



开发人员指南

# Amazon Pinpoint



# Amazon Pinpoint: 开发人员指南

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆或者贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

# Table of Contents

什么是 Amazon Pinpoint? .....	1
Amazon Pinpoint 功能 .....	1
定义受众客户细分 .....	1
通过消息传送活动与您的受众交互 .....	1
发送事务性消息 .....	1
分析用户行为 .....	2
区域可用性 .....	2
教程 .....	3
将 Postman 用于 Amazon Pinpoint API .....	3
关于本教程 .....	3
先决条件 .....	4
步骤 1：创建 IAM 策略和角色 .....	5
步骤 2：设置 Postman .....	8
步骤 3：发送其他请求 .....	15
设置短信注册系统 .....	21
关于双重选择加入 .....	21
关于此解决方案 .....	21
先决条件 .....	23
步骤 1：设置 Amazon Pinpoint .....	24
步骤 2：创建 IAM 策略和角色 .....	29
步骤 3：创建 Lambda 函数 .....	32
步骤 4：设置 Amazon API Gateway .....	43
步骤 5：创建和部署 Web 表单 .....	48
后续步骤 .....	55
与您的应用程序集成 .....	58
使用 AWS SDK .....	58
使用 AWS Amplify 连接您的前端应用程序 .....	59
后续步骤 .....	60
注册端点 .....	60
开始前的准备工作 .....	60
AWS 移动开发工具包 .....	60
AWS Amplify .....	61
后续步骤 .....	61
报告事件 .....	61

开始前的准备工作 .....	62
AWS 移动开发工具包 .....	62
Web 和 React Native .....	61
Amazon Pint 事件 API .....	63
后续步骤 .....	63
处理推送通知 .....	63
设置推送通知 .....	63
处理推送通知 .....	65
定义受众 .....	66
添加端点 .....	67
示例 .....	67
相关信息 .....	72
将用户与端点关联 .....	73
示例 .....	73
相关信息 .....	77
批量添加端点 .....	78
示例 .....	78
相关信息 .....	86
导入端点 .....	86
开始之前 .....	86
示例 .....	86
相关信息 .....	98
删除端点 .....	98
示例 .....	98
管理受众成员的端点 .....	101
访问受众数据 .....	103
查找端点 .....	103
示例 .....	104
相关信息 .....	109
导出端点 .....	109
开始之前 .....	110
示例 .....	110
相关信息 .....	121
列出端点 ID .....	121
创建分段 .....	124
构建客户细分 .....	124

使用 AWS SDK for Java 构建分段 .....	124
导入客户细分 .....	127
导入客户细分 .....	128
使用 AWS Lambda 自定义分段 .....	130
事件数据 .....	131
创建 Lambda 函数 .....	132
分配 Lambda 函数策略 .....	134
将 Lambda 函数分配给活动 .....	136
创建活动 .....	138
创建标准活动 .....	138
使用创建广告系列 AWS SDK for Java .....	138
创建 A/B 测试活动 .....	141
使用创建 A/B 测试广告系列 AWS SDK for Java .....	141
使用 SMS 和 Voice API .....	144
SMS 和 Voice API 第 2 版中的概念 .....	145
管理电话号码 .....	146
申请电话号码 .....	147
修改电话号码功能 .....	149
列出电话号码 .....	151
释放电话号码 .....	153
管理发件人 ID .....	154
将发件人 ID 与池关联 .....	155
为配置集设置默认发件人 ID .....	156
取消发件人 ID 与配置集的关联 .....	156
管理池 .....	157
创建池 .....	158
将发起身份添加到池中 .....	159
列出池 .....	161
列出池中的发起身份池 .....	163
删除池 .....	163
管理选择退出列表 .....	164
创建选择退出列表 .....	165
列出账户中的选择退出列表 .....	166
将目标号码添加到选择退出列表中 .....	167
从选择退出列表中删除目标号码 .....	168
管理配置集 .....	169

创建配置集 .....	170
列出配置集 .....	171
删除配置集 .....	172
管理关键字 .....	173
创建关键字 .....	174
列出关键字 .....	175
删除关键字 .....	175
管理事件目标 .....	176
事件类型 .....	177
创建事件目标 .....	179
更新事件目标 .....	179
删除事件目标 .....	180
管理 CloudWatch Logs 事件目标 .....	181
管理 Kinesis Data Firehose 事件目标 .....	185
管理 Amazon SNS 事件目标 .....	190
发送消息 .....	194
发送短信 .....	194
发送语音消息 .....	196
发送和验证一次性密码 (OTP) .....	199
发送 OTP 消息 .....	199
SendOtpMessage 响应 .....	202
验证 OTP 消息 .....	202
VerifyOtpMessage 响应 .....	203
代码示例 .....	204
生成参考 ID .....	204
发送 OTP 代码 .....	204
验证 OTP 代码 .....	206
发送和检索应用程序内消息 .....	207
检索端点的应用程序内消息 .....	207
了解 GetInAppMessages API 响应 .....	209
InAppMessageCampaigns 对象 .....	211
InAppMessage 对象 .....	212
HeaderConfig 对象 .....	213
BodyConfig 对象 .....	213
InAppMessageContent 对象 .....	214
Schedule 对象 .....	215

InAppMessageButton 对象 .....	215
DefaultButtonConfig 对象 .....	216
OverrideButtonConfig 对象 .....	217
验证电话号码 .....	218
电话号码验证使用案例 .....	218
使用电话号码验证服务 .....	219
电话号码验证响应 .....	220
发送消息 .....	224
发送电子邮件 .....	224
选择一种电子邮件发送方法 .....	225
在 Amazon Pinpoint 与 Amazon Simple Email Service (SES) 之间进行选择 .....	225
使用 API .....	225
发送 SMS 消息 .....	239
发送语音消息 .....	251
发送推送通知 .....	259
创建自定义渠道 .....	269
创建通过自定义渠道发送消息的活动 .....	269
了解事件数据 .....	270
配置 Webhook .....	271
配置 Lambda 函数 .....	271
示例 Lambda 函数 .....	272
Amazon Pinpoint Lambda 函数响应格式 .....	276
授予 Amazon Pinpoint 权限以调用 Lambda 函数 .....	277
流式传输事件 .....	279
设置事件流式传输 .....	280
先决条件 .....	280
AWS CLI .....	280
AWS SDK for Java .....	280
禁用事件流式传输 .....	281
应用程序事件 .....	282
示例 .....	282
应用程序事件属性 .....	283
活动事件 .....	287
示例事件 .....	287
活动事件属性 .....	288
旅程事件 .....	294

示例事件 .....	294
旅程事件属性 .....	295
电子邮件事件 .....	298
示例事件 .....	299
电子邮件事件属性 .....	305
短信事件 .....	310
示例 .....	311
短信事件属性 .....	311
查询分析数据 .....	319
受支持的指标 .....	319
查询基础知识 .....	320
IAM 策略 .....	321
标准指标 .....	324
活动的应用程序指标 .....	325
事务性电子邮件消息的应用程序指标 .....	329
事务性短信的应用程序指标 .....	335
活动指标 .....	340
旅程参与指标 .....	345
旅程执行指标 .....	350
旅程活动执行指标 .....	351
旅程和活动执行指标 .....	354
查询活动数据 .....	356
先决条件 .....	356
查询一个活动的数据 .....	357
查询多个活动的数据 .....	362
查询事务性消息数据 .....	367
先决条件 .....	367
查询事务性电子邮件消息的数据 .....	368
查询事务性短信的数据 .....	372
使用查询结果 .....	377
JSON 结构 .....	377
JSON 对象和字段 .....	382
记录 API 调用 .....	384
CloudTrail 中的 Amazon Pinpoint 信息 .....	384
CloudTrail 可以记录的 Amazon Pinpoint API 操作 .....	385
CloudTrail 可以记录的 Amazon Pinpoint Email API 操作 .....	389



CloudTrail 可以记录的 Amazon Pinpoint SMS 和 Voice API 第 1 版操作 .....	390
CloudTrail 可以记录的 Amazon Pinpoint SMS 和 Voice API 第 2 版操作 .....	391
示例：Amazon Pinpoint 日志文件条目 .....	393
为资源添加标签 .....	398
管理标签 .....	398
在 IAM 策略中使用标签 .....	399
向资源添加标签 .....	399
使用 API 添加标签 .....	400
使用 AWS CLI 添加标签 .....	400
显示资源的标签 .....	402
使用 API 显示标签 .....	402
使用 AWS CLI 显示标签 .....	403
更新资源的标签 .....	403
从资源中删除标签 .....	404
使用 API 删除标签 .....	404
使用 AWS CLI 删除标签 .....	405
相关信息 .....	405
使用 AWS Lambda 自定义建议 .....	406
在消息中使用建议 .....	406
创建 Lambda 函数 .....	408
输入事件数据 .....	408
响应数据和要求 .....	410
分配 Lambda 函数策略 .....	415
授权 Amazon Pinpoint 调用该函数 .....	416
配置推荐器模型 .....	417
删除数据 .....	418
删除端点 .....	418
从 Amazon S3 中删除分段和端点数据 .....	418
删除所有项目数据 .....	419
删除所有 AWS 数据 .....	419
代码示例 .....	421
Amazon Pinpoint 示例 .....	421
操作 .....	422
Amazon Pinpoint SMS 和 Voice API 示例 .....	501
操作 .....	502
安全性 .....	511

数据保护 .....	511
数据加密 .....	513
互连网络流量隐私 .....	514
为 Amazon Pinpoint 创建接口 VPC 端点 .....	514
Identity and Access Management .....	516
受众 .....	516
使用身份进行身份验证 .....	517
使用策略管理访问 .....	519
Amazon Pinpoint 如何与 IAM 配合使用 .....	521
Amazon Pinpoint 策略操作 .....	527
基于身份的策略示例 .....	563
常见任务的 IAM 角色 .....	575
问题排查 .....	588
日志记录和监控 .....	590
合规性验证 .....	591
韧性 .....	592
基础设施安全性 .....	592
配置和漏洞分析 .....	593
安全最佳实操 .....	593
限额 .....	594
项目限额 .....	594
API 请求限额 .....	595
SMS 和 Voice API 第 2 版请求限额 .....	597
活动限额 .....	599
电子邮件限额 .....	600
电子邮件限额 .....	601
电子邮件发送人和接收人限额 .....	601
电子邮件发送限额 .....	602
端点限额 .....	603
端点导入限额 .....	604
事件提取限额 .....	605
旅程限额 .....	606
Lambda 限额 .....	606
机器学习限额 .....	607
消息模板限额 .....	608
推送通知限额 .....	609

---

应用程序内消息限额 .....	610
分段限额 .....	610
短信限额 .....	611
10DLC 限额 .....	612
语音限额 .....	613
请求提高限额 .....	615
文档历史记录 .....	617
早期更新 .....	622
.....	dcxxvi

# 什么是 Amazon Pinpoint ?

Amazon Pinpoint 是一项 AWS 服务，您可以用它来跨多种消息传送渠道与客户交互。您可以使用 Amazon Pinpoint 发送推送通知、电子邮件、短信或语音消息。

本开发人员指南中的信息适用于应用程序开发人员。本指南包含有关以编程方式使用 Amazon Pinpoint 的功能的信息。它还包含移动应用程序开发人员特别感兴趣的信息，例如[将分析和消息收发功能与您的应用程序集成的过程](#)。

本文档有若干个随附文档。以下文档提供与 Amazon Pinpoint API 相关的参考信息：

- [Amazon Pinpoint API 参考](#)
- [Amazon Pinpoint SMS 和 Voice API](#)

如果您是刚开始使用 Amazon Pinpoint，或许先查看 [Amazon Pinpoint 用户指南](#) 再来阅读本文档会很有用。

## Amazon Pinpoint 功能

本节介绍 Amazon Pinpoint 的主要功能以及您可以使用它们执行的任务。

### 定义受众客户细分

通过[定义受众客户细分](#)联系合适的消息受众。客户细分指定哪些用户接收活动所发出的消息。您可以根据应用程序报告的数据（如操作系统或移动设备类型）定义动态客户细分。您还可以使用其他服务或应用程序来导入您定义的静态细分。

### 通过消息传送活动与您的受众交互

通过[创建消息传送活动](#)与受众交互。活动按照您定义的计划发送定制消息。您可以创建发送移动推送、电子邮件或短信的活动。

要体验备选活动战略，请将活动设置为 A/B 测试，并通过 Amazon Pinpoint 分析来分析结果。

### 发送事务性消息

通过向特定用户直接发送事务性移动推送和短信（如新账户激活消息、订单确认和密码重置通知等），及时向客户通报信息。您可以使用 Amazon Pinpoint REST API 发送事务性消息。

## 分析用户行为

使用 Amazon Pinpoint 提供的分析功能，洞察受众及活动有效性情况。您可以查看有关用户参与度、购买活动、人数统计等的趋势。您还可以通过查看指标（例如为活动或应用程序发送或打开的消息总数）来监控消息流量。通过 Amazon Pinpoint API，您的应用程序可报告自定义数据，Amazon Pinpoint 将分析这些数据，而您可以查询某些标准指标的分析数据。

要在 Amazon Pinpoint 外部分析或存储分析数据，您可以配置 Amazon Pinpoint，[将数据流式传输到 Amazon Kinesis](#)。

## 区域可用性

Amazon Pinpoint 在北美、欧洲、亚洲和大洋洲的多个 AWS 区域中均已推出。在每个区域中，AWS 将维护多个可用区。这些可用区的物理位置是相互隔离的，但可通过私有、低延迟、高吞吐量和高度冗余的网络连接联合在一起。这些可用区使我们能够提供极高水平的可用性和冗余，同时最大程度地减少延迟。

要了解有关 AWS 区域的更多信息，请参阅《Amazon Web Services 一般参考》中的[管理 AWS 区域](#)。有关目前可用 Amazon Pinpoint 的所有区域的列表，请参阅《Amazon Web Services 一般参考》中的[Amazon Pinpoint 端点和限额](#)以及[AWS 服务端点](#)。要详细了解每个区域中可用的可用区数量，请参阅[AWS 全球基础设施](#)。

# 教程

本部分中的教程旨在向新 Amazon Pinpoint 用户介绍如何完成几个重要任务。如果您是初次使用 Amazon Pinpoint，或者不熟悉某些功能，则这些教程是一个绝佳的切入点。

本指南中的教程包括面向开发人员或系统管理员的任务。这些教程介绍如何使用 Amazon Pinpoint API、AWS SDK 和 AWS CLI 来执行任务。如果您主要使用基于 Web 的控制台与 Amazon Pinpoint 交互，请参阅《Amazon Pinpoint 用户指南》的“教程”部分。

## 教程

- [教程：将 Postman 用于 Amazon Pinpoint API](#)
- [教程：设置短信注册系统](#)

## 教程：将 Postman 用于 Amazon Pinpoint API

Postman 是一款流行的工具，可在易于使用的图形环境中测试 API。您可以使用 Postman 向任何 REST API 发送 API 请求，并接收对请求的响应。使用 Postman 可以方便地测试和排查您对 Amazon Pinpoint API 进行的调用。本教程包含设置 Postman 和将 Postman 用于 Amazon Pinpoint API 的过程。

### Note

Postman 是由第三方公司开发的，而不是由 Amazon Web Services (AWS) 开发或支持的。要了解有关使用 Postman 的更多信息，或需要帮助以解决与 Postman 相关的问题，请参阅 Postman 网站上的[支持中心](#)。

## 关于本教程

此部分包含本教程的概述。

### 目标受众

本教程适用于开发人员和系统实施者。虽然您不必非得熟悉 Amazon Pinpoint 或 Postman 才能完成本教程中的步骤，但您应熟悉管理 IAM 策略和修改 JSON 代码示例。

本教程中的过程旨在防止新用户使用可永久删除 Amazon Pinpoint 资源的 API 操作。高级用户可通过修改与其用户关联的策略来删除此限制。

## 使用的功能

本教程包含以下 Amazon Pinpoint 功能的使用示例：

- 通过使用 Postman 与 Amazon Pinpoint API 交互

## 所需时间

完成本教程大约需要 15 分钟。

## 区域限制

没有与使用此解决方案关联的区域限制。

## 资源用量费用

创建 AWS 账户并不会收费；但是，如果您使用 Postman 执行以下任一操作，则实施此解决方案可能会产生 AWS 用量费用：

- 发送电子邮件、SMS、移动推送或语音消息
- 创建和发送活动
- 使用电话号码验证功能

有关与使用 Amazon Pinpoint 关联的费用的更多信息，请参阅 [Amazon Pinpoint 定价](#)。

## 先决条件

开始本教程之前，请完成以下先决条件：

- 您必须具有 AWS 账户。要创建 AWS 账户，请前往 <https://console.aws.amazon.com/> 并选择创建新的 AWS 账户。
- 确保用于登录 AWS Management Console 的账户能够创建新的 IAM 策略和角色。
- 确保您创建了至少一个示例项目，该项目已开启电子邮件并具有经过验证的电子邮件身份。请参阅《Amazon Pinpoint 用户指南》中的 [创建带有电子邮件支持的 Amazon Pinpoint 项目](#)。
- 确保您具有 AWS 账户 ID。可以在控制台的右上角找到您的 AWS 账户 ID，或者，您可以使用命令行界面 (CLI) 查找。请参阅 [查找您的 AWS 账户 ID](#)。
- 您必须在计算机上下载并安装 Postman。您可以从 [Postman 网站](#) 下载 Postman。

- 在计算机上安装 Postman 后，您必须创建一个 Postman 账户。当您首次启动 Postman 应用程序时，系统会提示您登录或创建新账户。请按照 Postman 提供的说明登录您的账户，或者，如果您还没有此账户，则创建一个。

## 步骤 1：创建 IAM 策略和角色

当您使用 Postman 测试 Amazon Pinpoint API 时，第一步是创建用户。在本部分，您将创建一个策略，以允许用户能够与所有 Amazon Pinpoint 资源交互。然后，您创建一个用户，并将该策略直接附加到该用户。

### 步骤 1.1：创建 IAM 策略

了解如何创建 IAM 策略。使用此策略的用户和角色能够与 Amazon Pinpoint API 中的所有资源交互。它还允许访问与 Amazon Pinpoint Email API 以及 Amazon Pinpoint SMS 和 Voice API 相关的资源。

#### 创建策略

1. 登录 AWS Management Console 并打开 IAM 控制台，[网址为 https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/)。
2. 在导航窗格中，选择 策略，然后选择 创建策略。
3. 在 JSON 选项卡上，粘贴以下代码。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:Update*",
        "mobiletargeting:Get*",
        "mobiletargeting:Send*",
        "mobiletargeting:Put*",
        "mobiletargeting:Create*"
      ],
      "Resource": [
        "arn:aws:mobiletargeting:*:123456789012:apps/*",
        "arn:aws:mobiletargeting:*:123456789012:apps/*/campaigns/*",
        "arn:aws:mobiletargeting:*:123456789012:apps/*/segments/*"
      ]
    }
  ]
}
```



