# Table of Contents

Welcome ................................................................................................................................. 1
Actions ........................................................................................................................................ 2

- BatchCheckLayerAvailability ................................................................................................. 3
  - Request Syntax .................................................................................................................... 3
  - Request Parameters ........................................................................................................... 3
  - Response Syntax ................................................................................................................ 4
  - Response Elements .......................................................................................................... 4
  - Errors ............................................................................................................................... 4
  - Example ............................................................................................................................ 5
  - See Also ............................................................................................................................ 6

- BatchDeleteImage .................................................................................................................. 7
  - Request Syntax .................................................................................................................. 7
  - Request Parameters .......................................................................................................... 7
  - Response Syntax ................................................................................................................ 8
  - Response Elements .......................................................................................................... 8
  - Errors ............................................................................................................................... 8
  - Examples ........................................................................................................................... 9
  - See Also ............................................................................................................................ 10

- BatchGetImage .................................................................................................................... 12
  - Request Syntax .................................................................................................................. 12
  - Request Parameters .......................................................................................................... 12
  - Response Syntax ................................................................................................................ 13
  - Response Elements .......................................................................................................... 13
  - Errors ............................................................................................................................... 13
  - Example ............................................................................................................................ 14
  - See Also ............................................................................................................................ 15

- CompleteLayerUpload .......................................................................................................... 16
  - Request Syntax .................................................................................................................. 16
  - Request Parameters .......................................................................................................... 16
  - Response Syntax ................................................................................................................ 17
  - Response Elements .......................................................................................................... 17
  - Errors ............................................................................................................................... 18
  - See Also ............................................................................................................................ 18

- CreateRepository .................................................................................................................. 20
  - Request Syntax .................................................................................................................. 20
  - Request Parameters .......................................................................................................... 20
  - Response Syntax ................................................................................................................ 20
  - Response Elements .......................................................................................................... 20
  - Errors ............................................................................................................................... 21
  - Example ............................................................................................................................ 21
  - See Also ............................................................................................................................ 22

- DeleteLifecyclePolicy ............................................................................................................ 23
  - Request Syntax .................................................................................................................. 23
  - Request Parameters .......................................................................................................... 23
  - Response Syntax ................................................................................................................ 23
  - Response Elements .......................................................................................................... 23
  - Errors ............................................................................................................................... 24
  - Example ............................................................................................................................ 25
  - See Also ............................................................................................................................ 25

- DeleteRepository ................................................................................................................ 27
  - Request Syntax .................................................................................................................. 27
  - Request Parameters .......................................................................................................... 27
  - Response Syntax ................................................................................................................ 27
  - Response Elements .......................................................................................................... 28
<table>
<thead>
<tr>
<th>Function</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetRepositoryPolicy</td>
<td>58</td>
</tr>
<tr>
<td>PutLifecyclePolicy</td>
<td>72</td>
</tr>
<tr>
<td>SetRepositoryPolicy</td>
<td>76</td>
</tr>
<tr>
<td>StartLifecyclePolicyPreview</td>
<td>80</td>
</tr>
<tr>
<td>UploadLayerPart</td>
<td>84</td>
</tr>
</tbody>
</table>

See Also  .......................................................... 57
Request Syntax  ..................................................... 58
Request Parameters ................................................ 58
Response Syntax  ..................................................... 58
Response Elements .................................................. 58
Errors  ................................................................. 59
Example  ............................................................... 59

See Also  .......................................................... 60
InitiateLayerUpload  ................................................. 62
Request Syntax  ..................................................... 62
Request Parameters ................................................ 62
Response Syntax  ..................................................... 62
Response Elements .................................................. 62
Errors  ................................................................. 63
See Also  ............................................................... 63
ListImages  ............................................................ 64
Request Syntax  ..................................................... 64
Request Parameters ................................................ 64
Response Syntax  ..................................................... 65
Response Elements .................................................. 65
Errors  ................................................................. 66
Example  ............................................................... 66
See Also  ............................................................... 67
PutImage  ............................................................... 68
Request Syntax  ..................................................... 68
Request Parameters ................................................ 68
Response Syntax  ..................................................... 69
Response Elements .................................................. 69
Errors  ................................................................. 69
Example  ............................................................... 70
See Also  ............................................................... 71
PutLifecyclePolicy  .................................................... 72
Request Syntax  ..................................................... 72
Request Parameters ................................................ 72
Response Syntax  ..................................................... 72
Response Elements .................................................. 73
Errors  ................................................................. 73
Example  ............................................................... 73
See Also  ............................................................... 74
SetRepositoryPolicy  .................................................. 76
Request Syntax  ..................................................... 76
Request Parameters ................................................ 76
Response Syntax  ..................................................... 77
Response Elements .................................................. 77
Errors  ................................................................. 77
Example  ............................................................... 78
See Also  ............................................................... 79
StartLifecyclePolicyPreview  .................................... 80
Request Syntax  ..................................................... 80
Request Parameters ................................................ 80
Response Syntax  ..................................................... 80
Response Elements .................................................. 81
Errors  ................................................................. 81
Example  ............................................................... 82
See Also  ............................................................... 83
Request Syntax .............................................................................................................................. 84
Request Parameters ....................................................................................................................... 84
Response Syntax .............................................................................................................................. 85
Response Elements .......................................................................................................................... 85
Errors ............................................................................................................................................. 86
See Also .......................................................................................................................................... 86
Data Types ....................................................................................................................................... 88
AuthorizationData ........................................................................................................................... 89
Contents .......................................................................................................................................... 89
See Also .......................................................................................................................................... 89
DescribeImagesFilter ....................................................................................................................... 90
Contents .......................................................................................................................................... 90
See Also .......................................................................................................................................... 90
Image .............................................................................................................................................. 91
Contents .......................................................................................................................................... 91
See Also .......................................................................................................................................... 91
ImageDetail ..................................................................................................................................... 92
Contents .......................................................................................................................................... 92
See Also .......................................................................................................................................... 93
ImageFailure .................................................................................................................................... 94
Contents .......................................................................................................................................... 94
See Also .......................................................................................................................................... 94
ImageIdentifier ............................................................................................................................... 95
Contents .......................................................................................................................................... 95
See Also .......................................................................................................................................... 95
Layer .............................................................................................................................................. 96
Contents .......................................................................................................................................... 96
See Also .......................................................................................................................................... 96
LayerFailure .................................................................................................................................... 97
Contents .......................................................................................................................................... 97
See Also .......................................................................................................................................... 97
LifecyclePolicyPreviewFilter .......................................................................................................... 98
Contents .......................................................................................................................................... 98
See Also .......................................................................................................................................... 98
LifecyclePolicyPreviewResult .......................................................................................................... 99
Contents .......................................................................................................................................... 99
See Also .......................................................................................................................................... 99
LifecyclePolicyPreviewSummary ..................................................................................................... 100
Contents .......................................................................................................................................... 100
See Also .......................................................................................................................................... 100
LifecyclePolicyRuleAction .............................................................................................................. 101
Contents .......................................................................................................................................... 101
See Also .......................................................................................................................................... 101
ListImagesFilter .............................................................................................................................. 102
Contents .......................................................................................................................................... 102
See Also .......................................................................................................................................... 102
Repository ...................................................................................................................................... 103
Contents .......................................................................................................................................... 103
See Also .......................................................................................................................................... 103
Common Parameters ....................................................................................................................... 105
Common Errors .............................................................................................................................. 107
Welcome

Amazon Elastic Container Registry (Amazon ECR) is a managed Docker registry service. Customers can use the familiar Docker CLI to push, pull, and manage images. Amazon ECR provides a secure, scalable, and reliable registry. Amazon ECR supports private Docker repositories with resource-based permissions using IAM so that specific users or Amazon EC2 instances can access repositories and images. Developers can use the Docker CLI to author and manage images.

This document was last published on August 2, 2018.
Actions

The following actions are supported:

- BatchCheckLayerAvailability (p. 3)
- BatchDeleteImage (p. 7)
- BatchGetImage (p. 12)
- CompleteLayerUpload (p. 16)
- CreateRepository (p. 20)
- DeleteLifecyclePolicy (p. 23)
- DeleteRepository (p. 27)
- DeleteRepositoryPolicy (p. 30)
- DescribeImages (p. 34)
- DescribeRepositories (p. 39)
- GetAuthorizationToken (p. 43)
- GetDownloadUrlForLayer (p. 46)
- GetLifecyclePolicy (p. 49)
- GetLifecyclePolicyPreview (p. 53)
- GetRepositoryPolicy (p. 58)
- InitiateLayerUpload (p. 62)
- ListImages (p. 64)
- PutImage (p. 68)
- PutLifecyclePolicy (p. 72)
- SetRepositoryPolicy (p. 76)
- StartLifecyclePolicyPreview (p. 80)
- UploadLayerPart (p. 84)
BatchCheckLayerAvailability

Check the availability of multiple image layers in a specified registry and repository.

