the Amazon VPC User Guide.

- **Outbound:** All traffic sources using any protocol and port.

You can set this behavior at the security group level. For an additional level of security, you can also use a network ACL. For more information, see Comparison of Security Groups and Network ACLs in the Amazon VPC User Guide.

For example, to add inbound and outbound rules to a security group, you could set up those rules as follows.

Inbound rules:

<table>
<thead>
<tr>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH (22)</td>
<td>TCP (6)</td>
<td>22</td>
<td>0.0.0.0 (But see the following note and Inbound SSH IP address ranges for AWS Cloud9 (p. 483).)</td>
</tr>
</tbody>
</table>

Note
For EC2 environments created on or after July 31 2018, AWS Cloud9 automatically adds an inbound rule to restrict inbound IP addresses using SSH over port 22 to only those addresses that AWS Cloud9 uses. For more information, see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).

Outbound rules:

<table>
<thead>
<tr>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL Traffic</td>
<td>ALL</td>
<td>ALL</td>
<td>0.0.0.0/0</td>
</tr>
</tbody>
</table>

If you also choose to add inbound and outbound rules to a network ACL, you could set up those rules as follows.

Inbound rules:

<table>
<thead>
<tr>
<th>Rule #</th>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
<th>Allow / Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>SSH (22)</td>
<td>TCP (6)</td>
<td>22</td>
<td>0.0.0.0 (But see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).)</td>
<td>ALLOW</td>
</tr>
<tr>
<td>Rule #</td>
<td>Type</td>
<td>Protocol</td>
<td>Port Range</td>
<td>Source</td>
<td>Allow / Deny</td>
</tr>
<tr>
<td>-------</td>
<td>---------------</td>
<td>----------</td>
<td>-----------------------------------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>200</td>
<td>Custom TCP</td>
<td>TCP (6)</td>
<td>32768-61000 (For Amazon Linux and Ubuntu Server instances. For other instance types, see Ephemeral Ports.)</td>
<td>0.0.0.0/0</td>
<td>ALLOW</td>
</tr>
<tr>
<td>*</td>
<td>ALL Traffic</td>
<td>ALL</td>
<td>ALL</td>
<td>0.0.0.0/0</td>
<td>DENY</td>
</tr>
</tbody>
</table>

**Outbound rules:**

<table>
<thead>
<tr>
<th>Rule #</th>
<th>Type</th>
<th>Protocol</th>
<th>Port Range</th>
<th>Source</th>
<th>Allow / Deny</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>ALL Traffic</td>
<td>ALL</td>
<td>ALL</td>
<td>0.0.0.0/0</td>
<td>ALLOW</td>
</tr>
<tr>
<td>*</td>
<td>ALL Traffic</td>
<td>ALL</td>
<td>ALL</td>
<td>0.0.0.0/0</td>
<td>DENY</td>
</tr>
</tbody>
</table>

For more information about security groups and network ACLs, see the following in the *Amazon VPC User Guide*.

- Security
- Security Groups for your VPC
- Network ACLs

### Create a Security Group in a VPC

To use the Amazon VPC or Amazon EC2 consoles, do one of the following.

- In the Amazon VPC console, choose **Security Groups** in the navigation pane. Choose **Create Security Group**, and then follow the on-screen directions.
- In the Amazon EC2 console, expand **Network & Security** in the navigation pane, and then choose **Security Groups**. Choose **Create Security Group**, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `create-security-group` command, for example as follows.

```
aws ec2 create-security-group --region us-east-2 --vpc-id vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command with the aws-shell, omit `aws`.

### Confirm Whether a VPC Has at Least One Network ACL

To use the Amazon VPC console, choose **Your VPCs** in the navigation pane. Select the box next to the VPC you want AWS Cloud9 to use. On the **Summary** tab, if there is a value for **Network ACL**, the VPC has at least one network ACL.
To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-network-acls` command, for example as follows.

```bash
aws ec2 describe-network-acls --output table --query
  'NetworkAcls[*].Associations[*].NetworkAclId' --region us-east-2 --filters Name=vpc-id,Values=vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

If the output contains at least one entry in the list, the VPC has at least one network ACL.

### View a List of Network ACLs For a VPC

To use the Amazon VPC console, choose **Network ACLs** in the navigation pane. In the **Search Network ACLs** box, type the VPC's ID or name, and then press Enter. Network ACLs for that VPC appear in the list of search results.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `describe-network-acls` command, for example as follows.

```bash
aws ec2 describe-network-acls --output table --query
  'NetworkAcls[*].Associations[*].NetworkAclId' --region us-east-2 --filters Name=vpc-id,Values=vpc-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the VPC, and replace `vpc-1234ab56` with the VPC ID. To run the preceding command in Windows, replace the single quotes with double quotes. To run the preceding command with the aws-shell, omit `aws`.

The output contains a list of network ACLs for that VPC.

### View or Change Settings For a Network ACL

To use the Amazon VPC console, choose **Network ACLs** in the navigation pane. Select the box next to the network ACL. To see the settings, look at each of the tabs. To change a setting on a tab, choose **Edit** if applicable, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell to see the settings, run the Amazon EC2 `describe-network-acls` command, for example as follows.

```bash
aws ec2 describe-network-acls --output table --region us-east-2 --network-acl-ids acl-1234ab56
```

In the preceding command, replace `us-east-2` with the AWS Region that contains the network ACL, and replace `acl-1234ab56` with the network ACL ID. To run the preceding command with the aws-shell, omit `aws`.

### Create a Network ACL

To use the Amazon VPC console, choose **Network ACLs** in the navigation pane. Choose **Create Network ACL**, and then follow the on-screen directions.

To use the AWS CLI or the aws-shell, run the Amazon EC2 `create-network-acl` command, for example as follows.
Create an Amazon VPC for AWS Cloud9

You can use the Amazon VPC console to create an Amazon VPC that is compatible with AWS Cloud9.

Note
For this procedure, we recommend you sign in to the AWS Management Console and open the Amazon VPC console using credentials for an IAM administrator user in your AWS account. If you can't do this, check with your AWS account administrator. Some organizations may not allow you to create VPCs on your own. If you cannot create a VPC, check with your AWS account administrator or network administrator.

1. If the Amazon VPC console isn't already open, sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc.
2. In the navigation bar, if the AWS Region isn't the same as the environment, choose the correct AWS Region.
3. Choose VPC Dashboard in the navigation pane, if the VPC Dashboard page isn't already displayed.
5. For Step 1: Select a VPC Configuration, with VPC with a Single Public Subnet already selected, choose Select.
6. For Step 2: VPC with a Single Public Subnet, we recommend that you leave the following default settings. (However, you can change the CIDR settings if you have custom CIDRs you want to use. For more information, see VPC and Subnet Sizing in the Amazon VPC User Guide.)
   - IPv4 CIDR block: 10.0.0.0/16
   - IPv6 CIDR block: No IPv6 CIDR Block
   - Public subnet's IPv4 CIDR: 10.0.0.0/24
   - Availability Zone: No Preference
   - Enable DNS hostnames: Yes
   - Hardware tenancy: Default
7. For VPC name, type a name for the VPC.
8. For Subnet name, type a name for the subnet in the VPC.
9. Choose Create new VPC.

Amazon VPC creates the following resources that are compatible with AWS Cloud9:

- A VPC.
- A public subnet for the VPC.
- A route table for the public subnet with the minimum required settings.
- An internet gateway for the public subnet.
- A network ACL for the public subnet with the minimum required settings.

10. By default, the VPC allows incoming traffic from all types, protocols, ports, and IP addresses. You can restrict this behavior to allow only IP addresses coming from AWS Cloud9 using SSH over port 22. One approach is to set incoming rules on the VPC's default network ACL, as follows.
   a. In the navigation pane of the Amazon VPC console, choose Your VPCs.
   b. Select the box for the VPC you just created.
c. On the Description tab, choose the link next to Network ACL.
d. Select the box next to the network ACL that is displayed.
e. On the Inbound Rules tab, choose Edit inbound rules.
f. For Rule # 100, for Type, choose SSH (22).
g. For Source, type one of the CIDR blocks in the Inbound SSH IP address ranges for AWS Cloud9 (p. 483) list that matches the AWS Region for this VPC.
h. Choose Add another rule.
i. For Rule #, type 200.
j. For Type, choose SSH (22).
k. For Source, type the other CIDR block in the Inbound SSH IP address ranges for AWS Cloud9 (p. 483) list that matches the AWS Region for this VPC.

At minimum, you must also allow incoming traffic from all IP addresses using TCP over ports 32768-61000 for Amazon Linux and Ubuntu Server instance types. (For background, and for port ranges for other Amazon EC2 instance types, see Ephemeral Ports in the Amazon VPC User Guide). To do this, choose Add another rule.
m. For Rule #, type 300.
n. For Type, choose Custom TCP Rule.
o. For Port Range, type 32768–61000 (for Amazon Linux and Ubuntu Server instance types).
p. For Source, type 0.0.0.0/0.
q. Choose Save.
r. You might need to add more inbound or outbound rules to the network ACL, depending on how you plan to use AWS Cloud9. See the documentation for the web services or APIs you want to allow to communicate into or out of the VPC for the Type, Protocol, Port Range, and Source values to specify for these rules.

Create a Subnet for AWS Cloud9

You can use the Amazon VPC console to create a subnet for a VPC that is compatible with AWS Cloud9.

If you followed the previous procedure to create a VPC for AWS Cloud9, you do not also need to follow this procedure. This is because the Create new VPC wizard creates a subnet for you automatically.

Important

- The AWS account must already have a compatible VPC in the same AWS Region for the environment. For more information, see the VPC requirements in Amazon VPC Requirements for AWS Cloud9 (p. 467).
- For this procedure, we recommend you sign in to the AWS Management Console, and then open the Amazon VPC console using credentials for an IAM administrator user in your AWS account. If you can't do this, check with your AWS account administrator.
- Some organizations may not allow you to create subnets on your own. If you cannot create a subnet, check with your AWS account administrator or network administrator.

1. If the Amazon VPC console isn't already open, sign in to the AWS Management Console and open the Amazon VPC console at https://console.aws.amazon.com/vpc.
2. In the navigation bar, if the AWS Region isn't the same as the AWS Region for the environment, choose the correct AWS Region.
3. Choose Subnets in the navigation pane, if the Subnets page isn't already displayed.
4. Choose Create Subnet.
5. In the Create Subnet dialog box, for Name tag, type a name for the subnet.
6. For VPC, choose the VPC to associate the subnet with.
7. For Availability Zone, choose the Availability Zone within the AWS Region for the subnet to use, or choose No Preference to let AWS choose an Availability Zone for you.
8. For IPv4 CIDR block, type the range of IP addresses for the subnet to use, in CIDR format. This range of IP addresses must be a subset of IP addresses in the VPC.

For information about CIDR blocks, see VPC and Subnet Sizing in the Amazon VPC User Guide. See also 3.1. Basic Concept and Prefix Notation in RFC 4632 or IPv4 CIDR blocks in Wikipedia.

9. After you create the subnet, be sure to associate it with a compatible route table and an internet gateway, as well as security groups, a network ACL, or both. For more information, see the requirements in Amazon VPC Requirements for AWS Cloud9 (p. 467).

AWS Cloud9 SSH Development Environment host requirements

To instruct AWS Cloud9 to connect an environment to an existing cloud compute instance or your own server, you create an AWS Cloud9 SSH development environment. However, before you create an SSH environment, you should consider the benefits of creating EC2 environments instead.

When you create an EC2 environment, AWS Cloud9 creates a new environment, requests Amazon EC2 to launch a new instance, and then connects the newly launched instance to the new environment. Creating an EC2 environment has the following benefits:

- **Automatic instance launching.** When you create an EC2 environment, AWS Cloud9 requests Amazon EC2 to create a new instance at the same time. In an SSH environment, you must provide an existing cloud compute instance (for example an Amazon EC2 instance) or your own server yourself.

- **Automatic instance shutdown.** By default, AWS Cloud9 automatically shuts down the EC2 environment 30 minutes after all web browser instances that are connected to the IDE for the EC2 environment are closed. (You can change this behavior at any time.) This helps reduce additional charges to your AWS account for using Amazon EC2.

- **Automatic instance cleanup.** When you delete an EC2 environment, the connected Amazon EC2 instance is automatically deleted. This also helps reduce additional charges to your AWS account for using Amazon EC2. In an SSH environment that is connected to cloud compute instance, you must remember to delete the instance yourself.

If you want to create an EC2 environment instead, see Creating an EC2 Environment (p. 56). Otherwise, continue reading for information about creating SSH environments.

**When and how to create an SSH Environment**

You must create an SSH environment instead of an EC2 environment whenever any of the following is true:

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Directions</th>
</tr>
</thead>
<tbody>
<tr>
<td>You don't want to incur additional charges to your AWS account for using AWS cloud compute instances, so you decide to connect AWS Cloud9 to an existing cloud compute instance outside of AWS or your own server instead.</td>
<td>1. Be sure your instance or server meets the requirements (p. 480) later in this topic. 2. Create an SSH environment (p. 55) for AWS Cloud9 to connect your instance or server to.</td>
</tr>
<tr>
<td>Requirement</td>
<td>Directions</td>
</tr>
<tr>
<td>----------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| You want to use an existing AWS cloud compute instance (for example an Amazon EC2 instance) in your AWS account instead of having AWS Cloud9 to launch a new instance at the same time the environment is created. | 1. Be sure the instance meets the requirements (p. 480) later in this topic.  
2. Create an SSH environment (p. 55) for AWS Cloud9 to connect the instance to. |
| You want to use an Amazon EC2 instance type that AWS Cloud9 currently doesn't support for an EC2 environment (for example, R4). | 1. Launch an Amazon EC2 instance based on the desired instance type. Or identify an existing instance in your AWS account that runs the desired instance type.  
2. Be sure the instance meets the requirements (p. 480) later in this topic.  
3. Create an SSH environment (p. 55) for AWS Cloud9 to connect the instance to. |
| You want to use an Amazon EC2 instance that is based on an Amazon Machine Image (AMI) other than Amazon Linux or Ubuntu Server. | 1. Launch an Amazon EC2 instance based on the desired AMI. Or identify an existing instance in your AWS account that is based on the desired AMI.  
2. Be sure the instance meets the requirements (p. 480) later in this topic.  
3. Create an SSH environment (p. 55) for AWS Cloud9 to connect the instance to. |
| You want to connect multiple environments to a single existing cloud compute instance or your own server. | 1. Be sure the instance or server meets the requirements (p. 480) later in this topic.  
2. Create an SSH environment (p. 55) for each environment you want AWS Cloud9 to connect the instance or server to. |

**Note**

Launching an Amazon EC2 instance might result in possible charges to your AWS account for Amazon EC2. For more information, see [Amazon EC2 Pricing](https://aws.amazon.com/ec2/pricing/).

**SSH host requirements**

The existing cloud compute instance or your own server must meet the following requirements for AWS Cloud9 to connect it to an SSH environment.

- It must run Linux. (AWS Cloud9 doesn't support Windows.)
- It must be reachable over the public internet by using SSH. If it is reachable only through a virtual private cloud (VPC) or virtual private network (VPN), that VPC or VPN must have access to the public internet.
- If the host is an existing AWS cloud compute instance that is part of an Amazon Virtual Private Cloud (Amazon VPC), there are additional requirements. See [Amazon VPC Settings](https://docs.aws.amazon.com/vpc/latest/userguide/vpc-settings.html) (p. 467).
- It must have Python installed, and the version must be 2.7. To check the version, from the existing instance's or server's terminal, run the command `python --version`. To install Python 2.7 on the instance or server, see one of the following:
  - Step 1: Install Required Tools (p. 419) in the [Python Sample](https://s3.amazonaws.com/aws-cloud9-samples/sample.html).
  - Download Python from the Python website and see [Installing Packages](https://docs.python.org/2/howto/logging.html) in the [Python Packaging User Guide](https://docs.python.org/2/howto/distribute.html).
Note
To connect to an existing AWS cloud compute instance to verify and meet requirements, see one or more of the following resources:

- For Amazon EC2, see Connect to Your Linux Instance in the Amazon EC2 User Guide for Linux Instances.
- For Amazon LightSail, see Connect to your Linux/Unix-based Lightsail instance in the Amazon LightSail Documentation.
- For AWS Elastic Beanstalk, see Listing and Connecting to Server Instances in the AWS Elastic Beanstalk Developer Guide.
- For AWS OpsWorks, see Using SSH to Log In to a Linux Instance in the AWS OpsWorks User Guide.
- For other AWS services, see the service’s documentation.

To connect to your own server to verify and meet requirements, you could search the internet using a phrase such as “connect to a server by using the ssh command” (from macOS or Linux) or “connect to a server by using PuTTY” (from Windows).

- It must have Node.js installed. We currently support versions from Node.js 0.6.16 to Node.js 12.x.

Warning
AWS Cloud9 installation problems may occur when creating an SSH environment if you use a Node.js version that’s not supported by AWS Cloud9.

To check the version, from the existing instance’s or server’s terminal, run the command `node --version`. To install Node.js on the instance or server, see one of the following:

- Step 1: Install Required Tools (p. 431) in the Node.js Sample.
- Installing Node.js via package manager on the Node.js website.
- Node Version Manager on GitHub.

The path to the directory on the existing instance or server that you want AWS Cloud9 to start from after login must have its access permissions set to `rwxr-xr-x`. This means read-write-execute permissions for the owner that corresponds to the login name that you specify in the create environment wizard (p. 65) for User on the Configure settings page, read-execute permissions for the group that this owner belongs to, and read-execute permissions for others.

For example, if the directory’s path is `~` (where `~` represents the home directory for the login name that you specify for User on the Configure settings page), you can set these permissions on the directory by running the `chmod` command on the instance or server, as follows.