```
    },
    {
      "Sid": "VisualEditor1",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:TagResource",
        "mobiletargeting:PhoneNumberValidate",
        "mobiletargeting:ListTagsForResource",
        "mobiletargeting>CreateApp"
      ],
      "Resource": "arn:aws:mobiletargeting:*:123456789012:*"
    },
    {
      "Sid": "VisualEditor2",
      "Effect": "Allow",
      "Action": [
        "ses:TagResource",
        "ses:Send*",
        "ses:Create*",
        "ses:Get*",
        "ses:List*",
        "ses:Put*",
        "ses:Update*",
        "sms-voice:SendVoiceMessage",
        "sms-voice:List*",
        "sms-voice:Create*",
        "sms-voice:Get*",
        "sms-voice:Update*"
      ],
      "Resource": "*"
    }
  ]
}
```

在前面的示例中，将 `123456789012` 替换为您的 AWS 账户 ID。

你的 AWS 账户 ID 可以在控制台的右上角找到，或者你可以使用 CLI，参见[查找你的 AWS 账户 ID](#)。

#### Note

为保护您的 Amazon Pinpoint 账户中的数据，此策略仅包含允许您读取、创建和修改资源的权限，不包含删除资源的权限。您可以在 IAM 控制台中使用可视化编辑器来修改此策

略。有关更多信息，请参阅《IAM 用户指南》中的[管理 IAM 策略](#)。您也可以使用 IAM API 中的[CreatePolicyVersion](#)操作来更新此策略。

另请注意，除了 mobiletargeting 服务外，此策略还包含允许您与 ses 和 sms-voice 服务交互的权限。ses 和 sms-voice 权限允许您分别与 Amazon Pinpoint Email API 以及 Amazon Pinpoint SMS 和 Voice API 进行交互。mobiletargeting 权限允许您与 Amazon Pinpoint API 交互。

选择下一步：标签。

4. （可选）您可以通过选择添加标签将标签添加到策略。
5. 选择下一步：审核。
6. 对于名称，输入策略的名称，如 **PostmanAccessPolicy**。选择创建策略。

## 步骤 1.2：创建 IAM 用户

创建策略后，您可以创建用户并将策略附加到该用户。当您创建用户时，IAM 会提供一组凭证，以允许 Postman 执行 Amazon Pinpoint API 操作。

### 创建用户

1. 通过 <https://console.aws.amazon.com/iam/> 打开 IAM 控制台。
2. 在 IAM 控制台的导航窗格中，选择用户，然后选择添加用户。
3. 在用户详细信息下，对于用户名，输入用于标识用户的名称，如 **PostmanUser**。然后选择下一步。
4. 在设置权限下，对于权限选项，选择直接附加策略。
5. 在权限策略下，选择您在[步骤 1.1](#)中创建的策略 (**PostmanAccessPolicy**)。然后选择下一步。
6. 在审核和创建页面上，您也可以添加有助于标识用户的标签。有关使用标签的更多信息，请参阅《IAM 用户指南》中的[标记 IAM 资源](#)。
7. 当您准备好创建用户时，请选择创建用户。

## 步骤 1.3：创建访问密钥

IAM 提供了一组凭证，您可以使用这些凭证来允许 Postman 执行 Amazon Pinpoint API 操作。

## 创建用户

1. 通过 <https://console.aws.amazon.com/iam/> 打开 IAM 控制台。
2. 在 IAM 控制台的导航窗格中，选择用户。选择在 [步骤 1.2](#) 中创建的用户 (**PostmanUser**)，然后选择安全凭证选项卡。
3. 在访问密钥部分，选择创建访问密钥。
4. 在访问密钥最佳实践管理门户备选方案页面上，选择在 AWS 之外运行的应用程序。然后选择下一步。
5. (可选) 您可以向策略中添加描述标签。
6. 选择创建访问密钥。
7. 在检索访问密钥页面上，复制访问密钥和秘密访问密钥列中显示的凭证。

### Note

在本教程的后面部分，您需要提供在此访问密钥 ID 和秘密访问密钥。这是您可以查看秘密访问密钥的唯一机会。建议您复制它并保存到安全的位置。

8. 保存两个密钥后，选择完成。

## 步骤 2：设置 Postman

现在，您已经创建了一个能够访问 Amazon Pinpoint API 的用户，接下来可以设置 Postman 了。在本部分，您将在 Postman 中创建一个或多个环境。接下来，您将导入一个集合，其中包含 Amazon Pinpoint API 中的每项操作的请求模板。

### 步骤 2.1：创建 Postman 工作区

在 Postman 中，工作区 是项目和环境的组织容器。在本节中，您将至少创建一个工作区以与 Amazon Pinpoint 配合使用。

#### 创建工作区

在 Postman 中，选择更多操作，选择文件，然后选择新建。

1. 在新建窗口中，选择工作区。
2. 输入名称和摘要，并将可见性设置为个人。选择创建工作区。

## 步骤 2.2：创建 Postman 环境

在 Postman 中，环境是一组存储为键/值对的变量。您可以使用环境来更改通过 Postman 发出的请求的配置，而无需更改 API 请求本身。

在本部分，您将创建至少一个环境以与 Amazon Pinpoint 配合使用。您创建的每个环境都包含一组变量，这些变量特定于您在单个 AWS 区域中的账户。如果您使用本部分中的过程创建多个环境，则可通过从 Postman 的环境菜单中选择不同的环境，在不同区域之间轻松切换。

### 创建环境

1. 在 Postman 中，选择更多操作菜单，选择文件，然后选择新建。
2. 在新建窗口中，选择环境。
3. 在管理环境窗口中，对于环境名称，输入 **Amazon Pinpoint - *Region Name***。将####替换为以下值之一：
  - 美国东部 (弗吉尼亚州北部)
  - 美国西部 (俄勒冈州)
  - 亚太地区 (孟买)
  - 亚太地区 (悉尼)
  - 欧洲地区 (法兰克福)
  - 欧洲地区 (爱尔兰)

#### Note

您至少需要为单个 AWS 区域创建一个环境，并且该 AWS 区域必须包含一个项目。如果您尚未在前面列出的 AWS 区域中创建项目，请参阅《Amazon Pinpoint 用户指南》中的[创建带有电子邮件支持的 Amazon Pinpoint 项目](#)。

4. 创建六个新变量：endpoint、region、serviceName、accountId、accessKey 和 secretAccessKey。使用下表确定要在每个变量的初始值和当前值列中输入哪个值。

区域	变量	初始值和当前值
美国东部 (弗吉尼亚州北部)	endpoint	<b>pinpoint.us-east-1</b> <b>.amazonaws.com</b>

区域	变量	初始值和当前值
	region	<b>us-east-1</b>
	serviceName	<b>mobiletargeting</b>
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )
美国西部 ( 俄勒冈州 )	endpoint	<b>pinpoint.us-west-2 .amazonaws.com</b>
	region	<b>us-west-2</b>
	serviceName	<b>mobiletargeting</b>
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )
亚太地区 ( 孟买 )	endpoint	<b>pinpoint.ap-south-1 .amazonaws.com</b>
	region	<b>ap-south-1</b>
	serviceName	<b>mobiletargeting</b>
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )

区域	变量	初始值和当前值
亚太地区 (悉尼)	endpoint	<b>pinpoint.ap-southeast-2.amazonaws.com</b>
	region	<b>ap-southeast-2</b>
	serviceName	<b>mobiletargeting</b>
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )
欧洲地区 (法兰克福)	endpoint	<b>pinpoint.eu-central-1.amazonaws.com</b>
	region	<b>eu-central-1</b>
	serviceName	<b>mobiletargeting</b>
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )
欧洲地区 (爱尔兰)	endpoint	<b>pinpoint.eu-west-1.amazonaws.com</b>
	region	<b>eu-west-1</b>
	serviceName	<b>mobiletargeting</b>

区域	变量	初始值和当前值
	accountId	( 您的 AWS 账户 ID )
	accessKey	( 您的 IAM 访问密钥 ID )
	secretAccessKey	( 您的 IAM 秘密访问密钥 )

创建这些变量后，管理环境窗口类似于下图所示的示例。

US East (N. Virginia)				Fork	0	Save	Share	...
	VARIABLE	TYPE ①	INITIAL VALUE ①	CURRENT VALUE ①	...	Persist All	Reset All	
<input checked="" type="checkbox"/>	endpoint	default	pinpoint.us-east-1.amazonaws.com	pinpoint.us-east-1.amazonaws.com				🗑️ ...
<input checked="" type="checkbox"/>	region	default	us-east-1	us-east-1				
<input checked="" type="checkbox"/>	serviceName	default	mobiletargeting	mobiletargeting				
<input checked="" type="checkbox"/>	accountId	default	123456789012	123456789012				
<input checked="" type="checkbox"/>	accessKey	default	AKIAIOSFODNN7EXAMPLE	AKIAIOSFODNN7EXAMPLE				
<input checked="" type="checkbox"/>	secretAccessKey	default	wJairXUtnFEMI/K7MDENG/bPxrFiCY...	wJairXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY				

完成后，选择保存。

### ⚠ Important

上图所示的访问密钥是虚构的。不要与他人共享您的 IAM 访问密钥。

Postman 包含允许您共享和导出环境的功能。如果您使用这些功能，切勿与不应访问这些凭证的任何人分享您的访问密钥 ID 和秘密访问密钥。

有关更多信息，请参阅《IAM 用户指南》中的 [IAM 最佳实践](#)。

5. ( 可选 ) 对于您要创建的每个额外环境，请重复步骤 1-4。

### 💡 Tip

在 Postman 中，您可以根据需要创建任意多个环境。您可以通过以下方式使用环境：

- 为必须进行 Amazon Pinpoint API 测试的每个区域创建一个单独的环境。
- 创建与不同的 AWS 账户账户关联的环境。
- 创建使用与其他用户关联的凭证的环境。

6. 创建完环境后，请继续完成下一部分。

## 步骤 2.3：在 Postman 中创建 Amazon Pinpoint 集合

在 Postman 中，集合是一组 API 请求。集合中的请求通常由一个共同的目的联系在一起。在本部分，您将创建一个新集合，其中包含 Amazon Pinpoint API 中的每项操作的请求模板。

### 创建 Amazon Pinpoint 集合

1. 在 Postman 中，选择更多操作菜单，选择文件，然后选择导入。
2. 在导入窗口中，选择从链接导入，然后输入以下 URL：[https://raw.githubusercontent.com/awsdocs/amazon-pinpoint-developer-guide/master/Amazon%20Pinpoint.postman\\_collection.json](https://raw.githubusercontent.com/awsdocs/amazon-pinpoint-developer-guide/master/Amazon%20Pinpoint.postman_collection.json)。

选择导入。Postman 将导入 Amazon Pinpoint 集合，其中包含 120 个示例请求。

## 步骤 2.4：测试 Postman 配置

导入 Amazon Pinpoint 集合后，建议您执行快速测试，以确认所有组件是否都配置正确。可以通过提交 GetApps 请求来测试您的配置。此请求将返回当前 AWS 区域内您的 Amazon Pinpoint 账户中存在的所有项目的列表。此请求不需要任何额外配置，因此这是一种测试配置的好办法。

### 测试 Amazon Pinpoint 集合的配置

1. 在左侧导航窗格中，选择集合，展开 Amazon Pinpoint 集合，然后展开应用程序文件夹。
2. 在请求列表中，选择 GetApps。
3. 环境选择器选择您在[步骤 2.2](#)中创建的环境。
4. 选择发送。如果请求成功发送，则响应窗格显示 200 OK 状态。您将看到一个类似于下图中的示例的响应。



Body Cookies Headers (11) Test Results Status: 200 OK Time: 152 ms Size: 1.1 KB Save Download

Pretty Raw Preview JSON ↕

```

1 - {
2 -   "Item": [
3 -     {
4 -       "Name": "SampleProject1",
5 -       "Id": "12345678901234567890123456789012",
6 -       "Arn": "arn:aws:mobiletargeting:us-west-2:123456789012:apps/12345678901234567890123456789012",
7 -       "tags": {}
8 -     },
9 -     {
10 -      "Name": "SampleProject2",
11 -      "Id": "98765432109876543210987654321098",
12 -      "Arn": "arn:aws:mobiletargeting:us-west-2:123456789012:apps/98765432109876543210987654321098",
13 -      "tags": {}
14 -     }
15 -   ]
16 - }

```

### Note

如果您没有在 AWS 区域中创建任何项目，那么 Amazon Pinpoint 就会返回 { "Item": [] }。

此响应显示您在步骤 3 中选择的区域内的账户中存在的所有 Amazon Pinpoint 项目的列表。

## 问题排查

当您提交请求时，可能会看到错误。请查看以下列表，了解可能会遇到的几个常见错误，以及可以采取的解决问题的步骤。

错误消息	问题	解析
无法得到任何响应 连接到 <code>https://%7B%7Bendpoint%7D%7D/v1/apps</code> 时出错。	<code>{{endpoint}}</code> 变量当前没有值，在您选择环境时才设置值。	使用环境选择器选择一个环境。
请求中包含的安全令牌无效。	Postman 未能找到您的访问密钥 ID 或秘密访问密钥的当前值。	选择环境选择器旁边的齿轮图标，然后选择当前环境。确认 <code>accessKey</code> 和 <code>secretAccessKey</code> 值显示在初始值和当

错误消息	问题	解析
		前值列中，并且您正确输入了凭证。
“消息”：“用户: arn:aws:iam::123456789012:user/PinpointPostmanUser 无权对资源: arn:aws:mobiletargeting:us-west-2:123456789012:* 执行: mobiletargeting:GetApps”	与您的用户关联的 IAM 策略不包含相应权限。	确认您的用户具有在 <a href="#">步骤 1.1</a> 中所述的权限，并且您在 <a href="#">步骤 2.1</a> 中创建环境时提供了正确的凭证。

### 步骤 3：发送其他请求

当您完成配置和测试 Postman 后，可以开始向 Amazon Pinpoint API 发送其他请求。本部分包括开始发送请求之前需要了解的信息。它还包括两个示例请求，描述了如何使用 Amazon Pinpoint 集合。

#### Important

当您完成本部分中的过程时，您就将请求提交给了 Amazon Pinpoint API。这些请求在您的 Amazon Pinpoint 账户中创建新的资源、修改现有资源、发送消息、更改 Amazon Pinpoint 项目配置并使用其他 Amazon Pinpoint 功能。执行这些请求时要小心。

### 关于 Amazon Pinpoint Postman 集合中的示例

对于 Amazon Pinpoint Postman 集合中的大多数操作，您必须先进行配置，然后才能使用它们。对于 GET 和 DELETE 操作，通常只需修改预请求脚本选项卡上设置的变量即可。

#### Note

当您使用[步骤 1.1](#)中显示的 IAM 策略时，您无法执行此集合中包含的任何 DELETE 请求。

例如，GetCampaign 操作要求您指定一个 projectId 和 campaignId。在预请求脚本选项卡上，这两个变量都存在，并填充了示例值。删除示例值，并将它们替换为您的 Amazon Pinpoint 项目和活动的相应值。

在这些变量中，最常用的是 `projectId` 变量。此变量的值应该是您的请求应用到的项目的唯一标识符。要获取项目的这些标识符的列表，您可以参考在本教程的前一步骤中发送的 `GetApps` 请求的响应。在该响应中，`Id` 字段提供了项目的唯一标识符。要详细了解 `GetApps` 操作和响应中每个字段的含义，请参阅《Amazon Pinpoint API 参考》中的[应用程序](#)。

#### Note

在 Amazon Pinpoint 中，“项目”与“应用程序”同义。

对于 `POST` 和 `PUT` 操作，您还需要修改请求正文，以包含要发送到 API 的值。例如，当您提交 `CreateApp` 请求（一个 `POST` 请求）时，必须为创建的项目指定一个名称。您可以在正文选项卡上修改请求。在本示例中，将 `"Name"` 旁边的值替换为项目的名称。如果您要将标签添加到项目中，可以在 `tags` 对象中指定它们。或者，如果您不想添加标签，可以删除整个 `tags` 对象。

#### Note

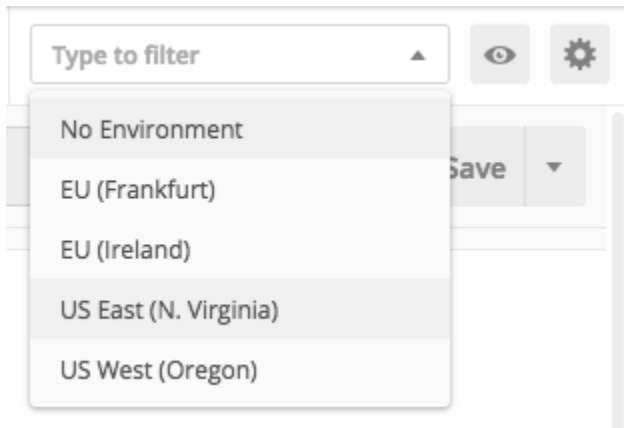
`UntagResource` 操作还要求您指定 `URL` 参数。您可以在参数选项卡上指定这些参数。将值列中的值替换为您要为指定资源删除的标签。

## 示例请求：通过使用 `CreateApp` 操作来创建项目

在 Amazon Pinpoint 中创建客户细分和活动之前，您必须先创建一个项目。在 Amazon Pinpoint 中，一个项目由出于一个共同目的而联合的客户细分、活动、配置和数据组成。例如，您可以使用一个项目来包含与特定应用程序或者与特定品牌或营销计划相关的所有内容。当您向 Amazon Pinpoint 添加客户信息时，该信息与一个项目相关联。

通过发送 `CreateApp` API 请求来创建项目

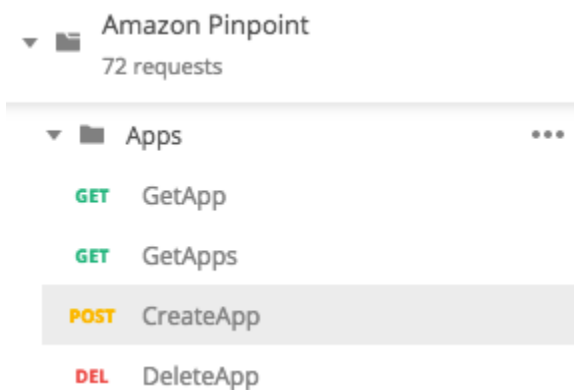
1. 在环境菜单中，选择您要在其中创建项目的 AWS 区域 区域，如下图所示。



在本示例中，Postman 已进行配置，因此环境菜单显示以下四个选项：

- 美国东部 ( 弗吉尼亚州北部 )
- 美国西部 ( 俄勒冈州 )
- 欧洲地区 ( 法兰克福 )
- 欧洲地区 ( 爱尔兰 )

2. 在应用程序文件夹中，选择 CreateApp 操作。



Amazon Pinpoint Postman 集合中的应用程序文件夹展开并显示以下请求：

- GetApp
- GetApps
- CreateApp
- DeleteApp

3. 在正文选项卡上，"Name" 的旁边，将占位符值 ("string") 替换为活动的名称，例如 **"MySampleProject"**。

4. 删除活动名称后面的逗号，然后删除第 3 至 5 行上的整个 tags 对象。完成后，您的请求应类似于以下代码片段中所示的示例。

```
{
  "Name": "MySampleProject"
}
```

Postman 配置为发送请求作为原始 JSON 负载。

5. 选择发送。如果活动成功创建，则响应窗格将显示 201 Created 状态。

```
{
  "Name": "MySampleProject"
  "Id": "12345678901234567890123456789012",
  "Arn": "arn:aws:mobiletargeting:us-
east-1:123456789012:apps/12345678901234567890123456789012",
  "tags": {}
}
```

## 示例：通过使用 **SendMessage** 操作来发送电子邮件

使用 Amazon Pinpoint **SendMessage** API 发送事务性消息非常常见。通过使用 **SendMessage** API（而不是创建活动）来发送消息的一个优势是，您可以将消息发送到任何地址，如电子邮件地址、电话号码或设备令牌。消息发往的地址不必存在于您的 Amazon Pinpoint 账户中。我们来将这种方法与通过创建活动发送消息的方法进行比较。在 Amazon Pinpoint 中发送活动之前，您必须先将端点添加到您的 Amazon Pinpoint 账户中，然后创建客户细分，创建活动，并执行活动。

本部分中的示例演示如何将事务性电子邮件直接发送到特定电子邮件地址。您可以修改此请求，以通过其他渠道（如短信、移动推送或语音）发送消息。

### 通过提交 **SendMessage** 请求来发送电子邮件

1. 确认项目已启用电子邮件渠道，并配置了要用于发送和接收邮件的电子邮件地址或域。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的[启用和禁用电子邮件渠道](#)和[验证电子邮件身份](#)。

#### Note

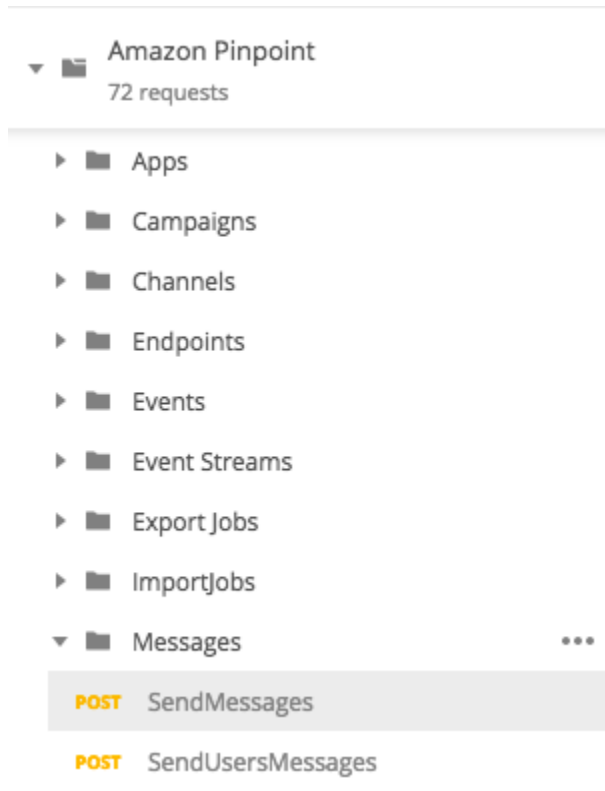
要完成本节中的过程，您必须验证电子邮件地址

2. 在环境菜单上，选择要从中发送消息的 AWS 区域 区域。

在本示例中，Postman 已进行配置，因此环境菜单显示以下四个选项：

- 美国东部 ( 弗吉尼亚州北部 )
- 美国西部 ( 俄勒冈州 )
- 欧洲地区 ( 法兰克福 )
- 欧洲地区 ( 爱尔兰 )

3. 在消息文件夹中，选择 SendMessages 操作。



4. 在预请求脚本选项卡上，将 `projectId` 变量的值替换为您在本部分步骤 2 中选择的区域内已存在的项目的 ID。

5. 在正文选项卡上，删除请求编辑器中显示的示例请求。粘贴以下代码：

```
{
  "MessageConfiguration":{
    "EmailMessage":{
      "FromAddress":"sender@example.com",
      "SimpleEmail":{
        "Subject":{
          "Data":"Sample Amazon Pinpoint message"
        },

```

```

        "HtmlPart":{
            "Data":"<h1>Test message</h1><p>This is a sample message sent
from <a href=\"https://aws.amazon.com/pinpoint\">Amazon Pinpoint</a> using the
SendMessages API.</p>"
        },
        "TextPart":{
            "Data":"This is a sample message sent from Amazon Pinpoint
using the SendMessages API."
        }
    }
},
"Addresses":{
    "recipient@example.com": {
        "ChannelType": "EMAIL"
    }
}
}

```

6. 在上述代码中，将 *sender@example.com* 替换为验证的电子邮件地址。将 *recipient@example.com* 替换为您要将消息发送到的电子邮件地址。

#### Note

如果您的账户仍在 Amazon Pinpoint 电子邮件沙盒中，则您只能将电子邮件发送到在您的 Amazon Pinpoint 账户中验证过的地址或域。有关将您的账户从沙盒移除的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[请求电子邮件的生产访问权限](#)。

7. 选择发送。如果消息成功发送，则响应窗格显示 200 OK 状态。