**Note**
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the `docker` CLI to pull, tag, and push images.

**Request Syntax**

```json
{
    "layerDigests": [ "string" ],
    "registryId": "string",
    "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

- **layerDigests** (p. 3)
  
  The digests of the image layers to check.
  
  Type: Array of strings
  
  Array Members: Minimum number of 1 item. Maximum number of 100 items.
  
  Length Constraints: Minimum length of 0. Maximum length of 1000.
  
  Required: Yes

- **registryId** (p. 3)
  
  The AWS account ID associated with the registry that contains the image layers to check. If you do not specify a registry, the default registry is assumed.
  
  Type: String
  
  Pattern: `[0-9]{12}`
  
  Required: No

- **repositoryName** (p. 3)
  
  The name of the repository that is associated with the image layers to check.
  
  Type: String
  
  
  Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]*)+/?)*[a-z0-9]+(?:[._-][a-z0-9]*)`
  
  Required: Yes
Response Syntax

```json
{
  "failures": [
    {
      "failureCode": "string",
      "failureReason": "string",
      "layerDigest": "string"
    }
  ],
  "layers": [
    {
      "layerAvailability": "string",
      "layerDigest": "string",
      "layerSize": number,
      "mediaType": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

- **failures (p. 4)**
  
  Any failures associated with the call.
  
  Type: Array of LayerFailure (p. 97) objects

- **layers (p. 4)**
  
  A list of image layer objects corresponding to the image layer references in the request.
  
  Type: Array of Layer (p. 96) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

- **InvalidParameterException**
  
  The specified parameter is invalid. Review the available parameters for the API request.
  
  HTTP Status Code: 400

- **RepositoryNotFoundException**
  
  The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.
  
  HTTP Status Code: 400

- **ServerException**
  
  These errors are usually caused by a server-side issue.
HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example checks the availability of an image layer in the amazonlinux repository.

Sample Request

POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 126
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.BatchCheckLayerAvailability
X-Amz-Date: 20161216T195733Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
    "layerDigests": [
        "sha256:8e3fa21c4cc40232e835a6761332d225c7af3235c5755f44ada2ed9d0e4ab7e8"
    ],
    "repositoryName": "amazonlinux"
}

Sample Response

HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 19:57:33 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 233
Connection: keep-alive
x-amzn-RequestId: e2422faf-c3c9-11e6-a3ee-63b3b5dcf3b9
{
    "failures": [],
    "layers": [
        {
            "layerAvailability": "AVAILABLE",
            "layerDigest": "sha256:8e3fa21c4cc40232e835a6761332d225c7af3235c5755f44ada2ed9d0e4ab7e8",
            "layerSize": 91768077,
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip"
        }
    ]
}
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
BatchDeleteImage

Deletes a list of specified images within a specified repository. Images are specified with either `imageTag` or `imageDigest`.

You can remove a tag from an image by specifying the image's tag in your request. When you remove the last tag from an image, the image is deleted from your repository.

You can completely delete an image (and all of its tags) by specifying the image's digest in your request.

### Request Syntax

```json
{
  "imageIds": [
    {
      "imageDigest": "string",
      "imageTag": "string"
    }
  ],
  "registryId": "string",
  "repositoryName": "string"
}
```

### Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**imageIds (p. 7)**

A list of image ID references that correspond to images to delete. The format of the `imageIds` reference is `imageTag=tag` or `imageDigest=digest`.

Type: Array of `ImageIdentifier (p. 95)` objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: Yes

**registryId (p. 7)**

The AWS account ID associated with the registry that contains the image to delete. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: `[0–9]{12}`

Required: No

**repositoryName (p. 7)**

The repository that contains the image to delete.

Type: String


Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]+)*)(?:[a-z0-9]+(?:[._-][a-z0-9]+)*)*`
Required: Yes

Response Syntax

```json
{
  "failures": [
    {
      "failureCode": "string",
      "failureReason": "string",
      "imageId": {
        "imageDigest": "string",
        "imageTag": "string"
      }
    }
  ],
  "imageIds": [
    {
      "imageDigest": "string",
      "imageTag": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**failures (p. 8)**

Any failures associated with the call.

Type: Array of ImageFailure (p. 94) objects

**imageIds (p. 8)**

The image IDs of the deleted images.

Type: Array of ImageIdentifier (p. 95) objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400
ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Examples

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

Example

This example deletes an image in the ubuntu repository with the imageTag value of xenial.

Sample Request

```plaintext
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 66
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.BatchDeleteImage
X-Amz-Date: 20161216T193711Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
    "repositoryName": "ubuntu",
    "imageIds": [
        {
            "imageTag": "xenial"
        }
    ]
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 19:37:11 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 138
Connection: keep-alive
x-amzn-RequestId: 09cc7023-c3c7-11e6-8acf-61b7dd8abe56

{
    "failures": [],
    "imageIds": [
        {
            "imageDigest": "sha256:7a64bc9c8843b0a8c88a7e4715b7615e4e1b0d8ca3c7a76ec8250899c397a",
            "imageTag": "xenial"
        }
    ]
}
```
Example

This example deletes an image (and all of its tags) in the ubuntu repository with the `imageDigest` value of `sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a`.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 134
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.BatchDeleteImage
X-Amz-Date: 20161216T194250Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "repositoryName": "ubuntu",
  "imageIds": [
    {
      "imageDigest": "sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a"
    }
  ]
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 19:42:50 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 248
Connection: keep-alive
x-amzn-RequestId: d441a9f6-c3c7-11e6-8acf-61b7dd8abe56

{
  "failures": [],
  "imageIds": [
    {
      "imageDigest": "sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a",
      "imageTag": "xenial"
    },
    {
      "imageDigest": "sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a",
      "imageTag": "latest"
    }
  ]
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
Amazon EC2 Container Registry API Reference

See Also

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
BatchGetImage

Gets detailed information for specified images within a specified repository. Images are specified with either imageTag or imageDigest.

Request Syntax

```json
{
  "acceptedMediaTypes": [ "string" ],
  "imageIds": [
    {
      "imageDigest": "string",
      "imageTag": "string"
    }
  ],
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**acceptedMediaTypes (p. 12)**

The accepted media types for the request.

Valid values: application/vnd.docker.distribution.manifest.v1+json | application/vnd.docker.distribution.manifest.v2+json | application/vnd.oci.image.manifest.v1+json

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: No

**imageIds (p. 12)**

A list of image ID references that correspond to images to describe. The format of the imageIds reference is imageTag=tag or imageDigest=digest.

Type: Array of ImageIdentifier (p. 95) objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: Yes

**registryId (p. 12)**

The AWS account ID associated with the registry that contains the images to describe. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No
**Response Syntax**

{  
  "failures": [  
    {  
      "failureCode": "string",  
      "failureReason": "string",  
      "imageId": {  
        "imageDigest": "string",  
        "imageTag": "string"  
      }  
    }  
  ],  
  "images": [  
    {  
      "imageId": {  
        "imageDigest": "string",  
        "imageTag": "string"  
      },  
      "imageManifest": "string",  
      "registryId": "string",  
      "repositoryName": "string"  
    }  
  ]
}

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

**failures (p. 13)**

Any failures associated with the call. Type: Array of ImageFailure (p. 94) objects

**images (p. 13)**

A list of image objects corresponding to the image references in the request. Type: Array of Image (p. 91) objects

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 107).
InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example describes an image in the amazonlinux repository with the imageTag value of latest.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 71
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.BatchGetImage
X-Amz-Date: 20161216T195356Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "repositoryName": "amazonlinux",
  "imageIds": [
    {
      "imageTag": "latest"
    }
  ]
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
```

API Version 2015-09-21
API Version 2015-09-21

15
CompleteLayerUpload

Informs Amazon ECR that the image layer upload has completed for a specified registry, repository name, and upload ID. You can optionally provide a sha256 digest of the image layer for data validation purposes.

**Note**
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the docker CLI to pull, tag, and push images.

**Request Syntax**

```json
{
    "layerDigests": [ "string" ],
    "registryId": "string",
    "repositoryName": "string",
    "uploadId": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**layerDigests (p. 16)**

The sha256 digest of the image layer.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Pattern: `[a-zA-Z0-9-._+\[\]:]+:[a-fA-F0-9]+`

Required: Yes

**registryId (p. 16)**

The AWS account ID associated with the registry to which to upload layers. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: `[0-9]{12}`

Required: No

**repositoryName (p. 16)**

The name of the repository to associate with the image layer.

Type: String


Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]*[a-z0-9]+)*[a-z0-9]+(?:[._-][a-z0-9]+)*)`
Required: Yes

**uploadId (p. 16)**

The upload ID from a previous *InitiateLayerUpload (p. 62)* operation to associate with the image layer.

Type: String

Pattern: `[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}`

Required: Yes

### Response Syntax

```
{
  "layerDigest": "string",
  "registryId": "string",
  "repositoryName": "string",
  "uploadId": "string"
}
```

### Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**layerDigest (p. 17)**

The sha256 digest of the image layer.

Type: String

Pattern: `[a-zA-Z0-9-._+]*:[a-fA-F0-9]+`

**registryId (p. 17)**

The registry ID associated with the request.

Type: String

Pattern: `[0-9]{12}`

**repositoryName (p. 17)**

The repository name associated with the request.

Type: String


Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]+)*[/])*[a-z0-9]+(?:[._-][a-z0-9]+)*`

**uploadId (p. 17)**

The upload ID associated with the layer.

Type: String
Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

EmptyUploadException

The specified layer upload does not contain any layer parts.

HTTP Status Code: 400

InvalidLayerException

The layer digest calculation performed by Amazon ECR upon receipt of the image layer does not match the digest specified.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

LayerAlreadyExistsException

The image layer already exists in the associated repository.