```
sudo chmod u=rwx,g=rx,o=rx ~
```

- Download and run the AWS Cloud9 Installer (p. 482) on the existing instance or server.
- Optionally, you can restrict inbound traffic over SSH to only the IP addresses that AWS Cloud9 uses. To do this, set inbound SSH traffic to the IP ranges as described in Inbound SSH IP address ranges for AWS Cloud9 (p. 483).

After you are sure your instance or server meets the preceding requirements, create an SSH environment (p. 65) for AWS Cloud9 to connect it to.

## Using the AWS Cloud9 Installer

Before you create an AWS Cloud9 SSH development environment, the cloud compute instance (for example an Amazon EC2 instance) or your own server that will connect to the environment must meet the SSH Host Requirements (p. 480). One of these requirements is to download and run the AWS Cloud9 Installer on the instance or server. The AWS Cloud9 Installer is a Linux shell script that checks
whether the instance or server is running on an operating system platform and architecture that AWS Cloud9 supports. If this check succeeds, the script then attempts to install components and their dependencies that AWS Cloud9 requires to be on the instance or server.

This topic describes how to download and run this installer script on the target instance or server.

- Download and Run the AWS Cloud9 Installer (p. 482)
- Troubleshooting the AWS Cloud9 Installer (p. 482)

**Download and Run the AWS Cloud9 Installer**

1. Make sure the cloud compute instance or your own server that will connect to the environment meets the SSH Host Requirements (p. 480). This includes having specific versions of Python and Node.js already installed; setting specific permissions on the directory that you want AWS Cloud9 to start from after login; and setting up any associated Amazon Virtual Private Cloud.

2. While you are connected to the instance or server, run one of the following commands on that instance or server.

   ```bash
```

3. If a Done message displays with no errors, you can create the SSH environment (p. 65).
   
   If an error message displays, see the next section for troubleshooting information.

**Troubleshooting the AWS Cloud9 Installer**

This section describes common issues, possible causes, and recommended solutions for troubleshooting AWS Cloud9 Installer errors.

If your issue is not listed, or if you need additional help, see the AWS Cloud9 Discussion Forum. (When you enter this forum, AWS might require you to sign in.) You can also contact us directly.

- -bash: wget: command not found (p. 482)
- Error: please install make to proceed (p. 482)
- Error: please install gcc to proceed (p. 483)
- configure: error: curses not found (p. 483)

- **-bash: wget: command not found**

  **Issue:** When you run the installer script, the following message displays: -bash: wget: command not found.

  **Possible cause:** The `wget` utility is not installed on the instance or server.

  **Recommended solution:** Run the installer script on the instance or server with the `curl` utility instead.

- **Error: please install make to proceed**

  **Issue:** When you run the installer script, the following message displays: Error: please install make to proceed.

  **Possible cause:** The `make` utility is not installed on the instance or server.
Recommended solution: Install the `make` utility, and then try running the installer script on the instance or server again.

To install the `make` utility, you could run a command on the instance or server such as the following.

- For Amazon Linux, Amazon Linux 2, and Red Hat Enterprise Linux (RHEL) running in Amazon EC2: `sudo yum -y groupinstall "Development Tools"
- For Ubuntu Server running in Amazon EC2: `sudo apt install -y build-essential`
- For SUSE: `sudo zypper install -y make`

**Error: please install gcc to proceed**

**Issue:** When you run the installer script, the following message displays: Error: please install gcc to proceed.

**Possible cause:** The `gcc` utility is not installed on the instance or server.

**Recommended solution:** Install the `gcc` utility, and then try running the installer script on the instance or server again.

To install the `gcc` utility, you could run a command on the instance or server such as the following.

- For Amazon Linux, Amazon Linux 2, and Red Hat Enterprise Linux (RHEL) running in Amazon EC2: `sudo yum -y groupinstall "Development Tools"
- For Ubuntu Server running in Amazon EC2: `sudo apt install -y build-essential`
- For SUSE: `sudo zypper install -y gcc`
- For other operating systems, see Installing GCC.

**configure: error: curses not found**

**Issue:** When you run the installer script, the following message displays: configure: error: curses not found.

**Possible cause:** The `ncurses` terminal control library is not installed on the instance or server.

**Recommended solution:** Install the `ncurses` terminal control library (and, on some operating systems, the `glibc-static` library), and then try running the installer script on the instance or server again.

To install the `ncurses` terminal control library (and, on some operating systems, the `glibc-static` library), you could run commands on the instance or server such as the following.

- For Amazon Linux, Amazon Linux 2, and Red Hat Enterprise Linux (RHEL) running in Amazon EC2: `sudo yum -y install ncurses-devel`
- For SUSE: `sudo zypper install -y ncurses-devel` and `sudo zypper install -y glibc-static`

**Inbound SSH IP address ranges for AWS Cloud9**

You can restrict incoming traffic to just the IP address ranges that AWS Cloud9 uses to connect over SSH to AWS cloud compute instances (for example Amazon EC2 instances) in an Amazon VPC or your own servers in your network.

**Note**
For an EC2 environment created on or after July 31 2018, you can skip this topic. This is because AWS Cloud9 automatically restricts inbound SSH traffic for that environment to only those IP
addresses that are described later in this topic. AWS Cloud9 does this by automatically adding a rule to the security group that is associated with the Amazon EC2 instance for the environment. This rule restricts inbound SSH traffic over port 22 to only those IP addresses for the associated AWS Region.

IP address ranges for most AWS Regions are in the `ip-ranges.json` file, as described in AWS IP Address Ranges in the AWS General Reference.

**Note**
See below (p. 484) for IP address ranges for the Asia Pacific (Hong Kong) and Middle East (Bahrain) Regions that are not currently included in the `ip-ranges.json` file.

To find the IP ranges in the `ip-ranges.json` file:

- For Windows, using the AWS Tools for Windows PowerShell, run the following command.

  ```powershell
  Get-AWSPublicIpAddressRange -ServiceKey CLOUD9
  ```

- For Linux, download the `ip-ranges.json` file. Then you can query it by using a tool such as `jq`, for example, by running the following command.

  ```bash
  jq '.prefixes[] | select(.service=="CLOUD9")' < ip-ranges.json
  ```

These IP ranges might change occasionally. Whenever there's a change, we send notifications to subscribers of the AmazonIpSpaceChanged topic. To get these notifications, see AWS IP Address Ranges Notifications in the AWS General Reference.

To use these IP address ranges when configuring environments that use AWS cloud compute instances, see VPC Settings for AWS Cloud9 Development Environments (p. 467). Also, if you choose to restrict incoming traffic for EC2 environments, or for SSH environments associated with Amazon EC2 instances running Amazon Linux or Ubuntu Server, be sure to also allow at minimum all IP addresses using TCP over ports 32768-61000. For more information, and port ranges for other AWS cloud compute instance types, see Ephemeral ports in the Amazon VPC User Guide.

To use these IP address ranges when configuring SSH environments that use your own network, see your network's documentation or your network administrator.

### IP addresses not in `ip-ranges.json`

AWS Cloud9 IP address ranges for the following AWS Regions are currently not provided in the `ip-ranges.json` file: Asia Pacific (Hong Kong) and Middle East (Bahrain). The following table lists the IP ranges for those Regions.

**Note**
Each Region has two IP address ranges to support the AWS Cloud9 control plane (information routing) and data plane (information processing) services.

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>Code</th>
<th>IP ranges (CIDR notation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Hong Kong)</td>
<td>ap-east1</td>
<td>18.163.201.96/27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>18.163.139.32/27</td>
</tr>
<tr>
<td>Middle East (Bahrain)</td>
<td>me-south-1</td>
<td>15.185.141.160/27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.185.91.32/27</td>
</tr>
</tbody>
</table>
Amazon Machine Image (AMI) Contents for an AWS Cloud9 EC2 Development Environment

Use the following information to get details about Amazon Machine Images (AMIs) that AWS Cloud9 uses for an EC2 environment.

**Topics**
- Amazon Linux (p. 485)
- Ubuntu Server (p. 486)

**Amazon Linux**

To display the version of an Amazon Linux instance, run the following command from the AWS Cloud9 IDE for the connected environment or from an SSH utility such as the `ssh` command or PuTTY.

```
cat /etc/system-release
```

To display a list of packages that are installed on an Amazon Linux instance, run one or more of the following commands.

To display all installed packages as a single list:

```
sudo yum list installed
```

To display a list of installed packages with package names containing the specified text:

```
sudo yum list installed | grep YOUR_SEARCH_TERM
```

In the preceding command, replace `YOUR_SEARCH_TERM` with some portion of the package name. For example, to display a list of all installed packages with names containing `sql`:

```
sudo yum list installed | grep sql
```

To display a list of all installed packages, displayed one page at a time:

```
sudo yum list installed | less
```

To scroll through the displayed pages:
- To move down a line, press `j`.
- To move up a line, press `k`.
- To move down a page, press `Ctrl-F`.
- To move up a page, press `Ctrl-B`.
- To quit, press `q`.

For additional options, run the `man yum` command. See also Amazon Linux AMI 2018.03 Packages on the Amazon Linux AMI website.
Ubuntu Server

To display the version of an Ubuntu Server instance, run the following command from the AWS Cloud9 IDE for the connected environment or from an SSH utility such as the `sshd` command or PuTTY.

```bash
lsb_release -a
```

The version will display next to the Description field.

To display a list of packages that are installed on an Ubuntu Server, run one or more of the following commands.

To display all installed packages as a single list:

```bash
sudo apt list --installed
```

To display a list of installed packages with package names containing the specified text:

```bash
sudo apt list --installed | grep YOUR_SEARCH_TERM
```

In the preceding command, replace `YOUR_SEARCH_TERM` with some portion of the package name. For example, to display a list of all installed packages with names containing `sql`:

```bash
sudo apt list --installed grep sql
```

To display a list of all installed packages, one page at a time:

```bash
sudo apt list --installed | less
```

To scroll through the displayed pages:

- To move down a line, press `j`.
- To move up a line, press `k`.
- To move down a page, press `Ctrl-F`.
- To move up a page, press `Ctrl-B`.
- To quit, press `q`.

For additional options, run the `man apt` command. See also Ubuntu Packages Search on the Ubuntu website.

Access Permissions Reference for AWS Cloud9

Access to AWS Cloud9 requires AWS access credentials. Those credentials must have permissions to do things such as create, share, or delete an AWS Cloud9 development environment. The following sections describe how you can use AWS Identity and Access Management (IAM) to allow or deny access to your AWS Cloud9 resources and then map those permissions to credentials.

**Topics**

- Overview (p. 487)
- AWS Managed (Predefined) Policies for AWS Cloud9 (p. 490)
Overview

This section provides an overview of the IAM authentication and access control model that applies to AWS Cloud9.

**Note**
If you just want to set up predefined sets of access permissions for common usage scenarios and user types, skip ahead to AWS Managed (Predefined) Policies for AWS Cloud9 (p. 490).

**Topics**
- Authentication (p. 487)
- Access Control (p. 488)
- AWS Cloud9 Resources and Operations (p. 488)
- Understanding Resource Ownership (p. 489)
- Managing Access to Resources (p. 489)

**Authentication**

You can access AWS as any of the following types of identities:

**AWS account root user**

When you sign up for AWS, you provide an email address and password that is associated with your AWS account. These are your root credentials, and they provide complete access to all of your AWS resources.

**Important**
As an AWS security best practice, we recommend that you use the root credentials only to create an IAM administrator group with an IAM administrator user. This is a group that gives the user full permissions to your AWS account. Then you can use this administrator user to create other IAM users and roles with limited permissions. For more information, see Create Individual IAM Users and Creating Your First IAM Admin User and Group in the IAM User Guide.

**IAM user**

An IAM user is simply an identity within your AWS account that has specific custom permissions (for example, permissions to create an AWS Cloud9 development environment). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Cloud9 console, AWS Management Console, AWS Discussion Forums, and AWS Support Support Center.

In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several AWS SDKs or by using the AWS Command Line Interface (AWS CLI) or the aws-shell. The AWS SDKs, the AWS CLI, and the aws-shell use these access keys to cryptographically sign your request. If you don't use these tools, you must sign the request yourself. AWS Cloud9 supports Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the Amazon Web Services General Reference.

**IAM role**

An IAM role is another IAM identity you can create in your account that has specific permissions. It's similar to an IAM user, but it isn't associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations:
AWS service access

You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows AWS Lambda to access an Amazon S3 bucket on your behalf, and then load data stored in the bucket into an Amazon Redshift. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the IAM User Guide.

Applications running on Amazon EC2

Instead of storing access keys within an Amazon EC2 instance for use by applications running on the instance and making AWS API requests, you can use an IAM role to manage temporary credentials for these applications. To assign an AWS role to an Amazon EC2 instance and make it available to all of its applications, you can create an instance profile that is attached to the instance. An instance profile contains the role and enables programs running on the Amazon EC2 instance to get temporary credentials. For more information, see Create and Use an Instance Profile to Manage Temporary Credentials and Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances in the IAM User Guide.

Note
Instead of attaching an instance profile to an Amazon EC2 instance that connects to an environment, AWS Cloud9 can automatically set up and manage temporary credentials on your behalf in an EC2 environment. For more information, see AWS Managed Temporary Credentials (p. 501).

Federated user access

Instead of creating an IAM user, you can use pre-existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information, see Federated Users and Roles in the IAM User Guide.

Access Control

You can have valid credentials to authenticate your requests, but unless you have permissions, you cannot create or access AWS Cloud9 resources. For example, you must have permissions to create, share, or delete an AWS Cloud9 development environment.

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies. An account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles).

When you grant permissions, you decide who is getting the permissions, the resources they can access, and the actions that can be performed on those resources.

AWS Cloud9 Resources and Operations

In AWS Cloud9, the primary resource is an AWS Cloud9 development environment. In a policy, you use an Amazon Resource Name (ARN) to identify the resource that the policy applies to. The following table lists environment ARNs. For more information, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the Amazon Web Services General Reference.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>ARN format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every environment owned by the specified account in the specified region</td>
<td>arn:aws:cloud9:REGION_ID:ACCOUNT_ID:environment:*</td>
</tr>
</tbody>
</table>
### Resource type | ARN format
---|---
Every environment owned by the specified account in the specified region | `arn:aws:cloud9:REGION_ID:ACCOUNT_ID:*`
Every AWS Cloud9 resource, regardless of account and region | `arn:aws:cloud9:*`

For example, you can indicate a specific environment in your statement using its ARN, as follows.

```
"Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:70d899206236474f9590d93b7c41dfEX"
```

To specify all resources, use the wildcard character (`*`) in the `Resource` element, as follows.

```
"Resource": "*"
```

To specify multiple resources in a single statement, separate their ARNs with commas, as follows.

```
"Resource": [  
  "arn:aws:cloud9:us-east-2:123456789012:environment:70d899206236474f9590d93b7c41dfEX",  
  "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
]
```

AWS Cloud9 provides a set of operations to work with AWS Cloud9 resources. For a list, see the AWS Cloud9 Permissions Reference (p. 500).

### Understanding Resource Ownership

The AWS account owns the resources that are created in the account, regardless of who created the resources. For example:

- If you use the root account credentials of your AWS account to create an AWS Cloud9 development environment (which, although possible, is not recommend as an AWS security best practice), your AWS account is the owner of the environment.
- If you create an IAM user in your AWS account and grant permissions to create an environment to that user, the user can create an environment. However, your AWS account, to which the user belongs, owns the environment.
- If you create an IAM role in your AWS account with permissions to create an environment, anyone who can assume the role can create an environment. Your AWS account, to which the role belongs, owns the environment.

### Managing Access to Resources

A permissions policy describes who has access to which resources.

#### Note

This section discusses the use of IAM in AWS Cloud9. It doesn't provide detailed information about the IAM service. For complete IAM documentation, see What Is IAM? in the IAM User Guide. For information about IAM policy syntax and descriptions, see the IAM JSON Policy Reference in the IAM User Guide.

Policies attached to an IAM identity are referred to as identity-based policies (or IAM policies). Policies attached to a resource are referred to as resource-based policies. AWS Cloud9 supports both identity-based and resource-based policies.
Each of the following API actions requires only an IAM policy to be attached to the IAM identity who wants to call these API actions.

- CreateEnvironmentEC2
- DescribeEnvironments

The following API actions require a resource-based policy. An IAM policy isn't required, but AWS Cloud9 will use an IAM policy if it is attached to the IAM identity who wants to call these API actions. The resource-based policy must be applied to the desired AWS Cloud9 resource.

- CreateEnvironmentMembership
- DeleteEnvironment
- DeleteEnvironmentMembership
- DescribeEnvironmentMemberships
- DescribeEnvironmentStatus
- UpdateEnvironment
- UpdateEnvironmentMembership

For details on what each of these API actions do, see the AWS Cloud9 API Reference.

You cannot attach a resource-based policy to an AWS Cloud9 resource directly. Instead, AWS Cloud9 attaches the appropriate resource-based policies to AWS Cloud9 resources as you add, modify, update, or delete environment members.

To grant a user permissions to perform actions on AWS Cloud9 resources, you attach a permissions policy to an IAM group that the user belongs to. We recommend you attach an AWS managed (predefined) policy for AWS Cloud9 whenever possible. AWS managed policies are easier and faster to attach. They also contain predefined sets of access permissions for common usage scenarios and user types, such as full administration of an environment, environment users, and users who have only read-only access to an environment. For a list of AWS managed policies for AWS Cloud9, see AWS Managed (Predefined) Policies for AWS Cloud9 (p. 490).

For more detailed usage scenarios and unique user types, you can create and attach your own customer-managed policies. See Additional Setup Options for AWS Cloud9 (Team and Enterprise) (p. 22) and Creating Customer-Managed Policies for AWS Cloud9 (p. 493).

To attach an IAM policy (AWS managed or customer-managed) to an IAM identity, see Attaching IAM Policies (Console) in the IAM User Guide.

AWS Managed (Predefined) Policies for AWS Cloud9

AWS addresses many common use cases by providing standalone IAM policies that AWS creates and administers. These AWS managed policies grant necessary permissions for common use cases so you can avoid having to investigate what permissions are needed. For example, you can use AWS managed policies for AWS Cloud9 to quickly and easily allow users to have full administration of an AWS Cloud9 development environment, act as an environment user, or use an environment they are added to. For more information, see AWS Managed Policies in the IAM User Guide.

To attach an AWS managed policy to an IAM identity, see Attaching IAM Policies (Console) in the IAM User Guide.

The following AWS managed policies, which you can attach to IAM identities in your account, are specific to AWS Cloud9.
• **AWSCloud9Administrator**: Provides the following permissions:
  - Amazon EC2: get information about multiple Amazon VPC and subnet resources in their AWS account.
  - AWS Cloud9: all AWS Cloud9 actions in their AWS account.
  - IAM: get information about IAM users in their AWS account, and create the AWS Cloud9 service-linked role in their AWS account as needed.

  The **AWSCloud9Administrator** managed policy contains the following permissions:

  ```json
  {
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
    "cloud9:*",
    "ec2:DescribeSubnets",
    "ec2:DescribeVpcs",
    "iam:GetUser",
    "iam:ListUsers"
    ],
    "Resource": "*"
  },
  {
    "Effect": "Allow",
    "Action": [
    "iam:CreateServiceLinkedRole"
    ],
    "Resource": "*",
    "Condition": {
    "StringLike": {
    "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
    }
  }
  ]
  }
  ```

• **AWSCloud9User**: Provides the following permissions:
  - Amazon EC2: get information about multiple Amazon VPC and subnet resources in their AWS account.
  - AWS Cloud9: create and get information about their environments, and get and change user settings for their environments.
  - IAM: get information about IAM users in their AWS account, and create the AWS Cloud9 service-linked role in their AWS account as needed.

  The **AWSCloud9User** managed policy contains the following permissions:

  ```json
  {
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
    "cloud9:CreateEnvironmentEC2",
    "cloud9:CreateEnvironmentSSH",
    "cloud9:GetUserPublicKey",
    "cloud9:GetUserSettings",
    "cloud9:UpdateUserSettings",
    "cloud9:ValidateEnvironmentName",
    "ec2:DescribeSubnets",
    ```
AWS Managed (Predefined) Policies for AWS Cloud9

• **AWSCloud9EnvironmentMember**: Provides the following permissions:
  - **AWS Cloud9**: get information about environments they've been invited to, and get user settings for environments they've been invited to.
  - **IAM**: get information about IAM users in their AWS account.