```

{
  "ApplicationId": "12345678901234567890123456789012",
  "RequestId": "<sampleValue>",
  "Result": {
    "recipient@example.com": {
      "DeliveryStatus": "SUCCESSFUL",
      "StatusCode": 200,
      "StatusMessage": "<sampleValue>",
      "MessageId": "<sampleValue>"
    }
  }
}

```

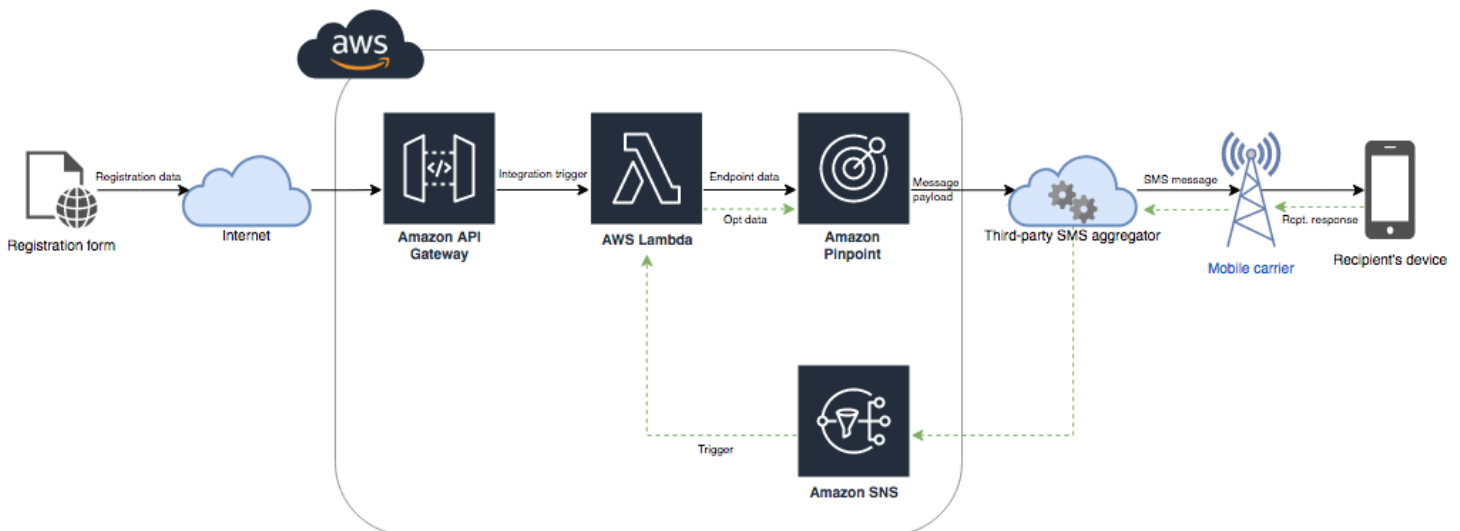
## 教程：设置短信注册系统

向客户发送时效性消息时可以使用短信（文本消息），这是一种非常不错的方式。如今，许多人时刻把手机带在身边。此外，短信消息往往比推送通知、电子邮件或电话更能吸引人们的注意力。

捕获客户手机号码的常用方法是使用基于 Web 的表单。验证完客户的电话号码并确认其订阅后，您可以开始向该客户发送营销性、事务性和信息性短信。

本教程介绍如何设置 Web 表单来捕获客户的联系信息。Web 表单向 Amazon Pinpoint 发送此信息。接下来，Amazon Pinpoint 会验证电话号码是否有效，并捕获与电话号码相关的其他元数据。之后，Amazon Pinpoint 会向客户发送一条消息，要求他们确认订阅。客户确认订阅后，Amazon Pinpoint 会选择让客户接收您的消息。

下面的架构示意图显示了此解决方案中的数据流。



### 关于双重选择加入

本教程介绍如何在 Amazon Pinpoint 中设置使用双向短信消息的双重选择加入系统。

在短信双重选择加入系统中，客户通过在 Web 表单或您的应用程序中提交电话号码来向您提供其电话号码。当您收到客户的请求时，您将在 Amazon Pinpoint 中创建一个新的端点。新的端点应选择退出您的通信。接下来，您向该电话号码发送消息。在消息中，您要求接收者通过回复特定的字词（如“是”或“确认”）来确认其订阅。如果客户用您指定的字词响应该消息，则将端点的状态更改为选择加入。否则，如果客户不响应，或者用不同的字词响应，您可以将端点的状态保留为选择退出。

### 关于此解决方案

本部分包含有关您在本教程中构建的解决方案的信息。



## 目标受众

本教程以开发人员和系统实施者为受众。虽然您不必非得熟悉 Amazon Pinpoint 才能完成本教程中的步骤，但是，您应该熟悉管理 IAM 策略、在 Node.js 中创建 Lambda 函数以及部署 Web 内容。

## 使用的功能

本教程包含以下 Amazon Pinpoint 功能的使用示例：

- 发送事务性短信
- 使用电话号码验证获取有关电话号码的信息
- 使用双向短信接收传入的短信
- 创建动态客户细分
- 创建活动
- 使用以下方式与亚马逊 Pinpoint API 互动 AWS Lambda

## 所需时间

完成本教程大概需要一小时的时间。实施此解决方案后，您可以执行其他步骤来细化解决方案，以适合您的独特使用案例。

## 区域限制

本教程要求您通过使用 Amazon Pinpoint 控制台来租赁长代码。您可以使用 Amazon Pinpoint 控制台来租赁基于几个国家/地区的专用长代码。但是，只有基于加拿大的长代码可用于发送短信。（您可以使用基于其他国家和地区的长代码来发送语音消息。）

我们在本教程中开发代码示例时考虑到了这一限制。例如，代码示例假定接收者的电话号码始终有 10 位数，以及国家/地区代码 1。如果您在美国或加拿大以外的国家或地区实施了此解决方案，则必须相应地修改代码示例。

## 资源用量费用

创建 AWS 账户不收取任何费用。但是，通过实施此解决方案，可能会产生以下费用：

- 长代码租赁费用 – 为完成本教程，您需要租赁长代码。加拿大的长代码费用为每月 1.00 美元。
- 电话号码验证使用费用 – 本教程中的解决方案使用 Amazon Pinpoint 的电话号码验证功能来验证您收到的每个号码是否有效以及格式是否正确，同时还会获取关于电话号码的其他信息。对于每个电话号码验证请求，您需要支付 0.006 美元。

- 消息发送费用 – 本教程中的解决方案会发送出站短信。您需要为通过 Amazon Pinpoint 发送的每条消息付费。您为每条消息支付的费用取决于接收者所在的国家或地区。如果您向位于美国 ( 不包括美国领地 ) 的接收者发送消息，需为每条消息支付 0.00645 美元。如果您向位于加拿大的接收者发送消息，则需支付 0.00109–0.02 美元，具体取决于接收者的运营商和位置。
- 消息接收费用 – 此解决方案还接收并处理传入的短信。您需要为发送到您的 Amazon Pinpoint 账户关联电话号码的每条传入消息付费。支付的费用取决于在哪里接收电话号码。如果您是在美国 ( 不包括美国领地 ) 接收号码，则需为每条传入消息支付 0.0075 美元。如果是在加拿大接收号码，则为每条传入消息支付 0.00155 美元。
- Lambda 使用费用 – 此解决方案使用两个与 Amazon Pinpoint API 交互的 Lambda 函数。当您调用 Lambda 函数时，系统将根据对函数发出的请求的数量、代码执行所需的时间以及函数使用的内存量来收取费用。本教程中的函数使用非常少的内存，并且通常运行 1-3 秒。此解决方案的部分或所有用量可能纳入 Lambda 免费使用套餐。有关更多信息，请参阅 [Lambda 定价](#)。
- API Gateway 使用费用 – 此解决方案中的 Web 表调用由 API Gateway 管理的 API。对于每百万次调用 API Gateway，您需要支付 3.50 至 3.70 美元，具体取决于您在哪个地区 AWS 使用 Amazon Pinpoint。有关更多信息，请参阅 [API Gateway 定价](#)。
- Web 托管费用 – 此解决方案包含一个基于 Web 的表单，您必须在您的网站上托管该表单。托管此内容的费用取决于您的 Web 托管提供商。

#### Note

此列表中显示的所有价格均以美元 (USD) 为单位。

下一步：[先决条件](#)

## 先决条件

开始本教程之前，您必须先完成以下先决条件：

- 您必须有一个 AWS 账户。要创建 AWS 账户，请前往 <https://console.aws.amazon.com/> 并选择创建新的 AWS 账户。
- 您用于登录 AWS Management Console 的账户必须能够执行以下任务：
  - 创建新的 IAM 策略和角色
  - 创建新的 Amazon Pinpoint 项目
  - 创建新的 Lambda 函数

- 在 API Gateway 中创建新的 API
- 您必须有一种托管网页的方法，并且您应了解如何发布网页。尽管可以使用 AWS 服务来托管您的网页，但并不强求。

 Tip

要了解有关使用 AWS 服务托管网页的更多信息，请参阅[托管静态网页](#)。

下一步：[设置 Amazon Pinpoint](#)

## 步骤 1：设置 Amazon Pinpoint

实施此解决方案的第一步是设置 Amazon Pinpoint。请在此部分中执行以下操作：

- 创建 Amazon Pinpoint 项目
- 启用短信渠道并租用电话号码
- 配置双向短信

开始本教程之前，您应阅读[先决条件](#)。

### 步骤 1.1：创建 Amazon Pinpoint 项目

首先，您需要创建一个 Amazon Pinpoint 项目。在 Amazon Pinpoint 中，一个项目由出于一个共同目的而联合的客户细分、活动、配置和数据组成。例如，您可以使用一个项目来包含与特定应用程序或者与特定品牌或营销计划相关的所有内容。当您向 Amazon Pinpoint 添加客户信息时，该信息与一个项目相关联。

根据您之前是否已在 Amazon Pinpoint 中创建了项目，创建新项目所涉及的步骤将有所不同。

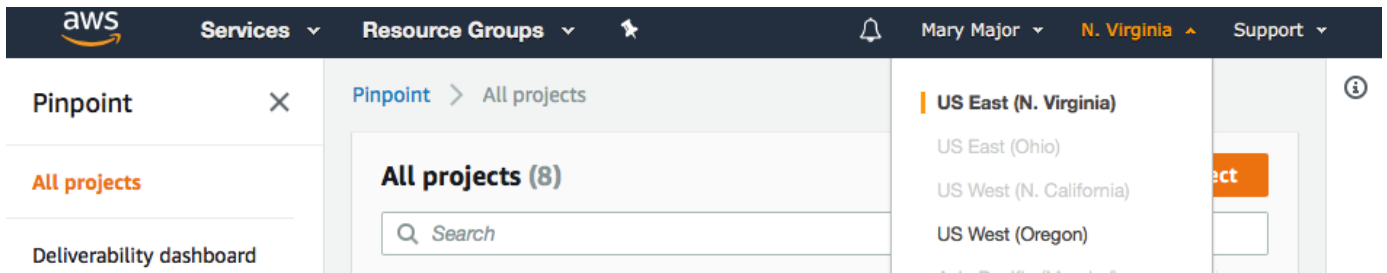
创建项目（对于 Amazon Pinpoint 新用户）

这些步骤描述了如果您从未在当前 AWS 地区创建过项目，则创建新 Amazon Pinpoint 项目的过程。

创建项目

1. [登录 AWS Management Console 并打开亚马逊 Pinpoint 控制台，网址为 https://console.aws.amazon.com/pinpoint/。](https://console.aws.amazon.com/pinpoint/)

2. 使用区域选择器选择要使用的 AWS 区域，如下图所示。如果您不确定，请选择位置离您最近的区域。



3. 在开始使用下，对于名称，输入活动的名称（如 **SMSRegistration**），然后选择创建项目。
4. 在配置功能页面上，选择跳过此步骤。
5. 在导航窗格中，选择所有项目。
6. 在所有项目页面上，在您刚刚创建的项目旁边，复制项目 ID 列中显示的值。

**i** Tip

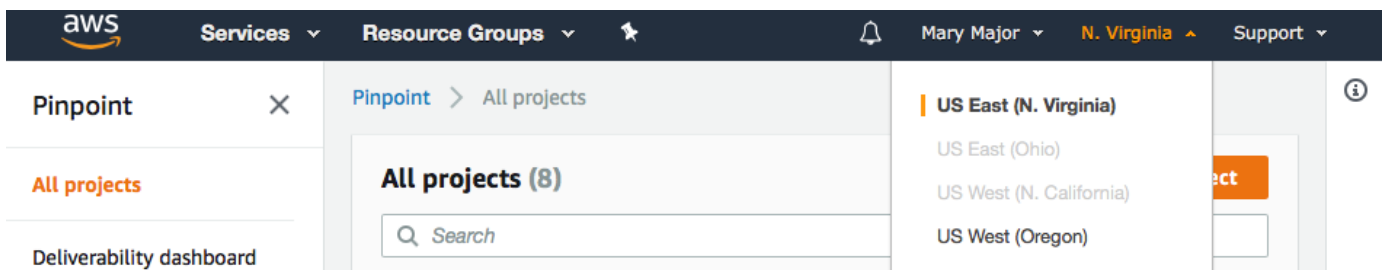
在本教程中，您需要在几个不同的地方使用此 ID。将项目 ID 保存在一个方便的位置，以便稍后复制。

### 创建项目（对于 Amazon Pinpoint 现有用户）

如果您已经在当前 AWS 地区创建了项目，则这些步骤描述了创建新 Amazon Pinpoint 项目的过程。

#### 创建项目

1. [登录 AWS Management Console 并打开亚马逊 Pinpoint 控制台，网址为 https://console.aws.amazon.com/pinpoint/。](https://console.aws.amazon.com/pinpoint/)
2. 使用区域选择器选择要使用的 AWS 区域，如下图所示。如果您不确定，请选择位置离您最近的区域。



3. 在所有项目页面上，选择创建项目。

4. 在创建项目窗口上，对于项目名称，输入项目的名称（例如 **SMSRegistration**）。选择创建。
5. 在配置功能页面上，选择跳过此步骤。
6. 在导航窗格中，选择所有项目。
7. 在所有项目页面上，在您刚刚创建的项目旁边，复制项目 ID 列中显示的值。

**i** Tip

在本教程中，您需要在几个不同的地方使用此 ID。将项目 ID 保存在一个方便的位置，以便稍后复制。

## 步骤 1.2：获取专用电话号码

**i** Note

Amazon Pinpoint 已更新其用户指南文档。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

创建项目后，您可以开始在该项目中配置功能。在本节中，您将启用短信渠道，并获取专用的长代码，以便在发送短信时使用。

**i** Note

本节假设您在注册品牌和活动之后租赁的是美国 10DLC 电话号码、美国免费电话号码或加拿大长代码。如果您执行本部分中的过程，但选择美国或加拿大以外的国家/地区，您将无法使用该号码发送短信。要了解有关在美国或加拿大以外的国家/地区租赁支持短信的长代码的更多信息，请参阅《Amazon Pinpoint SMS 用户指南》中的 [支持的国家/地区和区域（短信渠道）](#)。

要使用 Amazon Pinpoint 控制台启用短信渠道，请按照以下步骤操作：

1. [登录 AWS Management Console 并打开亚马逊 Pinpoint 控制台](https://console.aws.amazon.com/pinpoint/)，网址为 <https://console.aws.amazon.com/pinpoint/>。
2. 在导航窗格中的 Settings (设置) 下，选择 SMS and voice (SMS 和语音)。
3. 选择 SMS settings (SMS 设置) 旁边的 Edit (编辑)。


4. 在常规设置下，选择为此项目启用短信渠道，然后选择保存更改。

要使用 Amazon Pinpoint SMS 控制台申请电话号码，请按照以下步骤操作：

申请电话号码（控制台）

1. 打开 Amazon Pinpoint SMS 控制台，网址为：<https://console.aws.amazon.com/sms-voice/>。
2. 在导航窗格的配置下，选择电话号码，然后选择申请发起方。
3. 在消息目标国家/地区的选择国家/地区页面上，选择美国或加拿大。选择下一步。
4. 在消息收发使用案例部分，输入以下内容：

- 在号码功能下，选择短信。


 Important

一旦购买了电话号码，就无法更改短信和语音的功能。

- 对于双向消息收发，选择是。
5. 选择下一步。
  6. 在选择发起方类型下，选择长代码或 10DLC。

如果您选择 10DLC 并且已经有注册的市场活动，则可以从关联到已注册市场活动中选择该市场活动。

7. 选择下一步。
8. 在查看和请求中，您可以在提交请求之前对其进行验证和编辑。选择请求。
9. 根据您的申请的电话号码类型，可能会出现需要注册窗口。您的电话号码已与此注册关联，在注册获得批准之前，您无法发送消息。有关注册要求的更多信息，请参阅[注册](#)。
  - a. 对于注册表单名称，请输入友好名称。
  - b. 选择开始注册以完成电话号码注册，或者选择稍后注册。

 Important

在您的注册获得批准之前，您的电话号码无法发送消息。

无论注册状态如何，您仍然需要支付该电话号码的每月定期租赁费。有关注册要求的更多信息，请参阅[注册](#)。

## 步骤 1.3 : 启用双向 SMS

现在您已经有了一个专用电话号码，可以设置双向短信了。启用双向短信使您的客户能够回复您发送给他们的短信。在此解决方案中，您可以使用双向短信为您的客户提供一种方法，来确认他们想要订阅您的短信计划。

### Note

Amazon Pinpoint 已更新其用户指南文档。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

要使用 Amazon Pinpoint SMS 控制台启用双向短信，请执行以下步骤：

### 启用双向 SMS

1. 打开 Amazon Pinpoint SMS 控制台，网址为：<https://console.aws.amazon.com/sms-voice/>。
2. 在导航窗格的配置下，选择电话号码。
3. 在电话号码页面上，选择所需电话号码。
4. 在双向短信选项卡上，选择编辑设置按钮。
5. 在编辑设置页面上，选择启用双向消息。
6. 对于目标类型，选择 Amazon SNS。
  - 新 Amazon SNS 主题 – Amazon Pinpoint SMS 在您的账户中创建一个主题。该主题将自动创建，并具有所有必需的权限。有关 Amazon SNS 主题的更多信息，请参阅《Amazon SNS 开发人员指南》中的[配置 Amazon SNS](#)。
  - 对于传入消息目的地，输入主题名称，例如 **SMSRegistrationFormTopic**。
7. 对于双向渠道角色，请选择使用 SNS 主题策略。
8. 选择保存更改。

使用 Amazon Pinpoint SMS 控制台向您的电话号码添加关键字，以便客户发送给您以确认其订阅（例如 **Yes** 或 **Confirm**）。

### 添加关键字

1. 打开 Amazon Pinpoint SMS 控制台，网址为：<https://console.aws.amazon.com/sms-voice/>。

2. 在导航窗格的配置下，选择电话号码。
3. 在电话号码页面上，选择要向其中添加关键字的电话号码。
4. 在关键词选项卡上，选择添加关键字按钮。
5. 在自定义关键字窗格中添加以下内容：
  - 关键字 - 要添加的新关键字（例如 **Yes** 或 **Confirm**）。
  - 回复消息 - 要发回给接收人的消息。
  - 关键字操作 - 收到关键字时要执行的操作。选择自动回复。
6. 选择添加关键字。

下一步：[创建 IAM 策略和角色](#)

## 步骤 2：创建 IAM 策略和角色

实施 SMS 注册解决方案的下一步是在 AWS Identity and Access Management (IAM) 中配置策略和角色。对于此解决方案，您需要创建一个策略，该策略提供对与 Amazon Pinpoint 相关的某些资源的访问权限。然后，创建一个角色并将策略附加到该角色。在本教程的后面部分，您将创建一个 AWS Lambda 函数，该函数使用此角色在 Amazon Pinpoint API 中调用某些操作。

### 步骤 2.1：创建 IAM 策略

此部分将向您介绍如何创建 IAM 策略。使用此策略的用户和角色可以执行以下操作：

- 使用电话号码验证功能
- 查看、创建和更新 Amazon Pinpoint 端点
- 向 Amazon Pinpoint 端点发送消息

在本教程中，您需要使 Lambda 能够执行这些任务。但是，为了提高安全性，此策略使用授予最低权限的原则。也就是说，它只授予完成此解决方案所需的权限，而不授予任何其他权限。此策略受以下几方面的限制：

- 您只能用它来调用特定区域内的电话号码验证 API。
- 您只能用它来查看、创建或更新与特定 Amazon Pinpoint 项目关联的端点。
- 您只能用它将消息发送到与特定 Amazon Pinpoint 项目关联的端点。



## 创建策略

1. 登录 AWS Management Console 并打开 IAM 控制台，[网址为 https://console.aws.amazon.com/iam/](https://console.aws.amazon.com/iam/)。
2. 在导航窗格中，选择 策略，然后选择 创建策略。
3. 在 JSON 选项卡上，粘贴以下代码。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:CreateLogGroup"
      ],
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": "mobiletargeting:SendMessages",
      "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:GetEndpoint",
        "mobiletargeting:UpdateEndpoint",
        "mobiletargeting:PutEvents"
      ],
      "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/endpoints/*"
    },
    {
      "Effect": "Allow",
      "Action": "mobiletargeting:PhoneNumberValidate",
      "Resource": "arn:aws:mobiletargeting:region:accountId:phone/number/validate"
    }
  ]
}
```


在上述示例中，执行以下操作：

- 将##替换为您使用 Amazon Pinpoint 的 AWS 区域，例如us-east-1或。eu-central-1

 Tip

有关提供 Amazon Pinpoint 的 AWS 地区的完整列表，请参阅中的[AWS 区域和终端节点](#)。AWS 一般参考

- 将## ID 替换为账户的唯一 ID。AWS
- 将 *projectId* 替换为您在本教程[步骤 1.1](#) 中创建的项目的唯一 ID。

 Note


这些logs操作使 Lambda 能够将其输出记录在日志中。 CloudWatch

4. 选择查看策略。
5. 对于名称，输入策略的名称，如 **RegistrationFormPolicy**。选择创建策略。

## 步骤 2.2：创建 IAM 角色

### 创建角色

1. 通过 <https://console.aws.amazon.com/iam/> 打开 IAM 控制台。
2. 在 IAM 控制台的导航窗格中，选择角色，然后选择创建角色。
3. 在选择将使用此角色的服务下，选择 Lambda，然后选择下一步：权限。

 Note

您在此步骤中选择什么服务不重要，无论如何选择，都在下一步中应用您的策略。

4. 在附加权限策略下，选择您在上一部分中创建的策略，然后选择下一步：标签。
5. 选择 下一步: 审核。
6. 在审核下，对于名称，输入角色的名称，如 **SMSRegistrationForm**。选择创建角色。

下一步：[创建 Lambda 函数](#)