HTTP Status Code: 400

LayerPartTooSmallException

Layer parts must be at least 5 MiB in size.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

UploadNotFoundException

The upload could not be found, or the specified upload id is not valid for this repository.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
See Also

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
CreateRepository

Creates an image repository.

Request Syntax

```json
{
   "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

repositoryName (p. 20)

The name to use for the repository. The repository name may be specified on its own (such as nginx-web-app) or it can be prepended with a namespace to group the repository into a category (such as project-a/nginx-web-app).

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*

Required: Yes

Response Syntax

```json
{
   "repository": {
      "createdAt": number,
      "registryId": "string",
      "repositoryArn": "string",
      "repositoryName": "string",
      "repositoryUri": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

repository (p. 20)

The repository that was created.

Type: Repository (p. 103) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

LimitExceededException

The operation did not succeed because it would have exceeded a service limit for your account. For more information, see Amazon ECR Default Service Limits in the Amazon Elastic Container Registry User Guide.

HTTP Status Code: 400

RepositoryAlreadyExistsException

The specified repository already exists in the specified registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

Example

This example creates a repository called nginx-web-app inside the project-a namespace in the default registry for an account.

Sample Request

```plaintext
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 45
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.CreateRepository
X-Amz-Date: 20151130T203458Z
User-Agent: aws-cli/1.9.9 Python/2.7.10 Darwin/14.5.0 botocore/1.3.9
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "repositoryName": "project-a/nginx-web-app"
}
```
Sample Response

HTTP/1.1 200 OK
Server: Server
Date: Mon, 30 Nov 2015 20:34:58 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 175
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
   "repository": {
      "registryId": "012345678910",
      "repositoryArn": "arn:aws:ecr:us-east-1:012345678910:repository/project-a/nginx-web-app",
      "repositoryName": "project-a/nginx-web-app"
   }
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteLifecyclePolicy

Deletes the specified lifecycle policy.

Request Syntax

```json
{
    "registryId": "string",
    "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

registryId (p. 23)

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

repositoryName (p. 23)

The name of the repository.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]*)+)[a-z0-9]+(?:[._-][a-z0-9]*)+

Required: Yes

Response Syntax

```json
{
    "lastEvaluatedAt": number,
    "lifecyclePolicyText": "string",
    "registryId": "string",
    "repositoryName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

**lastEvaluatedAt (p. 23)**

The time stamp of the last time that the lifecycle policy was run.

Type: Timestamp

**lifecyclePolicyText (p. 23)**

The JSON lifecycle policy text.

Type: String

Length Constraints: Minimum length of 100. Maximum length of 10240.

**registryId (p. 23)**

The registry ID associated with the request.

Type: String

Pattern: [0-9]{12}

**repositoryName (p. 23)**

The repository name associated with the request.

Type: String


**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**LifecyclePolicyNotFoundException**

The lifecycle policy could not be found, and no policy is set to the repository.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500
Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

Example

This example deletes a lifecycle policy for a repository called project-a/amazon-ecs-sample in the default registry for an account.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.DeleteLifecyclePolicy
Content-Type: application/x-amz-json-1.1
User-Agent: aws-cli/1.11.144 Python/3.6.1 Darwin/16.6.0 botocore/1.7.2
X-Amz-Date: 20170901T223937Z
Authorization: AUTHPARAMS
Content-Length: 48

{
    "repositoryName": "project-a/amazon-ecs-sample",
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 01 Sep 2017 19:42:18 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 340
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
    "lastEvaluatedAt":1.504295007E9,
    "lifecyclePolicyText":"{"rules":[{"rulePriority":1,"description":"Expire images older than 14 days","selection":{"tagStatus":"untagged","countType":"sinceImagePushed","countUnit":"days","countNumber":14},"action":{"type":"expire"}}]},
    "registryId":"012345678910",
    "repositoryName":"project-a/amazon-ecs-sample"
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
DeleteRepository

Deletes an existing image repository. If a repository contains images, you must use the `force` option to delete it.

**Request Syntax**

```json
{
    "force": boolean,
    "registryId": "string",
    "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**force (p. 27)**

If a repository contains images, forces the deletion.

Type: Boolean

Required: No

**registryId (p. 27)**

The AWS account ID associated with the registry that contains the repository to delete. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: `[0-9]{12}`

Required: No

**repositoryName (p. 27)**

The name of the repository to delete.

Type: String


Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*`

Required: Yes

**Response Syntax**

```json
{
    "repository": {
        "createdAt": number,
        "registryId": "string",
    }
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

repository (p. 27)

The repository that was deleted.

Type: Repository (p. 103) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

RepositoryNotEmptyException

The specified repository contains images. To delete a repository that contains images, you must force the deletion with the force parameter.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.
Example

This example deletes a repository named `ubuntu` in the default registry for an account.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 43
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.DeleteRepository
X-Amz-Date: 20151130T201424Z
User-Agent: aws-cli/1.9.9 Python/2.7.10 Darwin/14.5.0 botocore/1.3.9
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
    "repositoryName": "ubuntu",
    "force": true
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 30 Nov 2015 20:14:24 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 141
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
    "repository": {
        "registryId": "012345678910",
        "repositoryArn": "arn:aws:ecr:us-east-1:012345678910:repository/ubuntu",
        "repositoryName": "ubuntu"
    }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DeleteRepositoryPolicy

Deletes the repository policy from a specified repository.

Request Syntax

```
{
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**registryId (p. 30)**

The AWS account ID associated with the registry that contains the repository policy to delete. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

**repositoryName (p. 30)**

The name of the repository that is associated with the repository policy to delete.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+*)*[a-z0-9]+(?:[._-][a-z0-9]+*)*

Required: Yes

Response Syntax

```
{
  "policyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
**policyText (p. 30)**

The JSON repository policy that was deleted from the repository.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 10240.

**registryId (p. 30)**

The registry ID associated with the request.

Type: String

Pattern: [0-9]{12}

**repositoryName (p. 30)**

The repository name associated with the request.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*/+

---

**Errors**

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**RepositoryPolicyNotFoundException**

The specified repository and registry combination does not have an associated repository policy.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

---

**Example**

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.
You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

**Example**

This example deletes the repository policy from the `ubuntu` repository.

**Sample Request**

```
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 28
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.DeleteRepositoryPolicy
X-Amz-Date: 20151215T003722Z
User-Agent: aws-cli/1.9.10 Python/2.7.10 Darwin/14.5.0 botocore/1.3.10
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
  "repositoryName": "ubuntu"
}
```

**Sample Response**

```
HTTP/1.1 200 OK
Server: Server
Date: Tue, 15 Dec 2015 00:37:22 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 301
Connection: keep-alive
x-amzn-RequestId: 01817918-a2c4-11e5-a19f-014c7a9aad99
{
```

**See Also**

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeImages

Returns metadata about the images in a repository, including image size, image tags, and creation date.

**Note**
Beginning with Docker version 1.9, the Docker client compresses image layers before pushing them to a V2 Docker registry. The output of the `docker images` command shows the uncompressed image size, so it may return a larger image size than the image sizes returned by DescribeImages (p. 34).

**Request Syntax**

```
{
  "filter": {
    "tagStatus": "string",
  },
  "imageIds": [
    {
      "imageDigest": "string",
      "imageTag": "string"
    }
  ],
  "maxResults": number,
  "nextToken": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**filter (p. 34)**

The filter key and value with which to filter your DescribeImages results.

Type: DescribeImagesFilter (p. 90) object

Required: No

**imageIds (p. 34)**

The list of image IDs for the requested repository.

Type: Array of ImageIdentifier (p. 95) objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: No

**maxResults (p. 34)**

The maximum number of repository results returned by DescribeImages in paginated output. When this parameter is used, DescribeImages only returns maxResults results in a single page along with a nextToken response element. The remaining results of the initial request can be seen by sending another DescribeImages request with the returned nextToken value. This value can
be between 1 and 100. If this parameter is not used, then DescribeImages returns up to 100 results and a nextToken value, if applicable. This option cannot be used when you specify images with imageIds.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

nextToken (p. 34)

The nextToken value returned from a previous paginated DescribeImages request where maxResults was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the nextToken value. This value is null when there are no more results to return. This option cannot be used when you specify images with imageIds.

Type: String

Required: No

registryId (p. 34)

The AWS account ID associated with the registry that contains the repository in which to describe images. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: \[0-9\]\{12\}

Required: No

repositoryName (p. 34)

A list of repositories to describe.

Type: String


Pattern: \(?:[a-z0-9]+\(\?:[._-][a-z0-9]+\)*\(a-z0-9\)+\(\?:[._-][a-z0-9]+\)*\)

Required: Yes

Response Syntax

```json
{
  "imageDetails": [
    {
      "imageDigest": "string",
      "imagePushedAt": number,
      "imageSizeInBytes": number,
      "imageTags": [ "string" ],
      "registryId": "string",
      "repositoryName": "string"
    }
  ],
  "nextToken": "string"
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**imageDetails (p. 35)**

A list of ImageDetail (p. 92) objects that contain data about the image.

Type: Array of ImageDetail (p. 92) objects

**nextToken (p. 35)**

The `nextToken` value to include in a future DescribeImages request. When the results of a DescribeImages request exceed maxResults, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**ImageNotFoundException**

The image requested does not exist in the specified repository.