The **AWSCloud9EnvironmentMember** managed policy contains the following permissions:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:GetUserSettings",
        "cloud9:UpdateUserSettings",
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloud9:DescribeEnvironmentMemberships"
      ],
      "Resource": "*",
      "Condition": {
        "Null": {
          "cloud9:UserArn": "true",
          "cloud9:EnvironmentId": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    }
  ]
}
```
Creating Customer-Managed Policies for AWS Cloud9

If none of the AWS managed policies meet your access control requirements, you can create and attach your own customer-managed policies.

To create a customer-managed policy, see Create an IAM Policy (Console) in the IAM User Guide.

Topics
• Specifying Policy Elements: Effects, Principals, Actions, and Resources (p. 493)
• Customer-Managed Policy Examples (p. 493)
• AWS Cloud9 Permissions Reference (p. 500)

Specifying Policy Elements: Effects, Principals, Actions, and Resources

For each AWS Cloud9 resource, the service defines a set of API operations. To grant permissions for these API operations, AWS Cloud9 defines a set of actions that you can specify in a policy.

The following are the basic policy elements:

• **Effect:** You specify the effect, either allow or deny, when the user requests the action. If you don't explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource. You might do this to ensure a user cannot access a resource, even if a different policy grants access.

• **Principal:** In identity-based policies (IAM policies), the user the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions.

• **Resource:** You use an ARN to identify the resource that the policy applies to.

• **Action:** You use action keywords to identify resource operations you want to allow or deny. For example, the cloud9:CreateEnvironmentEC2 permission gives the user permissions to perform the CreateEnvironmentEC2 operation.

To learn more about IAM policy syntax and descriptions, see the IAM JSON Policy Reference in the IAM User Guide.

For a table showing all of the AWS Cloud9 API actions and the resources they apply to, see the AWS Cloud9 Permissions Reference (p. 500).

Customer-Managed Policy Examples

In this section, you can find example policies that grant permissions for AWS Cloud9 actions. You can adapt the following example IAM policies to allow or explicitly deny AWS Cloud9 access for your IAM identities.

To create or attach a customer-managed policy to an IAM identity, see Create an IAM Policy (Console) and Attaching IAM Policies (Console) in the IAM User Guide.
Creating Customer-Managed Policies for AWS Cloud9

Note
The following examples use the US East (Ohio) Region (us-east-2), a fictitious AWS account ID (123456789012), and a fictitious AWS Cloud9 development environment ID (81e900317347585a0601e04c8d52eaEX).

Topics
- Get Information About Environments (p. 494)
- Create EC2 Environments (p. 494)
- Create EC2 Environments with Specific Amazon EC2 Instance Types (p. 495)
- Create EC2 Environments in Specific Amazon VPC Subnets (p. 495)
- Create an EC2 Environment with a Specific Environment Name (p. 496)
- Create SSH Environments Only (p. 496)
- Update Environments, or Prevent Updating an Environment (p. 496)
- Get Lists of Environment Members (p. 497)
- Share Environments Only with a Specific User (p. 497)
- Prevent Sharing Environments (p. 498)
- Change, or Prevent Changing, the Settings of Environment Members (p. 498)
- Remove, or Prevent Removing, Environment Members (p. 499)
- Delete Environments, or Prevent Deleting an Environment (p. 499)

Get Information About Environments

The following example IAM policy statement, attached to an IAM entity, allows that entity to get information about any environment in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DescribeEnvironments",
      "Resource": "*"
    }
  ]
}
```

Note that the preceding access permission is already included in the AWS managed policies AWSCloud9Administrator and AWSCloud9User.

Create EC2 Environments

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*"
    }
  ]
}
```
Note that the preceding access permission is already included in the AWS managed policies AWSCloud9Administrator and AWSCloud9User.

Create EC2 Environments with Specific Amazon EC2 Instance Types

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account. However, EC2 environments can use only the specified class of Amazon EC2 instance types.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:InstanceType": "t2.*"
        }
      }
    }
  ]
}
```

Note that if the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, those AWS managed policies will override the behavior of the preceding IAM policy statement, as those AWS managed policies are more permissive.

Create EC2 Environments in Specific Amazon VPC Subnets

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account. However, EC2 environments can use only the specified Amazon VPC subnets.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:SubnetId": [
            "subnet-12345678",
            "subnet-23456789"
          ]
        }
      }
    }
  ]
}
```

Note that if the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, those AWS managed policies will override the behavior of the preceding IAM policy statement, as those AWS managed policies are more permissive.
Create an EC2 Environment with a Specific Environment Name

The following example IAM policy statement, attached to an IAM entity, allows that entity to create an AWS Cloud9 EC2 development environment in their account. However, the EC2 environment can use only the specified name.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloud9:CreateEnvironmentEC2",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloud9:EnvironmentName": "my-demo-environment"
                }
            }
        }
    ]
}
```

Note that if the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, those AWS managed policies will override the behavior of the preceding IAM policy statement, as those AWS managed policies are more permissive.

Create SSH Environments Only

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 SSH development environments in their account. However, the entity cannot create AWS Cloud9 EC2 development environments.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloud9:CreateEnvironmentSSH",
            "Resource": "*"
        },
        {
            "Effect": "Deny",
            "Action": "cloud9:CreateEnvironmentEC2",
            "Resource": "*"
        }
    ]
}
```

Update Environments, or Prevent Updating an Environment

The following example IAM policy statement, attached to an IAM entity, allows that entity to change information about any AWS Cloud9 development environment in their account.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloud9:UpdateEnvironment",
            "Resource": "*"
        }
    ]
}
```
Note that the preceding access permission is already included in the AWS managed policy AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from changing information about the environment with the specified ARN.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "cloud9:UpdateEnvironment",
            "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
        }
    ]
}
```

Get Lists of Environment Members

The following example IAM policy statement, attached to an IAM entity, allows that entity to get a list of members for any environment in their account.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloud9:DescribeEnvironmentMemberships",
            "Resource": "*"
        }
    ]
}
```

Note that the preceding access permission is already included in the AWS managed policy AWSCloud9Administrator. Also note that the preceding access permission is more permissive than the equivalent access permission in the AWS managed policy AWSCloud9User.

Share Environments Only with a Specific User

The following example IAM policy statement, attached to an IAM entity, allows that entity to share any environment in their account with only the specified user.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloud9:CreateEnvironmentMembership"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloud9:UserArn": "arn:aws:iam::123456789012:user/MyDemoUser"
                }
            }
        }
    ]
}
```
Creating Customer-Managed Policies for AWS Cloud9

Note that if the AWS managed policy `AWSCloud9Administrator` or `AWSCloud9User` is already attached to the IAM entity, those AWS managed policies will override the behavior of the preceding IAM policy statement, as those AWS managed policies are more permissive.

**Prevent Sharing Environments**

The following example IAM policy statement, attached to an IAM entity, prevents that entity from sharing any environment in their account.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "cloud9:CreateEnvironmentMembership",
                "cloud9:UpdateEnvironmentMembership"
            ],
            "Resource": "*"
        }
    ]
}
```

**Change, or Prevent Changing, the Settings of Environment Members**

The following example IAM policy statement, attached to an IAM entity, allows that entity to change the settings of members in any environment in their account.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "cloud9:UpdateEnvironmentMembership",
            "Resource": "*"
        }
    ]
}
```

Note that the preceding access permission is already included in the AWS managed policy `AWSCloud9Administrator`.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from changing the settings of members in the environment with the specified ARN.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "cloud9:UpdateEnvironmentMembership",
            "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317947585a0601e04c8d52eaEX"
        }
    ]
}
```
Remove, or Prevent Removing, Environment Members

The following example IAM policy statement, attached to an IAM entity, allows that entity to remove any member from any environment in their account.

```
{
   "Version": "2012-10-17",
   "Statement": [ 
       {
           "Effect": "Allow",
           "Action": "cloud9:DeleteEnvironmentMembership",
           "Resource": "*"
       } 
   ]
}
```

Note that the preceding access permission is already included in the AWS managed policy AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from removing any member from the environment with the specified ARN.

```
{
   "Version": "2012-10-17",
   "Statement": [ 
       {
           "Effect": "Deny",
           "Action": "cloud9:DeleteEnvironmentMembership",
           "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
       } 
   ]
}
```

Delete Environments, or Prevent Deleting an Environment

The following example IAM policy statement, attached to an IAM entity, allows that entity to delete any environment in their account.

```
{
   "Version": "2012-10-17",
   "Statement": [ 
       {
           "Effect": "Allow",
           "Action": "cloud9:DeleteEnvironment",
           "Resource": "*"
       } 
   ]
}
```

Note that the preceding access permission is already included in the AWS managed policy AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from deleting the environment with the specified ARN.

```
{
   "Version": "2012-10-17",
   "Statement": [ 
       {
           "Effect": "Deny",
           "Action": "cloud9:DeleteEnvironment",
           "Resource": "*"
       } 
   ]
}
```
Creating Customer-Managed Policies for AWS Cloud9

AWS Cloud9 Permissions Reference

You can use the following table as a reference when you are setting up access control and writing permissions policies that you can attach to an IAM identity (identity-based policies).

You can use AWS-wide condition keys in your AWS Cloud9 policies to express conditions. For a list, see IAM JSON Policy Elements: Condition in the IAM User Guide.

You specify the actions in the policy's Action field. To specify an action, use the cloud9: prefix followed by the API operation name (for example, "Action": "cloud9:DescribeEnvironments"). To specify multiple actions in a single statement, separate them with commas (for example, "Action": [ "cloud9:UpdateEnvironment", "cloud9:DeleteEnvironment" ]).

Topics

- Using Wildcard Characters (p. 500)
- AWS Cloud9 API Operations and Required Permissions for Actions (p. 500)

Using Wildcard Characters

You specify an ARN, with or without a wildcard character (*), as the resource value in the policy's Resource field. You can use a wildcard to specify multiple actions or resources. For example, cloud9:* specifies all AWS Cloud9 actions and cloud9:Describe* specifies all AWS Cloud9 actions that begin with Describe.

The following example allows an IAM entity to get information about environments and environment memberships for any environment in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [ "cloud9:Describe*" ],
      "Resource": "*"
    }
  ]
}
```

Note that the preceding access permission is already included in the AWS managed policy AWSCloud9Administrator. Also note that the preceding access permission is more permissive than the equivalent access permission in the AWS managed policy AWSCloud9User.

AWS Cloud9 API Operations and Required Permissions for Actions

<table>
<thead>
<tr>
<th>AWS Cloud9 operation</th>
<th>Required permission (API action)</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateEnvironmentEC2</td>
<td>cloud9:CreateEnvironmentEC2*</td>
<td></td>
</tr>
</tbody>
</table>
### AWS Managed Temporary Credentials

If you're just looking for the list of actions that AWS managed temporary credentials supports, skip ahead to *Actions Supported by AWS Managed Temporary Credentials* (p. 503).

For an AWS Cloud9 EC2 development environment, AWS Cloud9 makes temporary AWS access credentials available to you in the environment. We call these *AWS managed temporary credentials*. This provides the following benefits:

<table>
<thead>
<tr>
<th>AWS Cloud9 operation</th>
<th>Required permission (API action)</th>
<th>Resource</th>
</tr>
</thead>
</table>
| CreateEnvironmentMembership | cloud9:CreateEnvironmentMembership | required
| DeleteEnvironment | cloud9:DeleteEnvironment | arn:aws:cloud9:REGION_ID:ACCOUNT_ID:
| DeleteEnvironmentMembership | cloud9:DeleteEnvironmentMembership | arn:aws:cloud9:REGION_ID:ACCOUNT_ID:
| DescribeEnvironmentMemberships | cloud9:DescribeEnvironmentMemberships | required |
| DescribeEnvironments | cloud9:DescribeEnvironments | arn:aws:cloud9:REGION_ID:ACCOUNT_ID:
| DescribeEnvironmentStatus | cloud9:DescribeEnvironmentStatus | arn:aws:cloud9:REGION_ID:ACCOUNT_ID:
| UpdateEnvironment | cloud9:UpdateEnvironment | arn:aws:cloud9:REGION_ID:ACCOUNT_ID:
| UpdateEnvironmentMembership | cloud9:UpdateEnvironmentMembership | arn:aws:cloud9:REGION_ID:ACCOUNT_ID: |
AWS Managed Temporary Credentials

- You don't need to store the permanent AWS access credentials of an AWS entity (for example, an IAM user) anywhere in the environment. This prevents those credentials from being accessed by environment members without your knowledge and approval.

- You don't need to manually set up, manage, or attach an instance profile to the Amazon EC2 instance that connects to the environment. (An instance profile is another approach for managing temporary AWS access credentials.)

- AWS Cloud9 continually renews its temporary credentials, so a single set of credentials can only be used for a limited time. This is an AWS security best practice. For more information, see Creating and Updating AWS Managed Temporary Credentials (p. 504).

- AWS Cloud9 puts additional restrictions on how its temporary credentials can be used to access AWS actions and resources from the environment. This is also an AWS security best practice.

Here's how AWS managed temporary credentials work whenever an EC2 environment tries to access an AWS service on behalf of an AWS entity (for example, an IAM user):

1. AWS Cloud9 checks to see if the calling AWS entity (for example, the IAM user) has permissions to take the requested action for the requested resource in AWS. If the permission doesn't exist or is explicitly denied, the request fails.

2. AWS Cloud9 checks AWS managed temporary credentials to see if its permissions allow the requested action for the requested resource in AWS. If the permission doesn't exist or is explicitly denied, the request fails. For a list of permissions that AWS managed temporary credentials support, see Actions Supported by AWS Managed Temporary Credentials (p. 503).

3. If both the AWS entity and AWS managed temporary credentials allow the requested action for the requested resource, the request succeeds.

4. If either the AWS entity or AWS managed temporary credentials explicitly deny (or fail to explicitly allow) the requested action for the requested resource, the request fails. This means that even if the calling AWS entity has the correct permissions, the request will fail if AWS Cloud9 doesn't also explicitly allow it. Likewise, if AWS Cloud9 allows a specific action to be taken for a specific resource, the request will fail if the AWS entity doesn't also explicitly allow it.

The owner of an EC2 environment can turn on or off AWS managed temporary credentials for that environment at any time, as follows:

1. With the environment open, in the AWS Cloud9 IDE, on the menu bar choose AWS Cloud9, Preferences.

2. In the Preferences tab, in the navigation pane, choose AWS Settings, Credentials.

3. Use AWS managed temporary credentials to turn AWS managed temporary credentials on or off.

If you turn off AWS managed temporary credentials, by default the environment cannot access any AWS services, regardless of the AWS entity who makes the request. If you cannot or do not want to turn on AWS managed temporary credentials for an environment, but you still need the environment to access AWS services, consider the following alternatives:

- Attach an instance profile to the Amazon EC2 instance that connects to the environment. For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71).

- Store your permanent AWS access credentials in the environment, for example, by setting special environment variables or by running the aws configure command. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

The preceding alternatives override all permissions that are allowed (or denied) by AWS managed temporary credentials in an EC2 environment.
Actions Supported by AWS Managed Temporary Credentials

For an AWS Cloud9 EC2 development environment, AWS managed temporary credentials allow all AWS actions for all AWS resources in the caller's AWS account, with the following restrictions:

- For AWS Cloud9, only the following actions are allowed:
  - cloud9:CreateEnvironmentEC2
  - cloud9:CreateEnvironmentSSH
  - cloud9:DescribeEnvironmentMemberships
  - cloud9:DescribeEnvironments
  - cloud9:DescribeEnvironmentStatus
  - cloud9:UpdateEnvironment

- For IAM, only the following actions are allowed:
  - iam:AttachRolePolicy
  - iam:ChangePassword
  - iam:CreatePolicy
  - iam:CreatePolicyVersion
  - iam:CreateRole
  - iam:CreateServiceLinkedRole
  - iam:DeletePolicy
  - iam:DeletePolicyVersion
  - iam:DeleteRole
  - iam:DeleteRolePolicy
  - iam:DeleteSSHPublicKey
  - iam:DetachRolePolicy
  - iam:GetInstanceProfile
  - iam:GetPolicy
  - iam:GetPolicyVersion
  - iam:GetRole
  - iam:GetRolePolicy
  - iam:GetSSHPublicKey
  - iam:GetUser
  - iam:List*
  - iam:PassRole
  - iam:PutRolePolicy
  - iam:SetDefaultPolicyVersion
  - iam:UpdateAssumeRolePolicy
  - iam:UpdateRoleDescription
  - iam:UpdateSSHPublicKey
  - iam:UploadSSHPublicKey

- All IAM actions that interact with roles are allowed only for role names starting with Cloud9-.
  However, iam:PassRole works with all role names.

- For AWS Security Token Service (AWS STS), only the following actions are allowed:
  - sts:GetCallerIdentity
  - sts:DecodeAuthorizationMessage

- All supported AWS actions are restricted to the IP address of the environment. This is an AWS security best practice.
If AWS Cloud9 doesn’t support an action or resource that you need an EC2 environment to access, or if AWS managed temporary credentials is turned off for an EC2 environment and you cannot turn it back on, consider the following alternatives:

- Attach an instance profile to the Amazon EC2 instance that connects to the EC2 environment. For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71).
- Store your permanent AWS access credentials in the EC2 environment, for example, by setting special environment variables or by running the `aws configure` command. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

The preceding alternatives override all permissions that are allowed (or denied) by AWS managed temporary credentials in an EC2 environment.

### Creating and Updating AWS Managed Temporary Credentials

For an AWS Cloud9 EC2 development environment, AWS managed temporary credentials are created the first time you open the environment.

AWS managed temporary credentials are updated under any of the following conditions:

- Whenever a certain period of time passes. Currently, this is every 5 minutes.
- Whenever you reload the web browser tab that displays the IDE for the environment.
- When the timestamp that is listed in the `~/.aws/credentials` file for the environment is reached.
- If the **AWS managed temporary credentials** setting is set to off, whenever you turn it back on. (To view or change this setting, choose **AWS Cloud9, Preferences** in the menu bar of the IDE. In the **Preferences** tab, in the navigation pane, choose **AWS Settings, Credentials**.)

### Using Service-Linked Roles for AWS Cloud9

AWS Cloud9 uses AWS Identity and Access Management (IAM) *service-linked roles*. A service-linked role is a unique type of IAM role that is linked directly to AWS Cloud9. Service-linked roles are predefined by AWS Cloud9 and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up AWS Cloud9 easier because you don’t have to manually add the necessary permissions. AWS Cloud9 defines the permissions of its service-linked roles, and only AWS Cloud9 can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

You can delete the roles only after first deleting their related resources. This protects your AWS Cloud9 resources because you can’t inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have a **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

- Service-Linked Role Permissions for AWS Cloud9 (p. 505)
- Creating a Service-Linked Role for AWS Cloud9 (p. 506)
- Editing a Service-Linked Role for AWS Cloud9 (p. 506)
- Deleting a Service-Linked Role for AWS Cloud9 (p. 507)
- Supported Regions for AWS Cloud9 Service-Linked Roles (p. 507)
Service-Linked Role Permissions for AWS Cloud9

AWS Cloud9 uses the service-linked role named AWSServiceRoleForAWSCloud9. This service-linked role trusts the service cloud9.amazonaws.com to assume the role.

The role permissions policy allows AWS Cloud9 to complete the following actions on the specified resources.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "cloudformation:CreateStack",
            "cloudformation:DescribeStacks",
            "ec2:CreateSecurityGroup",
            "ec2:DescribeInstances",
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeVpcs",
            "ec2:RunInstances"
         ],
         "Resource": "*"
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:AuthorizeSecurityGroupIngress",
            "ec2:DeleteSecurityGroup",
            "ec2:TerminateInstances"
         ],
         "Resource": "*"
      },
      {
         "Effect": "Allow",
         "Action": [
            "cloudformation:DeleteStack"
         ],
         "Condition": {
            "StringLike": {
               "aws:RequestTag/Name": "aws-cloud9-*"
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:CreateTags"
         ],
         "Resource": "arn:aws:ec2::*:instance/*",
         "Condition": {
            "StringLike": {
               "aws:ResourceTag/aws:cloudformation:stack-name": "aws-cloud9-*"
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:StartInstances"
         ],
         "Resource": "*",
         "Condition": {
            "StringLike": {
               "ec2:ResourceTag/aws:cloudformation:stack-name": "aws-cloud9-*"
            }
         }
      }
   ]
}
```
You must configure permissions to allow AWS Cloud9 to create a service-linked role on behalf of an IAM entity (such as a user, group, or role).