## 步骤 3：创建 Lambda 函数

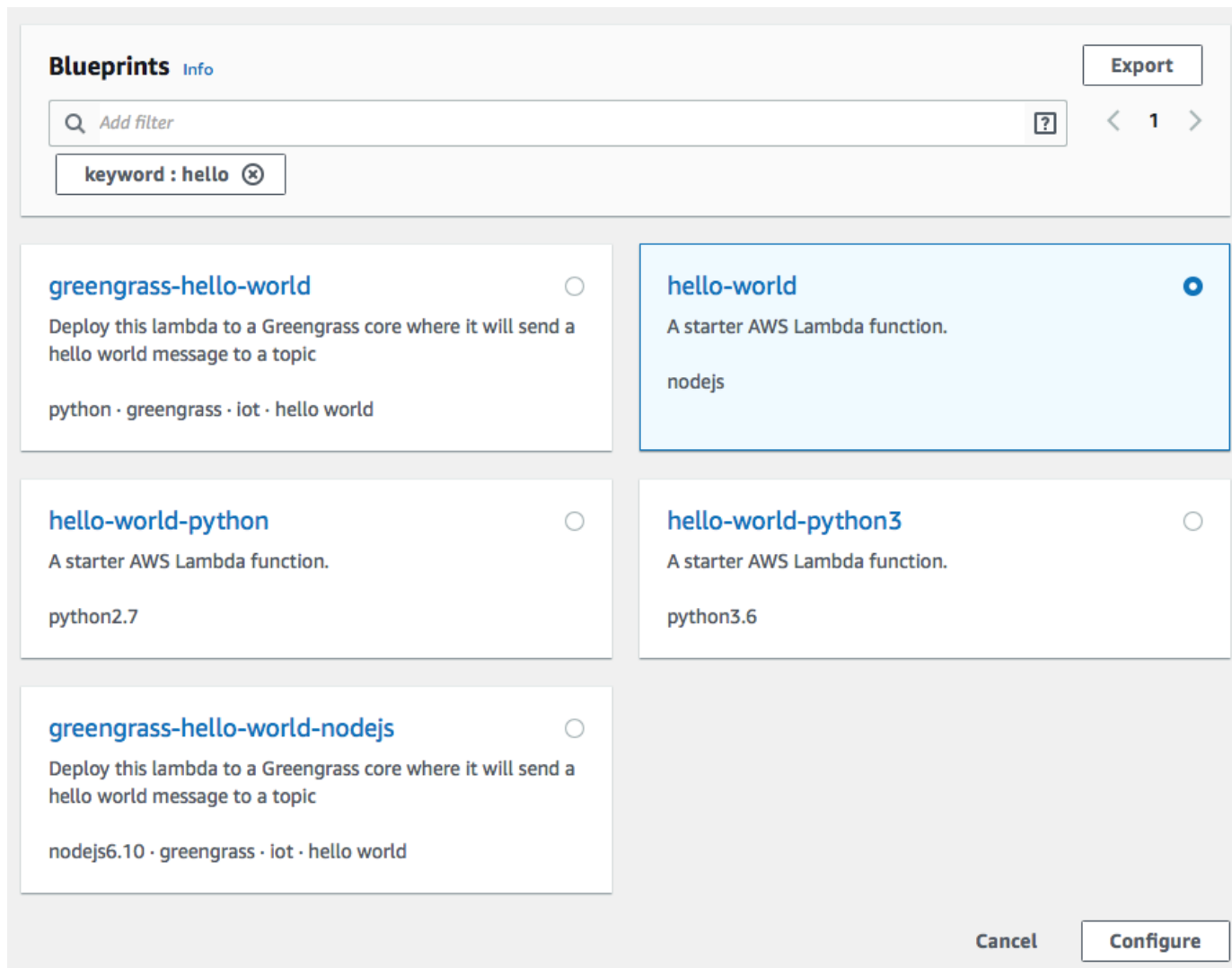
此解决方案使用两个 Lambda 函数。本部分将向您介绍如何创建和配置这些函数。稍后，您将设置 API Gateway 和 Amazon Pinpoint 以在发生特定事件时执行这些函数。这两个函数会在您指定的 Amazon Pinpoint 项目中创建和更新端点。第一个函数还使用电话号码验证功能。

### 步骤 3.1：创建可验证客户信息并可创建端点的函数

第一个函数从您的注册表单（从 Amazon API Gateway 接收）中获取输入。它使用这些信息，通过使用 Amazon Pinpoint 的[电话号码验证](#)功能来获取有关客户电话号码的信息。然后，该函数使用验证的数据在您指定的 Amazon Pinpoint 项目中创建一个新的端点。默认情况下，该函数创建的端点会选择不接收您的未来通信，但此状态可通过第二个函数进行更改。最后，此函数向客户发送一条消息，要求他们验证是否希望接收来自您的短信通信。

#### 创建 Lambda 函数

1. 打开 AWS Lambda 控制台，[网址为 https://console.aws.amazon.com/lambda/](https://console.aws.amazon.com/lambda/)。
2. 选择创建函数。
3. 在创建函数下，选择蓝图。
4. 在搜索字段中，输入 **hello**，然后按 Enter。在结果列表中，选择 hello-world Node.js 函数，如下图所示。选择 Configure（配置）。



5. 在 Basic information ( 基本信息 ) 中，执行以下操作：

- 对于 Name (名称)，输入函数的名称，如 **RegistrationForm**。
- 对于角色，选择选择现有角色。
- 对于现有角色，请选择您在[步骤 2.2](#) 中创建的 SMS RegistrationForm 角色。

完成后，选择 Create function (创建函数)。

6. 删除代码编辑器中的示例函数，然后粘贴以下代码：

```
var AWS = require('aws-sdk');
var pinpoint = new AWS.Pinpoint({region: process.env.region});

// Make sure the SMS channel is enabled for the projectId that you specify.
```

```
// See: https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
setup.html
var projectId = process.env.projectId;

// You need a dedicated long code in order to use two-way SMS.
// See: https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-voice-
manage.html#channels-voice-manage-request-phone-numbers
var originationNumber = process.env.originationNumber;

// This message is spread across multiple lines for improved readability.
var message = "ExampleCorp: Reply YES to confirm your subscription. 2 msgs per "
    + "month. No purchase req'd. Msg&data rates may apply. Terms: "
    + "example.com/terms-sms";

var messageType = "TRANSACTIONAL";

exports.handler = (event, context, callback) => {
    console.log('Received event:', event);
    validateNumber(event);
};

function validateNumber (event) {
    var destinationNumber = event.destinationNumber;
    if (destinationNumber.length == 10) {
        destinationNumber = "+1" + destinationNumber;
    }
    var params = {
        NumberValidateRequest: {
            IsoCountryCode: 'US',
            PhoneNumber: destinationNumber
        }
    };
    pinpoint.phoneNumberValidate(params, function(err, data) {
        if (err) {
            console.log(err, err.stack);
        }
        else {
            console.log(data);
            //return data;
            if (data['NumberValidateResponse']['PhoneTypeCode'] == 0) {
                createEndpoint(data, event.firstName, event.lastName, event.source);
            } else {
                console.log("Received a phone number that isn't capable of receiving "
                    + "SMS messages. No endpoint created.");
            }
        }
    });
}
```

```
    }
  }
});
}

function createEndpoint(data, firstName, lastName, source) {
  var destinationNumber = data['NumberValidateResponse']
['CleansedPhoneNumberE164'];
  var endpointId = data['NumberValidateResponse']
['CleansedPhoneNumberE164'].substring(1);

  var params = {
    ApplicationId: projectId,
    // The Endpoint ID is equal to the cleansed phone number minus the leading
    // plus sign. This makes it easier to easily update the endpoint later.
    EndpointId: endpointId,
    EndpointRequest: {
      ChannelType: 'SMS',
      Address: destinationNumber,
      // OptOut is set to ALL (that is, endpoint is opted out of all messages)
      // because the recipient hasn't confirmed their subscription at this
      // point. When they confirm, a different Lambda function changes this
      // value to NONE (not opted out).
      OptOut: 'ALL',
      Location: {
        PostalCode: data['NumberValidateResponse']['ZipCode'],
        City: data['NumberValidateResponse']['City'],
        Country: data['NumberValidateResponse']['CountryCodeIso2'],
      },
      Demographic: {
        Timezone: data['NumberValidateResponse']['Timezone']
      },
      Attributes: {
        Source: [
          source
        ]
      },
      User: {
        UserAttributes: {
          FirstName: [
            firstName
          ],
          LastName: [
            lastName
          ]
        }
      }
    }
  }
}
```

```
    ]
  }
}
};
pinpoint.updateEndpoint(params, function(err,data) {
  if (err) {
    console.log(err, err.stack);
  }
  else {
    console.log(data);
    //return data;
    sendConfirmation(destinationNumber);
  }
});
}

function sendConfirmation(destinationNumber) {
  var params = {
    ApplicationId: projectId,
    MessageRequest: {
      Addresses: {
        [destinationNumber]: {
          ChannelType: 'SMS'
        }
      },
      MessageConfiguration: {
        SMSMessage: {
          Body: message,
          MessageType: messageType,
          OriginationNumber: originationNumber
        }
      }
    }
  };

  pinpoint.sendMessage(params, function(err, data) {
    // If something goes wrong, print an error message.
    if(err) {
      console.log(err.message);
    }
    // Otherwise, show the unique ID for the message.
  } else {
    console.log("Message sent! "
      + data['MessageResponse']['Result'][destinationNumber]['StatusMessage']);
  }
});
}
```

```
    }  
  });  
}
```

7. 在 Environment variables (环境变量) 下，执行以下操作：

- 在第一行创建一个包含 **originationNumber** 键值的变量。接下来，将值设置为您在 [步骤 1.2](#) 中收到的专用长代码的电话号码。

**Note**

请务必包含加号 (+) 和电话号码的国家/地区代码。请勿包含任何其他特殊字符，如连字符 (-)、句点 (.) 或圆括号。

- 在第二行创建一个包含 **projectId** 键值的变量。接下来，将值设置为您在 [步骤 1.1](#) 中创建的项目的唯一 ID。
- 在第三行创建一个包含 **region** 键值的变量。接下来，将值设置为您使用 Amazon Pinpoint 所在的区域，如 **us-east-1** 或 **us-west-2**。

完成后，Environment Variables (环境变量) 部分应类似于下图中的示例。

### Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. [Learn more.](#)

originationNumber	+12065550199	Remove
projectId	33d643d9bexample9a5e726f6c4	Remove
region	us-east-1	Remove
Key	Value	Remove

► Encryption configuration

8. 在页面顶部，选择 Save (保存)。



### 步骤 3.1.1：测试函数

创建函数后，您应对其进行测试，以确保其配置正确。此外还要确保您创建的 IAM 角色具有适当权限。

#### 测试此函数

1. 选择测试。
2. 在 Configure test event (配置测试事件) 窗口上，执行以下操作：
  - 选择 Create new test event (新建测试事件)。
  - 对于事件名称，输入测试事件的名称，如 **MyPhoneNumber**。
  - 擦除代码编辑器中的示例代码。粘贴以下代码：

```
{
  "destinationNumber": "2065550142",
  "firstName": "Carlos",
  "lastName": "Salazar",
  "source": "Registration form test"
}
```

- 在上述代码示例中，将 destinationNumber、firstName 和 lastName 属性值替换为您想用于测试的值，例如您的个人联系详细信息。当您测试此函数时，它会向您 destinationNumber 属性中指定的电话号码发送短信。请确保您指定的电话号码能够接收短信。
  - 选择创建。
3. 再次选择 Test (测试)。
  4. 在 Execution result: succeeded (执行结果：成功) 下，选择 Details (详细信息)。在 Log output (日志输出) 部分中，查看函数的输出。确保该函数运行未出现错误。

检查与您指定的 destinationNumber 关联的设备，确保它收到测试消息。

5. 打开 Amazon Pinpoint 控制台，网址为：<https://console.aws.amazon.com/pinpoint/>。
6. 在 All projects (所有项目) 页面上，选择您在 [步骤 1.1](#) 中创建的项目。
7. 在导航窗格中，选择客户细分。在客户细分页面上，选择创建客户细分。
8. 在 Segment group 1 (分段组 1) 的 Add filters to refine your segment (添加筛选条件以细化分段) 下，选择 Filter by user (按用户筛选)。

- 在“选择用户属性”中，选择FirstName。然后，对于 Choose values (选择值)，选择您在测试事件中指定的名字。

客户细分估算部分应显示有零个合格端点，以及总共一个端点，如下图所示。这是预期结果。当函数创建一个新的端点时，该端点将选择退出。Amazon Pinpoint 中的客户细分会自动排除已选择退出的端点。

The screenshot displays the Amazon Pinpoint console interface for configuring a segment group. On the left, the 'Segment group 1' configuration is shown, including a description, a dropdown for 'Include endpoints that are in' (set to 'any'), and a dropdown for 'of the following segments' (set to 'All segments'). Below this, a filter configuration for 'Filter 1: User' is visible, showing 'FirstName' is 'Carlos'. On the right, the 'Segment estimate' panel shows '0 endpoints' and a red warning box stating 'No matches found. Your segment didn't produce any results. Remove or modify your segment filters until the segment contains at least one member.'

### 步骤 3.2：创建用于选择让客户接收您的通信的函数

仅当客户回复第一个函数发送的消息时才会执行第二个函数。如果客户的回复包含步骤 1.3 中指定的关键字，则该函数将更新其端点记录，选择让客户接收未来通信。Amazon Pinpoint 还会自动使用您在步骤 1.3 中指定的消息进行回复。

如果客户没有响应，或者使用指定关键字以外的任何内容进行响应，则什么都不会发生。客户的端点虽然仍在 Amazon Pinpoint 中，但不能被客户细分当作目标。

#### 创建 Lambda 函数

- 打开 AWS Lambda 控制台，[网址为 https://console.aws.amazon.com/lambda/](https://console.aws.amazon.com/lambda/)。
- 选择创建函数。
- 在创建函数下，选择蓝图。
- 在搜索字段中，输入 **hello**，然后按 Enter。在结果列表中，选择 hello-world Node.js 函数，如下图所示。选择 Configure (配置)。
- 在 Basic information (基本信息) 中，执行以下操作：

- 对于 Name (名称)，输入函数的名称，如 **RegistrationForm\_OptIn**。
- 对于角色，选择选择现有角色。
- 对于现有角色，请选择您在[步骤 2.2](#) 中创建的 SMS RegistrationForm 角色。

完成后，选择 Create function (创建函数)。

6. 删除代码编辑器中的示例函数，然后粘贴以下代码：

```
var AWS = require('aws-sdk');
var projectId = process.env.projectId;
var confirmKeyword = process.env.confirmKeyword.toLowerCase();
var pinpoint = new AWS.Pinpoint({region: process.env.region});

exports.handler = (event, context) => {
  console.log('Received event:', event);
  var timestamp = event.Records[0].Sns.Timestamp;
  var message = JSON.parse(event.Records[0].Sns.Message);
  var originationNumber = message.originationNumber;
  var response = message.messageBody.toLowerCase();

  if (response.includes(confirmKeyword)) {
    updateEndpointOptIn(originationNumber, timestamp);
  }
};

function updateEndpointOptIn (originationNumber, timestamp) {
  var endpointId = originationNumber.substring(1);

  var params = {
    ApplicationId: projectId,
    EndpointId: endpointId,
    EndpointRequest: {
      Address: originationNumber,
      ChannelType: 'SMS',
      OptOut: 'NONE',
      Attributes: {
        OptInTimestamp: [
          timestamp
        ]
      }
    },
  }
};
```

```
pinpoint.updateEndpoint(params, function(err, data) {
  if (err) {
    console.log("An error occurred.\n");
    console.log(err, err.stack);
  }
  else {
    console.log("Successfully changed the opt status of endpoint ID " +
endpointId);
  }
});
}
```

7. 在 Environment variables (环境变量) 下，执行以下操作：

- 在第一行创建一个包含 **projectId** 键值的变量。接下来，将值设置为您在[步骤 1.1](#) 中创建的项目的唯一 ID。
- 在第二行创建一个包含 **region** 键值的变量。接下来，将值设置为您使用 Amazon Pinpoint 所在的区域，如 **us-east-1** 或 **us-west-2**。
- 在第三行创建一个包含 **confirmKeyword** 键值的变量。接下来，将值设置为您在[步骤 1.3](#) 中创建的确认关键字。

**Note**

关键字不区分大小写。此函数会将传入消息转换为小写字母。

完成后，Environment Variables (环境变量) 部分应类似于下图中的示例。

### Environment variables

You can define environment variables as key-value pairs that are accessible from your function code. These are useful to store configuration settings without the need to change function code. [Learn more.](#)

projectId	33d643d9example9a5e726f6c4a	Remove
confirmKeyword	Yes	Remove
region	us-east-1	Remove
Key	Value	Remove

► Encryption configuration

## 8. 在页面顶部，选择 Save (保存)。

### 步骤 3.2.1：测试函数

创建函数后，您应对其进行测试，以确保其配置正确。此外还要确保您创建的 IAM 角色具有适当权限。

#### 测试此函数

1. 选择测试。
2. 在 Configure test event (配置测试事件) 窗口上，执行以下操作：
  - a. 选择 Create new test event (新建测试事件)。
  - b. 对于事件名称，输入测试事件的名称，如 **MyResponse**。
  - c. 擦除代码编辑器中的示例代码。粘贴以下代码：

```
{
  "Records": [
    {
      "Sns": {
        "Message": "{\"originationNumber\": \"+12065550142\", \"messageBody\": \"Yes\"}",
        "Timestamp": "2019-02-20T17:47:44.147Z"
      }
    }
  ]
}
```

在上述代码示例中，将 `originationNumber` 属性的值替换为您在测试之前的 Lambda 函数时使用的电话号码。将 `messageBody` 的值替换为您在 [步骤 1.3](#) 中指定的双向短信关键字。

(可选) 您可以将 `Timestamp` 的值替换为当前日期和时间。

- d. 选择创建。
3. 再次选择 Test (测试)。
  4. 在 Execution result: succeeded (执行结果：成功) 下，选择 Details (详细信息)。在 Log output (日志输出) 部分中，查看函数的输出。确保该函数运行未出现错误。
  5. 打开 Amazon Pinpoint 控制台，网址为：<https://console.aws.amazon.com/pinpoint/>。
  6. 在 All projects (所有项目) 页面上，选择您在 [步骤 1.1](#) 中创建的项目。

7. 在导航窗格中，选择客户细分。在客户细分页面上，选择创建客户细分。
8. 在 Segment group 1 (分段组 1) 的 Add filters to refine your segment (添加筛选条件以细化分段) 下，选择 Filter by user (按用户筛选)。
9. 在“选择用户属性”中，选择 FirstName。然后，对于 Choose values (选择值)，选择您在测试事件中指定的名字。

客户细分估算部分应显示有一个合格端点，以及总共一个端点。

下一步：[设置 Amazon API Gateway](#)

## 步骤 4：设置 Amazon API Gateway

在本部分，您将使用 Amazon API Gateway 创建一个新的 API。您在此解决方案中部署的注册表单将调用此 API。然后，API Gateway 会将注册表中捕获的信息传递给您在[步骤 3](#)中创建的 Lambda 函数。

### 步骤 4.1：创建 API

首先，您必须在 API Gateway 中创建一个新的 API。以下过程演示如何创建新的 REST API。

#### 创建新的 API

1. 打开 API Gateway 控制台，网址为：<https://console.aws.amazon.com/apigateway/>。
2. 选择创建 API。做出以下选择：
  - 在选择协议下，选择 REST。
  - 在创建新 API 下，选择新建 API。
  - 在设置下，对于名称，输入一个名称，如 **RegistrationForm**。对于描述，可以选择输入一些用于描述 API 用途的文本。对于端点类型，选择区域性。然后选择创建 API。

这些设置的示例如下图所示。

## Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

REST  WebSocket

## Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API  Clone from existing API  Import from Swagger or Open API 3  Example API

## Settings

Choose a friendly name and description for your API.

API name*	<input type="text" value="RegistrationForm"/>
Description	<input type="text" value="Collects input from a registration form. which is passed on to a"/>
Endpoint Type	<input type="text" value="Regional"/> ⓘ

\* Required


Create API

## 步骤 4.2 : 创建资源

现在，您已经创建了一个 API，可以开始向其添加资源。之后，向资源添加 POST 方法，并告知 API Gateway 将您从此方法中接收的数据传递到您的 Lambda 函数。

1. 在操作菜单上，选择创建资源。在新建子资源窗格中，对于资源名称，输入 **register**，如下图所示。选择创建资源。

## New Child Resource

Use this page to create a new child resource for your resource. 

Configure as [proxy resource](#)  

Resource Name\*

Resource Path\*

You can add path parameters using brackets. For example, the resource path **{username}** represents a path parameter called 'username'. Configuring **/{proxy+}** as a proxy resource catches all requests to its sub-resources. For example, it works for a GET request to /foo. To handle requests to /, add a new ANY method on the / resource.

Enable API Gateway CORS  

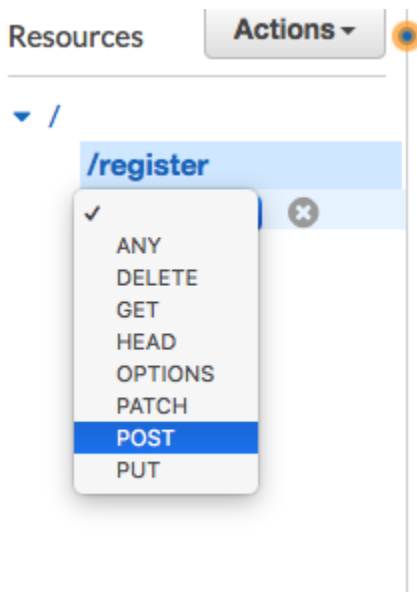
\* Required

Cancel

Create Resource

- 在操作菜单上，选择创建方法。从出现的菜单中选择 POST，如下图所示。然后，选择复选标记

 按钮。



- 在 /register - POST - 设置窗格中，进行以下选择：

- 对于集成类型，选择 Lambda 函数。



- 选择使用 Lambda 代理集成。
- 对于 Lambda 区域，请选择您在其中创建了 Lambda 函数的区域。
- 对于 Lambda 函数，请选择您在[步骤 3](#)中创建的 RegisterEndpoint 函数。

这些设置的示例如下图所示。

## /register - POST - Setup



Choose the integration point for your new method.