HTTP Status Code: 400

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.
Example

This example describes the images in a repository named `ubuntu` in the default account. Note that the image with the digest `sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a` is tagged as `latest` and `xenial`.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 28
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.DescribeImages
X-Amz-Date: 20161216T193133Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
  "repositoryName": "ubuntu"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 19:31:33 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 1107
Connection: keep-alive
x-amzn-RequestId: 404826b1-c3c6-11e6-a9e5-e3c203a2f07f
{
  "imageDetails": [
    {
      "imageDigest": "sha256:7c70a5ebcc7fcaaa22974a71175ba674efce3951fbc624943c891e9d256927c1",
      "imagePushedAt": 1452721263,
      "imageSizeInBytes": 44194573,
      "registryId": "012345678910",
      "repositoryName": "ubuntu"
    },
    {
      "imageDigest": "sha256:abdc090336ba4503bd72d0961a4f3d45134900d9a793d3f0c6a64d2555fbaa7",
      "imagePushedAt": 1481916613,
      "imageSizeInBytes": 39142127,
      "imageTags": ["precise"],
      "registryId": "012345678910",
      "repositoryName": "ubuntu"
    },
    {
      "imageDigest": "sha256:78dea3347768ba553ee8971cf2db2d048c96db84ebfcdada59ba8d6057941b",
      "imagePushedAt": 1452721207,
    }
  ]
}```
<table>
<thead>
<tr>
<th>Image Digest</th>
<th>Image Pushed At</th>
<th>Image Size In Bytes</th>
<th>Image Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>sha256:7a64bc9c8843b0a8c8b8a7e4715b7615e4e1b0d8ca3c7e7a76ec8250899c397a</td>
<td>1481916523</td>
<td>65747044</td>
<td>xenial, latest</td>
</tr>
<tr>
<td>sha256:881befbe6f54c1e85029fe3a11554342bf765a0849600ecb8fa2f922798b4925</td>
<td>1481916647</td>
<td>50223482</td>
<td>xenial, latest</td>
</tr>
<tr>
<td>sha256:881befbe6f54c1e85029fe3a11554342bf765a0849600ecb8fa2f922798b4925</td>
<td>1481916647</td>
<td>65770577</td>
<td>trusty</td>
</tr>
</tbody>
</table>

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
DescribeRepositories

Describes image repositories in a registry.

Request Syntax

```
{
  "maxResults": number,
  "nextToken": "string",
  "registryId": "string",
  "repositoryNames": [ "string" ]
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**maxResults (p. 39)**

The maximum number of repository results returned by DescribeRepositories in paginated output. When this parameter is used, DescribeRepositories only returns maxResults results in a single page along with a nextToken response element. The remaining results of the initial request can be seen by sending another DescribeRepositories request with the returned nextToken value. This value can be between 1 and 100. If this parameter is not used, then DescribeRepositories returns up to 100 results and a nextToken value, if applicable. This option cannot be used when you specify repositories with repositoryNames.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

**nextToken (p. 39)**

The nextToken value returned from a previous paginated DescribeRepositories request where maxResults was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the nextToken value. This value is null when there are no more results to return. This option cannot be used when you specify repositories with repositoryNames.

*Note*

This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String

Required: No

**registryId (p. 39)**

The AWS account ID associated with the registry that contains the repositories to be described. If you do not specify a registry, the default registry is assumed.

Type: String
Response Syntax

```json
{
  "nextToken": "string",
  "repositories": [
    {
      "createdAt": number,
      "registryId": "string",
      "repositoryArn": "string",
      "repositoryName": "string",
      "repositoryUri": "string"
    }
  ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

nextToken (p. 40)

The `nextToken` value to include in a future DescribeRepositories request. When the results of a DescribeRepositories request exceed `maxResults`, this value can be used to retrieve the next page of results. This value is `null` when there are no more results to return.

Type: String

repositories (p. 40)

A list of repository objects corresponding to valid repositories.

Type: Array of Repository (p. 103) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).
**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

## Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see [Signature Version 4 Signing Process](#) in the [AWS General Reference](#).

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the [AWS Command Line Interface (AWS CLI)](https://aws.amazon.com/cli/) or one of the [AWS SDKs](https://aws.amazon.com/sdk/) to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

### Example

This example describes the repositories in the default registry for an account.

#### Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.DescribeRepositories
X-Amz-Date: 20161216T200121Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

#### Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 20:01:21 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 972
Connection: keep-alive
x-amzn-RequestId: 6a1ca1a2-c3ca-11e6-aacd-d3a31c48f08c

{}
```
"repositories": [
  {
    "createdAt": 1450223349,
    "registryId": "012345678910",
    "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/ubuntu",
    "repositoryName": "ubuntu",
    "repositoryUri": "012345678910.dkr.ecr.us-west-2.amazonaws.com/ubuntu"
  },
  {
    "createdAt": 1481915414,
    "registryId": "012345678910",
    "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/amazonlinux",
    "repositoryName": "amazonlinux",
    "repositoryUri": "012345678910.dkr.ecr.us-west-2.amazonaws.com/amazonlinux"
  },
  {
    "createdAt": 1481845593,
    "registryId": "012345678910",
    "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/iis",
    "repositoryName": "iis",
    "repositoryUri": "012345678910.dkr.ecr.us-west-2.amazonaws.com/iis"
  },
  {
    "createdAt": 1479253336,
    "registryId": "012345678910",
    "repositoryArn": "arn:aws:ecr:us-west-2:012345678910:repository/windows_sample_app",
    "repositoryName": "windows_sample_app",
    "repositoryUri": "012345678910.dkr.ecr.us-west-2.amazonaws.com/windows_sample_app"
  }
]
GetAuthorizationToken

Retrieves a token that is valid for a specified registry for 12 hours. This command allows you to use the `docker` CLI to push and pull images with Amazon ECR. If you do not specify a registry, the default registry is assumed.

The `authorizationToken` returned for each registry specified is a base64 encoded string that can be decoded and used in a `docker login` command to authenticate to a registry. The AWS CLI offers an `aws ecr get-login` command that simplifies the login process.

**Request Syntax**

```json
{
   "registryIds": [ "string" ]
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**registryIds** (p. 43)

A list of AWS account IDs that are associated with the registries for which to get authorization tokens. If you do not specify a registry, the default registry is assumed.

Type: Array of strings

Array Members: Minimum number of 1 item. Maximum number of 10 items.

Pattern: `[0-9]{12}`

Required: No

**Response Syntax**

```json
{
   "authorizationData": [
      {
         "authorizationToken": "string",
         "expiresAt": number,
         "proxyEndpoint": "string"
      }
   ]
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
authorizationData (p. 43)

A list of authorization token data objects that correspond to the registryIds values in the request.

Type: Array of AuthorizationData (p. 89) objects

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example gets an authorization token for your default registry.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.GetAuthorizationToken
X-Amz-Date: 20151129T221940Z
User-Agent: aws-cli/1.9.9 Python/2.7.10 Darwin/14.5.0 botocore/1.3.9
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Sun, 29 Nov 2015 22:19:39 GMT
```
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
GetDownloadUrlForLayer

Retrieves the pre-signed Amazon S3 download URL corresponding to an image layer. You can only get URLs for image layers that are referenced in an image.

Note
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the docker CLI to pull, tag, and push images.

Request Syntax

```json
{
  "layerDigest": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**layerDigest (p. 46)**

The digest of the image layer to download.

Type: String

Pattern: [a-zA-Z0-9-_+\.]+:\([a-fA-F0-9]+\)

Required: Yes

**registryId (p. 46)**

The AWS account ID associated with the registry that contains the image layer to download. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

**repositoryName (p. 46)**

The name of the repository that is associated with the image layer to download.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*[a-z0-9]+(?!.*\([a-z0-9]+\))+

Required: Yes
Response Syntax

```
{
    "downloadUrl": "string",
    "layerDigest": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

downloadUrl (p. 47)

The pre-signed Amazon S3 download URL for the requested layer.

Type: String

layerDigest (p. 47)

The digest of the image layer to download.

Type: String

Pattern: [a-zA-Z0-9-_.]+:[a-fA-F0-9]+  

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

LayerInaccessibleException

The specified layer is not available because it is not associated with an image. Unassociated image layers may be cleaned up at any time.

HTTP Status Code: 400

LayersNotFoundException

The specified layers could not be found, or the specified layer is not valid for this repository.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.
HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
GetLifecyclePolicy

Retrieves the specified lifecycle policy.

Request Syntax

```json
{
    "registryId": "string",
    "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**registryId (p. 49)**

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: \[0-9\]\{12\}

Required: No

**repositoryName (p. 49)**

The name of the repository.

Type: String


Pattern: (?:\[a-z0-9\]+(?:\[._-\]\[a-z0-9\]+)*/)*\[a-z0-9\]+(?:\[._-\]\[a-z0-9\]+)*

Required: Yes

Response Syntax

```json
{
    "lastEvaluatedAt": number,
    "lifecyclePolicyText": "string",
    "registryId": "string",
    "repositoryName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

**lastEvaluatedAt (p. 49)**

The time stamp of the last time that the lifecycle policy was run.

Type: Timestamp

**lifecyclePolicyText (p. 49)**

The JSON lifecycle policy text.

Type: String

Length Constraints: Minimum length of 100. Maximum length of 10240.