To allow AWS Cloud9 to create the AWSServiceRoleForAWSCloud9 service-linked role, add the following statement to the permissions policy for the IAM entity on whose behalf AWS Cloud9 needs to create the service-linked role.

```json
{
  "Effect": "Allow",
  "Action": [
    "iam:CreateServiceLinkedRole"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
  }
}
```

Alternatively, you can add the AWS managed policies AWSCloud9User or AWSCloud9Administrator to the IAM entity.

To allow an IAM entity to delete the AWSServiceRoleForAWSCloud9 service-linked role, add the following statement to the permissions policy for the IAM entity that needs to delete a service-linked role.

```json
{
  "Effect": "Allow",
  "Action": [
    "iam:DeleteServiceLinkedRole",
    "iam:GetServiceLinkedRoleDeletionStatus"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "iam:AWSServiceName": "cloud9.amazonaws.com"
    }
  }
}
```

Creating a Service-Linked Role for AWS Cloud9

You don't need to manually create a service-linked role. When you create an AWS Cloud9 development environment, AWS Cloud9 creates the service-linked role for you.

Editing a Service-Linked Role for AWS Cloud9

AWS Cloud9 doesn't allow you to edit the AWSServiceRoleForAWSCloud9 service-linked role. For example, after you create a service-linked role, you can't change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the IAM User Guide.
Deleting a Service-Linked Role for AWS Cloud9

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete that role. That way you don’t have an unused entity that is not actively monitored or maintained.

Deleting a Service-Linked Role in IAM

Before you can use IAM to delete a service-linked role, you must remove any AWS Cloud9 resources used by the role. To remove AWS Cloud9 resources, see Deleting an Environment (p. 95).

You can use the IAM console to delete the AWSServiceRoleForAWSCloud9 service-linked role. For more information, see Deleting a Service-Linked Role in the IAM User Guide.

Supported Regions for AWS Cloud9 Service-Linked Roles

AWS Cloud9 supports using service-linked roles in in all of the regions where the service is available. For more information, see AWS Cloud9 in the Amazon Web Services General Reference.

Logging AWS Cloud9 API Calls with AWS CloudTrail

AWS Cloud9 is integrated with CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Cloud9. CloudTrail captures all API calls for AWS Cloud9 as events. The calls captured include calls from the AWS Cloud9 console and from code calls to the AWS Cloud9 APIs. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for AWS Cloud9. If you don’t configure a trail, you can still view the most recent events in the CloudTrail console in Event history. Using the information collected by CloudTrail, you can determine the request that was made to AWS Cloud9, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS Cloud9 Information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Cloud9, that activity is recorded in a CloudTrail event along with other AWS service events in Event history. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for AWS Cloud9, create a trail. A trail enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- Overview for Creating a Trail
- CloudTrail Supported Services and Integrations
- Configuring Amazon SNS Notifications for CloudTrail
Understanding AWS Cloud9 Log File Entries

AWS Cloud9 supports logging the following actions as events in CloudTrail log files:

- CreateEnvironmentEC2
- CreateEnvironmentSSH
- CreateEnvironmentMembership
- DeleteEnvironment
- DeleteEnvironmentMembership
- DescribeEnvironmentMemberships
- DescribeEnvironments
- ListEnvironments
- UpdateEnvironment
- UpdateEnvironmentMembership

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity Element.

Understanding AWS Cloud9 Log File Entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

- CreateEnvironmentEC2 (p. 508)
- CreateEnvironmentSSH (p. 509)
- CreateEnvironmentMembership (p. 510)
- DeleteEnvironment (p. 511)
- DeleteEnvironmentMembership (p. 511)
- DescribeEnvironmentMemberships (p. 512)
- DescribeEnvironments (p. 513)
- ListEnvironments (p. 514)
- UpdateEnvironment (p. 514)
- UpdateEnvironmentMembership (p. 515)

CreateEnvironmentEC2

The following example shows a CloudTrail log entry that demonstrates the CreateEnvironmentEC2 action.
CreateEnvironmentSSH

The following example shows a CloudTrail log entry that demonstrates the CreateEnvironmentSSH action.
CreateEnvironmentMembership

The following example shows a CloudTrail log entry that demonstrates the CreateEnvironmentMembership action.

```json
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventName": "CreateEnvironmentMembership",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "userArn": "arn:aws:iam::111122223333:user/MyUser",
        "permissions": "read-write"
      },
      "responseElements": {
        "environmentId": "5c39cc4a85d74a8bb6e23ed6EXAMPLE"
      },
      "requestID": "f0e629fbd-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906f455-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```
DeleteEnvironment

The following example shows a CloudTrail log entry that demonstrates the DeleteEnvironment action.

```json
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DeleteEnvironment",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE"
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e982EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

DeleteEnvironmentMembership

The following example shows a CloudTrail log entry that demonstrates the DeleteEnvironmentMembership action.
DescribeEnvironmentMemberships

The following example shows a CloudTrail log entry that demonstrates the DescribeEnvironmentMemberships action.
DescribeEnvironments

The following example shows a CloudTrail log entry that demonstrates the DescribeEnvironments action.

```json
{
  "Records": [ 
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "DescribeEnvironments",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentIds": [
          "2f5ff70a640f49398f67e3bdeb811ab2"
        ]
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```
AWS Cloud9 User Guide
Understanding AWS Cloud9 Log File Entries

ListEnvironments

The following example shows a CloudTrail log entry that demonstrates the ListEnvironments action.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mFAAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "ListEnvironments",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "nextToken": "NEXT_TOKEN_EXAMPLE",
        "maxResults": 15
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```

UpdateEnvironment

The following example shows a CloudTrail log entry that demonstrates the UpdateEnvironment action.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mFAAuthenticated": "false",
            "creationDate": "2019-01-11T12:29:47Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "UpdateEnvironment",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "nextToken": "NEXT_TOKEN_EXAMPLE",
        "maxResults": 15
      },
      "responseElements": null,
      "requestID": "f0e629fb-fd37-49bd-b2cc-e9822EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "readOnly": true,
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```
UpdateEnvironmentMembership

The following example shows a CloudTrail log entry that demonstrates the UpdateEnvironmentMembership action.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/MyUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "MyUser",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2019-01-14T11:29:47Z"
          }
        }
      },
      "invokedBy": "signin.amazonaws.com"
    },
    {
      "eventTime": "2019-01-14T11:33:27Z",
      "eventSource": "cloud9.amazonaws.com",
      "eventName": "UpdateEnvironmentMembership",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "environmentId": "2f5ff70a640f49398f67e3bdeEXAMPLE",
        "description": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "name": "my-test-environment-renamed"
      },
      "responseElements": null,
      "requestId": "f0e6296f-fd37-49bd-b2cc-e982EXAMPLE",
      "eventID": "8a906445-1b2a-47e9-8d7c-5b242EXAMPLE",
      "eventType": "AwsApiCall",
      "recipientAccountId": "111122223333"
    }
  ]
}
```
Tags

A tag is a label or attribute that you or AWS attaches to an AWS resource. Each tag consists of a key and a paired value. You can use tags to control access to your AWS Cloud9 resources, as described in Control Access Using AWS Resource Tags in the IAM User Guide. Tags can also help you manage billing information, as described in User-Defined Cost Allocation Tags.

When you create an AWS Cloud9 EC2 development environment (p. 56), AWS Cloud9 includes certain system tags that it needs to manage the environment. System tags start with "aws:". During that creation process, you can also add your own resource tags.

After the environment is created, you can view the tags that are attached to the environment, add new resource tags to the environment, or modify or remove the tags that you added earlier. You can attach up to 50 user-defined tags to an AWS Cloud9 environment.

View or update tags using one or more of the following methods.

• In the AWS Cloud9 console, select the environment you're interested in, and then choose View Details.

• Use the following AWS Cloud9 CLI commands: list-tags-for-resource, tag-resource, and untag-resource.

• Use the following AWS Cloud9 API actions: ListTagsForResource, TagResource, and UntagResource.

Warning
Tags that you create or update for AWS Cloud9 by using the preceding methods are NOT automatically propagated to underlying resources. For information about how to do this, see the next section, Propagating Tag Updates to Underlying Resources (p. 516).

Propagating Tag Updates to Underlying Resources

When you use AWS Cloud9 CLI commands or API actions to add, modify, or remove the tags that are attached to an AWS Cloud9 environment, those changes aren't automatically propagated to underlying resources such as the AWS CloudFormation stack, the Amazon EC2 instance, and Amazon EC2 security groups. You must manually propagate those changes.

To make it easier to use the following procedures, you can obtain the environment ID for the environment you're interested in. If you wish to do this, do so as follows:

1. In the AWS Cloud9 console, select the environment you're interested in, and then choose View Details.
2. Look for the Environment ARN property and record the environment ID, which is the part of the environment ARN after "environment:"

You will need to propagate tag updates to one or more of the following locations, depending on what you'll use the tags for.

**Propagating Tag Updates to the AWS CloudFormation Stack**

*Note*

When you update tags to the AWS CloudFormation stack, those updates are automatically propagated to the Amazon EC2 instance and Amazon EC2 security groups that are associated with the stack.

1. Navigate to the AWS CloudFormation console.
2. Find and choose the stack that corresponds to the AWS Cloud9 environment you're interested in. If you recorded the environment ID, you can use it to filter for the environment.
3. On the Stack info tab, in the Tags section, review the list of tags.
4. If you need to update the tags, choose Update near the top of the page, and follow the instructions. For more information, see Updating Stacks Directly in the AWS CloudFormation User Guide.

You can also update tags using the describe-stacks and update-stack CLI commands.

**Propagating Tag Updates to the Amazon EC2 Instance**

1. Navigate to the Amazon EC2 Instances console.
2. Find and select the Amazon EC2 instance that corresponds to the AWS Cloud9 environment you're interested in. If you recorded the environment ID earlier, you can use it to filter for the environment.
3. On the Tags tab, view and update tags as necessary.

You can also update tags using the describe-tags, create-tags, and delete-tags CLI commands.

**Propagating Tag Updates to Amazon EC2 Security Groups**

1. Navigate to the Amazon EC2 Security Groups console.
2. Find and select the security group that corresponds to the AWS Cloud9 environment you're interested in. If you recorded the environment ID earlier, you can use it to filter for the environment.
3. Open the Tags tab to view and update tags as necessary.

You can also update tags using the describe-tags, create-tags, and delete-tags CLI commands.
Security for AWS Cloud9

Cloud security at Amazon Web Services (AWS) is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations. Security is a shared responsibility between AWS and you. The Shared Responsibility Model describes this as Security of the Cloud and Security in the Cloud.

Security of the Cloud – AWS is responsible for protecting the infrastructure that runs all of the services offered in the AWS Cloud and providing you with services that you can use securely. Our security responsibility is the highest priority at AWS, and the effectiveness of our security is regularly tested and verified by third-party auditors as part of the AWS Compliance Programs.

Security in the Cloud – Your responsibility is determined by the AWS service you are using, and other factors including the sensitivity of your data, your organization’s requirements, and applicable laws and regulations.

AWS Cloud9 follows the shared responsibility model through the specific AWS services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.

The following topics show you how to configure AWS Cloud9 to meet your security and compliance objectives.

Topics
- Data protection in AWS Cloud9 (p. 518)
- Identity and access management in AWS Cloud9 (p. 520)
- Logging and monitoring in AWS Cloud9 (p. 538)
- Compliance validation for AWS Cloud9 (p. 539)
- Resilience in AWS Cloud9 (p. 539)
- Infrastructure security in AWS Cloud9 (p. 540)
- Configuration and vulnerability analysis in AWS Cloud9 (p. 540)
- Security best practices for AWS Cloud9 (p. 540)

Data protection in AWS Cloud9

AWS Cloud9 conforms to the AWS shared responsibility model, which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, and all default security controls within AWS services.
• Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon Simple Storage Service (Amazon S3).

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a Name field. This includes when you work with AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the AWS Shared Responsibility Model and GDPR blog post on the AWS Security Blog.

Data encryption

Data encryption refers to protecting data while in transit (as it travels between AWS Cloud9 and your AWS account) and at rest (while it is stored in AWS Cloud9 configuration stores and AWS cloud-compute instances).

In the context of AWS Cloud9, the following types of data may require protection through encryption:

**Your content and data**

Information that you manipulate, collect, and store. The following are examples of this type of data:

- Your code files
- Configuration, applications, and data for the attached EC2 environment or SSH environment

**AWS Cloud9 metadata**

Data that AWS Cloud9 manipulates, collects, and stores. The following are examples of this type of data:

- IDE settings such as tab states, open files, and IDE preferences
- AWS Cloud9 development environment metadata such as environment names and descriptions
- AWS Cloud service API, and console logs
- Service logs such as HTTP requests

AWS Cloud9 also transmits some of your content and data through its data plane service. This includes your files, terminal input, output text, and some IDE commands (for example, for saving files).

Encryption at rest

Encryption at rest refers to protecting your data from unauthorized access by encrypting data while stored. Any customer data stored in an AWS Cloud9 environment such as code files, packages, or dependencies is always stored in the customer's resources. If the customer uses an Amazon EC2 environment, the data is stored in the associated Amazon Elastic Block Store (Amazon EBS) volume that exists in their AWS account. If the customer uses an SSH environment, the data is stored in local storage on their Linux server.

When Amazon EC2 instances are created for an AWS Cloud9 development environment, an unencrypted Amazon EBS volume is created and attached to that instance. Customers who want to encrypt their data need to create an encrypted EBS volume and attach it to the EC2 instance.

Metadata about the AWS Cloud9 development environments, such as environment names, members of the environments, and IDE settings, is stored by AWS, not in customer resources. Customer-specific information, such as environment descriptions and IDE settings, is encrypted.
Encryption in transit

Encryption in transit refers to protecting your data from being intercepted while it moves between communication endpoints. All data transmitted between the customer's client and the AWS Cloud9 service is encrypted through HTTPS, WSS, and encrypted SSH.

- **HTTPS** – Ensures secure requests between the customer's web browser and the AWS Cloud9 service. AWS Cloud9 also loads assets from Amazon CloudFront sent over HTTPS from the customer's browser.
- **WSS (WebSocket Secure)** – Enables secure two-way communications through WebSockets between the customer's web browser and the AWS Cloud9 service.
- **Encrypted SSH (Secure Shell)**: Enables secure transmission of data between the client's web browser and the AWS Cloud9 service.

Use of HTTPS, WSS, and SSH protocols depends on your using a browser supported by AWS Cloud9. See Supported Browsers for AWS Cloud9 (p. 557).

**Note**


Key management

AWS Key Management Service (AWS KMS) is a managed service for creating and controlling customer master keys (CMKs), the encryption keys used to encrypt the customer's data. AWS Cloud9 generates and manages cryptographic keys for encrypting data on behalf of customers.

Internetwork traffic privacy

SSH environments connect to on-premises, customer-owned compute and storage. Encrypted SSH, HTTPS, and WSS connections support data transit between the service and SSH environment.

You can configure AWS Cloud9 EC2 development environments (backed by Amazon EC2 instances) to be launched within specific VPCs and subnets. For more information about Amazon Virtual Private Cloud settings, see VPC Settings for AWS Cloud9 Development Environments (p. 467).

Identity and access management in AWS Cloud9

AWS Identity and Access Management (IAM) is an Amazon Web Services (AWS) service that helps an administrator securely control access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use resources in AWS services. IAM is an AWS service that you can use with no additional charge.

To use AWS Cloud9 to access AWS, you need an AWS account and AWS credentials. To increase the security of your AWS account, we recommend that you use an IAM user to provide access credentials instead of using your AWS account credentials.

For details about working with IAM, see AWS Identity and Access Management.

For an overview of IAM users and why they are important for the security of your account, see AWS Security Credentials in the Amazon Web Services General Reference.

AWS Cloud9 follows the shared responsibility model through the specific Amazon Web Services (AWS) services it supports. For AWS service security information, see the AWS service security documentation page and AWS services that are in scope of AWS compliance efforts by compliance program.
Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work you do in AWS Cloud9.

Service user - If you use the AWS Cloud9 service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Cloud9 features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Cloud9, see Troubleshooting AWS Cloud9 (p. 542).

Service administrator - If you're in charge of AWS Cloud9 resources at your company, you probably have full access to AWS Cloud9. It's your job to determine which AWS Cloud9 features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS Cloud9, see How AWS Cloud9 works with IAM (p. 522).

IAM administrator - If you use the AWS Cloud9 service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Cloud9 features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Cloud9, see Troubleshooting AWS Cloud9 (p. 542).

Authenticating with identities

You can access AWS as any of the following types of identities.

AWS account root user

When you sign up for AWS, you provide an email address and password that is associated with your AWS account. These are your root credentials, and they provide complete access to all of your AWS resources.

Important

As an AWS security best practice, we recommend that you use the root credentials only to create an IAM administrator group with an IAM administrator user. This is a group that gives the user full permissions to your AWS account. Then you can use this administrator user to create other IAM users and roles with limited permissions. For more information, see Create Individual IAM Users and Creating Your First IAM Admin User and Group in the IAM User Guide.