**Integration type**  Lambda Function ⓘ

HTTP ⓘ

Mock ⓘ

AWS Service ⓘ

VPC Link ⓘ

**Use Lambda Proxy integration**  ⓘ

**Lambda Region** us-east-1

**Lambda Function**

EndpointRegistration ⓘ

**Use Default Timeout**  ⓘ

**Save**

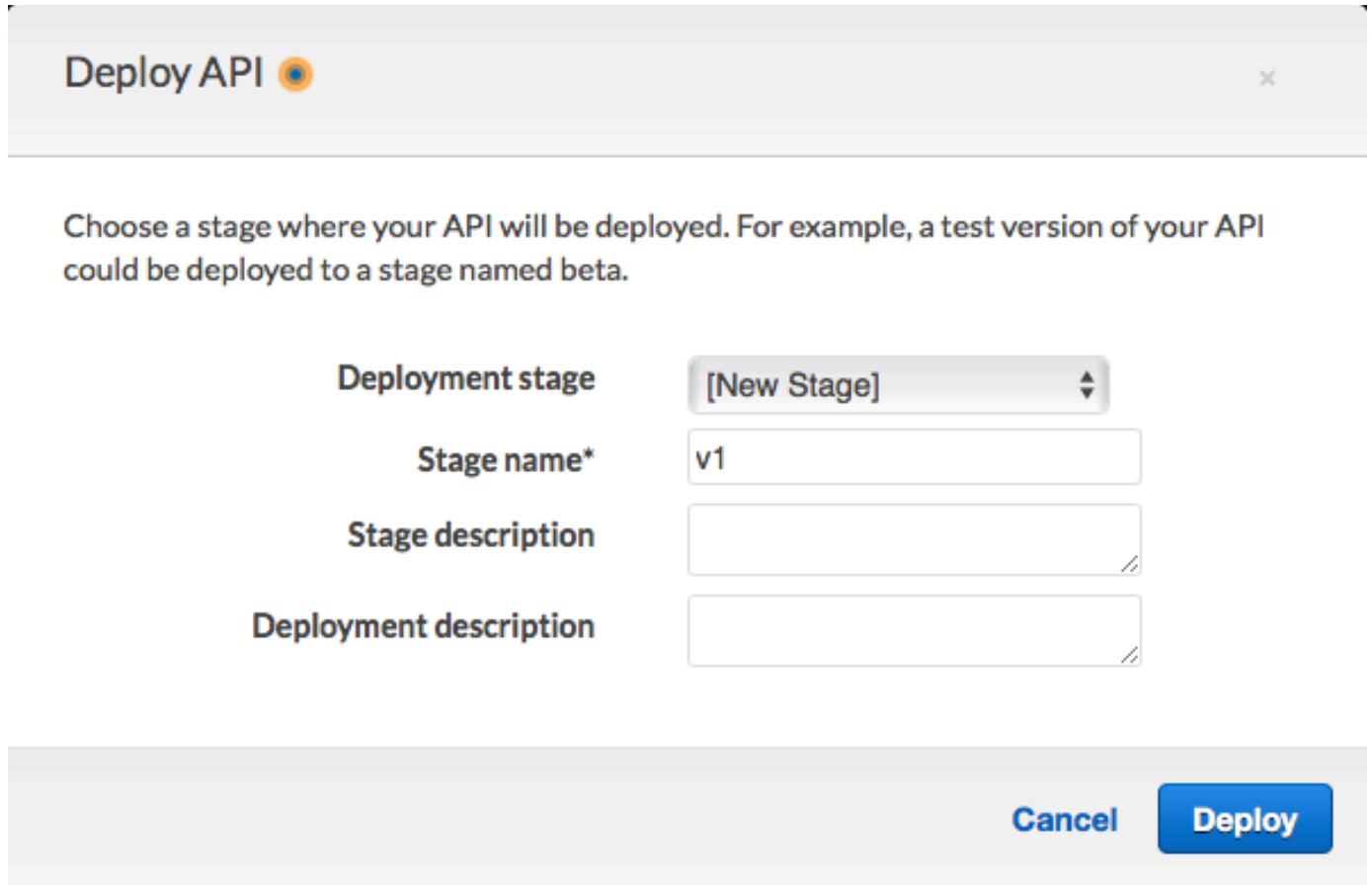
选择保存。在显示的窗口中，选择确定以授予 API Gateway 执行您的 Lambda 函数的权限。

## 步骤 4.3：部署 API

该 API 现已准备就绪，可供使用。此时，您必须对其进行部署，以便创建可公开访问的端点。

1. 在操作菜单上，选择部署 API。在部署 API 窗口中，进行以下选择：
  - 对于部署阶段，选择[新阶段]。
  - 对于阶段名称，输入 **v1**。
  - 选择部署。

这些选择的示例如下图所示。



Deploy API

Choose a stage where your API will be deployed. For example, a test version of your API could be deployed to a stage named beta.

Deployment stage [New Stage]

Stage name\* v1

Stage description

Deployment description

Cancel Deploy

- 在 v1 阶段编辑器窗格中，选择 /register 资源，然后选择 POST 方法。复制调用 URL 旁边显示的地址，如下图所示。

v1 - POST - /register

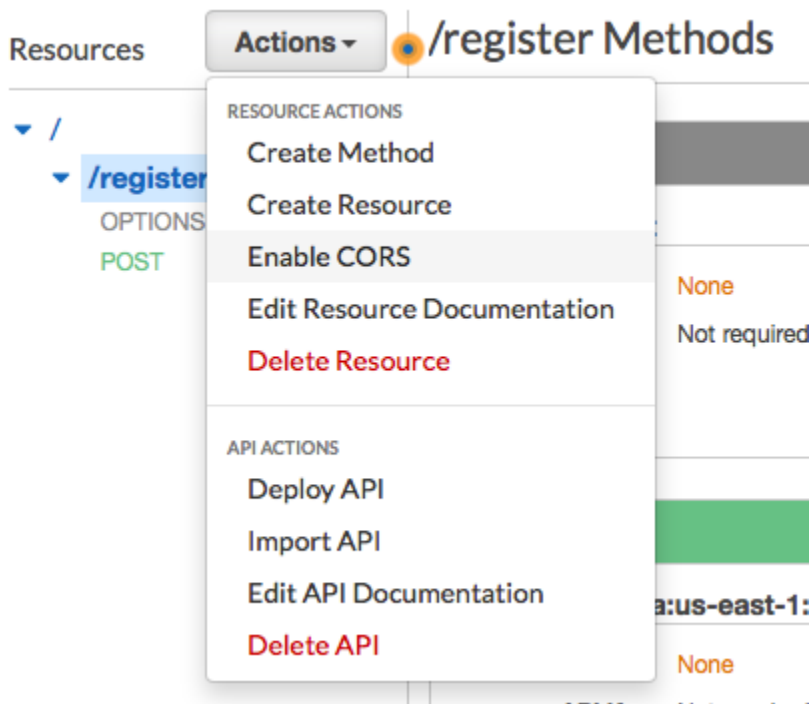
Invoke URL: <https://example.execute-api.us-east-1.amazonaws.com/v1/register>

Use this page to override the v1 stage settings for the POST to /register method.

Settings  Inherit from stage  
 Override for this method

Save Changes

- 在导航窗格中，选择资源。在资源列表中，选择 /register 资源。最后，在操作菜单上，选择启用 CORS，如下图所示。



4. 在启用 CORS 窗格中，选择启用 CORS 并替换现有 CORS 标头。

下一步：[创建和部署 Web 表单](#)

## 步骤 5：创建和部署 Web 表单

此解决方案所有使用 AWS 服务的组成部分现在均已就位。最后一步是创建和部署捕获客户数据的 Web 表单。

### 步骤 5.1：创建 JavaScript 表单处理程序

在本部分，您将创建一个 JavaScript 函数，该函数会解析您在下一部分中创建的 Web 表单的内容。解析内容后，此函数将数据发送到您在[第 4 部分](#)中创建的 API。

#### 创建表单处理程序

1. 在文本编辑器中，创建一个新文件。
2. 在编辑器中，粘贴以下代码。

```
$(document).ready(function() {  
  
    // Handle form submission.  
    $("#submit").click(function(e) {
```

```
var firstName = $("#firstName").val(),
    lastName = $("#lastName").val(),
    source = window.location.pathname,
    optTimestamp = undefined,
    utcSeconds = Date.now() / 1000,
    timestamp = new Date(0),
    phone = $("#areaCode").val()
        + $("#phone1").val()
        + $("#phone2").val();

e.preventDefault();

if (firstName == "") {
    $('#form-response').html('<div class="mt-3 alert alert-info"
role="alert">Please enter your first name.</div>');
} else if (lastName == "") {
    $('#form-response').html('<div class="mt-3 alert alert-info"
role="alert">Please enter your last name.</div>');
} else if (phone.match(/[^0-9]/gi)) {
    $('#form-response').html('<div class="mt-3 alert alert-info"
role="alert">Your phone number contains invalid characters. Please check the phone
number that you supplied.</div>');
} else if (phone.length < 10) {
    $('#form-response').html('<div class="mt-3 alert alert-info"
role="alert">Please enter your phone number.</div>');
} else if (phone.length > 10) {
    $('#form-response').html('<div class="mt-3 alert alert-info"
role="alert">Your phone number contains too many digits. Please check the phone
number that you supplied.</div>');
} else {
    $('#submit').prop('disabled', true);
    $('#submit').html('<span class="spinner-border spinner-border-sm"
role="status" aria-hidden="true"></span> Saving your preferences</button>');

    timestamp.setUTCSeconds(utcSeconds);

    var data = JSON.stringify({
        'destinationNumber': phone,
        'firstName': firstName,
        'lastName': lastName,
        'source': source,
        'optTimestamp': timestamp.toString()
    });
});
```

```
$.ajax({
  type: 'POST',
  url: 'https://example.execute-api.us-east-1.amazonaws.com/v1/register',
  contentType: 'application/json',
  data: data,
  success: function(res) {
    $('#form-response').html('<div class="mt-3 alert alert-success"
role="alert"><p>Congratulations! You've successfully registered for SMS
Alerts from ExampleCorp.</p><p>We just sent you a message. Follow the instructions
in the message to confirm your subscription. We won't send any additional
messages until we receive your confirmation.</p><p>If you decide you don't
want to receive any additional messages from us, just reply to one of our messages
with the keyword STOP.</p></div>');
    $('#submit').prop('hidden', true);
    $('#unsubAll').prop('hidden', true);
    $('#submit').text('Preferences saved!');
  },
  error: function(jqxhr, status, exception) {
    $('#form-response').html('<div class="mt-3 alert alert-danger"
role="alert">An error occurred. Please try again later.</div>');
    $('#submit').text('Save preferences');
    $('#submit').prop('disabled', false);
  }
});
});
});
```

3. 在上述示例中，将 `https://example.execute-api.us-east-1.amazonaws.com/v1/register` 替换为您在[步骤 4.3](#) 中获取的调用 URL。
4. 保存该文件。

## 步骤 5.2：创建表单文件

在本部分，您将创建一个 HTML 文件，其中包含客户注册您的短信计划时使用的表单。此文件使用您在上一部分中创建的 JavaScript 表单处理程序将表单数据传输到您的 Lambda 函数。

### ⚠ Important

当用户提交此表单时，它将触发一个 Lambda 函数，该函数会调用多个 Amazon Pinpoint API 操作。恶意用户可以对您的表单发起攻击，这可能会导致发出大量请求。如果您计划将此解决方案用于生产使用案例，应通过使用 [Google reCAPTCHA](#) 之类的系统来确保其安全。

## 创建表单

1. 在文本编辑器中，创建一个新文件。
2. 在编辑器中，粘贴以下代码。

```
<!doctype html>
<html lang="en">

<head>
  <!-- Meta tags required by Bootstrap -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQU0hcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-U02eT0CpHqdSJK6hJty5KVphtPhzWj9W01cllHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy60rQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

  <script type="text/javascript" src="SMSFormHandler.js"></script>
  <title>SMS Registration Form</title>
</head>

<body>
```

```

<div class="container">
  <div class="row justify-content-center mt-3">
    <div class="col-md-6">
      <h1>Register for SMS Alerts</h1>
      <p>Enter your phone number below to sign up for PromotionName messages from ExampleCorp.</p>
      <p>We don't share your contact information with anyone else. For more information, see our <a href="http://example.com/privacy">Privacy Policy</a>.</p>
      <p>ExampleCorp alerts are only available to recipients in the United States.</p>
    </div>
  </div>
  <div class="row justify-content-center">
    <div class="col-md-6">
      <form>
        <div class="form-group">
          <label for="firstName" class="font-weight-bold">First name</label>
          <input type="text" class="form-control" id="firstName" placeholder="Your first name" required>
        </div>
        <div class="form-group">
          <label for="lastName" class="font-weight-bold">Last name</label>
          <input type="text" class="form-control" id="lastName" placeholder="Your last name" required>
        </div>
        <label for="areaCode" class="font-weight-bold">Phone number</label>
        <div class="input-group">
          <span class="h3">(&nbsp;&nbsp;&nbsp;;</span>
          <input type="tel" class="form-control" id="areaCode" placeholder="Area code" required>
          <span class="h3">&nbsp;&nbsp;&nbsp;)&nbsp;&nbsp;&nbsp;</span>
          <input type="tel" class="form-control" id="phone1" placeholder="555" required>
          <span class="h3">&nbsp;&nbsp;&nbsp;-&nbsp;&nbsp;&nbsp;</span>
          <input type="tel" class="form-control" id="phone2" placeholder="0199" required>
        </div>
        <div id="form-response"></div>
        <button id="submit" type="submit" class="btn btn-primary btn-block mt-3">Submit</button>
      </form>
    </div>
  </div>
</div>
<div class="row mt-3">

```

```
<div class="col-md-12 text-center">
  <small class="text-muted">Copyright © 2019, ExampleCorp or its
affiliates.</small>
</div>
</div>
</div>
</body>
</html>
```

3. 在上述示例中，将 *SMSFormHandler.js* 替换为您在上一部分中创建的表单处理程序 JavaScript 文件的完整路径。
4. 保存该文件。

## 步骤 5.2：上传表单文件

现在，您已经创建了 HTML 表单和 JavaScript 表单处理程序，最后一步是将这些文件发布到 Internet。本部分假定您已经有一个 Web 托管提供商。如果您没有托管提供商，您可以使用 Amazon Route 53、Amazon Simple Storage Service (Amazon S3) 和 Amazon CloudFront 来启动网站。有关更多信息，请参阅[托管静态网站](#)。

如果您使用另一个 Web 托管提供商，请参阅该提供商的文档，以了解有关发布网页的更多信息。

## 步骤 5.3：测试表单

发布表单后，您应提交一些测试事件，以确保该表单按预期工作。

### 测试注册表单

1. 在 Web 浏览器中，转至您上传注册表单的位置。如果您使用了[步骤 5.1](#) 中的代码示例，则会看到一个类似于下图示例的表单。



# Register for SMS Alerts

Enter your phone number below to sign up for PromotionName messages from ExampleCorp.

We don't share your contact information with anyone else. For more information, see our [Privacy Policy](#).

ExampleCorp alerts are only available to recipients in the United States.

## First name

## Last name

## Phone number

(  )  -

Copyright © 2019, ExampleCorp or its affiliates.

2. 在名字、姓氏和电话号码字段中输入您的联系信息。

### Note

当您提交表单时，Amazon Pinpoint 会尝试向您指定的电话号码发送消息。由于有这种功能，您应该从头到尾使用真实的电话号码来测试该解决方案。

如果您在[步骤 3](#)中测试了 Lambda 函数，则您的 Amazon Pinpoint 项目已经包含至少一个端点。当您测试此表单时，应在表单上提交一个不同的电话号码，或者通过使用 [DeleteEndpoint](#) API 操作删除现有的端点。

3. 检查与您指定的电话号码关联的设备，以确保它收到消息。
4. 打开 Amazon Pinpoint 控制台，网址为：<https://console.aws.amazon.com/pinpoint/>。
5. 在所有项目页面上，选择您在[步骤 1.1](#)中创建的项目。

6. 在导航窗格中，选择客户细分。在客户细分页面上，选择创建客户细分。
7. 在客户细分组 1 的添加筛选条件以细化客户细分下，选择按用户筛选。
8. 对于选择用户属性，选择 FirstName。然后，对于选择值，选择您在提交表单时指定的名字。

客户细分估算部分应显示有零个合格端点以及一个端点（在总计端点下），如以下示例所示。这是预期结果。当 Lambda 函数创建一个新的端点时，该端点默认情况下选择退出。

The screenshot displays the Amazon Pinpoint console interface for configuring a segment. On the left, under 'Segment group 1', there is a section for 'Endpoints that match any of the following filters:'. A filter named 'Filter 1: User' is active, with the attribute 'FirstName' set to 'is' and the value 'Carlos'. Below this, there is an 'OR' section for adding more filters. On the right, the 'Segment estimate' sidebar shows '0 endpoints' and a red warning box stating 'No matches found. Your segment didn't produce any results. Remove or modify your segment filters until the segment contains at least one member.'

9. 在设备收到消息时，使用您在 [步骤 1.3](#) 中指定的双向短信关键字来回复该消息。Amazon Pinpoint 会立即发送回复消息。
10. 在 Amazon Pinpoint 控制台中，重复步骤 4 至 8。这次，当您创建一个客户细分时，将看到一个合格端点，以及总共一个端点。这是预期结果，因为端点现已选择接收消息。

## 后续步骤

完成本教程后，您完成了以下操作：

- 创建一个 Amazon Pinpoint 项目，配置短信渠道，并获得专用的长代码。
- 创建一个 IAM 策略（该策略使用最低权限原则授予访问权限），并将该策略与角色关联。
- 创建两个 Lambda 函数，这两个函数使用 Amazon Pinpoint API 中的 PhoneNumberValidate、UpdateEndpoint 和 SendMessages 操作。
- 使用 API Gateway 创建 REST API。
- 创建和部署基于 Web 的表单，用于收集客户的联系信息。
- 对解决方案进行测试，以确保其有效。

本部分讨论使用客户信息（通过使用此解决方案收集）可以采取的几种方式。还包括一些建议，告诉您可以采用哪些方法来自定义此解决方案以适合特定的使用案例。

## 创建客户细分

您通过此表单收集的所有客户详细信息都将作为端点进行存储。此解决方案会创建包含几个属性的端点，您可以用它们进行客户细分。

例如，此解决方案会捕获名为 `Source` 的端点属性。该属性包含指向表单托管位置的完整路径。创建客户细分时，可以按端点来筛选客户细分，然后通过选择 `Source` 属性进一步细化筛选。

根据 `Source` 属性创建客户细分在许多方面都很有用。首先，这可以让您快速对已注册从您那里接收短信的客户创建客户细分。此外，Amazon Pinpoint 中的客户细分工具可自动排除未选择接收消息的端点。

如果您决定在多个不同的位置托管注册表单，`Source` 属性也很有用。例如，您的营销材料可能引用托管在某个位置中的表单，而浏览您网站时发现该表单的客户可能看到的是托管在其他某个位置中的表单。当您执行此操作时，看到营销材料之后填写表单的客户的源属性不同于在网站上找到表单后进行填写的客户的源属性。您可以利用这种差异来创建不同的客户细分，然后向每个受众发送定制通信。

## 发送个性化活动消息

创建客户细分后，您可以开始向这些客户细分发送活动。创建活动消息时，您可以通过指定要在消息中包含的端点属性来对其进行个性化。例如，此解决方案中使用的 Web 表单要求客户输入其名字和姓氏。这些值存储在端点关联的用户记录中。

例如，如果您使用 `GetEndpoint API` 操作来检索有关使用此解决方案创建的端点的信息，则会看到与以下示例类似的部分：

```
...
"User": {
  "UserAttributes": {
    "FirstName": [
      "Carlos"
    ],
    "LastName": [
      "Salazar"
    ]
  }
}
...
```

如果您想在活动消息中包含这些属性的值，则可以使用点表示法来引用属性。然后，用一对大括号将整个引用括起来。例如，要在活动消息中包含每个收件人的名字，请在消息中包含以下字符串：`{{User.UserAttributes.FirstName}}`。当 Amazon Pinpoint 发送消息时，它将该字符串替换为 `FirstName` 属性的值。

## 使用表单收集其他信息

您可以修改此解决方案，以收集有关注册表单的其他信息。例如，您可以要求客户提供他们的地址，然后使用地址数据填充 Endpoint 资源中的 `Location.City`、`Location.Country`、`Location.Region` 和 `Location.PostalCode` 字段。收集注册表单上的地址信息可能会使端点包含更准确的信息。要作出此更改，您需要将相应字段添加到 Web 表单。您还必须修改 JavaScript 代码，以便表单传递新值。最后，您必须修改创建端点的 Lambda 函数以处理新的传入信息。

您也可以修改表单，以便它通过其他渠道收集联系信息。例如，您不仅可以使使用表单来收集客户的电话号码，还可以收集其电子邮件地址。要进行此项更改，您需要修改 Web 表单的 HTML 和 JavaScript。您还必须修改创建端点的 Lambda 函数，以便创建两个单独的端点（一个作为电子邮件端点，另一个作为短信端点）。您还应修改 Lambda 函数，以便为 `User.UserId` 属性生成一个唯一值，然后将该值与这两个端点进行关联。

## 记录其他属性以用于审核

此解决方案在创建和更新端点时会记录两个重要属性。首先，当第一个 Lambda 函数最初创建端点时，它会在 `Attributes.Source` 属性中记录表单自身的 URL。如果客户响应这条消息，则第二个 Lambda 函数会创建一个 `Attributes.OptInTimestamp` 属性。此属性包含客户对于接收您的消息表示同意的确切日期和时间。

如果移动运营商或监管机构要求您提供客户同意的证据，那么这两个字段很有用。通过使用 [GetEndpoint](#) API 操作，您可以随时检索此信息。

您也可以修改 Lambda 函数以记录可能对审核有用的其他数据，如从中提交注册请求的 IP 地址。

# 将 Amazon Pinpoint 与您的应用程序集成

将 Amazon Pinpoint 与您的客户端代码集成以了解用户并与之互动。

在您完成集成后，当用户启动您的应用程序时，该应用程序将连接到 Amazon Pinpoint 服务以添加或更新端点。端点表示您可以向其发送消息的目标，例如用户设备、电子邮件地址或电话号码。

此外，您的应用程序将提供使用率数据或事件。在 Amazon Pinpoint 控制台中查看事件数据，以了解您拥有的用户的数量、这些用户使用您的应用程序的频率和时间等。

在您的应用程序提供端点和事件后，您可以使用此信息为特定受众（即分段）定制消息收发活动。（您也可以直接发送收件人的简单列表而不创建活动。）

使用本节中的主题以将 Amazon Pinpoint 与移动或 Web 应用程序集成。这些主题包含与 JavaScript、Android、Swift 或 Flutter 应用程序集成的代码示例和过程。要开始集成应用程序，请参阅[the section called “使用 AWS Amplify 连接您的前端应用程序”](#)。

在客户端外，您可以使用[受支持的 AWS SDK](#) 或 [Amazon Pinpoint API](#) 导入端点、导出事件数据、定义客户分段、创建和运行活动，等等。

## 主题

- [通过 AWS SDK 使用 Amazon Pinpoint](#)
- [使用 AWS Amplify 将您的前端应用程序连接到 Amazon Pinpoint](#)
- [在应用程序中注册端点](#)
- [在应用程序中报告事件](#)
- [处理推送通知](#)