**registryId (p. 49)**

The registry ID associated with the request.

Type: String

Pattern: [0-9]{12}

**repositoryName (p. 49)**

The repository name associated with the request.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*[a-z0-9]+(?:[._-][a-z0-9]+)*)

## Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**LifecyclePolicyNotFoundException**

The lifecycle policy could not be found, and no policy is set to the repository.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500
Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example retrieves the lifecycle policy for a repository called project-a/amazon-ecs-sample in the default registry for an account.

Sample Request

<table>
<thead>
<tr>
<th>POST / HTTP/1.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host: ecr.us-west-2.amazonaws.com</td>
</tr>
<tr>
<td>Accept-Encoding: identity</td>
</tr>
<tr>
<td>X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.GetLifecyclePolicy</td>
</tr>
<tr>
<td>Content-Type: application/x-amz-json-1.1</td>
</tr>
<tr>
<td>User-Agent: aws-cli/1.11.144 Python/3.6.1 Darwin/16.6.0 botocore/1.7.2</td>
</tr>
<tr>
<td>X-Amz-Date: 20170901T210647Z</td>
</tr>
<tr>
<td>Authorization: AUTHPARAMS</td>
</tr>
<tr>
<td>Content-Length: 48</td>
</tr>
</tbody>
</table>

{  
  "repositoryName": "project-a/amazon-ecs-sample"  
}

Sample Response

HTTP/1.1 200 OK  
Server: Server  
Date: Fri, 01 Sep 2017 21:06:48 GMT  
Content-Type: application/x-amz-json-1.1  
Content-Length: 372  
Connection: keep-alive  
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f  

{  
  "lastEvaluatedAt":1.504295007E9,  
  "lifecyclePolicyText":"{"rules":[{"rulePriority":1,"description":"Expire images older than 14 days","selection":{"tagStatus":"untagged","countType":"sinceImagePushed","countUnit":"days","countNumber":14},"action":{"type":"expire\"}}]  
  "registryId":"012345678910",  
  "repositoryName":"project-a/amazon-ecs-sample"  
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
• AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
GetLifecyclePolicyPreview

Retrieves the results of the specified lifecycle policy preview request.

Request Syntax

```json
{
  "filter": {
    "tagStatus": "string"
  },
  "imageIds": [
    {
      "imageDigest": "string",
      "imageTag": "string"
    }
  ],
  "maxResults": number,
  "nextToken": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**filter (p. 53)**

An optional parameter that filters results based on image tag status and all tags, if tagged.

Type: LifecyclePolicyPreviewFilter (p. 98) object

Required: No

**imageIds (p. 53)**

The list of imageIDs to be included.

Type: Array of ImageIdentifier (p. 95) objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

Required: No

**maxResults (p. 53)**

The maximum number of repository results returned by GetLifecyclePolicyPreviewRequest in paginated output. When this parameter is used, GetLifecyclePolicyPreviewRequest only returns maxResults results in a single page along with a nextToken response element. The remaining results of the initial request can be seen by sending another GetLifecyclePolicyPreviewRequest request with the returned nextToken value. This value can be between 1 and 100. If this parameter is not used, then GetLifecyclePolicyPreviewRequest returns up to 100 results and a nextToken value, if
applicable. This option cannot be used when you specify images with imageIds.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

nextToken (p. 53)

The nextToken value returned from a previous paginated GetLifecyclePolicyPreviewRequest request where maxResults was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the nextToken value. This value is null when there are no more results to return. This option cannot be used when you specify images with imageIds.

Type: String

Required: No

registryId (p. 53)

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

repositoryName (p. 53)

The name of the repository.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*

Required: Yes

Response Syntax

```json
{
  "lifecyclePolicyText": "string",
  "nextToken": "string",
  "previewResults": [
    {
      "action": {
        "type": "string"
      },
      "appliedRulePriority": number,
      "imageDigest": "string",
      "imagePushedAt": number,
      "imageTags": [ "string" ]
    }
  ],
  "registryId": "string",
  "repositoryName": "string",
}```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**lifecyclePolicyText (p. 54)**

The JSON lifecycle policy text.

- **Type:** String
  - **Length Constraints:** Minimum length of 100. Maximum length of 10240.

**nextToken (p. 54)**

The nextToken value to include in a future GetLifecyclePolicyPreview request. When the results of a GetLifecyclePolicyPreview request exceed maxResults, this value can be used to retrieve the next page of results. This value is null when there are no more results to return.

- **Type:** String

**previewResults (p. 54)**

The results of the lifecycle policy preview request.

- **Type:** Array of LifecyclePolicyPreviewResult (p. 99) objects

**registryId (p. 54)**

The registry ID associated with the request.

- **Type:** String
  - **Pattern:** [0-9]{12}

**repositoryName (p. 54)**

The repository name associated with the request.

- **Type:** String
  - **Length Constraints:** Minimum length of 2. Maximum length of 256.
  - **Pattern:** (?::[a-z0-9]+(?:[._-][a-z0-9]+)+)[a-z0-9]+(?:[._-][a-z0-9]+)*

**status (p. 54)**

The status of the lifecycle policy preview request.

- **Type:** String
  - **Valid Values:** IN_PROGRESS | COMPLETE | EXPIRED | FAILED

**summary (p. 54)**

The list of images that is returned as a result of the action.

- **Type:** LifecyclePolicyPreviewSummary (p. 100) object
Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**LifecyclePolicyPreviewNotFoundException**

There is no dry run for this repository.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

**ServerException**

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

Example

This example retrieves the result of a lifecycle policy preview for a repository called project-a/amazon-ecs-sample in the default registry for an account.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.GetLifecyclePolicyPreview
Content-Type: application/x-amz-json-1.1
User-Agent: aws-cli/1.11.144 Python/3.6.1 Darwin/16.6.0 botocore/1.7.2
X-Amz-Date: 20170901T222304Z
Authorization: AUTHPARAMS
Content-Length: 48

{
    "repositoryName": "project-a/amazon-ecs-sample"
}
```
Sample Response

HTTP/1.1 200 OK
Server: Server
Date: Fri, 01 Sep 2017 22:23:06 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 640
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
    "lifecyclePolicyText": "{
        "rules": [
            {
                "rulePriority": 1,
                "description": "Expire images older than 14 days",
                "selection": {
                    "tagStatus": "untagged",
                    "countType": "sinceImagePushed",
                    "countUnit": "days",
                    "countNumber": 14
                },
                "action": {
                    "type": "expire"
                }
            }
        ],
    "registryId": "012345678910",
    "repositoryName": "project-a/amazon-ecs-sample",
    "status": "COMPLETE",
    "summary": {
        "expiringImageTotalCount": 0
    }
}"

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
GetRepositoryPolicy

Retrieves the repository policy for a specified repository.

Request Syntax

```
{
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

registryId (p. 58)

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

repositoryName (p. 58)

The name of the repository with the policy to retrieve.

Type: String


Required: Yes

Response Syntax

```
{
  "policyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.
policyText (p. 58)

The JSON repository policy text associated with the repository.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 10240.

registryId (p. 58)

The registry ID associated with the request.

Type: String

Pattern: [0-9]{12}

repositoryName (p. 58)

The repository name associated with the request.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

RepositoryPolicyNotFoundException

The specified repository and registry combination does not have an associated repository policy.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.
You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example gets the repository policy for the `ubuntu` repository.

Sample Request

```plaintext
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 28
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.GetRepositoryPolicy
X-Amz-Date: 20151215T002404Z
User-Agent: aws-cli/1.9.10 Python/2.7.10 Darwin/14.5.0 botocore/1.3.10
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS
{
  "repositoryName": "ubuntu"
}
```

Sample Response

```plaintext
HTTP/1.1 200 OK
Server: Server
Date: Tue, 15 Dec 2015 00:24:04 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 301
Connection: keep-alive
x-amzn-RequestId: 25da0b72-a2c2-11e5-8543-ebda6fb1393b
{
  "policyText": "{\n    "Version": "2012-10-17",\n    "Statement": [ {\n      "Sid": "AllowPull",\n      "Effect": "Allow",\n      "Principal": "*",\n      "Action": [ "ecr:BatchGetImage", "ecr:GetDownloadUrlForLayer" ],\n      "registerId": "012345678910",\n      "repositoryName": "ubuntu"\n    } ]\n  }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
InitiateLayerUpload

Notify Amazon ECR that you intend to upload an image layer.

**Note**
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the `docker` CLI to pull, tag, and push images.

**Request Syntax**

```json
{
   "registryId": "string",
   "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

`registryId` (p. 62)

The AWS account ID associated with the registry to which you intend to upload layers. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: `[0-9]{12}`

Required: No

`repositoryName` (p. 62)

The name of the repository to which you intend to upload layers.

Type: String


Pattern: `(?:[a-z0-9]+(?:[._-][a-z0-9]+)*[a-z0-9]+(?:[._-][a-z0-9]+)*{0,1})$`

Required: Yes

**Response Syntax**

```json
{
   "partSize": number,
   "uploadId": "string"
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.
The following data is returned in JSON format by the service.

partSize (p. 62)

The size, in bytes, that Amazon ECR expects future layer part uploads to be.

Type: Long

Valid Range: Minimum value of 0.

uploadId (p. 62)

The upload ID for the layer upload. This parameter is passed to further UploadLayerPart (p. 84) and CompleteLayerUpload (p. 16) operations.