IAM user

An IAM user is simply an identity within your AWS account that has specific custom permissions (for example, permissions to create an AWS Cloud9 development environment). You can use an IAM user name and password to sign in to secure AWS webpages like the AWS Cloud9 console, AWS Management Console, AWS Discussion Forums, and AWS Support Center.

In addition to a user name and password, you can also generate access keys for each user. You can use these keys when you access AWS services programmatically, either through one of the several AWS SDKs or by using the AWS Command Line Interface (AWS CLI) or the aws-shell. The AWS SDKs, the AWS CLI, and the aws-shell use these access keys to cryptographically sign your request. If you don't use these
tools, you must sign the request yourself. AWS Cloud9 supports Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 Signing Process in the Amazon Web Services General Reference.

IAM role

An IAM role is another IAM identity you can create in your account that has specific permissions. It's similar to an IAM user, but it isn't associated with a specific person. An IAM role enables you to obtain temporary access keys that can be used to access AWS services and resources. IAM roles with temporary credentials are useful in the following situations.

AWS service access

You can use an IAM role in your account to grant an AWS service permissions to access your account's resources. For example, you can create a role that allows AWS Lambda to access an Amazon S3 bucket on your behalf, and then load data stored in the bucket into an Amazon Redshift data warehouse. For more information, see Creating a Role to Delegate Permissions to an AWS Service in the IAM User Guide.

Applications running on Amazon EC2

Instead of storing access keys within an Amazon EC2 instance for use by applications running on the instance and making AWS API requests, you can use an IAM role to manage temporary credentials for these applications. To assign an AWS role to an Amazon EC2 instance and make it available to all of its applications, you can create an instance profile that is attached to the instance. An instance profile contains the role and enables programs running on the Amazon EC2 instance to get temporary credentials. For more information, see Create and Use an Instance Profile to Manage Temporary Credentials and Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances in the IAM User Guide.

Note

Instead of attaching an instance profile to an Amazon EC2 instance that connects to an environment, AWS Cloud9 can automatically set up and manage temporary credentials on your behalf in an EC2 environment. For more information, see AWS managed temporary credentials (p. 536).

Federated user access

Instead of creating an IAM user, you can use pre-existing user identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information, see Federated Users and Roles in the IAM User Guide.

Managing access using policies

You can have valid credentials to authenticate your requests, but unless you have permissions, you cannot create or access AWS Cloud9 resources. For example, you must have permissions to create, share, or delete an AWS Cloud9 development environment.

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies. An account administrator can attach permissions policies to IAM identities (that is, users, groups, and roles).

When you grant permissions, you decide who is getting the permissions, the resources they can access, and the actions that can be performed on those resources.

How AWS Cloud9 works with IAM

AWS Identity and Access Management is used to manage the permissions that allow you to work with both AWS Cloud9 development environments and other AWS services and resources.
AWS Cloud9 resources and operations

In AWS Cloud9, the primary resource is an AWS Cloud9 development environment. In a policy, you use an Amazon Resource Name (ARN) to identify the resource that the policy applies to. The following table lists environment ARNs. For more information, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the Amazon Web Services General Reference.

<table>
<thead>
<tr>
<th>Resource type</th>
<th>ARN format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Every environment owned by the specified account in the specified AWS Region</td>
<td>arn:aws:cloud9:REGION_ID:ACCOUNT_ID:environment:*</td>
</tr>
<tr>
<td>Every environment owned by the specified account in the specified Region</td>
<td>arn:aws:cloud9:REGION_ID:ACCOUNT_ID:*</td>
</tr>
<tr>
<td>Every AWS Cloud9 resource, regardless of account and Region</td>
<td>arn:aws:cloud9:*</td>
</tr>
</tbody>
</table>

For example, you can indicate a specific environment in your statement using its ARN, as follows.

"Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:70d899206236474f9590d93b7c41dfEX"

To specify all resources, use the wildcard character (*) in the Resource element, as follows.

"Resource": "*

To specify multiple resources in a single statement, separate their ARNs with commas, as follows.


AWS Cloud9 provides a set of operations to work with AWS Cloud9 resources. For a list, see the AWS Cloud9 permissions reference (p. 534).

Understanding resource ownership

The AWS account owns the resources that are created in the account, regardless of who created the resources.

For example:

- If you use the root account credentials of your AWS account to create an AWS Cloud9 development environment (which, although possible, is not recommend as an AWS security best practice), your AWS account is the owner of the environment.
- If you create an IAM user in your AWS account and grant permissions to create an environment to that user, the user can create an environment. However, your AWS account, to which the user belongs, owns the environment.
- If you create an IAM role in your AWS account with permissions to create an environment, anyone who can assume the role can create an environment. Your AWS account, to which the role belongs, owns the environment.
Managing access to resources

A permissions policy describes who has access to which resources.

**Note**
This section discusses the use of IAM in AWS Cloud9. It doesn't provide detailed information about the IAM service. For complete IAM documentation, see What Is IAM? in the IAM User Guide. For information about IAM policy syntax and descriptions, see the IAM JSON Policy Reference in the IAM User Guide.

Policies attached to an IAM identity are referred to as identity-based policies (or IAM policies). Policies attached to a resource are referred to as resource-based policies. AWS Cloud9 supports both identity-based and resource-based policies.

Each of the following API actions requires only an IAM policy to be attached to the IAM identity that wants to call these API actions:

- CreateEnvironmentEC2
- DescribeEnvironments

The following API actions require a resource-based policy. An IAM policy isn't required, but AWS Cloud9 will use an IAM policy if it is attached to the IAM identity that wants to call these API actions. The resource-based policy must be applied to the desired AWS Cloud9 resource:

- CreateEnvironmentMembership
- DeleteEnvironment
- DeleteEnvironmentMembership
- DescribeEnvironmentMemberships
- DescribeEnvironmentStatus
- UpdateEnvironment
- UpdateEnvironmentMembership

For details on what each of these API actions does, see the AWS Cloud9 API Reference.

You cannot attach a resource-based policy to an AWS Cloud9 resource directly. Instead, AWS Cloud9 attaches the appropriate resource-based policies to AWS Cloud9 resources as you add, modify, update, or delete environment members.

To grant a user permissions to perform actions on AWS Cloud9 resources, you attach a permissions policy to an IAM group that the user belongs to. We recommend you attach an AWS managed (predefined) policy for AWS Cloud9 whenever possible. AWS managed policies are easier and faster to attach. They also contain predefined sets of access permissions for common usage scenarios and user types, such as full administration of an environment, environment users, and users who have only read-only access to an environment. For a list of AWS managed policies for AWS Cloud9, see AWS managed (predefined) policies for AWS Cloud9 (p. 524).

For more detailed usage scenarios and unique user types, you can create and attach your own customer-managed policies. See Additional Setup Options for AWS Cloud9 (Team and Enterprise) (p. 22) and Creating Customer-Managed Policies for AWS Cloud9 (p. 493).

To attach an IAM policy (AWS managed or customer-managed) to an IAM identity, see Attaching IAM Policies (Console) in the IAM User Guide.

**AWS managed (predefined) policies for AWS Cloud9**

AWS addresses many common use cases by providing standalone IAM policies that AWS creates and administers. These AWS managed policies grant necessary permissions for common use cases so you
can avoid having to investigate what permissions are needed. For example, you can use AWS managed policies for AWS Cloud9 to quickly and easily allow users to have full administration of an AWS Cloud9 development environment, act as an environment user, or use an environment they are added to. For more information, see AWS Managed Policies in the IAM User Guide.

To attach an AWS managed policy to an IAM identity, see Attaching IAM Policies (Console) in the IAM User Guide.

The following AWS managed policies, which you can attach to IAM identities in your account, are specific to AWS Cloud9:

- **AWSCloud9Administrator** – Provides the following permissions:
  - Amazon EC2 – Get information about multiple Amazon VPC and subnet resources in their AWS account.
  - AWS Cloud9 – All AWS Cloud9 actions in their AWS account.
  - IAM – Get information about IAM users in their AWS account, and create the AWS Cloud9 service-linked role in their AWS account as needed.

  The **AWSCloud9Administrator** managed policy contains the following permissions.

  ```json
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "cloud9:*",
          "ec2:DescribeSubnets",
          "ec2:DescribeVpcs",
          "iam:GetUser",
          "iam:ListUsers"
        ],
        "Resource": "*
      },
      {
        "Effect": "Allow",
        "Action": [
          "iam:CreateServiceLinkedRole"
        ],
        "Resource": "*
      },
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    ]
  }
  ```

- **AWSCloud9User** – Provides the following permissions:
  - Amazon EC2 – Get information about multiple Amazon VPC and subnet resources in their AWS account.
  - – Create and get information about their environments, and get and change user settings for their environments.
  - IAM – Get information about IAM users in their AWS account, and create the AWS Cloud9 service-linked role in their AWS account as needed.

  The **AWSCloud9User** managed policy contains the following permissions.

  ```json
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "cloud9:*",
          "ec2:DescribeSubnets",
          "ec2:DescribeVpcs",
          "iam:GetUser",
          "iam:ListUsers"
        ],
        "Resource": "*
      },
      {
        "Effect": "Allow",
        "Action": [
          "iam:CreateServiceLinkedRole"
        ],
        "Resource": "*
      },
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    ]
  }
  ```
• AWSCloud9EnvironmentMember – Provides the following permissions:

  • AWS Cloud9 – Get information about and user settings for environments they’ve been invited to.
  • IAM – Get information about IAM users in their AWS account.

The AWSCloud9EnvironmentMember managed policy contains the following permissions.

```
{  
  "Version": "2012-10-17",
  "Statement": [
    
    {  
      "Effect": "Allow",
      "Action": [  
        "cloud9:GetUserSettings",
        "cloud9:UpdateUserSettings",
        "iam:GetUser",
        "iam:ListUsers"
      ],
      "Resource": "*"
    },
    
    {  
      "Effect": "Allow",
      "Action": [  
        "cloud9:GetUserSettings",
        "cloud9:UpdateUserSettings"
      ],
      "Resource": "*",
      "Condition": {
        "Null": {
          "cloud9:UserArn": "true",
          "cloud9:EnvironmentId": "true"
        }
      }
    },
    
    {  
      "Effect": "Allow",
      "Action": [  
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "cloud9.amazonaws.com"
        }
      }
    }
  ]
}
```
Creating customer-managed policies for AWS Cloud9

If none of the AWS managed policies meet your access control requirements, you can create and attach your own customer-managed policies.

To create a customer-managed policy, see Create an IAM Policy (Console) in the IAM User Guide.

Topics
- Specifying policy elements: effects, principals, actions, and resources (p. 527)
- Customer-managed policy examples (p. 528)
- AWS Cloud9 permissions reference (p. 534)

Specifying policy elements: effects, principals, actions, and resources

For each AWS Cloud9 resource, the service defines a set of API operations. To grant permissions for these API operations, AWS Cloud9 defines a set of actions that you can specify in a policy.

The following are the basic policy elements:

- **Effect** – You specify the effect, either allow or deny, when the user requests the action. If you don't explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource. You might do this to ensure a user cannot access a resource, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions.
- **Resource** – Use an ARN to identify the resource that the policy applies to.
- **Action** – Use action keywords to identify resource operations you want to allow or deny. For example, the `cloud9:CreateEnvironmentEC2` permission gives the user permissions to perform the `CreateEnvironmentEC2` operation.

To learn more about IAM policy syntax and descriptions, see the IAM JSON Policy Reference in the IAM User Guide.

For a table showing all of the AWS Cloud9 API actions and the resources they apply to, see the AWS Cloud9 permissions reference (p. 534).
Customer-managed policy examples

In this section, you can find example policies that grant permissions for AWS Cloud9 actions. You can adapt the following example IAM policies to allow or explicitly deny AWS Cloud9 access for your IAM identities.

To create or attach a customer-managed policy to an IAM identity, see Create an IAM Policy (Console) and Attaching IAM Policies (Console) in the IAM User Guide.

Note
The following examples use the US East (Ohio) Region (us-east-2), a fictitious AWS account ID (123456789012), and a fictitious AWS Cloud9 development environment ID (81e900317347585a0601e04c8d52eaEX).

Topics

- Get information about environments (p. 528)
- Create EC2 environments (p. 528)
- Create EC2 environments with specific Amazon EC2 instance types (p. 529)
- Create EC2 environments in specific Amazon VPC subnets (p. 529)
- Create an EC2 environment with a specific environment name (p. 530)
- Create SSH environments only (p. 530)
- Update environments or prevent updating an environment (p. 530)
- Get lists of environment members (p. 531)
- Share environments only with a specific user (p. 531)
- Prevent sharing environments (p. 532)
- Change, or prevent changing, the settings of environment members (p. 532)
- Remove, or prevent removing, environment members (p. 533)
- Delete, or prevent deleting, an environment (p. 533)

Get information about environments

The following example IAM policy statement, attached to an IAM entity, allows that entity to get information about any environment in their account.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "cloud9:DescribeEnvironments",
         "Resource": "*
      }
   ]
}
```

Note
The preceding access permission is already included in the AWS managed policies AWSCloud9Administrator and AWSCloud9User.

Create EC2 environments

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "cloud9:CreateEnvironments",
         "Resource": "*
      }
   ]
}
```
"Statement": [
  {
    "Effect": "Allow",
    "Action": "cloud9:CreateEnvironmentEC2",
    "Resource": "*"
  }
]

Note
The preceding access permission is already included in the AWS managed policies AWSCloud9Administrator and AWSCloud9User.

Create EC2 environments with specific Amazon EC2 instance types

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account. However, EC2 environments can use only the specified class of Amazon EC2 instance types.

{  
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:InstanceType": "t3.*"
        }
      }
    }
  ]
}

Note
If the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, that AWS managed policy will override the behavior of the preceding IAM policy statement. This is because those AWS managed policies are more permissive.

Create EC2 environments in specific Amazon VPC subnets

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 EC2 development environments in their account. However, EC2 environments can use only the specified Amazon VPC subnets.

{  
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloud9:SubnetId": [  
            "subnet-12345678",
            "subnet-23456789"
          ]
        }
      }
    }
  ]
}
Note
If the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, that AWS managed policy will override the behavior of the preceding IAM policy statement. This is because those AWS managed policies are more permissive.

Create an EC2 environments with a specific environment name

The following example IAM policy statement, attached to an IAM entity, allows that entity to create an AWS Cloud9 EC2 development environment in their account. However, the EC2 environment can use only the specified name.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloud9:EnvironmentName": "my-demo-environment"
        }
      }
    }
  ]
}
```

Note
If the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, that AWS managed policy will override the behavior of the preceding IAM policy statement. This is because those AWS managed policies are more permissive.

Create SSH environments only

The following example IAM policy statement, attached to an IAM entity, allows that entity to create AWS Cloud9 SSH development environments in their account. However, the entity cannot create AWS Cloud9 EC2 development environments.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentSSH",
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": "cloud9:CreateEnvironmentEC2",
      "Resource": "*"
    }
  ]
}
```

Update environments or prevent updating an environment

The following example IAM policy statement, attached to an IAM entity, allows that entity to change information about any AWS Cloud9 development environment in their account.
AWS Cloud9 User Guide
How AWS Cloud9 works with IAM

Note
The preceding access permission is already included in the AWS managed policy
AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity
from changing information about the environment with the specified ARN.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:UpdateEnvironment",
      "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```

Get lists of environment members

The following example IAM policy statement, attached to an IAM entity, allows that entity to get a list of
members for any environment in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DescribeEnvironmentMemberships",
      "Resource": "*
```

Note
The preceding access permission is already included in the AWS managed policy
AWSCloud9Administrator. Also, the preceding access permission is more permissive than the
equivalent access permission in the AWS managed policy AWSCloud9User.

Share environments only with a specific user

The following example IAM policy statement, attached to an IAM entity, allows that entity to share any
environment in their account with only the specified user.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:CreateEnvironmentShare",
      "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```
"Effect": "Allow",
"Action": [
"cloud9:CreateEnvironmentMembership"
],
"Resource": "*",
"Condition": {
"StringEquals": {
"cloud9:UserArn": "arn:aws:iam::123456789012:user/MyDemoUser"
}
]
}

Note
If the AWS managed policy AWSCloud9Administrator or AWSCloud9User is already attached to the IAM entity, those AWS managed policies will override the behavior of the preceding IAM policy statement. This is because those AWS managed policies are more permissive.

Prevent sharing environments
The following example IAM policy statement, attached to an IAM entity, prevents that entity from sharing any environment in their account.

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Deny",
"Action": [
"cloud9:CreateEnvironmentMembership",
"cloud9:UpdateEnvironmentMembership"
],
"Resource": "*"
}
]
}
```

Change, or prevent changing, the settings of environment members
The following example IAM policy statement, attached to an IAM entity, allows that entity to change the settings of members in any environment in their account.

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": "cloud9:UpdateEnvironmentMembership",
"Resource": "*"
}
]
}
```

Note
The preceding access permission is already included in the AWS managed policy AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from changing the settings of members in the environment with the specified ARN.
Remove, or prevent removing, environment members

The following example IAM policy statement, attached to an IAM entity, allows that entity to remove any member from any environment in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DeleteEnvironmentMembership",
      "Resource": "*"
    }
  ]
}
```

Note

The preceding access permission is already included in the AWS managed policy AWSCloud9Administrator.

Delete, or prevent deleting, an environment

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity from removing any member from the environment with the specified ARN.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "cloud9:DeleteEnvironmentMembership",
      "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"
    }
  ]
}
```

Delete, or prevent deleting, an environment

The following example IAM policy statement, attached to an IAM entity, allows that entity to delete any environment in their account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "cloud9:DeleteEnvironment",
      "Resource": "*"
    }
  ]
}
```
Note
The preceding access permission is already included in the AWS managed policy
AWSCloud9Administrator.

The following example IAM policy statement, attached to an IAM entity, explicitly prevents that entity
from deleting the environment with the specified ARN.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Deny",  
      "Action": "cloud9:DeleteEnvironment",  
      "Resource": "arn:aws:cloud9:us-east-2:123456789012:environment:81e900317347585a0601e04c8d52eaEX"  
    }  
  ]
}
```

AWS Cloud9 permissions reference

You can use the following table as a reference when you are setting up access control and writing
permissions policies to attach to an IAM identity (identity-based policies).

You can use AWS-wide condition keys in your AWS Cloud9 policies to express conditions. For a list, see
IAM JSON Policy Elements: Condition in the IAM User Guide.

You specify the actions in the policy's Action field. To specify an action, use the cloud9: prefix
followed by the API operation name (for example, "Action": "cloud9:DescribeEnvironments").
To specify multiple actions in a single statement, separate them with commas (for example, "Action":
[ "cloud9:UpdateEnvironment", "cloud9:DeleteEnvironment" ]).

Topics
• Using wildcard characters (p. 534)
• AWS Cloud9 API operations and required permissions for actions (p. 535)

Using wildcard characters

You specify an ARN, with or without a wildcard character (*), as the resource value in the policy's
Resource field. You can use a wildcard to specify multiple actions or resources. For example, cloud9:*
specifies all AWS Cloud9 actions and cloud9:Describe* specifies all AWS Cloud9 actions that begin
with Describe.

The following example allows an IAM entity to get information about environments and environment
memberships for any environment in their account.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "cloud9:Describe*"
      ],  
      "Resource": "*"
    }  
  ]
}
```
### How AWS Cloud9 works with IAM

**Note**

The preceding access permission is already included in the AWS managed policy AWSCloud9Administrator. Also, that the preceding access permission is more permissive than the equivalent access permission in the AWS managed policy AWSCloud9User.