## 通过 AWS SDK 使用 Amazon Pinpoint

AWS 软件开发工具包（SDK）适用于许多常用编程语言。每个软件开发工具包都提供 API、代码示例和文档，使开发人员能够更轻松地以其首选语言构建应用程序。

软件开发工具包文档	代码示例
<a href="#">AWS SDK for C++</a>	<a href="#">AWS SDK for C++ 代码示例</a>

软件开发工具包文档	代码示例
<a href="#">AWS SDK for Go</a>	<a href="#">AWS SDK for Go 代码示例</a>
<a href="#">AWS SDK for Java</a>	<a href="#">AWS SDK for Java 代码示例</a>
<a href="#">AWS SDK for JavaScript</a>	<a href="#">AWS SDK for JavaScript 代码示例</a>
<a href="#">AWS SDK for Kotlin</a>	<a href="#">AWS SDK for Kotlin 代码示例</a>
<a href="#">AWS SDK for .NET</a>	<a href="#">AWS SDK for .NET 代码示例</a>
<a href="#">AWS SDK for PHP</a>	<a href="#">AWS SDK for PHP 代码示例</a>
<a href="#">AWS SDK for Python (Boto3)</a>	<a href="#">AWS SDK for Python (Boto3) 代码示例</a>
<a href="#">AWS SDK for Ruby</a>	<a href="#">AWS SDK for Ruby 代码示例</a>
<a href="#">AWS SDK for Rust</a>	<a href="#">AWS SDK for Rust 代码示例</a>
<a href="#">适用于 SAP ABAP 的 AWS SDK</a>	<a href="#">适用于 SAP ABAP 的 AWS SDK 代码示例</a>
<a href="#">AWS SDK for Swift</a>	<a href="#">AWS SDK for Swift 代码示例</a>

有关特定于 Amazon Pinpoint 的示例，请参阅[将 Amazon Pinpoint 与 AWS SDK 结合使用的代码示例](#)。

#### 示例可用性

找不到所需的内容？通过使用此页面底部的提供反馈链接请求代码示例。

## 使用 AWS Amplify 将您的前端应用程序连接到 Amazon Pinpoint

使用 AWS Amplify 将应用程序与 AWS 集成。对于 Swift 应用程序，请参阅 Amplify for Swift 文档中的[入门](#)。对于 Android 应用程序，请参阅 Amplify for Android SDK 文档中的[入门](#)。对于 React Native 应用程序，请参阅 Amplify JavaScript 文档中的[入门](#)。对于 Flutter 应用程序，请参阅 Flutter SDK 文档中的[入门](#)。这些主题将帮助您：

- 设置后端资源。

- 使用 Amplify 库将您的应用程序连接到后端资源。

要详细了解如何将您的前端应用程序连接到 Amazon Pinpoint，以进行分析、应用程序内消息收发和推送通知，请参阅 [AWS Amplify](#)。

## 后续步骤

您已将 AWS Amplify 与您的应用程序集成。接下来，请更新您的代码以将您用户的设备注册为端点。请参阅 [在应用程序中注册端点](#)。

## 在应用程序中注册端点

当用户启动了一个会话（例如，通过启动您的移动应用程序）时，您的移动或 Web 应用程序可以自动向 Amazon Pinpoint 注册（或更新）端点。端点表示用户用来启动会话的设备。它包含描述设备的属性，还可包含您定义的自定义属性。端点也可表示其他客户通信方法，如电子邮件地址或手机号码。

在您的应用程序注册端点后，可根据端点属性对受众进行分段。您随后可以让这些分段参与定制的消息收发活动。您还可以使用 Amazon Pinpoint 控制台中的分析页面查看有关端点注册和活动的图表，例如新端点和每日活动端点。

您可以将一个用户 ID 分配给多个端点。用户 ID 表示单个用户，而被分配了用户 ID 的每个端点表示用户的设备之一。将用户 ID 分配给您的端点之后，您可以在控制台中查看有关用户活动的图表，例如 Daily active users 和 Monthly active users。

## 开始前的准备工作

如果您尚没有将适用于 Android 或 iOS 的 AWS 移动软件开发工具包或者 AWS Amplify JavaScript 库与您的应用程序集成，请执行此操作。请参阅 [使用 AWS Amplify 将您的前端应用程序连接到 Amazon Pinpoint](#)。

## 将端点注册到适用于 Android 或 iOS 的 AWS 移动开发工具包

您可以使用适用于 Android 或 iOS 的 AWS 移动开发工具包注册和自定义端点。有关更多信息以及要查看代码示例，请参阅以下文档：

- Android 开发工具包文档中的 [在应用程序中注册端点](#)。
- iOS 开发工具包文档中的 [在应用程序中注册端点](#)。

## 将端点注册到 AWS Amplify JavaScript 库

您可以使用 AWS Amplify JavaScript 库在应用程序中注册和更新端点。有关更多信息以及要查看代码示例，请参阅 AWS Amplify JavaScript 文档中的[更新端点](#)。

### 后续步骤

您已更新您的应用程序以注册端点。现在，当用户启动您的应用程序时，系统会将设备信息和自定义属性提供给 Amazon Pinpoint。您可以使用此信息定义受众分段。在控制台中，您可以查看有关端点的指标以及获得了用户 ID 的用户（如果适用）。

接下来，完成[在应用程序中报告事件](#)中的步骤来更新您的应用程序以报告使用情况数据。

## 在应用程序中报告事件

在您的移动或 Web 应用程序中，可以使用 AWS Mobile SDK 或 [Amazon Pinpoint 事件 API](#) 将使用率数据或事件报告给 Amazon Pinpoint。您可以报告事件以捕获会话时间、用户购买行为、登录尝试或您需要的任何自定义事件类型之类的信息。

在您的应用程序报告事件之后，您可在 Amazon Pinpoint 控制台中查看分析。分析页面上的图表提供了用户行为的多个方面的指标。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 [Amazon Pinpoint 分析图表参考](#)。

要在 Amazon Pinpoint 外部分分析和存储事件数据，您可以将 Amazon Pinpoint 配置为将数据流式传输到 Amazon Kinesis。有关更多信息，请参阅[将 Amazon Pinpoint 事件流式传输到 Kinesis](#)。

通过使用 AWS Mobile SDK 和 AWS Amplify JavaScript 库，您可以调用 Amazon Pinpoint API 来报告以下类型的事件：

#### 会话事件

指示用户打开和关闭您的应用程序的时间及频率。

应用程序报告会话事件之后，您可以使用 Amazon Pinpoint 控制台中的分析页面来查看会话、每日活动端点、7 天保留率等的图表。

#### 自定义事件

通过分配自定义事件类型定义的非标准事件。您可以将自定义属性和指标添加到自定义事件。



在控制台中的分析页面上，事件选项卡显示了您的应用程序报告的所有自定义事件的指标。

## 货币化事件

报告您的应用程序产生的收入以及用户购买的商品数。

在分析页面上，收入选项卡显示了收入、付费用户、销售数量等的图表。

## 身份验证事件

指示用户对您的应用程序进行身份验证的频率。

在分析页面上，用户选项卡显示了登录、注册和身份验证失败的图表。

## 开始前的准备工作

执行以下操作（如果您尚未这样做）：

- 将应用程序与 AWS Amplify 集成。请参阅[使用 AWS Amplify 将您的前端应用程序连接到 Amazon Pinpoint](#)。
- 更新您的应用程序以注册端点。请参阅[在应用程序中注册端点](#)。

## 使用适用于 Android 或 iOS 的 AWS 移动开发工具包报告事件

您可以使用适用于 iOS 和 Android 的 AWS Mobile SDK，来使移动应用程序能够向 Amazon Pinpoint 报告事件。

有关更新您的应用程序以记录事件并将事件提交到 Amazon Pinpoint 的更多信息，请参阅 AWS Amplify 文档中的以下页面：

- iOS SDK 文档中的[分析](#)
- Android SDK 文档中的[分析](#)

## 使用 AWS Amplify JavaScript 库报告事件

您可以使用 AWS Amplify JavaScript 库，以使 JavaScript 和 React Native 应用程序能够向 Amazon Pinpoint 报告应用程序使用情况事件。有关更新应用程序以向 Amazon Pinpoint 提交事件的更多信息，请参阅 AWS Amplify JavaScript 文档中的[分析](#)。

## 使用 Amazon Pinpoint API 报告事件

您可以使用 Amazon Pinpoint API 或 AWS SDK 向 Amazon Pinpoint 批量提交事件。有关更多信息，请参阅《Amazon Pinpoint API 参考》中的 [事件](#)。

### 后续步骤

您已更新您的应用程序以报告事件。现在，当用户与您的应用程序交互时，您的应用程序会将使用率数据发送到 Amazon Pinpoint。您可以在控制台中查看此数据，并将其流式传输到 Amazon Kinesis。

接下来，更新您的应用程序以处理通过 Amazon Pinpoint 发送的推送通知。请参阅[处理推送通知](#)。

## 处理推送通知

以下主题介绍了如何修改您的 Swift、Android、React Native 或 Flutter 应用程序，以使其接收通过 Amazon Pinpoint 发送的推送通知。

### 主题

- [为 Amazon Pinpoint 设置推送通知](#)
- [处理推送通知](#)

## 为 Amazon Pinpoint 设置推送通知

为了设置 Amazon Pinpoint 以便它可以向应用程序发送推送通知，必须首先提供授权 Amazon Pinpoint 向应用程序发送消息的凭证。您提供的凭证取决于您使用的推送通知系统：

- 对于 iOS 应用程序，需提供从 Apple 开发人员门户获取的 SSL 证书。该证书授权 Amazon Pinpoint 通过 Apple Push Notification service (APNs) 将消息发送给应用程序。
- 对于 Android 应用程序，您需要提供从 Firebase 控制台获取的 Web API 密钥。这些凭证授权 Amazon Pinpoint 通过 Firebase Cloud Messaging 将消息发送给应用程序。

获取推送通知渠道的凭证后，您必须在 Amazon Pinpoint 中创建一个项目，并向此项目提供推送通知服务的凭证。

### 主题

- [设置 Swift 推送通知](#)
- [设置 Android 推送通知](#)

- [设置 Flutter 推送通知](#)
- [设置 React Native 推送通知](#)
- [在 Amazon Pinpoint 中创建项目](#)

## 设置 Swift 推送通知

使用 Apple Push Notification service (APNs) 来发送针对 iOS 应用程序的推送通知。必须先在 Apple 开发人员门户上创建应用程序 ID，并且必须创建所需证书，然后才能将推送通知发送给 iOS 设备。您可以在 AWS Amplify 文档的[设置推送通知服务](#)中了解完成这些步骤的更多信息。

### 使用 APNs 令牌

作为最佳实践，您应该开发自己的应用程序，以便在重新安装应用程序时重新生成客户的设备令牌。

如果接收者将其设备升级到 iOS 新的主要版本（例如，从 iOS 12 升级到 iOS 13），之后又重新安装了您的应用程序，则该应用程序会生成一个新的令牌。如果您的应用程序未刷新令牌，则会使用旧令牌来发送通知。结果，Apple Push Notification service (APNs) 拒绝该通知，因为该令牌现在无效。当您尝试发送通知时，会收到来自 APNs 的消息失败通知。

## 设置 Android 推送通知

针对 Android 应用程序的推送通知使用 Firebase Cloud Messaging (FCM) 来发送，FCM 取代了 Google Cloud Messaging (GCM)。您必须先获取 FCM 凭证，然后才能向 Android 设备发送推送通知。然后，您可以使用这些凭证创建 Android 项目并启动可以接收推送通知的示例应用程序。您可以在 AWS Amplify 文档的[推送通知](#)部分了解完成这些步骤的更多信息。

## 设置 Flutter 推送通知

Flutter 应用程序的推送通知使用 Firebase Cloud Messaging (FCM) for Android 和 APNs for iOS 发送。您可以在 [AWS Amplify Flutter 文档](#)的“推送通知”部分了解完成这些步骤的更多信息。

## 设置 React Native 推送通知

React Native 应用程序的推送通知使用 Firebase Cloud Messaging (FCM) for Android 和 APNs for iOS 发送。您可以在 [AWS Amplify JavaScript 文档](#)的“推送通知”部分了解完成这些步骤的更多信息。

## 在 Amazon Pinpoint 中创建项目

在 Amazon Pinpoint 中，项目是一组具有共同目的的设置、数据、活动以及分段。在 Amazon Pinpoint API 中，项目也称为应用程序。本节使用单词“项目”专指该概念。

要在 Amazon Pinpoint 中开始发送推送通知，您必须创建一个项目。接下来，您必须通过提供适当的凭证启用要使用的推送通知渠道。

您可以使用 Amazon Pinpoint 控制台创建新项目并设置推送通知渠道。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的[设置 Amazon Pinpoint 推送通知渠道](#)。

您还可以使用 [Amazon Pinpoint API](#)、[AWS SDK](#) 或 [AWS Command Line Interface](#) (AWS CLI) 创建和设置项目。要创建项目，请使用 Apps 资源。要配置推送通知渠道，请使用以下资源：

- [APNs 渠道](#) 使用 Apple Push Notification service 将消息发送给 iOS 设备的用户。
- [ADM 渠道](#) 将消息发送给 Amazon Kindle Fire 设备的用户。
- [百度渠道](#) 将消息发送给百度用户。
- [GCM 渠道](#) 使用 Firebase Cloud Messaging (FCM) 将消息发送给 Android 设备，该渠道取代了 Google Cloud Messaging (GCM)。

## 处理推送通知

获取发送推送通知所需的凭证后，您可以更新您的应用程序，以便他们能够接收推送通知。有关更多信息，请参阅 AWS Amplify 文档中的[推送通知—入门](#)。

# 向 Amazon Pinpoint 定义您的受众

在 Amazon Pinpoint 中，每个受众成员由一个或多个端点表示。当您使用 Amazon Pinpoint 发送一条消息时，即是将该消息定向到表示目标受众成员的端点。每个端点定义都包含一个消息目标，例如设备令牌、电子邮件地址或电话号码。它还包含有关您的用户及其设备的数据。在分析受众、细分受众或与受众互动之前，第一步是将端点添加到 Amazon Pinpoint 项目中。

要添加端点，您可以：

- 将 Amazon Pinpoint 与您的 Android、iOS 或 JavaScript 客户端集成，这样，当用户访问您的应用程序时，可自动添加端点。
- 使用 Amazon Pinpoint API 单独或批量添加端点。
- 导入存储在 Amazon Pinpoint API 外部的端点定义。

在添加端点后，您可以：

- 在 Amazon Pinpoint 控制台中查看有关受众的分析。
- 通过查找或导出端点数据来了解受众。
- 根据端点属性定义受众分段，例如人口统计数据或用户兴趣。
- 利用定制消息传送活动与目标受众互动。
- 直接向端点列表发送消息。

使用本节中的主题，通过 Amazon Pinpoint API 添加、更新和删除端点。如果要从 Android、iOS 或 JavaScript 客户端自动添加端点，请改为参阅[在应用程序中注册端点](#)。

主题

- [向 Amazon Pinpoint 中添加端点](#)
- [将用户与 Amazon Pinpoint 端点关联](#)
- [将端点批量添加到 Amazon Pinpoint](#)
- [将端点导入 Amazon Pinpoint](#)
- [从 Amazon Pinpoint 中删除端点](#)
- [管理受众成员的最大端点数](#)

## 向 Amazon Pinpoint 中添加端点

端点就是消息送达的目的地，如移动设备、电话号码或电子邮件地址。必须先为受众成员定义一个或多个端点，然后才能为此个体发送消息。

定义端点时，指定渠道和地址。渠道是向端点发送消息所使用的平台的类型。渠道的示例包括推送通知服务、短信或电子邮件。地址指定向端点发送消息时发送到哪里，如设备令牌、电话号码或电子邮件地址。

要添加有关受众的更多信息，可以使用自定义属性和标准属性丰富端点。这些属性可能包含有关您的用户、其首选项、其设备、其所用客户端的版本及其位置的数据。将此类数据添加到端点后，将能够：

- 在 Amazon Pinpoint 控制台中查看有关受众的图表。
- 基于端点属性细分受众，以便可以将消息发送到正确的目标受众。
- 通过包含将被端点属性值所替换的消息变量来个性化设置消息。

如果您已使用 AWS Mobile SDK 或 AWS Amplify JavaScript 库集成了 Amazon Pinpoint，则移动或 JavaScript 客户端应用程序会自动注册端点。客户端将为每个新用户注册一个端点，并且它将更新再次使用用户的端点。要通过移动客户端或 JavaScript 客户端注册端点，请参阅[在应用程序中注册端点](#)。

### 示例

以下示例演示如何将端点添加到 Amazon Pinpoint 项目。此端点表示一个居住在西雅图、使用 iPhone 的受众成员。可通过 Apple Push Notification Service (APNs) 为此人发送消息。端点的地址是 APNs 提供的设备令牌。

#### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

Example 更新端点命令

要添加或更新端点，请使用 [update-endpoint](#) 命令：

```
$ aws pinpoint update-endpoint \  
> --application-id application-id \  
> --endpoint-id endpoint-id \  
> --endpoint-request file://endpoint-request-file.json
```

其中：

- application-id 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。
- example-endpoint 是要分配给新端点的 ID，或者是要更新的现有端点的 ID。
- endpoint-request-file.json 是包含 --endpoint-request 参数输入的本地 JSON 文件的文件路径。

### Example 端点请求文件

示例 update-endpoint 命令使用 JSON 文件作为 --endpoint-request 形参 (parameter) 的实参 (argument)。此文件包含与下类似的端点定义：

```
{
  "ChannelType": "APNS",
  "Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
  "Attributes": {
    "Interests": [
      "Technology",
      "Music",
      "Travel"
    ]
  },
  "Metrics": {
    "technology_interest_level": 9.0,
    "music_interest_level": 6.0,
    "travel_interest_level": 4.0
  },
  "Demographic": {
    "AppVersion": "1.0",
    "Make": "apple",
    "Model": "iPhone",
    "ModelVersion": "8",
    "Platform": "ios",
    "PlatformVersion": "11.3.1",
    "Timezone": "America/Los_Angeles"
  },
  "Location": {
    "Country": "US",
    "City": "Seattle",
    "PostalCode": "98121",
    "Latitude": 47.61,
    "Longitude": -122.33
  }
}
```

```
}  
}
```

有关可用于定义一个端点的属性，请参阅《Amazon Pinpoint API 参考》中的 [EndpointBatchRequest](#) 架构。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要添加端点，请初始化 [EndpointRequest](#) 对象并将其传递到 AmazonPinpoint 客户端的 [updateEndpoint](#) 方法：

```
import com.amazonaws.regions.Regions;  
import com.amazonaws.services.pinpoint.AmazonPinpoint;  
import com.amazonaws.services.pinpoint.AmazonPinpointClientBuilder;  
import com.amazonaws.services.pinpoint.model.*;  
import java.util.Arrays;  
  
public class AddExampleEndpoint {  
  
    public static void main(String[] args) {  
  
        final String USAGE = "\n" +  
            "AddExampleEndpoint - Adds an example endpoint to an Amazon Pinpoint  
application." +  
            "Usage: AddExampleEndpoint <applicationId>" +  
            "Where:\n" +  
            "  applicationId - The ID of the Amazon Pinpoint application to add the example  
" +  
            "endpoint to.";  
  
        if (args.length < 1) {  
            System.out.println(USAGE);  
            System.exit(1);  
        }  
  
        String applicationId = args[0];  
  
        // The device token assigned to the user's device by Apple Push Notification
```



```
// service (APNs).
String deviceToken =
"1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f";

// Initializes an endpoint definition with channel type and address.
EndpointRequest wangXiulansIphoneEndpoint = new EndpointRequest()
    .withChannelType(ChannelType.APNS)
    .withAddress(deviceToken);

// Adds custom attributes to the endpoint.
wangXiulansIphoneEndpoint.addAttributeEntry("interests", Arrays.asList(
    "technology",
    "music",
    "travel"));

// Adds custom metrics to the endpoint.
wangXiulansIphoneEndpoint.addMetricEntry("technology_interest_level", 9.0);
wangXiulansIphoneEndpoint.addMetricEntry("music_interest_level", 6.0);
wangXiulansIphoneEndpoint.addMetricEntry("travel_interest_level", 4.0);

// Adds standard demographic attributes.
wangXiulansIphoneEndpoint.setDemographic(new EndpointDemographic()
    .withAppVersion("1.0")
    .withMake("apple")
    .withModel("iPhone")
    .withModelVersion("8")
    .withPlatform("ios")
    .withPlatformVersion("11.3.1")
    .withTimezone("America/Los_Angeles"));

// Adds standard location attributes.
wangXiulansIphoneEndpoint.setLocation(new EndpointLocation()
    .withCountry("US")
    .withCity("Seattle")
    .withPostalCode("98121")
    .withLatitude(47.61)
    .withLongitude(-122.33));

// Initializes the Amazon Pinpoint client.
AmazonPinpoint pinpointClient = AmazonPinpointClientBuilder.standard()
    .withRegion(Regions.US_EAST_1).build();

// Updates or creates the endpoint with Amazon Pinpoint.
```

```
UpdateEndpointResult result = pinpointClient.updateEndpoint(new
UpdateEndpointRequest()
    .withApplicationId(applicationId)
    .withEndpointId("example_endpoint")
    .withEndpointRequest(wangXiulansIphoneEndpoint));

System.out.format("Update endpoint result: %s\n",
result.getMessageBody().getMessage());

}
}
```

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example PUT 端点请求

要添加端点，请向位于以下 URI 的[端点](#)资源发出 PUT 请求：

```
/v1/apps/application-id/endpoints/endpoint-id
```

其中：

- `application-id` 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。
- `endpoint-id` 是要分配给新端点的 ID，或者是要更新的现有端点的 ID。

在您的请求中，添加所需标头并提供 [EndpointRequest](#) JSON 作为正文：

```
PUT /v1/apps/application_id/endpoints/example_endpoint HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180415T182538Z
Content-Type: application/json
Accept: application/json
X-Amz-Date: 20180428T004705Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180428/us-
east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-
type;host;x-amz-date,
Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{
  "ChannelType": "APNS",
```

```
"Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
"Attributes": {
  "Interests": [
    "Technology",
    "Music",
    "Travel"
  ]
},
"Metrics": {
  "technology_interest_level": 9.0,
  "music_interest_level": 6.0,
  "travel_interest_level": 4.0
},
"Demographic": {
  "AppVersion": "1.0",
  "Make": "apple",
  "Model": "iPhone",
  "ModelVersion": "8",
  "Platform": "ios",
  "PlatformVersion": "11.3.1",
  "Timezone": "America/Los_Angeles"
},
"Location": {
  "Country": "US",
  "City": "Seattle",
  "PostalCode": "98121",
  "Latitude": 47.61,
  "Longitude": -122.33
}
}
```