Type: String

Pattern: \[0-9a-fA-F\]{8}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{12}\]

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
ListImages

Lists all the image IDs for a given repository.

You can filter images based on whether or not they are tagged by setting the `tagStatus` parameter to `TAGGED` or `UNTAGGED`. For example, you can filter your results to return only `UNTAGGED` images and then pipe that result to a `BatchDeleteImage` (p. 7) operation to delete them. Or, you can filter your results to return only `TAGGED` images to list all of the tags in your repository.

**Request Syntax**

```json
{
    "filter": {
        "tagStatus": "string"
    },
    "maxResults": number,
    "nextToken": "string",
    "registryId": "string",
    "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see [Common Parameters](#).

The request accepts the following data in JSON format.

**filter (p. 64)**

The filter key and value with which to filter your `ListImages` results.

Type: [ListImagesFilter](#) object

Required: No

**maxResults (p. 64)**

The maximum number of image results returned by `ListImages` in paginated output. When this parameter is used, `ListImages` only returns `maxResults` results in a single page along with a `nextToken` response element. The remaining results of the initial request can be seen by sending another `ListImages` request with the returned `nextToken` value. This value can be between 1 and 100. If this parameter is not used, then `ListImages` returns up to 100 results and a `nextToken` value, if applicable.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 100.

Required: No

**nextToken (p. 64)**

The `nextToken` value returned from a previous paginated `ListImages` request where `maxResults` was used and the results exceeded the value of that parameter. Pagination continues from the end of the previous results that returned the `nextToken` value. This value is null when there are no more results to return.
**Note**
This token should be treated as an opaque identifier that is only used to retrieve the next items in a list and not for other programmatic purposes.

Type: String
Required: No

**registryId (p. 64)**
The AWS account ID associated with the registry that contains the repository in which to list images. If you do not specify a registry, the default registry is assumed.

Type: String
Pattern: \[0-9\]{12}
Required: No

**repositoryName (p. 64)**
The repository with image IDs to be listed.

Type: String
Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*+[a-z0-9]+(?:[._-][a-z0-9]+)*
Required: Yes

---

**Response Syntax**

```json
{
    "imageIds": [
        {
            "imageDigest": "string",
            "imageTag": "string"
        }
    ],
    "nextToken": "string"
}
```

**Response Elements**

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**imageIds (p. 65)**
The list of image IDs for the requested repository.

Type: Array of ImageIdentifier (p. 95) objects

Array Members: Minimum number of 1 item. Maximum number of 100 items.

**nextToken (p. 65)**
The `nextToken` value to include in a future ListImages request. When the results of a ListImages request exceed maxResults, this value can be used to retrieve the next page of results. This value is null when there are no more results to return.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example lists all of the images in the amazonlinux repository.

Sample Request

```plaintext
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 33
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.ListImages
X-Amz-Date: 20161216T200542Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AWUTHPARAMS

{
  "repositoryName": "amazonlinux"
}
```

Sample Response

```
HTTP/1.1 200 OK
```

API Version 2015-09-21
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
PutImage

Creates or updates the image manifest and tags associated with an image.

**Note**
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the `docker` CLI to pull, tag, and push images.

**Request Syntax**

```
{
  "imageManifest": "string",
  "imageTag": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

- **imageManifest (p. 68)**
  - The image manifest corresponding to the image to be uploaded.
  - Type: String
  - Required: Yes

- **imageTag (p. 68)**
  - The tag to associate with the image. This parameter is required for images that use the Docker Image Manifest V2 Schema 2 or OCI formats.
  - Type: String
  - Required: No

- **registryId (p. 68)**
  - The AWS account ID associated with the registry that contains the repository in which to put the image. If you do not specify a registry, the default registry is assumed.
  - Type: String
  - Pattern: `[0-9]{12}`
  - Required: No

- **repositoryName (p. 68)**
  - The name of the repository in which to put the image.
  - Type: String
Response Syntax

```
{
    "image": {
        "imageId": {
            "imageDigest": "string",
            "imageTag": "string"
        },
        "imageManifest": "string",
        "registryId": "string",
        "repositoryName": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

image (p. 69)

Details of the image uploaded.

Type: Image (p. 91) object

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

ImageAlreadyExistsException

The specified image has already been pushed, and there were no changes to the manifest or image tag after the last push.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

LayersNotFoundException

The specified layers could not be found, or the specified layer is not valid for this repository.

HTTP Status Code: 400

LimitExceededException

The operation did not succeed because it would have exceeded a service limit for your account. For more information, see Amazon ECR Default Service Limits in the Amazon Elastic Container Registry User Guide.

API Version 2015-09-21

69
HTTP Status Code: 400

RepositoryNotFoundException
The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException
These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example
In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example
This example puts an image to the amazonlinux repository with the tag 2016.09.

Sample Request

```plaintext
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
Content-Length: 653
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.PutImage
X-Amz-Date: 20161216T201255Z
User-Agent: aws-cli/1.11.22 Python/2.7.12 Darwin/16.3.0 botocore/1.4.79
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
   "imageManifest": 
      "schemaVersion": 2,
      "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
      "config": 
         {"mediaType": "application/vnd.docker.container.image.v1+json",
          "size": 102,
          "digest": "sha256:5b52b314511a611975c2c69e695d920acdf8ae8948fe0ef0b7d01881f10b64"
         }
      ,
   "layers": [  
      {"mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
       "size": 91768077,
       "digest": "sha256:8e3fa21c4e40232e835a6761332d225c7af335c5755f44ada2ed9d0e4ab7e8"
      }
   ],
   "repositoryName": "amazonlinux",
   "imageTag": "2016.09"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 16 Dec 2016 20:12:56 GMT
```
Amazon EC2 Container Registry API Reference

See Also

Content-Type: application/x-amz-json-1.1
Content-Length: 786
Connection: keep-alive
x-amzn-RequestId: 084038f1-c3cc-11e6-8d10-9da51cf53fd3

{
   "image": {
      "imageId": {
         "imageDigest": "sha256:f1d4ae3f7261a72e98c6ebeefe9985cf10a0e5a5bd762585a43e0700ed99863807",
         "imageTag": "2016.09"
      },
      "imageManifest": "{
         "schemaVersion": 2,
         "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
         "config": {
            "mediaType": "application/vnd.docker.container.image.v1+json",
            "size": 1486,
            "digest": "sha256:5b52b31451a611975c2c5e695d920acdf8ae8848fe0ef0b7d018df1186b64"
         },
         "layers": [{
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 91768077,
            "digest": "sha256:8e3fa31c440232e835a6761332d225c7af3335c5755f44ada2ed9d0e4ab7e8"
         }]
      }
   },
   "registryId": "012345678910",
   "repositoryName": "amazonlinux"
}

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
PutLifecyclePolicy

Creates or updates a lifecycle policy. For information about lifecycle policy syntax, see Lifecycle Policy Template.

Request Syntax

```
{
  "lifecyclePolicyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**lifecyclePolicyText (p. 72)**

The JSON repository policy text to apply to the repository.

Type: String

Length Constraints: Minimum length of 100. Maximum length of 10240.

Required: Yes

**registryId (p. 72)**

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

**repositoryName (p. 72)**

The name of the repository to receive the policy.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*[a-z0-9]+(?:[._-][a-z0-9]+)*)

Required: Yes

Response Syntax

```
{
  "lifecyclePolicyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

lifecyclePolicyText (p. 72)

The JSON repository policy text.
Type: String
Length Constraints: Minimum length of 100. Maximum length of 10240.

registryId (p. 72)

The registry ID associated with the request.
Type: String
Pattern: [0-9]{12}

repositoryName (p. 72)

The repository name associated with the request.
Type: String
Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*)*[a-z0-9]+(?:[._-][a-z0-9]+)*

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.
HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.
HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.
HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.
You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example creates a lifecycle policy to expire images older than 14 days for a repository called project-a/amazon-ecs-sample in the default registry for an account.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.PutLifecyclePolicy
Content-Type: application/x-amz-json-1.1
User-Agent: aws-cli/1.11.144 Python/3.6.1 Darwin/16.6.0 botocore/1.7.2
X-Amz-Date: 20170901T194217Z
Authorization: AUTHPARAMS
Content-Length: 535

{
    "repositoryName": "project-a/amazon-ecs-sample",
    "lifecyclePolicyText": "{\n        "rules": [{\n            "rulePriority": 1,
            "description": "Expire images older than 14 days",
            "selection": {
                "tagStatus": "untagged",
                "countType": "sinceImagePushed",
                "countUnit": "days",
                "countNumber": 14
            },
            "action": {
                "type": "expire"
            }
        }
    ]
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Fri, 01 Sep 2017 19:42:18 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 340
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

{
    "lifecyclePolicyText": "{\n        "rules": [{\n            "rulePriority": 1,
            "description": "Expire images older than 14 days",
            "selection": {
                "tagStatus": "untagged",
                "countType": "sinceImagePushed",
                "countUnit": "days",
                "countNumber": 14
            },
            "action": {
                "type": "expire"
            }
        }
    ]
},

    "registryId": "012345678910",
    "repositoryName": "project-a/amazon-ecs-sample"
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
• AWS SDK for Go
• AWS SDK for Java
• AWS SDK for JavaScript
• AWS SDK for PHP V3
• AWS SDK for Python
• AWS SDK for Ruby V2
SetRepositoryPolicy

Applies a repository policy on a specified repository to control access permissions.

Request Syntax

```
{
  "force": boolean,
  "policyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**force (p. 76)**

If the policy you are attempting to set on a repository policy would prevent you from setting another policy in the future, you must force the SetRepositoryPolicy (p. 76) operation. This is intended to prevent accidental repository lock outs.