### AWS Cloud9 API operations and required permissions for actions

<table>
<thead>
<tr>
<th>AWS Cloud9 operation</th>
<th>Required permission (API action)</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateEnvironmentEC2</td>
<td>cloud9:CreateEnvironmentEC2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to create an AWS Cloud9 EC2 development environment.</td>
<td></td>
</tr>
<tr>
<td>CreateEnvironmentMembership</td>
<td>cloud9:CreateEnvironmentMembership</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to add a member to an environment.</td>
<td></td>
</tr>
<tr>
<td>DeleteEnvironment</td>
<td>cloud9:DeleteEnvironment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to delete an environment.</td>
<td></td>
</tr>
<tr>
<td>DeleteEnvironmentMembership</td>
<td>cloud9:DeleteEnvironmentMembership</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to remove a member from an environment.</td>
<td></td>
</tr>
<tr>
<td>DescribeEnvironmentMemberships</td>
<td>cloud9:DescribeEnvironmentMemberships</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to get a list of members in an environment.</td>
<td></td>
</tr>
<tr>
<td>DescribeEnvironments</td>
<td>cloud9:DescribeEnvironments</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to get information about an environment.</td>
<td></td>
</tr>
<tr>
<td>DescribeEnvironmentStatus</td>
<td>cloud9:DescribeEnvironmentStatus</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to get information about the status of an environment.</td>
<td></td>
</tr>
<tr>
<td>UpdateEnvironment</td>
<td>cloud9:UpdateEnvironment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to update settings for an environment.</td>
<td></td>
</tr>
<tr>
<td>UpdateEnvironmentMembership</td>
<td>cloud9:UpdateEnvironmentMembership</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Required to update settings for a member in an environment.</td>
<td></td>
</tr>
</tbody>
</table>
AWS managed temporary credentials

If you’re just looking for the list of actions that AWS managed temporary credentials supports, skip ahead to Actions supported by AWS managed temporary credentials (p. 537).

For an AWS Cloud9 EC2 development environment, AWS Cloud9 makes temporary AWS access credentials available to you in the environment. We call these AWS managed temporary credentials. This provides the following benefits:

- You don’t need to store the permanent AWS access credentials of an AWS entity (for example, an IAM user) anywhere in the environment. This prevents those credentials from being accessed by environment members without your knowledge and approval.
- You don’t need to manually set up, manage, or attach an instance profile to the Amazon EC2 instance that connects to the environment. (An instance profile is another approach for managing temporary AWS access credentials.)
- AWS Cloud9 continually renews its temporary credentials, so a single set of credentials can be used only for a limited time. This is an AWS security best practice. For more information, see Creating and updating AWS managed temporary credentials (p. 538).
- AWS Cloud9 puts additional restrictions on how its temporary credentials can be used to access AWS actions and resources from the environment. This is also an AWS security best practice.

Here’s how AWS managed temporary credentials work whenever an EC2 environment tries to access an AWS service on behalf of an AWS entity (for example, an IAM user):

1. AWS Cloud9 checks to see if the calling AWS entity (for example, the IAM user) has permissions to take the requested action for the requested resource in AWS. If the permission doesn’t exist or is explicitly denied, the request fails.
2. AWS Cloud9 checks AWS managed temporary credentials to see if its permissions allow the requested action for the requested resource in AWS. If the permission doesn’t exist or is explicitly denied, the request fails. For a list of permissions that AWS managed temporary credentials support, see Actions supported by AWS managed temporary credentials (p. 537).

- If both the AWS entity and AWS managed temporary credentials allow the requested action for the requested resource, the request succeeds.
- If either the AWS entity or AWS managed temporary credentials explicitly deny (or fail to explicitly allow) the requested action for the requested resource, the request fails. This means that even if the calling AWS entity has the correct permissions, the request will fail if AWS Cloud9 doesn’t also explicitly allow it. Likewise, if AWS Cloud9 allows a specific action to be taken for a specific resource, the request will fail if the AWS entity doesn’t also explicitly allow it.

The owner of an EC2 environment can turn on or off AWS managed temporary credentials for that environment at any time, as follows:

1. With the environment open, in the AWS Cloud9 IDE, on the menu bar choose AWS Cloud9, Preferences.
2. On the Preferences tab, in the navigation pane, choose AWS Settings, Credentials.
3. Use AWS managed temporary credentials to turn AWS managed temporary credentials on or off.

If you turn off AWS managed temporary credentials, by default the environment cannot access any AWS services, regardless of the AWS entity who makes the request. If you cannot or do not want to turn on
AWS managed temporary credentials for an environment, but you still need the environment to access AWS services, consider the following alternatives:

- Attach an instance profile to the Amazon EC2 instance that connects to the environment. For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71).
- Store your permanent AWS access credentials in the environment, for example, by setting special environment variables or by running the `aws configure` command. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

The preceding alternatives override all permissions that are allowed (or denied) by AWS managed temporary credentials in an EC2 environment.

**Actions supported by AWS managed temporary credentials**

For an AWS Cloud9 EC2 development environment, AWS managed temporary credentials allow all AWS actions for all AWS resources in the caller's AWS account, with the following restrictions:

- For AWS Cloud9, only the following actions are allowed:
  - `cloud9:CreateEnvironmentEC2`
  - `cloud9:CreateEnvironmentSSH`
  - `cloud9:DescribeEnvironmentMemberships`
  - `cloud9:DescribeEnvironments`
  - `cloud9:DescribeEnvironmentStatus`
  - `cloud9:UpdateEnvironment`
- For IAM, only the following actions are allowed:
  - `iam:AttachRolePolicy`
  - `iam:ChangePassword`
  - `iam:CreatePolicy`
  - `iam:CreatePolicyVersion`
  - `iam:CreateRole`
  - `iam:CreateServiceLinkedRole`
  - `iam:DeletePolicy`
  - `iam:DeletePolicyVersion`
  - `iam:DeleteRole`
  - `iam:DeleteRolePolicy`
  - `iam:DeleteSSHPublicKey`
  - `iam:DetachRolePolicy`
  - `iam:GetInstanceProfile`
  - `iam:GetPolicy`
  - `iam:GetPolicyVersion`
  - `iam:GetRole`
  - `iam:GetRolePolicy`
  - `iam:GetSSHPublicKey`
  - `iam:GetUser`
  - `iam:List*`
  - `iam:PassRole`
  - `iam:PutRolePolicy`
  - `iam:SetDefaultPolicyVersion`
  - `iam:UpdateAssumeRolePolicy`
• `iam:UpdateRoleDescription`
• `iam:UpdateSSHPublicKey`
• `iam:UploadSSHPublicKey`
• All IAM actions that interact with roles are allowed only for role names starting with `Cloud9-`. However, `iam:PassRole` works with all role names.
• For AWS Security Token Service (AWS STS), only the following actions are allowed:
  • `sts:GetCallerIdentity`
  • `sts:DecodeAuthorizationMessage`
• All supported AWS actions are restricted to the IP address of the environment. This is an AWS security best practice.

If AWS Cloud9 doesn't support an action or resource that you need an EC2 environment to access, or if AWS managed temporary credentials is turned off for an EC2 environment and you cannot turn it back on, consider the following alternatives:

• Attach an instance profile to the Amazon EC2 instance that connects to the EC2 environment. For instructions, see Create and Use an Instance Profile to Manage Temporary Credentials (p. 71).
• Store your permanent AWS access credentials in the EC2 environment, for example, by setting special environment variables or by running the `aws configure` command. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

The preceding alternatives override all permissions that are allowed (or denied) by AWS managed temporary credentials in an EC2 environment.

**Creating and updating AWS managed temporary credentials**

For an AWS Cloud9 EC2 development environment, AWS managed temporary credentials are created the first time you open the environment.

AWS managed temporary credentials are updated under any of the following conditions:

• Whenever a certain period of time passes. Currently, this is every five minutes.
• Whenever you reload the web browser tab that displays the IDE for the environment.
• When the timestamp that is listed in the `~/.aws/credentials` file for the environment is reached.
• If the **AWS managed temporary credentials** setting is set to off, whenever you turn it back on. (To view or change this setting, choose **AWS Cloud9, Preferences** in the menu bar of the IDE. On the **Preferences** tab, in the navigation pane, choose **AWS Settings, Credentials**.)

**Logging and monitoring in AWS Cloud9**

**Monitoring activity with CloudTrail**

AWS Cloud9 is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Cloud9. CloudTrail captures all API calls for AWS Cloud9 as events. The calls captured include calls from the AWS Cloud9 console and from code calls to the AWS Cloud9 APIs.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for AWS Cloud9.

If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event History**. Using the information collected by CloudTrail, you can determine the request that was made.
to AWS Cloud9, the IP address from which the request was made, who made the request, when it was made, and additional details.

For more information, see Logging AWS Cloud9 API Calls with AWS CloudTrail (p. 507).

**Monitoring EC2 environment performance**

If you’re using an AWS Cloud9 EC2 development environment, you can monitor the reliability, availability, and performance of the associated Amazon EC2 instance. With instance status monitoring, for example, you can quickly determine whether Amazon EC2 has detected any problems that might prevent your instances from running applications.

For more information, see Monitoring Amazon EC2 in the *Amazon EC2 User Guide for Linux Instances*.

**Compliance validation for AWS Cloud9**

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs.

**At present, AWS Cloud9 is not in scope of any specific compliance programs.**

For a list of AWS services that are in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports by using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS Cloud9 is determined by the sensitivity of your data, your company’s compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- **Security and Compliance Quick Start Guides** – These deployment guides discuss architectural considerations and provide steps for deploying security-focused and compliance-focused baseline environments on AWS.
- **Architecting for HIPAA Security and Compliance Whitepaper** – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- **AWS Compliance Resources** – This collection of workbooks and guides might apply to your industry and location.
- **Evaluating Resources with Rules** in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- **AWS Security Hub** – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

**Resilience in AWS Cloud9**

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.
In addition to the AWS global infrastructure, AWS Cloud9 supports specific features to support your data resiliency and backup needs.

- Integrate AWS Cloud9 with AWS CodeCommit, a version control service hosted by Amazon Web Services that you can use to privately store and manage assets (such as documents, source code, and binary files) in the cloud. For more information, see Integrate AWS Cloud9 with AWS CodeCommit in the AWS CodeCommit User Guide.

- Use the Git version control system on AWS Cloud9 development environments to back up files and data on a remote GitHub repository. For more information, see GitHub Sample for AWS Cloud9 (p. 361).

Infrastructure security in AWS Cloud9

As a managed service, AWS Cloud9 is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access AWS Cloud9 through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend TLS 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Note
By default, AWS Cloud9 EC2 development environments automatically install security patches for the instances’ system packages.

Configuration and vulnerability analysis in AWS Cloud9

AWS Cloud9 development environments run on top of cloud-compute resources. The cloud-compute resource can be an Amazon EC2 instance (for an EC2 environment) or your own cloud-compute resource (for an SSH environment). These possibilities are described in What Is AWS Cloud9? (p. 1) in the Environments and Computing Resources section.

Regardless of the underlying cloud-compute resource, it is the responsibility of the AWS Cloud9 user or their AWS Cloud9 administrator to keep the cloud-compute resource patched and up to date. If an EC2 environment is chosen when creating the development environment, the initial creation of that environment will have the latest updates. But even that environment will need to be checked from time to time, depending on the requirements of your organization.

Security best practices for AWS Cloud9

The following best practices are general guidelines and don’t represent a complete security solution. Because these best practices might not be appropriate or sufficient for your environment, treat them as helpful considerations instead of prescriptions.

Some security best practices for AWS Cloud9

- Store your code securely in a version control system, for example, AWS CodeCommit.
• For your AWS Cloud9 EC2 development environments, configure and use Amazon Elastic Block Store encrypted volumes.
• For your EC2 environments, use tags (p. 516) to control access to your AWS Cloud9 resources.
• For your shared AWS Cloud9 development environments, follow the best practices (p. 91) for them.
Troubleshooting AWS Cloud9

Use the following information to help you identify and address issues with AWS Cloud9.

If your issue is not listed, or if you need additional help, see the AWS Cloud9 Discussion Forum. (When you enter this forum, AWS might require you to sign in.) You can also contact us directly.

Topics

- Environment Creation Error: "We are unable to create EC2 instances ..." (p. 542)
- Environment Creation Error: "Not authorized to perform sts:AssumeRole" (p. 543)
- Console Error: "User is not authorized to perform action on resource" (p. 543)
- Federated Identities Cannot Create Environments (p. 544)
- Cannot Open an Environment (p. 544)
- The AWS Cloud9 Installer Hangs or Fails (p. 545)
- SSH Environment Error: "Python version 2.7 is required to install pty.js" (p. 545)
- Application Preview or File Preview Notice: "Third-party cookies disabled" (p. 546)
- Application Preview Tab Displays an Error or is Blank (p. 548)
- Cannot Display Your Running Application Outside of the IDE (p. 549)
- After Reloading an Environment, You Must Refresh Application Preview (p. 550)
- Unable to Preview Application in the AWS Cloud9 IDE with HTTP (p. 551)
- Cannot Run Some Commands or Scripts in an EC2 Environment (p. 551)
- AWS CLI / aws-shell Error: "The security token included in the request is invalid" in an EC2 environment (p. 551)
- Amazon EC2 Instances Are Not Automatically Updated (p. 552)
- Lambda Local Function Run Error: Cannot Install SAM Local (p. 552)
- IDE Warning: "This Environment is Running Low on Memory" or "This Environment Has High CPU Load" (p. 553)
- Previewing a File Returns a 499 Error (p. 554)
- Environment Deletion Error: "One or more environments failed to delete" (p. 554)
- Console Warning: "Switching to the minimal code completion engine..." (p. 555)
- AWS Cloud9 Installer doesn’t finish after displaying: “Package Cloud9 IDE 1" (p. 555)
- VPC error for EC2-Classic accounts: "Unable to access your environment" (p. 556)

Environment Creation Error: "We are unable to create EC2 instances ..."

Issue: When you try to create an AWS Cloud9 development environment, a message appears with the phrase "We are unable to create EC2 instances in your account during account verification and activation."

Cause: AWS is currently verifying and activating your AWS account. Until activation is complete, which could take up to 24 hours, you can't create this or other environments.

Solution: Try creating the environment again later. If you're still receiving this message after 24 hours, email aws-verification@amazon.com. Be advised that AWS CloudFormation creates a related stack in your account, even though the attempt to create an environment fails. These stacks count against the stack creation limit in your account. To help avoid the stack creation limit, you can safely delete these
Environment Creation Error: "Not authorized to perform sts:AssumeRole"

**Issue:** When you try to create a new environment, you see this error: "Not authorized to perform sts:AssumeRole," and the environment is not created.

**Possible causes:** An AWS Cloud9 service-linked role doesn't exist in your AWS account.

**Recommended solutions:** Create an AWS Cloud9 service-linked role in your AWS account by running the following command with the AWS Command Line Interface (AWS CLI) or the aws-shell.

```bash
aws iam create-service-linked-role --aws-service-name cloud9.amazonaws.com # For the AWS CLI.
iam create-service-linked-role --aws-service-name cloud9.amazonaws.com # For the aws-shell.
```

If you cannot do this, check with your AWS account administrator.

After you run this command, try creating the environment again.

---

Console Error: "User is not authorized to perform action on resource"

**Issue:** When you try to use the AWS Cloud9 console to create or manage an AWS Cloud9 development environment, you see an error that contains a phrase similar to "User arn:aws:iam::123456789012:user/MyUser is not authorized to perform cloud9:action on resource arn:aws:cloud9:us-east-2:123456789012:environment:12a34567b8cd9012345ef67abcd890e1," where:

- **arn:aws:iam::123456789012:user/MyUser** is the Amazon Resource Name (ARN) of the requesting user.
- **action** is the name of the operation that the user requested.
- **arn:aws:cloud9:us-east-2:123456789012:environment:12a34567b8cd9012345ef67abcd890e1** is the ARN of the environment that the user requested to run the operation.

**Cause:** The user you signed in to the AWS Cloud9 console with doesn't have the correct AWS access permissions to perform the action.

**Solution:** Ensure the user has the correct AWS access permissions, and then try to perform the action again. For more information, see one or more of the following:

- **Step 3: Add AWS Cloud9 Access Permissions to the Group (p. 12)** in *Team Setup*
- **Step 6. Enable Groups and Users within the Organization to Use AWS Cloud9 (p. 20)** in *Enterprise Setup*
- **About Environment Member Access Roles (p. 82)** in *Working with Shared Environments*
Federated Identities Cannot Create Environments

**Issue:** When you try to use an AWS federated identity to create an AWS Cloud9 development environment, an access error message is displayed, and the environment isn’t created.

**Cause:** AWS Cloud9 uses service-linked roles. The service-linked role is created the first time an environment is created in an account using the `iam:CreateServiceLinkedRole` call. However, federated users can't call IAM APIs. For more information, see `GetFederationToken` in the AWS Security Token Service API Reference.

**Solution:** Ask an AWS account administrator to create the service-linked role for AWS Cloud9 either in the IAM console or by running this command with the AWS Command Line Interface (AWS CLI):

```
aws iam create-service-linked-role --aws-service-name cloud9.amazonaws.com
```

Or this command with the aws-shell:

```
iam create-service-linked-role --aws-service-name cloud9.amazonaws.com
```

For more information, see Using Service-Linked Roles in the IAM User Guide.

---

Cannot Open an Environment

**Issue:** When you try to open an environment, the IDE does not display for a long time (after at least five minutes).

**Possible causes:**

- The IAM user that is signed in to the AWS Cloud9 console does not have the required AWS access permissions to open the environment.
- If the environment is associated with an AWS cloud compute instance (for example an Amazon EC2 instance):
  - The instance's associated VPC is not set to the correct settings for AWS Cloud9.
  - The instance is transitioning between states or is failing automated status checks, during the time when AWS Cloud9 is trying to connect to the instance.
- If the environment is an SSH environment, the associated cloud compute instance or your own server is not set up correctly to allow AWS Cloud9 to access it.