如果您的请求成功，将收到与下类似的响应：

```
{
  "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
  "Message": "Accepted"
}
```

## 相关信息

有关 Amazon Pinpoint API 中的端点资源的更多信息，包括支持的 HTTP 方法和请求参数，请参阅《Amazon Pinpoint API 参考》中的[端点](#)。

有关使用变量个性化设置消息的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[消息变量](#)。

有关应用于端点的限额的信息（例如可分配的属性数），请参阅[the section called “端点限额”](#)。

## 将用户与 Amazon Pinpoint 端点关联

端点可以包含定义用户（表示您的受众中的一个人）的属性。例如，用户可能表示已安装您的移动应用程序的某个人，或在您的网站上具有账户的某个人。

可以通过指定一个唯一用户 ID 并（可选）自定义用户属性来定义用户。如果某个人在多台设备上使用您的应用程序，或者可通过多个地址为此人发送消息，则可将同一用户 ID 分配给多个端点。在此情况下，Amazon Pinpoint 跨端点同步用户属性。因此，如果您将一个用户属性添加到一个端点，则 Amazon Pinpoint 会将该属性添加到包含相同用户 ID 的每个端点。

可以添加用户属性来跟踪适用于个人且不会因此人所用设备而变化的数据。例如，可以添加人员的姓名、年龄或账户状态属性。

### Tip

如果您的应用程序使用 Amazon Cognito 用户池来处理用户身份验证，则 Amazon Cognito 可以将用户 ID 和属性自动添加到您的端点。对于端点用户 ID 值，Amazon Cognito 将分配已在用户池中分配给用户的 sub 值。要了解如何使用 Amazon Cognito 添加用户，请参阅《Amazon Cognito 开发人员指南》中的[将 Amazon Pinpoint 分析用于 Amazon Cognito 用户池](#)。

将用户定义添加到端点之后，细分受众的方式有了更多选择。可以基于用户属性定义分段，也可以通过导入用户 ID 列表来定义分段。当您向基于用户的分段发送消息时，可能的目标地址包括与分段中的每个用户关联的每个端点。

为受众发送消息的方式也有更多选择。可以使用活动为用户分段发送消息，也可以将消息直接发送到用户 ID 的列表。要个性化设置消息，可以包括将替换为用户属性值的消息变量。

## 示例

以下示例演示如何将用户定义添加到端点。

### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

## Example 更新端点命令

要将用户添加到端点，请使用 [update-endpoint](#) 命令。对于 `--endpoint-request` 参数，可以定义一个包含用户的新端点。或者，要更新现有端点，可以只提供要更改的属性。以下示例通过仅提供用户属性来将用户添加到现有端点：

```
$ aws pinpoint update-endpoint \  
> --application-id application-id \  
> --endpoint-id endpoint-id \  
> --endpoint-request file://endpoint-request-file.json
```

其中：

- *application-id* 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。
- *endpoint-id* 是要分配给新端点的 ID，或者是要更新的现有端点的 ID。
- *endpoint-request-file.json* 是包含 `--endpoint-request` 参数输入的本地 JSON 文件的文件路径。

## Example 端点请求文件

示例 `update-endpoint` 命令使用 JSON 文件作为 `--endpoint-request` 形参 (parameter) 的实参 (argument)。此文件包含与下类似的用户定义：

```
{  
  "User":{  
    "UserId":"example_user",  
    "UserAttributes":{  
      "FirstName":["Wang"],  
      "LastName":["Xiulan"],  
      "Gender":["Female"],  
      "Age":["39"]  
    }  
  }  
}
```

有关可用于定义一个用户的属性，请参阅《Amazon Pinpoint API 参考》中的 [EndpointRequest](#) 架构中的 `User` 对象。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要将用户添加到端点，请初始化 [EndpointRequest](#) 对象并将其传递到 AmazonPinpoint 客户端的 [updateEndpoint](#) 方法。可以使用此对象定义一个可以包含用户的新端点。或者，要更新现有端点，可以只更新要更改的属性。以下示例通过将 [EndpointUser](#) 对象添加到 EndpointRequest 对象来将用户添加到现有端点：

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
public static void updatePinpointEndpoint(PinpointClient pinpoint, String
applicationId, String endPointId) {
    try {
        List<String> wangXiList = new ArrayList<>();
        wangXiList.add("cooking");
        wangXiList.add("running");
        wangXiList.add("swimming");

        Map myMapWang = new HashMap<>();
        myMapWang.put("interests", wangXiList);

        List<String> myNameWang = new ArrayList<>();
        myNameWang.add("Wang ");
        myNameWang.add("Xiulan");

        Map wangName = new HashMap<>();
        wangName.put("name", myNameWang);
```

```
EndpointUser wangMajor = EndpointUser.builder()
    .userId("example_user_10")
    .userAttributes(wangName)
    .build();

// Create an EndpointBatchItem object for Mary Major.
EndpointRequest wangXiulanEndpoint = EndpointRequest.builder()
    .channelType(ChannelType.EMAIL)
    .address("wang_xiulan@example.com")
    .attributes(myMapWang)
    .user(wangMajor)
    .build();

// Adds multiple endpoint definitions to a single request object.
UpdateEndpointRequest endpointList = UpdateEndpointRequest.builder()
    .applicationId(applicationId)
    .endpointRequest(wangXiulanEndpoint)
    .endpointId(endPointId)
    .build();

UpdateEndpointResponse result = pinpoint.updateEndpoint(endpointList);
System.out.format("Update endpoint result: %s\n",
result.messageBody().message());

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [AddExampleUser.java](#)。

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

Example Put 包含用户定义的端点请求

要将用户添加到端点，请向位于以下 URI 的[端点](#)资源发出 PUT 请求：

`/v1/apps/application-id/endpoints/endpoint-id`

其中：

- *application-id* 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。

- `endpoint-id` 是要分配给新端点的 ID，或者是要更新的现有端点的 ID。

在您的请求中，包含所需标头并提供 [EndpointRequest](#) JSON 作为正文。请求正文可以定义一个可以包含用户的新端点。或者，要更新现有端点，可以只提供要更改的属性。以下示例通过仅提供用户属性来将用户添加到现有端点：

```
PUT /v1/apps/application_id/endpoints/example_endpoint HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180415T182538Z
Content-Type: application/json
Accept: application/json
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180501/us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-type;host;x-amz-date,
  Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{
  "User":{
    "UserId":"example_user",
    "UserAttributes":{
      "FirstName":"Wang",
      "LastName":"Xiulan",
      "Gender":"Female",
      "Age":"39"
    }
  }
}
```

如果请求成功，将收到与下类似的响应：

```
{
  "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
  "Message": "Accepted"
}
```

## 相关信息

有关 Amazon Pinpoint API 中的端点资源的更多信息，包括支持的 HTTP 方法和请求参数，请参阅《Amazon Pinpoint API 参考》中的[端点](#)。



有关使用变量个性化设置消息的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[消息变量](#)。

要通过导入用户 ID 列表来定义分段，请参阅《Amazon Pinpoint 用户指南》中的[导入分段](#)。

有关向最多 100 个用户 ID 发送直接消息的信息，请参阅《Amazon Pinpoint API 参考》中的[用户消息](#)。

有关应用于端点的限额的信息（包括可分配的用户属性数），请参阅[the section called “端点限额”](#)。

## 将端点批量添加到 Amazon Pinpoint

您可以通过批量提供端点的方式在单个操作中添加或更新多个端点。每个批处理请求最多可以包含 100 个端点定义。

如果要在一个操作中添加或更新 100 个以上的端点，请参阅[将端点导入 Amazon Pinpoint](#)。

### 示例

以下示例演示如何通过将两个端点包含在一个批处理请求中来一次添加两个端点。

#### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

Example 更新端点批处理命令

要提交端点批处理请求，请使用 [update-endpoints-batch](#) 命令：

```
$ aws pinpoint update-endpoints-batch \  
> --application-id application-id \  
> --endpoint-batch-request file://endpoint_batch_request_file.json
```

其中：

- *application-id* 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。
- *endpoint\_batch\_request\_file.json* 是至包含 --endpoint-batch-request 参数输入的本地 JSON 文件的文件路径。

Example 端点批处理请求文件

示例 update-endpoints-batch 命令使用 JSON 文件作为 --endpoint-request 形参 (parameter) 的实参 (argument)。此文件包含一批与下类似的端点定义：

```
{
  "Item": [
    {
      "ChannelType": "EMAIL",
      "Address": "richard_roe@example.com",
      "Attributes": {
        "Interests": [
          "Music",
          "Books"
        ]
      },
      "Metrics": {
        "music_interest_level": 3.0,
        "books_interest_level": 7.0
      },
      "Id": "example_endpoint_1",
      "User": {
        "UserId": "example_user_1",
        "UserAttributes": {
          "FirstName": "Richard",
          "LastName": "Roe"
        }
      }
    },
    {
      "ChannelType": "SMS",
      "Address": "+16145550100",
      "Attributes": {
        "Interests": [
          "Cooking",
          "Politics",
          "Finance"
        ]
      },
      "Metrics": {
        "cooking_interest_level": 5.0,
        "politics_interest_level": 8.0,
        "finance_interest_level": 4.0
      },
      "Id": "example_endpoint_2",
      "User": {
        "UserId": "example_user_2",
        "UserAttributes": {
```

```
        "FirstName": "Mary",
        "LastName": "Major"
    }
}
]
```

有关可用于定义一批端点的属性，请参阅《Amazon Pinpoint API 参考》中的 [EndpointBatchRequest](#) 架构。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要提交端点批处理请求，请初始化 [EndpointBatchRequest](#) 对象并将其传递到 AmazonPinpoint 客户端的 [updateEndpointsBatch](#) 方法。以下示例使用两个 EndpointBatchItem 对象填充 EndpointBatchRequest 对象：

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchItem;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchRequest;
import java.util.Map;
import java.util.List;
import java.util.ArrayList;
import java.util.HashMap;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchItem;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.EndpointBatchRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointsBatchRequest;
import java.util.Map;
import java.util.List;
import java.util.ArrayList;
import java.util.HashMap;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class AddExampleEndpoints {

    public static void main(String[] args) {
        final String usage = ""

            Usage:    <appId>

            Where:
                appId - The ID of the application.

            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String applicationId = args[0];
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        updateEndpointsViaBatch(pinpoint, applicationId);
        pinpoint.close();
    }

    public static void updateEndpointsViaBatch(PinpointClient pinpoint, String
applicationId) {
        try {
```

```
List<String> myList = new ArrayList<>();
myList.add("music");
myList.add("books");

Map myMap = new HashMap<String, List>();
myMap.put("attributes", myList);

List<String> myNames = new ArrayList<String>();
myList.add("Richard");
myList.add("Roe");

Map myMap2 = new HashMap<String, List>();
myMap2.put("name", myNames);

EndpointUser richardRoe = EndpointUser.builder()
    .userId("example_user_1")
    .userAttributes(myMap2)
    .build();

// Create an EndpointBatchItem object for Richard Roe.
EndpointBatchItem richardRoesEmailEndpoint =
EndpointBatchItem.builder()
    .channelType(ChannelType.EMAIL)
    .address("richard_roe@example.com")
    .id("example_endpoint_1")
    .attributes(myMap)
    .user(richardRoe)
    .build();

List<String> myListMary = new ArrayList<String>();
myListMary.add("cooking");
myListMary.add("politics");
myListMary.add("finance");

Map myMapMary = new HashMap<String, List>();
myMapMary.put("interests", myListMary);

List<String> myNameMary = new ArrayList<String>();
myNameMary.add("Mary ");
myNameMary.add("Major");

Map maryName = new HashMap<String, List>();
myMapMary.put("name", myNameMary);
```

```
EndpointUser maryMajor = EndpointUser.builder()
    .userId("example_user_2")
    .userAttributes(maryName)
    .build();

// Create an EndpointBatchItem object for Mary Major.
EndpointBatchItem maryMajorsSmsEndpoint =
EndpointBatchItem.builder()
    .channelType(ChannelType.SMS)
    .address("+16145550100")
    .id("example_endpoint_2")
    .attributes(myMapMary)
    .user(maryMajor)
    .build();

// Adds multiple endpoint definitions to a single request
object.

EndpointBatchRequest endpointList =
EndpointBatchRequest.builder()
    .item(richardRoesEmailEndpoint)
    .item(maryMajorsSmsEndpoint)
    .build();

// Create the UpdateEndpointsBatchRequest.
UpdateEndpointsBatchRequest batchRequest =
UpdateEndpointsBatchRequest.builder()
    .applicationId(applicationId)
    .endpointBatchRequest(endpointList)
    .build();

// Updates the endpoints with Amazon Pinpoint.
UpdateEndpointsBatchResponse result =
pinpoint.updateEndpointsBatch(batchRequest);
System.out.format("Update endpoints batch result: %s\n",
result.messageBody().message());

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [AddExampleEndpoints.java](#)。

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example Put 端点请求

要提交端点批处理请求，请向位于以下 URI 的[端点](#)资源发出 PUT 请求：

```
/v1/apps/application-id/endpoints
```

其中，*application-id* 是要在其中添加或更新端点的 Amazon Pinpoint 项目的 ID。

在您的请求中，包括所需标头，并提供 [EndpointBatchRequest](#) JSON 作为正文：

```
PUT /v1/apps/application_id/endpoints HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
X-Amz-Date: 20180501T184948Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180501/us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;content-length;content-type;host;x-amz-date,
  Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{
  "Item": [
    {
      "ChannelType": "EMAIL",
      "Address": "richard_roe@example.com",
      "Attributes": {
        "Interests": [
          "Music",
          "Books"
        ]
      },
      "Metrics": {
        "music_interest_level": 3.0,
        "books_interest_level": 7.0
      },
      "Id": "example_endpoint_1",
      "User":{
```

```
        "UserId": "example_user_1",
        "UserAttributes": {
            "FirstName": "Richard",
            "LastName": "Roe"
        }
    },
    {
        "ChannelType": "SMS",
        "Address": "+16145550100",
        "Attributes": {
            "Interests": [
                "Cooking",
                "Politics",
                "Finance"
            ]
        },
        "Metrics": {
            "cooking_interest_level": 5.0,
            "politics_interest_level": 8.0,
            "finance_interest_level": 4.0
        },
        "Id": "example_endpoint_2",
        "User": {
            "UserId": "example_user_2",
            "UserAttributes": {
                "FirstName": "Mary",
                "LastName": "Major"
            }
        }
    }
]
}
```

如果您的请求成功，将收到与下类似的响应：

```
{
  "RequestID": "67e572ed-41d5-11e8-9dc5-db288f3cbb72",
  "Message": "Accepted"
}
```



## 相关信息

有关 Amazon Pinpoint API 中的端点资源的更多信息，包括支持的 HTTP 方法和请求参数，请参阅《Amazon Pinpoint API 参考》中的[端点](#)。

## 将端点导入 Amazon Pinpoint

可以通过从 Amazon S3 存储桶导入大量端点的方式来添加或更新端点。如果您在 Amazon Pinpoint 之外拥有受众的记录，并且想要将这些信息添加到 Amazon Pinpoint 项目中，则导入端点的方法很有用。在此情况下，请：

1. 创建基于您自己的受众数据的端点定义。
2. 将这些端点定义保存到一个或多个文件中，然后将这些文件上传到 Amazon S3 存储桶。
3. 通过从存储桶导入端点来将端点添加到 Amazon Pinpoint 项目中。

每个导入任务可传输多达 1 GB 数据。在典型任务（其中每个端点为 4 KB 或更小）中，可以导入约 250,000 个端点。一个 AWS 账户可以运行多达两个并发导入任务。如果导入作业需要更多带宽，您可以向 AWS Support 提交提升服务限额请求。有关更多信息，请参阅[请求提高限额](#)。

## 开始之前

要导入端点，您的 AWS 账户中需要具备以下资源：

- Amazon S3 存储桶。要创建存储桶，请参阅《Amazon Simple Storage Service 用户指南》中的[创建存储桶](#)。
- AWS Identity and Access Management (IAM) 角色，用于授予 Amazon Pinpoint 对您的 Amazon S3 存储桶的读取权限。要创建该角色，请参阅[用于导入端点或分段的 IAM 角色](#)。

## 示例

以下示例演示如何将端点定义添加到您的 Amazon S3 存储桶，然后再将这些端点导入 Amazon Pinpoint 项目。

### 包含端点定义的文件

添加到您的 Amazon S3 存储桶的文件可以包含 CSV 或以换行符分隔的 JSON 格式的端点定义。有关可用于定义端点的属性，请参阅《Amazon Pinpoint API 参考》中的[EndpointRequest](#) JSON 架构。

## CSV

您可以导入在 CSV 文件中定义的端点，如以下示例中所示：

```
ChannelType,Address,Location.Country,Demographic.Platform,Demographic.Make,User.UserId
SMS,12065550182,CN,Android,LG,example-user-id-1
APNS,1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f,US,iOS,Apple,example-user-id-2
EMAIL,john.stiles@example.com,US,iOS,Apple,example-user-id-2
```

第一行是标头，其中包含端点属性。使用点标记（如 `Location.Country` 中所示）指定嵌套属性。

后续行通过为标头中的每个属性提供值来定义端点。

要在值中包含逗号或双引号，请将值括在双引号内，如 `"aaa,bbb"`。

CSV 中的值中不支持换行符。

## JSON

可以导入在以换行符分隔的 JSON 文件中定义的端点，如以下示例中所示：

```
{"ChannelType":"SMS","Address":"12065550182","Location":
{"Country":"CN"},"Demographic":{"Platform":"Android","Make":"LG"},"User":
{"UserId":"example-user-id-1"}}
{"ChannelType":"APNS","Address":"1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f","Location":
{"Country":"US"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
{"ChannelType":"EMAIL","Address":"john.stiles@example.com","Location":
{"Country":"US"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
```

在此格式中，每一行都是一个完整的 JSON 对象，其中包含单独的端点定义。

## 导入任务请求

以下示例演示如何通过将本地文件上传到存储桶，来将端点定义添加到 Amazon S3。然后，示例将端点定义导入 Amazon Pinpoint 项目。

## AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

## Example S3 CP 命令

要将本地文件上传到 Amazon S3 存储桶，请使用 Amazon S3 [cp](#) 命令：

```
$ aws s3 cp ./endpoints-file s3://bucket-name/prefix/
```

其中：

- *./endpoints-file* 是包含端点定义的本地文件的文件路径。
- *bucket-name/prefix/* 是 Amazon S3 存储桶的名称，并且还可以选择使用前缀，以帮助您按层次组织存储桶中的对象。例如，*pinpoint/imports/endpoints/* 就是一个有用的前缀。

## Example 创建导入任务命令

要从 Amazon S3 存储桶导入端点定义，请使用 [create-import-job](#) 命令：

```
$ aws pinpoint create-import-job \  
> --application-id application-id \  
> --import-job-request \  
> S3Url=s3://bucket-name/prefix/key, \  
> RoleArn=iam-import-role-arn, \  
> Format=format, \  
> RegisterEndpoints=true
```

其中：

- *application-id* 是要为之导入端点的 Amazon Pinpoint 项目的 ID。
- *bucket-name/prefix/key* 是 Amazon S3 中包含一个或多个导入对象的位置。此位置可以用单个对象的键结尾，也可以用符合多个对象条件的前缀结尾。
- *iam-import-role-arn* 是 IAM 角色的 Amazon 资源名称 (ARN)，用于授予 Amazon Pinpoint 对存储桶的读取访问权限。
- *format* 可以为 JSON 或 CSV，具体情况取决于定义端点时使用的格式。如果 Amazon S3 位置包含多个混合格式的对象，则 Amazon Pinpoint 将仅导入与指定格式匹配的对象。
- *RegisterEndPoints* 可以为 *true* 或 *false*。当设置为 *true* 时，导入任务会在导入端点定义时向 Amazon Pinpoint 注册端点。

## RegisterEndpoints 和 DefineSegments 组合

RegisterEndpoints	DefineSegments	描述
true	true	Amazon Pinpoint 将导入端点并创建一个包含端点的分段。
true	false	Amazon Pinpoint 将导入端点但不创建分段。
false	true	Amazon Pinpoint 将导入端点并创建一个包含端点的分段。将不保存端点，也不覆盖现有的端点。
false	false	Amazon Pinpoint 将拒绝此请求。

响应包含有关导入任务的详细信息：

```
{
  "ImportJobResponse": {
    "CreationDate": "2018-05-24T21:26:33.995Z",
    "Definition": {
      "DefineSegment": false,
      "ExternalId": "463709046829",
      "Format": "JSON",
      "RegisterEndpoints": true,
      "RoleArn": "iam-import-role-arn",
      "S3Url": "s3://bucket-name/prefix/key"
    },
    "Id": "d5ecad8e417d498389e1d5b9454d4e0c",
    "JobStatus": "CREATED",
    "Type": "IMPORT"
  }
}
```

响应通过 Id 属性提供任务 ID。可以使用此 ID 检查导入任务的当前状态。

## Example 获取导入任务命令

要检查导入任务的当前状态，请使用 `get-import-job` 命令：

```
$ aws pinpoint get-import-job \  
> --application-id application-id \  
> --job-id job-id
```

其中：

- `application-id` 是要为之启动导入任务的 Amazon Pinpoint 项目的 ID。
- `job-id` 是正检查的导入任务的 ID。

此命令的响应提供导入任务的当前状态：

```
{  
  "ImportJobResponse": {  
    "ApplicationId": "application-id",  
    "CompletedPieces": 1,  
    "CompletionDate": "2018-05-24T21:26:45.308Z",  
    "CreationDate": "2018-05-24T21:26:33.995Z",  
    "Definition": {  
      "DefineSegment": false,  
      "ExternalId": "463709046829",  
      "Format": "JSON",  
      "RegisterEndpoints": true,  
      "RoleArn": "iam-import-role-arn",  
      "S3Url": "s3://s3-bucket-name/prefix/endpoint-definitions.json"  
    },  
    "FailedPieces": 0,  
    "Id": "job-id",  
    "JobStatus": "COMPLETED",  
    "TotalFailures": 0,  
    "TotalPieces": 1,  
    "TotalProcessed": 3,  
    "Type": "IMPORT"  
  }  
}
```