Type: Boolean

Required: No

**policyText (p. 76)**

The JSON repository policy text to apply to the repository.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 10240.

Required: Yes

**registryId (p. 76)**

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: \[0-9\]\{12\}

Required: No

**repositoryName (p. 76)**

The name of the repository to receive the policy.

Type: String


Pattern: (?::[a-z0-9]+)(?:[._-][a-z0-9]+)+?\*[a-z0-9]+(?::[._-][a-z0-9]+)*
Response Syntax

```json
{
   "policyText": "string",
   "registryId": "string",
   "repositoryName": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

**policyText (p. 77)**

The JSON repository policy text applied to the repository.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 10240.

**registryId (p. 77)**

The registry ID associated with the request.

Type: String

Pattern: [0-9]{12}

**repositoryName (p. 77)**

The repository name associated with the request.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*[^a-z0-9]+(?:[._-][a-z0-9]+)*)

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**RepositoryNotFoundException**

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400
ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don’t need to learn how to sign requests yourself.

Example

The following example sets a repository policy on the ubuntu repository that allows all AWS accounts to pull from it.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 223
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.SetRepositoryPolicy
X-Amz-Date: 20151214T235302Z
User-Agent: aws-cli/1.9.10 Python/2.7.10 Darwin/14.5.0 botocore/1.3.10
Content-Type: application/x-amz-json-1.1
Authorization: AWUTHPARAMS
{
  "policyText": "\n  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowPull",
    "Effect": "Allow",
    "Principal": "*",
    "Action": ["ecr:BatchGetImage", "ecr:GetDownloadUrlForLayer"]
  }],
  "repositoryName": "ubuntu"
}
```

Sample Response

```
HTTP/1.1 200 OK
Server: Server
Date: Mon, 14 Dec 2015 23:53:02 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 301
Connection: keep-alive
x-amzn-RequestId: cfc3ead9-a2bd-11e5-91c7-7126cb670c2b
{
  "policyText": "\n  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowPull",
    "Effect": "Allow",
    "Principal": "*",
    "Action": ["ecr:BatchGetImage", "ecr:GetDownloadUrlForLayer"]
  }],
  "repositoryName": "ubuntu"
}
```
See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
StartLifecyclePolicyPreview

Starts a preview of the specified lifecycle policy. This allows you to see the results before creating the lifecycle policy.

Request Syntax

```
{
  "lifecyclePolicyText": "string",
  "registryId": "string",
  "repositoryName": "string"
}
```

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

**lifecyclePolicyText (p. 80)**

The policy to be evaluated against. If you do not specify a policy, the current policy for the repository is used.

Type: String

Length Constraints: Minimum length of 100. Maximum length of 10240.

Required: No

**registryId (p. 80)**

The AWS account ID associated with the registry that contains the repository. If you do not specify a registry, the default registry is assumed.

Type: String

Pattern: [0-9]{12}

Required: No

**repositoryName (p. 80)**

The name of the repository to be evaluated.

Type: String


Pattern: (?:[a-z0-9]+(\:[_\-][a-z0-9]+)*[a-z0-9]+(?:[_\-][a-z0-9]+)*)

Required: Yes

Response Syntax

```
{
}
```
Response Elements

If the action is successful, the service sends back an HTTP 200 response. The following data is returned in JSON format by the service.

**lifecyclePolicyText (p. 80)**

The JSON repository policy text.

Type: String

Length Constraints: Minimum length of 100. Maximum length of 10240.

**registryId (p. 80)**

The registry ID associated with the request.

Type: String

Pattern: \[0-9\]\{12\}

**repositoryName (p. 80)**

The repository name associated with the request.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*/)*[a-z0-9]+(?:[._-][a-z0-9]+)*

**status (p. 80)**

The status of the lifecycle policy preview request.

Type: String

Valid Values: IN_PROGRESS | COMPLETE | EXPIRED | FAILED

Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

**InvalidParameterException**

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

**LifecyclePolicyNotFoundException**

The lifecycle policy could not be found, and no policy is set to the repository.

HTTP Status Code: 400
LifecyclePolicyPreviewInProgressException

The previous lifecycle policy preview request has not completed. Please try again later.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

Example

In the following example or examples, the Authorization header contents (AUTHPARAMS) must be replaced with an AWS Signature Version 4 signature. For more information about creating these signatures, see Signature Version 4 Signing Process in the AWS General Reference.

You only need to learn how to sign HTTP requests if you intend to manually create them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

Example

This example creates a lifecycle policy preview to expire images older than 14 days for a repository called project-a/amazon-ecs-sample in the default registry for an account.

Sample Request

```
POST / HTTP/1.1
Host: ecr.us-west-2.amazonaws.com
Accept-Encoding: identity
X-Amz-Target: AmazonEC2ContainerRegistry_V20150921.StartLifecyclePolicyPreview
Content-Type: application/x-amz-json-1.1
User-Agent: aws-cli/1.11.144 Python/3.6.1 Darwin/16.6.0 botocore/1.7.2
X-Amz-Date: 20170901T221604Z
Authorization: AUTHPARAMS
Content-Length: 535

{
    "repositoryName": "project-a/amazon-ecs-sample",
    "lifecyclePolicyText": "{\n        \"rules\": [{\n            \"rulePriority\": 1,
            \"description\": \"Expire images older than 14 days\",
            \"selection\": {
                \"tagStatus\": \"untagged\",
                \"countType\": \"sinceImagePushed\",
                \"countUnit\": \"days\",
                \"countNumber\": 14
            },
            \"action\": {
                \"type\": \"expire\"
            }
        }], \n    }
}
```

Sample Response

```
HTTP/1.1 200 OK
```
Server: Server
Date: Fri, 01 Sep 2017 22:16:05 GMT
Content-Type: application/x-amz-json-1.1
Content-Length: 583
Connection: keep-alive
x-amzn-RequestId: 123a4b56-7c89-01d2-3ef4-example5678f

```json
{
    "lifecyclePolicyText": "{
        "rules": [
            {
                "rulePriority": 1,
                "description": "Expire images older than 14 days",
                "selection": {
                    "tagStatus": "untagged",
                    "countType": "sinceImagePushed",
                    "countUnit": "days",
                    "countNumber": 14
                },
                "action": {
                    "type": "expire"
                }
            }
        ]
    }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
UploadLayerPart

Uploads an image layer part to Amazon ECR.

**Note**
This operation is used by the Amazon ECR proxy, and it is not intended for general use by customers for pulling and pushing images. In most cases, you should use the `docker` CLI to pull, tag, and push images.

**Request Syntax**

```json
{
    "layerPartBlob": blob,
    "partFirstByte": number,
    "partLastByte": number,
    "registryId": "string",
    "repositoryName": "string",
    "uploadId": "string"
}
```

**Request Parameters**

For information about the parameters that are common to all actions, see Common Parameters (p. 105).

The request accepts the following data in JSON format.

- **layerPartBlob (p. 84)**
  
  The base64-encoded layer part payload.
  
  Type: Base64-encoded binary data object
  
  Required: Yes

- **partFirstByte (p. 84)**
  
  The integer value of the first byte of the layer part.
  
  Type: Long
  
  Valid Range: Minimum value of 0.
  
  Required: Yes

- **partLastByte (p. 84)**
  
  The integer value of the last byte of the layer part.
  
  Type: Long
  
  Valid Range: Minimum value of 0.
  
  Required: Yes

- **registryId (p. 84)**
  
  The AWS account ID associated with the registry to which you are uploading layer parts. If you do not specify a registry, the default registry is assumed.
  
  Type: String
Pattern: \[0-9\]{12}\n
Required: No
repositoryName (p. 84)

The name of the repository to which you are uploading layer parts.
Type: String
Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*[a-z0-9]+(?:[._-][a-z0-9]+)+)*

Required: Yes
uploadId (p. 84)

The upload ID from a previous InitiateLayerUpload (p. 62) operation to associate with the layer part upload.
Type: String
Pattern: \[0-9a-fA-F\]{8}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{4}-\[0-9a-fA-F\]{12}\n
Required: Yes

Response Syntax

```
{
  "lastByteReceived": number,
  "registryId": "string",
  "repositoryName": "string",
  "uploadId": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

lastByteReceived (p. 85)

The integer value of the last byte received in the request.
Type: Long
Valid Range: Minimum value of 0.
registryId (p. 85)

The registry ID associated with the request.
Type: String
Pattern: \[0-9\]{12}\nrepositoryName (p. 85)

The repository name associated with the request.
Errors

For information about the errors that are common to all actions, see Common Errors (p. 107).

InvalidLayerPartException

The layer part size is not valid, or the first byte specified is not consecutive to the last byte of a previous layer part upload.

HTTP Status Code: 400

InvalidParameterException

The specified parameter is invalid. Review the available parameters for the API request.

HTTP Status Code: 400

LimitExceededException

The operation did not succeed because it would have exceeded a service limit for your account. For more information, see Amazon ECR Default Service Limits in the Amazon Elastic Container Registry User Guide.