**Recommended solutions:**

- Make sure the IAM user that is signed in to the AWS Cloud9 console has the required AWS access permissions to open the environment, and then try opening the environment again. For more information see the following, or check with your AWS account administrator:
  - Step 3: Add AWS Cloud9 Access Permissions to the Group (p. 12) in Team Setup
  - AWS managed (predefined) policies for AWS Cloud9 (p. 524) in Authentication and Access Control
  - Customer-Managed Policy Examples for Teams Using AWS Cloud9 (p. 24) in Advanced Team Setup
  - Customer-Managed Policy Examples (p. 493) in Authentication and Access Control
  - Changing Permissions for an IAM User in the IAM User Guide
• **Troubleshoot IAM Policies in the IAM User Guide**

If the signed-in IAM user still cannot open the environment, you could try signing out and then signing back in as either the AWS account root user or an IAM administrator user in the account. Then try opening the environment again. If you are able to open the environment in this way, then there is most likely a problem with the IAM user's access permissions.

• If the environment is associated with an AWS cloud compute instance (for example an Amazon EC2 instance):

• Make sure the instance's associated VPC is set to the correct settings for AWS Cloud9, and then try opening the environment again. For details, see Amazon VPC Requirements for AWS Cloud9 (p. 467).

If the AWS cloud compute instance's associated VPC is set to the correct settings for AWS Cloud9 and you still cannot open the environment, the instance's security group might be preventing access to AWS Cloud9. As a troubleshooting technique only, check the security group to make sure that at minimum, inbound SSH traffic is allowed over port 22 for all IP addresses (Anywhere or 0.0.0.0/0). For instructions, see Describing Your Security Groups and Updating Security Group Rules in the Amazon EC2 User Guide for Linux Instances.

For additional VPC troubleshooting steps, watch the related 5-minute video AWS Knowledge Center Videos: What can I check if I cannot connect to an instance in a VPC? on the YouTube website.

**Warning**
When you have finished troubleshooting, be sure to set the inbound rules to an appropriate address range, as described in the section called “Inbound SSH IP address ranges” (p. 483).

• Restart the instance, make sure the instance is running and has passed all system checks, and then try opening the environment again. For details, see Reboot Your Instance and Viewing Status Checks in the Amazon EC2 User Guide for Linux Instances.

• If the environment is an SSH environment, make sure the associated cloud compute instance or your own server is set up correctly to allow AWS Cloud9 to access it, and then try opening the environment again. For details, see AWS Cloud9 SSH Development Environment host requirements (p. 479).

(back to top (p. 542))

### The AWS Cloud9 Installer Hangs or Fails

**Issue:** When you download and run the AWS Cloud9 Installer (p. 482), one or more error messages display, and the installer script does not show *Done*.

**Cause:** The AWS Cloud9 Installer has encountered one or more errors that it cannot recover from and therefore fails.

**Solution:** See common issues, their possible causes, and recommended solutions, in Troubleshooting the AWS Cloud9 Installer (p. 482).

(back to top (p. 542))

### SSH Environment Error: "Python version 2.7 is required to install pty.js"

**Issue:** After you open an AWS Cloud9 SSH development environment, the terminal in the AWS Cloud9 IDE displays a message that begins with "Python version 2.7 is required to install pty.js."

**Cause:** To work as expected, an SSH environment requires that Python version 2.7 is installed.
**Solution:** Install Python version 2.7 in the environment. To check your version, from your server’s terminal, run the command `python --version`. To install Python 2.7 on your server, see one of the following:

- **Step 1:** Install Python 3.6 (p. 419) in the *Python Sample*.
- **Download Python** on the Python website and **Installing Packages** in the *Python Packaging User Guide*.

---

**Application Preview or File Preview Notice: "Third-party cookies disabled"**

**Issue:** When you attempt to preview an application (p. 136) or a file (p. 134), a notice is displayed with the following message: "Preview functionality is disabled because your browser has third-party cookies disabled."

**Cause:** Although third-party cookies are not needed to open the AWS Cloud9 IDE, you must enable third-party cookies to use the Application Preview or File Preview features.

**Solution:** Enable third-party cookies in your web browser, reload your IDE, and then try opening the preview again.

- **Google Chrome:** [Change your cookie settings](https://support.google.com/chrome/answer/95645) in Clear, enable, and manage cookies in Chrome on the Google Chrome Help website.
- **Internet Explorer:** [Block or allow cookies](https://support.microsoft.com/kb/307887) in Delete and manage cookies on the Microsoft Support website.
- **Microsoft Edge:** [Blocking third-party cookies](https://support.microsoft.com/kb/4033110) on the Microsoft Support website.
- **Mozilla Firefox:** [Accept third party cookies](https://support.mozilla.org/kb/accept-third-party-cookies) setting in Enable and disable cookies that websites use to track your preferences on the Mozilla Support website.
- **Any other web browser:** see that web browser’s documentation.

To enable third-party cookies only for AWS Cloud9 (if your web browser allows this granularity), specify the following domains, depending on the supported AWS Regions where you want to use AWS Cloud9.

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VS.CLOUD9.US-EAST-1.AMAZONAWS.COM</td>
</tr>
<tr>
<td></td>
<td>VS.CLOUD9.US-EAST-2.AMAZONAWS.COM</td>
</tr>
<tr>
<td></td>
<td>VS.CLOUD9.US-WEST-1.AMAZONAWS.COM</td>
</tr>
<tr>
<td></td>
<td>VS.CLOUD9.US-WEST-2.AMAZONAWS.COM</td>
</tr>
</tbody>
</table>
### AWS Region

<table>
<thead>
<tr>
<th>AWS Region</th>
<th>Domains</th>
</tr>
</thead>
<tbody>
<tr>
<td>Asia Pacific (Hong Kong)</td>
<td>*.vfs.cloud9.ap-east-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-east-1.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Mumbai)</td>
<td>*.vfs.cloud9.ap-south-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-south-1.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Seoul)</td>
<td>*.vfs.cloud9.ap-northeast-2.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-northeast-2.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Singapore)</td>
<td>*.vfs.cloud9.ap-southeast-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-southeast-1.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Sydney)</td>
<td>*.vfs.cloud9.ap-southeast-2.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-southeast-2.amazonaws.com</td>
</tr>
<tr>
<td>Asia Pacific (Tokyo)</td>
<td>*.vfs.cloud9.ap-northeast-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ap-northeast-1.amazonaws.com</td>
</tr>
<tr>
<td>Canada (Central)</td>
<td>*.vfs.cloud9.ca-central-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.ca-central-1.amazonaws.com</td>
</tr>
<tr>
<td>Europe (Frankfurt)</td>
<td>*.vfs.cloud9.eu-central-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.eu-central-1.amazonaws.com</td>
</tr>
<tr>
<td>Europe (Ireland)</td>
<td>*.vfs.cloud9.eu-west-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.eu-west-1.amazonaws.com</td>
</tr>
<tr>
<td>Europe (London)</td>
<td>*.vfs.cloud9.eu-west-2.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.eu-west-2.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.eu-west-3.amazonaws.com</td>
</tr>
<tr>
<td>Europe (Stockholm)</td>
<td>*.vfs.cloud9.eu-north-1.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>vfs.cloud9.eu-north-1.amazonaws.com</td>
</tr>
</tbody>
</table>
Application Preview Tab Displays an Error or is Blank

**Issue:** On the menu bar in the IDE, when you choose Preview, Preview Running Application or Tools, Preview, Preview Running Application to try to display your application on a preview tab in the IDE, the tab displays an error, or the tab is blank.

**Possible causes:**

- Your application is not running in the IDE.
- Your application is not running using HTTP.
- Your application is running over more than one port.
- Your application is running over a port other than 8080, 8081, or 8082.
- Your application is running with an IP other than 127.0.0.1, localhost, or 0.0.0.0.
- The port (8080, 8081, or 8082) is not specified in the URL on the preview tab.
- Your network blocks inbound traffic to ports 8080, 8081, or 8082.
- You are trying to go to an address that contains an IP of 127.0.0.1, localhost, or 0.0.0.0. The default, built-in behavior of the AWS Cloud9 IDE is that this will attempt to go to your local computer instead of attempting to go the instance or your own server that is connected to the environment.

**Recommended solutions:**

- Ensure that the application is running in the IDE.
- Ensure that the application is running using HTTP. For examples in Node.js and Python, see Run an Application (p. 136).
- Ensure that the application is running over only one port. For examples in Node.js and Python, see Run an Application (p. 136).
- Ensure that the application is running over port 8080, 8081, or 8082. For examples in Node.js and Python, see Run an Application (p. 136).
- Ensure that the application is running with an IP of 127.0.0.1, localhost, or 0.0.0.0. For examples in Node.js and Python, see Run an Application (p. 136).
- Add :8080, :8081, or :8082 to the URL on the preview tab.
- Ensure that your network allows inbound traffic over ports 8080, 8081, or 8082. If you cannot make changes to your network, see your network administrator.
- If you are trying to go to an address that contains an IP of 127.0.0.1, localhost, or 0.0.0.0, try going to the following address instead:
Cannot Display Your Running Application Outside of the IDE

**Issue:** When you or others try to display your running application in a web browser tab outside of the IDE, that web browser tab displays an error, or the tab is blank.

**Possible causes:**

- The application is not running in the IDE.
- The application is running with an IP of `127.0.0.1` or `localhost`.
- The application is running in an AWS Cloud9 EC2 development environment, and one or more security groups that are associated with the corresponding Amazon EC2 instance do not allow inbound traffic over the protocols, ports, or IP addresses that the application requires.
- The application is running in an AWS Cloud9 SSH development environment for an AWS cloud compute instance (for example an Amazon EC2 instance), and the network ACL for the subnet in the virtual private cloud (VPC) that is associated with the corresponding instance does not allow inbound traffic over the protocols, ports, or IP addresses that the application requires.
- The URL is incorrect.
- The URL in the application preview tab is being requested instead of the instance's public IP address.
- You are trying to go to an address that contains an IP of `127.0.0.1` or `localhost`. These IPs will attempt to access resources on your local computer instead of resources in the environment.
- The instance's public IP address has changed.
- The web request originates from a virtual private network (VPN) that blocks traffic over the protocols, ports, or IP addresses that the application requires.
- The application is running in an SSH environment, and your server or the associated network does not allow traffic over the protocols, ports, or IP addresses that the application requires.

**Recommended solutions:**

- Ensure that the application is running in the IDE.
- Ensure that the application is not running with an IP of `127.0.0.1` or `localhost`. For some examples in Node.js and Python, see Run an Application (p. 136).
- If the application is running on an AWS cloud compute instance (for example an Amazon EC2 instance), ensure all security groups that are associated with the corresponding instance allow inbound traffic.
After Reloading an Environment, You Must Refresh Application Preview

**Issue:** After you reload an environment that displays an application preview tab, the tab doesn't display the application preview.

**Cause:** Sometimes users write code that can run an infinite loop or that otherwise uses so much memory that the AWS Cloud9 IDE can pause or stop when the application preview is running. To keep this from happening, AWS Cloud9 doesn't reload application preview tabs whenever an environment is reloaded.

**Solution:** After you reload an environment that displays an application preview tab, to display the application preview, choose the **Click to load the page** button on the tab.
Unable to Preview Application in the AWS Cloud9 IDE with HTTP

**Issue:** In the address box of an application preview tab in the AWS Cloud9 IDE, the URL always starts with https. If you try to change https in the box to http and then press Enter, the tab doesn't display the application preview.

**Cause:** To help improve code safety, in the address box of the application preview tab in the IDE, AWS Cloud9 always uses https. This behavior cannot be changed.

**Solution:** To view an application preview with an address starting with http instead of https, change https in the address box of the tab to http and then press Enter. Then choose the Open your page in a new tab button. This displays the application preview in a separate web browser tab using HTTP.

Cannot Run Some Commands or Scripts in an EC2 Environment

**Issue:** After you open an AWS Cloud9 EC2 development environment, you cannot install some types of packages, run commands such as yum or apt, or run scripts containing commands that typically work with other Linux operating systems.

**Cause:** The Amazon EC2 instances that AWS Cloud9 uses for an EC2 environment rely on either Amazon Linux (which is based on Red Hat Enterprise Linux (RHEL)) or Ubuntu Server.

**Solution:** If you install or manage packages or run commands or scripts in the IDE for an EC2 environment, ensure they are compatible with either RHEL (for Amazon Linux) or Ubuntu Server, depending on the instance for that environment.

AWS CLI / aws-shell Error: "The security token included in the request is invalid" in an EC2 environment

**Issue:** When you try to use the AWS Command Line Interface (AWS CLI) or the aws-shell to run a command in the AWS Cloud9 IDE for an EC2 environment, an error displays: "The security token included in the request is invalid."

**Possible causes:**

- If you have AWS managed temporary credentials enabled, you are trying to run a command that is not allowed with those AWS managed temporary credentials. For a list of allowed commands, see Actions supported by AWS managed temporary credentials (p. 537).
- If you have AWS managed temporary credentials enabled and the environment is a shared environment, the environment owner has not opened the environment within the past 12 hours so that AWS Cloud9 can refresh AWS managed temporary credentials in the environment. (AWS Cloud9 sets this 12-hour limit as an AWS security best practice.)
Recommended solutions:

- If you have AWS managed temporary credentials enabled, run allowed commands only. If you must run a command that is not allowed by AWS managed temporary credentials, one approach would be to configure the AWS CLI or aws-shell in the environment with a set of permanent credentials, which removes this limitation. For instructions, see Create and Store Permanent Access Credentials in an Environment (p. 75).

- Have the environment owner open the environment so that AWS Cloud9 can refresh temporary credentials in the environment.

For more information, see AWS managed temporary credentials (p. 536).

Lambda Local Function Run Error: Cannot Install SAM Local

Issue: After you try to run the local version of an AWS Lambda function in the AWS Cloud9 IDE, a dialog box is displayed, stating that AWS Cloud9 is having trouble installing SAM Local. AWS Cloud9 needs SAM Local to run local versions of AWS Lambda functions in the IDE. Until SAM Local is installed, you cannot run local versions of Lambda functions in the IDE.

Cause: AWS Cloud9 can't find SAM Local at the expected path in the environment, which is ~/.c9/bin/sam. This is because SAM Local is not yet installed, or if it is installed, AWS Cloud9 can't find it at that location.

Recommended solutions: You can wait for AWS Cloud9 to try to finish installing SAM Local, or you can install it yourself.
To see how AWS Cloud9 is doing with attempting to install SAM Local, choose Window, Installer on the menu bar.

To install SAM Local yourself, run the following commands, one at a time in the following order, from a terminal session in the IDE.

```
npm install -g aws-sam-local        # Use Node Package Manager (npm) to install SAM Local as a global package in the environment.
ln -sfn $(which sam) ~/.c9/bin/sam  # Create a symbolic link (a shortcut) from the path that AWS Cloud9 expects to where SAM Local is installed.
```

If, after running the previous commands, you're still having SAM Local install issues, try running the following additional commands, one at a time in the following order, from a terminal session in the IDE.

```
npm uninstall -g aws-sam-local  # Use npm to uninstall the globally-installed SAM Local from the environment.
rm -rf $(which sam)             # Remove the related symbolic link.
pip install --user aws-sam-cli  # Use pip to re-install the AWS SAM CLI from the context of the user (not globally).
hash -r                        # Reset the bash cache (removes all current tracked aliases).
sam --version                  # Verify that your installation worked.
```

For more information, see the aws labs/aws-sam-cli repository on the GitHub website.

(Back to top (p. 542))

IDE Warning: "This Environment is Running Low on Memory" or "This Environment Has High CPU Load"

**Issue:** While the IDE is running, you see a message that contains the phrase "this environment is running low on memory" or "this environment has high CPU load."

**Cause:** The IDE might not have enough compute resources available to continue running without delays or hangs.

**Recommended solutions:**

- Stop one or more running processes to free up available memory. To do this, on the menu bar in the IDE for the environment, choose Tools, Process List. For each process you want to stop, choose the process, and then choose Force Kill.
- Create a swap file in the environment. A swap file is a file in the environment that the operating system can use as virtual memory.

To confirm whether the environment is currently using swap memory, run the top command in a terminal session in the environment. If swap memory is being used, the output displays non-zero Swap memory statistics (for example, Swap: 499996k total, 1280k used, 498716 free, 110672k cached). To stop showing real-time memory information, press Ctrl + C.

To create a swap file, you could run a command such as the following in the environment.

```
sudo fallocate --length 512MB /var/swapfile && sudo chmod 600 /var/swapfile && sudo mkswap /var/swapfile && echo '/var/swapfile swap swap defaults 0 0' > /dev/null
```
The preceding command does the following:
1. Creates a 512 MB file named `swapfile` in the `/var` directory.
2. Changes access permissions for the `swapfile` file to read-write for the owner only.
3. Sets up the `swapfile` file as a swap file.
4. Writes information to the `/etc/fstab` file, which makes this swap file available whenever the system reboots.

After you run the preceding command, to make this swap file available immediately instead of waiting for a reboot, run the following command.

```
sudo swapon /var/swapfile
```

- Move or resize the environment to an instance or server with more compute resources. To move or resize Amazon EC2 instances, see Moving an environment or resizing an Amazon EBS volume (p. 92). For other instance or server types, refer to your instance’s or server’s documentation.

Previewing a File Returns a 499 Error

**Issue:** When you try to use the AWS Cloud9 IDE to preview a file that contains a `<script>` element containing the `src` attribute and with the `type` attribute set to `module`, a 499 error occurs and the script doesn't run as expected.

**Cause:** File preview fetch requests in the AWS Cloud9 IDE require cookies to be sent by the web browser to authenticate. By default, web browsers send cookies for regular script requests, but not for module script requests, unless you add the `crossorigin` attribute.

**Solution:** Add the `crossorigin` attribute to the `<script>` element. For example, `<script type="module" src="index.js" crossorigin></script>`. Then save the changed file, and try to preview the it again.

Environment Deletion Error: "One or more environments failed to delete"

**Issue:** When you try to delete one or more environments in the AWS Cloud9 console, a message is displayed that reads "one or more environments failed to delete," and at least one of the environments is not deleted.

**Possible cause:** AWS CloudFormation might have a problem deleting one or more of the environments. (AWS Cloud9 relies on AWS CloudFormation to create and delete environments.)

**Recommended solution:** Try using AWS CloudFormation to delete each of the undeleted environments, as follows.

2. On the AWS navigation bar, choose the AWS Region for the environment.
3. In the list of AWS CloudFormation stacks, select the entry where `Stack name` contains the undeleted environment name and `Status` is `DELETE_FAILED`. For example, if the environment name is
my-demo-environment, choose the stack that begins with the name `aws-cloud9-my-demo-environment`. (Choose the box or option next to the environment name, not the environment name itself.)

4. Choose Actions, Delete Stack.
5. If prompted, choose Yes, Delete.

The process of deleting a stack might take a few minutes.

If the stack disappears from the list, the environment is now deleted.

If the stack is still displayed displayed with DELETE_FAILED after a few minutes, the environment is still not deleted. In this case, you can try to manually delete each of the failed stack's resources.

**Note**
Manually deleting a failed stack's resources doesn't remove the stack itself from your AWS account.

To manually delete these resources, in the AWS CloudFormation console, choose the failed stack, and then select the Resources section. Go to the console in AWS for each resource in this list, and then use that console to manually delete the resource.

(Back to top (p. 542))

**Console Warning: "Switching to the minimal code completion engine..."

**Issue:** When working in the AWS Cloud9 console (for example, when opening the IDE or refreshing the IDE's web page), you see this message: "One or more sessions or collaborators are active on this environment. Switching to the minimal code completion engine to conserve memory." In correlation with this message, the code-completion behavior might be slow or intermittent.

**Cause:** Running the code-completion engine takes memory and CPU cycles from the environment. Additionally, a separate code-completion engine is required for each collaborator and each additional session. To avoid using too many resources, especially on small instance sizes like t2.nano and t2.micro, AWS Cloud9 switches to the minimal code-completion engine.

**Recommended solution:** If you will be collaborating often and for long periods of time, choose a larger Amazon EC2 instance when creating your EC2 environment (or connect your SSH environment to an instance with more capacity).

**Note**
Choosing a larger Amazon EC2 instance might result in additional charges to your AWS account.
For more information, see Amazon EC2 Pricing.

(Back to top (p. 542))

**AWS Cloud9 Installer doesn't finish after displaying: "Package Cloud9 IDE 1"

**Issue:** AWS Cloud9 is installed on your existing Amazon EC2 instance or on your own server as part of the process of creating an SSH development environment. The installation stalls after you see this message in the **AWS Cloud9 Installer** dialog box: "Package Cloud9 IDE 1". If you choose Cancel, you see the
following message: "Installation Failed." This error occurs when AWS Cloud9 packages can't be installed on the customer's SSH host.

**Cause:** An SSH host requires that you have Node.js installed. We currently support versions from Node.js 0.6.16 to Node.js 12.x. An installation error can occur if you have a version of Node.js on your host that AWS Cloud9 doesn't support.

**Recommended solution:** Install a version of Node.js that AWS Cloud9 supports on your SSH host.

**VPC error for EC2-Classic accounts: "Unable to access your environment"**

**Issue:** EC2-Classic was introduced in the original release of Amazon EC2. If you're using an AWS account that was set up before December 4, 2013, this error might occur if you don't explicitly configure a virtual private cloud (Amazon VPC) and subnet when creating an AWS Cloud9 EC2 development environment.

If you accept the default VPC settings, the Amazon EC2 instance is launched into the EC2-Classic network and not into a subnet of the default VPC. The following message is displayed when the creation of the environment fails:

Environment Error

Unable to access your environment

The environment creation failed with the error: The following resource(s) failed to create: [Instance]. Rollback requested by user.

You can confirm that the error is caused by the EC2 instance not being in the default VPC. Use AWS CloudFormation to view the stack event history for the development environment.

1. Open the AWS CloudFormation console. For more information, see Logging in to the AWS CloudFormation console.
2. In the AWS CloudFormation console, choose **Stacks**.
3. On the **Stacks** page, choose the name of the development environment that failed to create.
4. On the **Stack details** page, choose the **Events** tab and check for the following entry:

   **Status:** CREATE_FAILED
   **Status reason:** The AssociatePublicIpAddress parameter is only supported by VPC launches. [...]

**Cause:** An AWS Cloud9 development environment must be associated with an Amazon VPC that meets specific VPC requirements. For accounts with EC2-Classic enabled, accepting the default network settings when creating an EC2 environment (p. 55) means that the required EC2 instance isn't launched into the VPC. Instead, the instance is launched into the EC2-Classic network.

**Recommended solution:** With an EC2-Classic account, you must select a VPC and subnet when creating an EC2 environment (p. 55). On the **Configure settings** page, in the **Network settings (advanced)** section, select the VPC and subnet that you can launch your EC2 instance into.
## Supported Browsers for AWS Cloud9

The following table lists the supported browsers for AWS Cloud9.

<table>
<thead>
<tr>
<th>Browser</th>
<th>Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Chrome</td>
<td>Latest three versions</td>
</tr>
<tr>
<td>Mozilla Firefox</td>
<td>Latest three versions</td>
</tr>
<tr>
<td>Microsoft Edge</td>
<td>Latest three versions</td>
</tr>
<tr>
<td>Apple Safari for macOS</td>
<td>Latest two versions</td>
</tr>
</tbody>
</table>
Limits for AWS Cloud9

The following tables list limits in AWS Cloud9 and related AWS services.

- AWS Cloud9 Limits (p. 558)
- Related AWS Service Limits (p. 559)

AWS Cloud9 Limits

The following table provides the default limits for AWS Cloud9 for an AWS account. Unless otherwise noted, each limit is Region-specific.

To request an increase for a limit that is adjustable, do the following:

1. Sign in to AWS and open the Create case page on the AWS Support Center website.
2. Select Service limit increase.
3. Under Case classification, choose cloud9 from the Limit type list.
4. Fill out the rest of the fields as appropriate. If the request is urgent, choose Phone as the method of contact instead of Web.
5. Choose Submit.

These increases are not granted immediately, so it might take a couple of days for your increase to become effective.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Limit</th>
<th>Adjustable</th>
</tr>
</thead>
</table>
| Maximum number of AWS Cloud9 EC2 development environments | • 100 per user  
• 200 per account | Yes        |
| Maximum number of SSH environments            | • 100 per user  
• 200 per account | Yes        |
| Maximum number of members in an environment   | The default maximum number of members is equal to the memory of the instance for that environment divided by 60 MB, with results rounded down. For example, an instance with 1 GiB of memory can have a maximum of 17 members (which is 1 GiB divided by 60 MB, rounded down). If AWS Cloud9 cannot determine the memory of an instance, it defaults to a maximum of 8 users for each environment associated with that instance. | No¹        |
### Related AWS Service Limits

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default Limit</th>
<th>Adjustable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of members for an environment</td>
<td>25</td>
<td>No</td>
</tr>
<tr>
<td>Maximum editable file size</td>
<td>8 MB</td>
<td>No</td>
</tr>
</tbody>
</table>

1 You can move an environment (p. 92) to attempt to increase the default maximum number of members. However, the absolute maximum number of members for an environment is still 25.
Document History for the AWS Cloud9 User Guide

This topic contains a list of significant changes to the AWS Cloud9 User Guide. For notification about updates to this documentation, you can subscribe to the RSS feed.

Recent Updates

The following table describes important changes to the AWS Cloud9 User Guide after March 2019.

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added Region support to AWS Cloud9 (p. 560)</td>
<td>AWS Cloud9 is now also available in the following Regions: US West (N. California), Asia Pacific (Hong Kong), Europe (Paris), Middle East (Bahrain), and South America (São Paulo). For more information, see AWS Cloud9 in the Amazon Web Services General Reference.</td>
<td>May 7, 2020</td>
</tr>
<tr>
<td>Tags (p. 516)</td>
<td>Use tags to help you control access to AWS Cloud9 resources and help you manage billing information.</td>
<td>January 22, 2020</td>
</tr>
<tr>
<td>Added Region support to AWS Cloud9 (p. 560)</td>
<td>AWS Cloud9 is now also available in the following Regions: Asia Pacific (Mumbai), Asia Pacific (Seoul), Asia Pacific (Sydney), Canada (Central), Europe (London), and Europe (Stockholm). For more information, see AWS Cloud9 in the Amazon Web Services General Reference.</td>
<td>December 18, 2019</td>
</tr>
<tr>
<td>Updated: Troubleshooting, Cannot Open an Environment (p. 544)</td>
<td>Third-party cookies are no longer needed to open the IDE.</td>
<td>November 6, 2019</td>
</tr>
<tr>
<td>Added: Troubleshooting, Third-party cookies disabled (p. 546)</td>
<td>Although third-party cookies are no longer needed to open the IDE, they are needed to use the Application Preview or File Preview features. Information about this behavior has been added to the Troubleshooting topic.</td>
<td>November 6, 2019</td>
</tr>
</tbody>
</table>
## AWS Cloud9 User Guide

### Earlier Updates

The following table describes important changes to the *AWS Cloud9 User Guide* before April 2019.

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>August 15, 2019</td>
<td>Organization changes were applied to the user guide to assist in navigation, especially for first-time users.</td>
</tr>
<tr>
<td>May 15, 2019</td>
<td>AWS Cloud9 is now also available in the Europe (Frankfurt) Region. For more information, see <a href="#">AWS Cloud9 in the Amazon Web Services General Reference</a>.</td>
</tr>
<tr>
<td>May 10, 2019</td>
<td>Added a new sample demonstrating how to use AWS Cloud9 with LAMP (Linux, Apache HTTP Server, MySQL, and PHP). For more information, see the <a href="#">LAMP Sample for AWS Cloud9</a>.</td>
</tr>
<tr>
<td>April 19, 2019</td>
<td>Added a new sample demonstrating how to use AWS Cloud9 with WordPress. For more information, see the <a href="#">WordPress Sample for AWS Cloud9</a>.</td>
</tr>
<tr>
<td>April 4, 2019</td>
<td>AWS Cloud9 is now also available in the Asia Pacific (Tokyo) Region. For more information, see <a href="#">AWS Cloud9 in the Amazon Web Services General Reference</a>.</td>
</tr>
<tr>
<td>April 2, 2019</td>
<td>Instructions for using the AWS Cloud9 console to create AWS Cloud9 EC2 development environments that connect to Ubuntu Server were added. For more information, see <a href="#">Creating an EC2 Environment</a>.</td>
</tr>
</tbody>
</table>

Note that currently you cannot use code to create AWS Cloud9 EC2 development environments that connect to Ubuntu Server, for example by using the AWS CLI, AWS CloudFormation, the AWS SDKs, the Tools for Windows PowerShell, or the AWS Cloud9 API. Support for these methods is expected in the future.
<table>
<thead>
<tr>
<th>Change</th>
<th>Description</th>
<th>Date Changed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting started instructions added for students, educators, and enterprises</td>
<td>Instructions for getting started with AWS Cloud9 have been expanded to include steps for students, educators, and enterprises. For more information, see <em>Setting Up AWS Cloud9</em> (p. 6).</td>
<td>February 7, 2019</td>
</tr>
<tr>
<td>AWS CloudTrail support added</td>
<td>AWS CloudTrail now supports AWS Cloud9. For more information, see <em>Logging AWS Cloud9 API Calls with AWS CloudTrail</em> (p. 507).</td>
<td>January 21, 2019</td>
</tr>
<tr>
<td>Shared VPCs support added</td>
<td>AWS Cloud9 now supports shared VPCs in Amazon VPC. For more information, see <em>Amazon VPC Requirements for AWS Cloud9</em> (p. 467).</td>
<td>December 7, 2018</td>
</tr>
<tr>
<td>AWS RoboMaker integration added</td>
<td>AWS Cloud9 now supports AWS RoboMaker, a service that makes it easy to develop, test, and deploy intelligent robotics applications at scale. For more information, see <em>Getting Started with AWS RoboMaker and Developing with AWS Cloud9</em> in the <em>AWS RoboMaker Developer Guide</em>.</td>
<td>November 26, 2018</td>
</tr>
<tr>
<td>Information about additional productivity features for language projects added</td>
<td>The AWS Cloud9 IDE now provides additional productivity features for some languages in the context of a language project. For more information, see <em>Working with Language Projects in the AWS Cloud9 Integrated Development Environment (IDE)</em> (p. 154).</td>
<td>October 2, 2018</td>
</tr>
<tr>
<td>Go window added; Navigate and Commands windows removed</td>
<td>The Go window was added to the AWS Cloud9 IDE for environments created on or after October 2, 2018. This new window replaces the Navigate and Commands windows, which were both removed from the IDE for environments created on or after October 2, 2018. For more information, see <em>Step 10: Go Window</em> (p. 109) in <em>Tour the IDE</em> (p. 100).</td>
<td>October 2, 2018</td>
</tr>
<tr>
<td>AWS CDK sample added</td>
<td>Added a new sample demonstrating how to use AWS Cloud9.</td>
<td>August 30, 2018</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Change9 with the AWS Cloud Development Kit (AWS CDK). For more information, see the AWS CDK Sample for AWS Cloud9 (p. 377).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information about SSH IP address restrictions automatically added to EC2 environments added</td>
<td>For AWS Cloud9 EC2 development environments created on or after July 31 2018, AWS Cloud9 now automatically restricts incoming SSH traffic to just the IP address ranges that AWS Cloud9 uses to connect over SSH. For more information, see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).</td>
<td>July 31, 2018</td>
</tr>
<tr>
<td>Docker sample added</td>
<td>Added new sample demonstrating how to use AWS Cloud9 with Docker. For more information, see the Docker Sample for AWS Cloud9 (p. 458).</td>
<td>June 19, 2018</td>
</tr>
<tr>
<td>Additional Lambda function deployment options added</td>
<td>Added information about additional options for deploying AWS Lambda functions from the AWS Cloud9 IDE, depending on how the Lambda function was originally created. For more information, see Upload Code for a Lambda Function (p. 312).</td>
<td>May 29, 2018</td>
</tr>
<tr>
<td>Samples added for Java, .NET Core, and TypeScript</td>
<td>Added new samples demonstrating how to use AWS Cloud9 with Java, .NET Core, and TypeScript. For more information, see the Java Sample for AWS Cloud9 (p. 400), .NET Core Sample for AWS Cloud9 (p. 423), and TypeScript Sample for AWS Cloud9 (p. 452).</td>
<td>May 29, 2018</td>
</tr>
<tr>
<td>Supported browsers list added</td>
<td>Added information about supported browsers for AWS Cloud9. For more information, see Supported Browsers for AWS Cloud9 (p. 557).</td>
<td>May 23, 2018</td>
</tr>
<tr>
<td>Lambda function tutorials added</td>
<td>Added new tutorials demonstrating how to create Lambda functions with AWS Cloud9. For more information, see the AWS Lambda Tutorial for AWS Cloud9 (p. 325) and Advanced AWS Lambda Tutorial for AWS Cloud9 (p. 342).</td>
<td>April 19, 2018</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>SSH IP traffic restriction information added</td>
<td>Added information about how to restrict incoming traffic to just the IP address ranges that AWS Cloud9 uses to connect to hosts over SSH. For more information, see Inbound SSH IP address ranges for AWS Cloud9 (p. 483).</td>
<td>April 19, 2018</td>
</tr>
<tr>
<td>AWS Serverless Application Repository information added</td>
<td>Added information about how to use the AWS Serverless Application Repository with AWS Cloud9. For more information, see Create and Deploy Lambda Functions with the AWS Serverless Application Repository (p. 293).</td>
<td>April 19, 2018</td>
</tr>
<tr>
<td>Troubleshooters added for previewing applications and sharing running applications</td>
<td>Added new troubleshooters for previewing applications and sharing running applications. For more information, see Application Preview Tab Displays an Error or is Blank (p. 548) and Cannot Display Your Running Application Outside of the IDE (p. 549).</td>
<td>April 19, 2018</td>
</tr>
<tr>
<td>File Revision History information added</td>
<td>Added information about how to use the File Revision History pane in the IDE. For more information, see Working with File Revisions in the AWS Cloud9 Integrated Development Environment (IDE) (p. 143).</td>
<td>April 19, 2018</td>
</tr>
<tr>
<td>Lambda Python function debugging information added</td>
<td>Added information about how to debug Lambda functions that use Python. For more information, see Debug the Local Version of a Lambda Function or Its Related API Gateway API (p. 309).</td>
<td>March 22, 2018</td>
</tr>
<tr>
<td>Troubleshooter added for opening environments</td>
<td>Added a new troubleshooter for opening AWS Cloud9 development environments. For more information, see Cannot Open an Environment (p. 544).</td>
<td>March 19, 2018</td>
</tr>
<tr>
<td>Troubleshooter added for AWS Cloud9 Installer</td>
<td>Added a new troubleshooter for the AWS Cloud9 Installer. For more information, see The AWS Cloud9 Installer Hangs or Fails (p. 545).</td>
<td>March 19, 2018</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>AWS CodePipeline information added</td>
<td>Added information about how to use AWS Cloud9 with AWS CodePipeline. For more information, see Working with AWS CodePipeline in the AWS Cloud9 Integrated Development Environment (IDE) (p. 322).</td>
<td>February 13, 2018</td>
</tr>
<tr>
<td>aws-shell information added</td>
<td>Added information about how to use AWS Cloud9 with the aws-shell. For more information, see the AWS Command Line Interface and aws-shell Sample for AWS Cloud9 (p. 352).</td>
<td>January 19, 2018</td>
</tr>
<tr>
<td>Documentation availability on GitHub added</td>
<td>This guide is now available on GitHub. You can also use GitHub to submit feedback and change requests for this guide's content. For more information, choose the Edit on GitHub icon in the guide's navigation bar, or see the awsdocs/aws-cloud9-user-guide repository on the GitHub website.</td>
<td>January 10, 2018</td>
</tr>
<tr>
<td>Kindle format availability</td>
<td>This guide is now available in Amazon Kindle format. For more information, choose the Open Kindle icon in the guide's navigation bar, or see AWS Cloud9: User Guide Kindle Edition on the Amazon website.</td>
<td>January 2, 2018</td>
</tr>
<tr>
<td>Amazon Lightsail information added</td>
<td>Added information about how to use AWS Cloud9 with Amazon Lightsail. For more information, see Working with Amazon Lightsail Instances in the AWS Cloud9 Integrated Development Environment (IDE) (p. 276).</td>
<td>December 19, 2017</td>
</tr>
<tr>
<td>Added environment settings descriptions for AWS</td>
<td>Added descriptions of specific AWS settings for AWS Cloud9 development environments. For more information, see Working with AWS Project and User Settings in the AWS Cloud9 Integrated Development Environment (IDE) (p. 172).</td>
<td>December 7, 2017</td>
</tr>
<tr>
<td>Change</td>
<td>Description</td>
<td>Date Changed</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Getting started instructions added for AWS account root users and advanced setup steps for teams</td>
<td>Added setup steps for using AWS Cloud9 with an AWS account root user. Added advanced setup steps for using AWS Cloud9 with teams. For more information, see Setting Up AWS Cloud9 (p. 6).</td>
<td>December 5, 2017</td>
</tr>
<tr>
<td>Coverage expanded for environment requirements</td>
<td>Expanded coverage of requirements for an Amazon EC2 instance or your own server to connect to an AWS Cloud9 SSH development environment. For more information, see AWS Cloud9 SSH Development Environment host requirements (p. 479).</td>
<td>December 4, 2017</td>
</tr>
<tr>
<td>Initial documentation release</td>
<td>This is the initial release of the AWS Cloud9 User Guide.</td>
<td>November 30, 2017</td>
</tr>
</tbody>
</table>