HTTP Status Code: 400

RepositoryNotFoundException

The specified repository could not be found. Check the spelling of the specified repository and ensure that you are performing operations on the correct registry.

HTTP Status Code: 400

ServerException

These errors are usually caused by a server-side issue.

HTTP Status Code: 500

UploadNotFoundException

The upload could not be found, or the specified upload id is not valid for this repository.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:
See Also

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2
Data Types

The Amazon EC2 Container Registry API contains several data types that various actions use. This section describes each data type in detail.

**Note**
The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

- `AuthorizationData (p. 89)`
- `DescribeImagesFilter (p. 90)`
- `Image (p. 91)`
- `ImageDetail (p. 92)`
- `ImageFailure (p. 94)`
- `ImageIdentifier (p. 95)`
- `Layer (p. 96)`
- `LayerFailure (p. 97)`
- `LifecyclePolicyPreviewFilter (p. 98)`
- `LifecyclePolicyPreviewResult (p. 99)`
- `LifecyclePolicyPreviewSummary (p. 100)`
- `LifecyclePolicyRuleAction (p. 101)`
- `ListImagesFilter (p. 102)`
- `Repository (p. 103)`
AuthorizationData

An object representing authorization data for an Amazon ECR registry.

Contents

authorizationToken

A base64-encoded string that contains authorization data for the specified Amazon ECR registry. When the string is decoded, it is presented in the format user:password for private registry authentication using docker login.

Type: String
Pattern: ^\S+$
Required: No

expiresAt

The Unix time in seconds and milliseconds when the authorization token expires. Authorization tokens are valid for 12 hours.

Type: Timestamp
Required: No

proxyEndpoint

The registry URL to use for this authorization token in a docker login command. The Amazon ECR registry URL format is https://aws_account_id.dkr.ecr.region.amazonaws.com. For example, https://012345678910.dkr.ecr.us-east-1.amazonaws.com..

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
DescribeImagesFilter

An object representing a filter on a DescribeImages (p. 34) operation.

Contents

tagStatus

The tag status with which to filter your DescribeImages (p. 34) results. You can filter results based on whether they are TAGGED or UNTAGGED.

Type: String

Valid Values: TAGGED | UNTAGGED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Image
An object representing an Amazon ECR image.

Contents

imageId
An object containing the image tag and image digest associated with an image.
Type: ImageIdentifier (p. 95) object
Required: No

imageManifest
The image manifest associated with the image.
Type: String
Required: No

registryId
The AWS account ID associated with the registry containing the image.
Type: String
Pattern: [0-9]{12}
Required: No

repositoryName
The name of the repository associated with the image.
Type: String
Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*)*[a-z0-9]+(?:[._-][a-z0-9]+)*
Required: No

See Also
For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ImageDetail

An object that describes an image returned by a DescribeImages (p. 34) operation.

Contents

imageDigest
The sha256 digest of the image manifest.
Type: String
Required: No

imagePushedAt
The date and time, expressed in standard JavaScript date format, at which the current image was pushed to the repository.
Type: Timestamp
Required: No

imageSizeInBytes
The size, in bytes, of the image in the repository.
Note
Beginning with Docker version 1.9, the Docker client compresses image layers before pushing them to a V2 Docker registry. The output of the docker images command shows the uncompressed image size, so it may return a larger image size than the image sizes returned by DescribeImages (p. 34).
Type: Long
Required: No

imageTags
The list of tags associated with this image.
Type: Array of strings
Required: No

registryId
The AWS account ID associated with the registry to which this image belongs.
Type: String
Pattern: [0-9]{12}
Required: No

repositoryName
The name of the repository to which this image belongs.
Type: String
Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]*)*/)[a-z0-9]+(?:[._-][a-z0-9]*)*
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ImageFailure

An object representing an Amazon ECR image failure.

Contents

failureCode

The code associated with the failure.

Type: String

Valid Values: InvalidImageDigest | InvalidImageTag | ImageTagDoesNotMatchDigest | ImageNotFound | MissingDigestAndTag

Required: No

failureReason

The reason for the failure.

Type: String

Required: No

imageId

The image ID associated with the failure.

Type: ImageIdentifier (p. 95) object

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ImageIdentifier

An object with identifying information for an Amazon ECR image.

Contents

imageDigest

The sha256 digest of the image manifest.

Type: String
Required: No

imageTag

The tag used for the image.

Type: String
Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Layer

An object representing an Amazon ECR image layer.

Contents

layerAvailability

The availability status of the image layer.

Type: String

Valid Values: AVAILABLE | UNAVAILABLE

Required: No

layerDigest

The sha256 digest of the image layer.

Type: String

Pattern: [a-zA-Z0-9-_+.]+:[a-fA-F0-9]+

Required: No

layerSize

The size, in bytes, of the image layer.

Type: Long

Required: No

mediaType

The media type of the layer, such as application/vnd.docker.image.rootfs.diff.tar.gz or application/vnd.oci.image.layer.v1.tar+gzip.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LayerFailure

An object representing an Amazon ECR image layer failure.

Contents

failureCode

The failure code associated with the failure.

Type: String

Valid Values: InvalidLayerDigest | MissingLayerDigest

Required: No

failureReason

The reason for the failure.

Type: String

Required: No

layerDigest

The layer digest associated with the failure.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 1000.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LifecyclePolicyPreviewFilter

The filter for the lifecycle policy preview.

Contents

tagStatus

The tag status of the image.

Type: String

Valid Values: TAGGED | UNTAGGED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LifecyclePolicyPreviewResult

The result of the lifecycle policy preview.

Contents

action

The type of action to be taken.

Type: LifecyclePolicyRuleAction (p. 101) object

Required: No

appliedRulePriority

The priority of the applied rule.

Type: Integer

Valid Range: Minimum value of 1.

Required: No

imageDigest

The sha256 digest of the image manifest.

Type: String

Required: No

imagePushedAt

The date and time, expressed in standard JavaScript date format, at which the current image was pushed to the repository.

Type: Timestamp

Required: No

imageTags

The list of tags associated with this image.

Type: Array of strings

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LifecyclePolicyPreviewSummary

The summary of the lifecycle policy preview request.

Contents

expiringImageTotalCount

The number of expiring images.

Type: Integer

Valid Range: Minimum value of 0.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
LifecyclePolicyRuleAction

The type of action to be taken.

Contents

type

The type of action to be taken.

Type: String

Valid Values: EXPIRE

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
ListImagesFilter

An object representing a filter on a ListImages (p. 64) operation.

Contents

tagStatus

The tag status with which to filter your ListImages (p. 64) results. You can filter results based on whether they are TAGGED or UNTAGGED.

Type: String

Valid Values: TAGGED | UNTAGGED

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Repository

An object representing a repository.

Contents

createdAt

The date and time, in JavaScript date format, when the repository was created.

Type: Timestamp

Required: No

registryId

The AWS account ID associated with the registry that contains the repository.

Type: String

Pattern: [0-9]{12}

Required: No

repositoryArn

The Amazon Resource Name (ARN) that identifies the repository. The ARN contains the arn:aws:ecr namespace, followed by the region of the repository, AWS account ID of the repository owner, repository namespace, and repository name. For example, arn:aws:ecr:region:012345678910:repository/test.

Type: String

Required: No

repositoryName

The name of the repository.

Type: String


Pattern: (?:[a-z0-9]+(?:[._-][a-z0-9]+)*)*[a-z0-9]+(?:[._-][a-z0-9]+)*

Required: No

repositoryUri

The URI for the repository. You can use this URI for Docker push or pull operations.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
See Also

- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2
Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signature Version 4 Signing Process in the Amazon Web Services General Reference.

**Action**

The action to be performed.

Type: string

Required: Yes

**Version**

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

**X-Amz-Algorithm**

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: AWS4-HMAC-SHA256

Required: Conditional

**X-Amz-Credential**

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string (“aws4_request”). The value is expressed in the following format: access_key/YYYYMMDD/region/service/aws4_request.

For more information, see Task 2: Create a String to Sign for Signature Version 4 in the Amazon Web Services General Reference.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

**X-Amz-Date**

The date that is used to create the signature. The format must be ISO 8601 basic format (‘YYYYMMDD’T’HHMMSS’Z’). For example, the following date time is a valid X-Amz-Date value: 20120325T120000Z.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is
not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Handling Dates in Signature Version 4 in the Amazon Web Services General Reference.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS Security Token Service, go to AWS Services That Work with IAM in the IAM User Guide.

Condition: If you're using temporary security credentials from the AWS Security Token Service, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Task 1: Create a Canonical Request For Signature Version 4 in the Amazon Web Services General Reference.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional
Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

**AccessDeniedException**

You do not have sufficient access to perform this action.

HTTP Status Code: 400

**IncompleteSignature**

The request signature does not conform to AWS standards.

HTTP Status Code: 400

**InternalFailure**

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

**InvalidAction**

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

**InvalidClientTokenId**

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

**InvalidParameterCombination**

Parameters that must not be used together were used together.

HTTP Status Code: 400

**InvalidParameterValue**

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

**InvalidQueryParameter**

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

**MalformedQueryString**

The query string contains a syntax error.

HTTP Status Code: 404

**MissingAction**

The request is missing an action or a required parameter.

HTTP Status Code: 400
MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400