响应通过 `JobStatus` 属性提供任务状态。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要将包含端点定义的文件上传到 Amazon S3，请使用 AmazonS3 客户端的 [putObject](#) 方法。

要将端点导入 Amazon Pinpoint 项目，请初始化 [CreateImportJobRequest](#) 对象。然后，将此对象传递到 AmazonPinpoint 客户端的 [createImportJob](#) 方法。

```
package com.amazonaws.examples.pinpoint;

import com.amazonaws.AmazonServiceException;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.pinpoint.AmazonPinpoint;
import com.amazonaws.services.pinpoint.AmazonPinpointClientBuilder;
import com.amazonaws.services.pinpoint.model.CreateImportJobRequest;
import com.amazonaws.services.pinpoint.model.CreateImportJobResult;
import com.amazonaws.services.pinpoint.model.Format;
import com.amazonaws.services.pinpoint.model.GetImportJobRequest;
import com.amazonaws.services.pinpoint.model.GetImportJobResult;
import com.amazonaws.services.pinpoint.model.ImportJobRequest;
import com.amazonaws.services.s3.AmazonS3;
import com.amazonaws.services.s3.AmazonS3ClientBuilder;
import com.amazonaws.services.s3.model.AmazonS3Exception;
import java.io.File;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.List;
import java.util.concurrent.TimeUnit;

public class ImportEndpoints {

    public static void main(String[] args) {

        final String USAGE = "\n" +
            "ImportEndpoints - Adds endpoints to an Amazon Pinpoint application\n" +
            "by: \n" +
            "1.) Uploading the endpoint definitions to an Amazon S3 bucket. \n" +
            +
```

```

        "2.) Importing the endpoint definitions from the bucket to an Amazon
Pinpoint " +
        "application.\n\n" +
        "Usage: ImportEndpoints <endpointsFileLocation> <s3BucketName>
<iamImportRoleArn> " +
        "<applicationId>\n\n" +
        "Where:\n" +
        "  endpointsFileLocation - The relative location of the JSON file
that contains the " +
        "endpoint definitions.\n" +
        "  s3BucketName - The name of the Amazon S3 bucket to upload the
JSON file to. If the " +
        "bucket doesn't exist, a new bucket is created.\n" +
        "  iamImportRoleArn - The ARN of an IAM role that grants Amazon
Pinpoint read " +
        "permissions to the S3 bucket.\n" +
        "  applicationId - The ID of the Amazon Pinpoint application to add
the endpoints to.";

    if (args.length < 1) {
        System.out.println(USAGE);
        System.exit(1);
    }

    String endpointsFileLocation = args[0];
    String s3BucketName = args[1];
    String iamImportRoleArn = args[2];
    String applicationId = args[3];

    Path endpointsFilePath = Paths.get(endpointsFileLocation);
    File endpointsFile = new
File(endpointsFilePath.toAbsolutePath().toString());
    uploadToS3(endpointsFile, s3BucketName);

    importToPinpoint(endpointsFile.getName(), s3BucketName, iamImportRoleArn,
applicationId);
}

private static void uploadToS3(File endpointsFile, String s3BucketName) {

    // Initializes Amazon S3 client.
    final AmazonS3 s3 = AmazonS3ClientBuilder.defaultClient();

```

```
// Checks whether the specified bucket exists. If not, attempts to create
one.
if (!s3.doesBucketExistV2(s3BucketName)) {
    try {
        s3.createBucket(s3BucketName);
        System.out.format("Created S3 bucket %s.\n", s3BucketName);
    } catch (AmazonS3Exception e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}

// Uploads the endpoints file to the bucket.
String endpointsFileName = endpointsFile.getName();
System.out.format("Uploading %s to S3 bucket %s . . .\n", endpointsFileName,
s3BucketName);
try {
    s3.putObject(s3BucketName, "imports/" + endpointsFileName,
endpointsFile);
    System.out.println("Finished uploading to S3.");
} catch (AmazonServiceException e) {
    System.err.println(e.getMessage());
    System.exit(1);
}
}

private static void importToPinpoint(String endpointsFileName, String
s3BucketName,
    String iamImportRoleArn, String applicationId) {

    // The S3 URL that Amazon Pinpoint requires to find the endpoints file.
    String s3Url = "s3://" + s3BucketName + "/imports/" + endpointsFileName;

    // Defines the import job that Amazon Pinpoint runs.
    ImportJobRequest importJobRequest = new ImportJobRequest()
        .withS3Url(s3Url)
        .withRegisterEndpoints(true)
        .withRoleArn(iamImportRoleArn)
        .withFormat(Format.JSON);
    CreateImportJobRequest createImportJobRequest = new CreateImportJobRequest()
        .withApplicationId(applicationId)
        .withImportJobRequest(importJobRequest);

    // Initializes the Amazon Pinpoint client.
```



```
AmazonPinpoint pinpointClient = AmazonPinpointClientBuilder.standard()
    .withRegion(Regions.US_EAST_1).build();

System.out.format("Importing endpoints in %s to Amazon Pinpoint application
%s . . .\n",
    endpointsFileName, applicationId);

try {

    // Runs the import job with Amazon Pinpoint.
    CreateImportJobResult importResult =
pinpointClient.createImportJob(createImportJobRequest);

    String jobId = importResult.getImportJobResponse().getId();
    GetImportJobResult getImportJobResult = null;
    String jobStatus = null;

    // Checks the job status until the job completes or fails.
    do {
        getImportJobResult = pinpointClient.getImportJob(new
GetImportJobRequest()
            .withJobId(jobId)
            .withApplicationId(applicationId));
        jobStatus =
getImportJobResult.getImportJobResponse().getJobStatus();
        System.out.format("Import job %s . . .\n", jobStatus.toLowerCase());
        TimeUnit.SECONDS.sleep(3);
    } while (!jobStatus.equals("COMPLETED") && !jobStatus.equals("FAILED"));

    if (jobStatus.equals("COMPLETED")) {
        System.out.println("Finished importing endpoints.");
    } else {
        System.err.println("Failed to import endpoints.");
        System.exit(1);
    }

    // Checks for entries that failed to import.
    // getFailures provides up to 100 of the first failed entries for the
job, if
    // any exist.
    List<String> failedEndpoints =
getImportJobResult.getImportJobResponse().getFailures();
    if (failedEndpoints != null) {
        System.out.println("Failed to import the following entries:");
```

```

        for (String failedEndpoint : failedEndpoints) {
            System.out.println(failedEndpoint);
        }
    }

    } catch (AmazonServiceException | InterruptedException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
}
}

```

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example S3 PUT 对象请求

要将端点定义添加到存储桶，请使用 Amazon S3 [PUT 对象](#) 操作，并提供端点定义作为正文：

```

PUT /prefix/key HTTP/1.1
Content-Type: text/plain
Accept: application/json
Host: bucket-name.s3.amazonaws.com
X-Amz-Content-Sha256:
    c430dc094b0cec2905bc88d96314914d058534b14e2bc6107faa9daa12fdff2d
X-Amz-Date: 20180605T184132Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/
us-east-1/s3/aws4_request, SignedHeaders=accept;cache-control;content-
length;content-type;host;postman-token;x-amz-content-sha256;x-amz-date,
    Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{"ChannelType":"SMS","Address":"2065550182","Location":
{"Country":"CAN"},"Demographic":{"Platform":"Android","Make":"LG"},"User":
{"UserId":"example-user-id-1"}}
{"ChannelType":"APNS","Address":"1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f","Location":
{"Country":"USA"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}
{"ChannelType":"EMAIL","Address":"john.stiles@example.com","Location":
{"Country":"USA"},"Demographic":{"Platform":"iOS","Make":"Apple"},"User":
{"UserId":"example-user-id-2"}}

```

其中：

- `/prefix/key` 是上传后将包含端点定义的对象的前缀和键名。可以使用此前缀分层次组织对象。例如，`pinpoint/imports/endpoints/` 就是一个有用的前缀。
- `bucket-name` 是要将端点定义添加到的 Amazon S3 存储桶的名称。

### Example POST 导入任务请求

要从 Amazon S3 存储桶导入端点定义，请向[导入任务](#)资源发出 POST 请求。在您的请求中，包含所需标头并提供 [ImportJobRequest](#) JSON 作为正文：

```
POST /v1/apps/application_id/jobs/import HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180605T214912Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/
us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-
control;content-length;content-type;host;postman-token;x-amz-date,
Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{
  "S3Url": "s3://bucket-name/prefix/key",
  "RoleArn": "iam-import-role-arn",
  "Format": "format",
  "RegisterEndpoints": true
}
```

其中：

- `application-id` 是要为之导入端点的 Amazon Pinpoint 项目的 ID。
- `bucket-name/prefix/key` 是 Amazon S3 中包含一个或多个导入对象的位置。此位置可以用单个对象的键结尾，也可以用符合多个对象条件的前缀结尾。
- `iam-import-role-arn` 是 IAM 角色的 Amazon 资源名称 (ARN)，用于授予 Amazon Pinpoint 对存储桶的读取访问权限。
- `format` 可以为 JSON 或 CSV，具体情况取决于定义端点时使用的格式。如果 Amazon S3 位置包含多个混合格式的文件，则 Amazon Pinpoint 将仅导入与指定格式匹配的文件。

如果您的请求成功，将收到与下类似的响应：

```
{
  "Id": "a995ce5d70fa44adb563b7d0e3f6c6f5",
  "JobStatus": "CREATED",
  "CreationDate": "2018-06-05T21:49:15.288Z",
  "Type": "IMPORT",
  "Definition": {
    "S3Url": "s3://bucket-name/prefix/key",
    "RoleArn": "iam-import-role-arn",
    "ExternalId": "external-id",
    "Format": "JSON",
    "RegisterEndpoints": true,
    "DefineSegment": false
  }
}
```

响应通过 Id 属性提供任务 ID。可以使用此 ID 检查导入任务的当前状态。

#### Example GET 导入任务请求

要检查导入任务的当前状态，请向[导入任务](#)资源发出 GET 请求：

```
GET /v1/apps/application_id/jobs/import/job_id HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180605T220744Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180605/us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-control;content-type;host;postman-token;x-amz-date,
  Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache
```

其中：

- *application\_id* 是已为之启动导入任务的 Amazon Pinpoint 项目的 ID。
- *job\_id* 是正检查的导入任务的 ID。

如果您的请求成功，将收到与下类似的响应：

```
{
```

```
"ApplicationId": "application_id",
"Id": "70a51b2cf442447492d2c8e50336a9e8",
"JobStatus": "COMPLETED",
"CompletedPieces": 1,
"FailedPieces": 0,
"TotalPieces": 1,
"CreationDate": "2018-06-05T22:04:49.213Z",
"CompletionDate": "2018-06-05T22:04:58.034Z",
"Type": "IMPORT",
"TotalFailures": 0,
"TotalProcessed": 3,
"Definition": {
  "S3Url": "s3://bucket-name/prefix/key.json",
  "RoleArn": "iam-import-role-arn",
  "ExternalId": "external-id",
  "Format": "JSON",
  "RegisterEndpoints": true,
  "DefineSegment": false
}
}
```

响应通过 `JobStatus` 属性提供任务状态。

## 相关信息

有关 Amazon Pinpoint API 中的导入任务资源的更多信息，包括支持的 HTTP 方法和请求参数，请参阅《Amazon Pinpoint API 参考》中的[导入任务](#)。

## 从 Amazon Pinpoint 中删除端点

当您不想再向某个目标发送消息时（例如目标变得不可访问或客户关闭了账户时），可以删除该端点。

## 示例

以下示例说明如何删除端点。

### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

## Example 删除端点命令

要删除端点，请使用 [delete-endpoint](#) 命令：

```
$ aws pinpoint delete-endpoint \  
> --application-id application-id \  
> --endpoint-id endpoint-id
```

其中：

- application-id 是包含端点的 Amazon Pinpoint 项目的 ID。
- endpoint-id 是要删除的终端节点的 ID。

对此命令的响应是您删除的端点的 JSON 定义。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要删除端点，请使用 AmazonPinpoint 客户端的 [deleteEndpoint](#) 方法。提供 [DeleteEndpointRequest](#) 对象作为方法参数：

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.pinpoint.PinpointClient;  
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;  
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;  
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.pinpoint.PinpointClient;  
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;  
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;  
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
/**  
 * Before running this Java V2 code example, set up your development  
 * environment, including your credentials.  
 *  
 * For more information, see the following documentation topic:
```

```
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class DeleteEndpoint {
    public static void main(String[] args) {
        final String usage = ""

            Usage:  <appName> <endpointId >

            Where:
                appId - The id of the application to delete.
                endpointId - The id of the endpoint to delete.
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String endpointId = args[1];
        System.out.println("Deleting an endpoint with id: " + endpointId);
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        deletePinEndpoint(pinpoint, appId, endpointId);
        pinpoint.close();
    }

    public static void deletePinEndpoint(PinpointClient pinpoint, String appId,
String endpointId) {
        try {
            DeleteEndpointRequest appRequest = DeleteEndpointRequest.builder()
                .applicationId(appId)
                .endpointId(endpointId)
                .build();

            DeleteEndpointResponse result = pinpoint.deleteEndpoint(appRequest);
            String id = result.endpointResponse().id();
            System.out.println("The deleted endpoint id " + id);

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }
}
```

```
        System.exit(1);
    }
    System.out.println("Done");
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [DeleteEndpoint.java](#)。

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example DELETE 端点请求

要删除端点，请向[端点](#)资源发出 DELETE 请求：

```
DELETE /v1/apps/application-id/endpoints/endpoint-id HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

其中：

- *application-id* 是包含端点的 Amazon Pinpoint 项目的 ID。
- *endpoint-id* 是要删除的终端节点的 ID。

对此请求的响应是您删除的端点的 JSON 定义。

## 管理受众成员的最大端点数

您的每位受众成员最多可以具有 15 个与其 UserId 关联的端点，更多信息请参阅[端点限额](#)。如果您试图添加第 16 个端点，则根据 ChannelType，您要么遇到 BadRequestException 问题，要么只有移除具有最早 EffectiveDate 的端点才能成功添加。

### 添加第 16 个端点

- 如果端点的新渠道类型是短信、推送、语音、电子邮件、自定义或 IN\_APP，则会返回 BadRequestException，因为受众成员已达到最大端点数。您需要移除一个与受众成员关联的端点，然后重试，请参阅[从 Amazon Pinpoint 中删除端点](#)。



- 如果端点的新渠道类型是 ADM、GCM、APNS、APNS\_VOIP、APNS\_VOIP\_SANDBOX 或 Baidu，则：
  - 检查当前与受众成员关联的端点至少有一个具有 ADM、GCM、APNS、APNS\_VOICE、APNS\_VOIP\_SANDBOX 或 Baidu 的 ChannelType。如果没有，则返回 BadRequestException，并且您需要在重试之前移除一个端点，请参阅[从 Amazon Pinpoint 中删除端点](#)。
  - 否则，如果 ChannelType 为 ADM、GCM、APNS、APNS\_VOIP、APNS\_VOIP\_SANDBOX 或 Baidu，则将具有最早 EffectiveDate 的端点设置为 INACTIVE。
    - 删除旧端点中的 UserId。
    - 新的端点与受众成员关联，并且他们仍具有最大数量的端点。

通过将状态设置为 ACTIVE 可以重新启用端点，并且还可为其重新添加 UserId。

## 在 Amazon Pinpoint 中访问受众数据

当您添加端点到 Amazon Pinpoint 时，它将逐渐增长成为一个受众数据的存储库。此数据包含：

- 您使用 Amazon Pinpoint API 添加或更新的端点。
- 当用户进入您的应用程序时，您的客户端代码添加或更新的端点。

当您的受众增长和改变时，您的端点数据也会随之增长和改变。要查看 Amazon Pinpoint 拥有的关于您的受众的最新信息，您可以查找单个端点，也可以导出 Amazon Pinpoint 项目的所有端点。通过查看端点数据，您可以了解到记录在端点中的受众特征，例如：

- 用户的设备和平台。
- 用户的时区。
- 用户的设备上安装的您的应用程序的版本。
- 用户的位置，例如所在城市或国家/地区。
- 您记录的任何自定义属性或指标。

Amazon Pinpoint 控制台还提供了对于在端点中捕获的人口统计数据 and 自定义属性的分析。

要查找端点，必须先将它们添加到 Amazon Pinpoint 项目中。要添加端点，请参阅[向 Amazon Pinpoint 定义您的受众](#)。

使用本节中的主题，通过 Amazon Pinpoint API 查找或导出端点。

### 主题

- [使用 Amazon Pinpoint 查找端点](#)
- [从 Amazon Pinpoint 导出端点](#)
- [使用 Amazon Pinpoint 列出端点 ID](#)

## 使用 Amazon Pinpoint 查找端点

您可以查找已添加到 Amazon Pinpoint 项目的任何单个端点的详细信息。这些详细信息可能包含消息的目标地址、消息收发渠道、有关用户设备的数据、有关用户位置的数据以及记录在端点中的任何自定义属性。

要查找端点，需要用到端点 ID。如果您不知道该 ID，则可以改为导出来获取端点数据。要导出端点，请参阅[the section called “导出端点”](#)。

## 示例

以下示例演示如何通过指定 ID 来查找某个端点。

### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

Example 获取端点命令

要查找端点，请使用 [get-endpoint](#) 命令：

```
$ aws pinpoint get-endpoint \  
> --application-id application-id \  
> --endpoint-id endpoint-id
```

其中：

- *application-id* 是包含该端点的 Amazon Pinpoint 项目的 ID。
- *endpoint-id* 是要查找的端点的 ID。

对此命令的响应是端点的 JSON 定义，如以下示例所示：

```
{  
  "EndpointResponse": {  
    "Address":  
    "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",  
    "ApplicationId": "application-id",  
    "Attributes": {  
      "Interests": [  
        "Technology",  
        "Music",  
        "Travel"  
      ]  
    },  
    "ChannelType": "APNS",  
    "CohortId": "63",
```

```
"CreationDate": "2018-05-01T17:31:01.046Z",
"Demographic": {
  "AppVersion": "1.0",
  "Make": "apple",
  "Model": "iPhone",
  "ModelVersion": "8",
  "Platform": "ios",
  "PlatformVersion": "11.3.1",
  "Timezone": "America/Los_Angeles"
},
"EffectiveDate": "2018-05-07T19:03:29.963Z",
"EndpointStatus": "ACTIVE",
"Id": "example_endpoint",
"Location": {
  "City": "Seattle",
  "Country": "US",
  "Latitude": 47.6,
  "Longitude": -122.3,
  "PostalCode": "98121"
},
"Metrics": {
  "music_interest_level": 6.0,
  "travel_interest_level": 4.0,
  "technology_interest_level": 9.0
},
"OptOut": "ALL",
"RequestId": "7f546cac-6858-11e8-adcd-2b5a07aab338",
"User": {
  "UserAttributes": {
    "Gender": "Female",
    "FirstName": "Wang",
    "LastName": "Xiulan",
    "Age": "39"
  },
  "UserId": "example_user"
}
}
```

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

## Example 代码

要查找端点，请初始化 [GetEndpointRequest](#) 对象。然后，将此对象传递到 AmazonPinpoint 客户端的 [getEndpoint](#) 方法：

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;
```

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;
```

```
/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class LookUpEndpoint {
    public static void main(String[] args) {
        final String usage = ""

                Usage:  <appId> <endpoint>

                Where:
                    appId - The ID of the application to delete.
                    endpoint - The ID of the endpoint.\s
                "";
```

```
    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String appId = args[0];
    String endpoint = args[1];
    System.out.println("Looking up an endpoint point with ID: " + endpoint);
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    lookupPinpointEndpoint(pinpoint, appId, endpoint);
    pinpoint.close();
}

public static void lookupPinpointEndpoint(PinpointClient pinpoint, String appId,
String endpoint) {
    try {
        GetEndpointRequest appRequest = GetEndpointRequest.builder()
            .applicationId(appId)
            .endpointId(endpoint)
            .build();

        GetEndpointResponse result = pinpoint.getEndpoint(appRequest);
        EndpointResponse endResponse = result.endpointResponse();

        // Uses the Google Gson library to pretty print the endpoint JSON.
        Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .setPrettyPrinting()
            .create();

        String endpointJson = gson.toJson(endResponse);
        System.out.println(endpointJson);

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    System.out.println("Done");
}
```

```
}
```

为了以可读格式打印端点数据，本示例使用 Google GSON 库将 `EndpointResponse` 对象转换为 JSON 字符串。

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example GET 端点请求

要查找端点，请向[端点](#)资源发出 GET 请求：

```
GET /v1/apps/application_id/endpoints/endpoint_id HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

其中：

- *application-id* 是包含该端点的 Amazon Pinpoint 项目的 ID。
- *endpoint-id* 是要查找的端点的 ID。

对此请求的响应是端点的 JSON 定义，如以下示例所示：

```
{
  "ChannelType": "APNS",
  "Address": "1a2b3c4d5e6f7g8h9i0j1k2l3m4n5o6p7q8r9s0t1u2v3w4x5y6z7a8b9c0d1e2f",
  "EndpointStatus": "ACTIVE",
  "OptOut": "NONE",
  "RequestId": "b720cfa8-6924-11e8-aeda-0b22e0b0fa59",
  "Location": {
    "Latitude": 47.6,
    "Longitude": -122.3,
    "PostalCode": "98121",
    "City": "Seattle",
    "Country": "US"
  },
  "Demographic": {
    "Make": "apple",
```

```
    "Model": "iPhone",
    "ModelVersion": "8",
    "Timezone": "America/Los_Angeles",
    "AppVersion": "1.0",
    "Platform": "ios",
    "PlatformVersion": "11.3.1"
  },
  "EffectiveDate": "2018-06-06T00:58:19.865Z",
  "Attributes": {
    "Interests": [
      "Technology",
      "Music",
      "Travel"
    ]
  },
  "Metrics": {
    "music_interest_level": 6,
    "travel_interest_level": 4,
    "technology_interest_level": 9
  },
  "User": {},
  "ApplicationId": "application_id",
  "Id": "example_endpoint",
  "CohortId": "39",
  "CreationDate": "2018-06-06T00:58:19.865Z"
}
```

## 相关信息

有关 Amazon Pinpoint API 中端点资源的更多信息，请参阅《Amazon Pinpoint API 参考》中的[端点](#)。

## 从 Amazon Pinpoint 导出端点

要获取 Amazon Pinpoint 拥有的关于受众的所有信息，您可以导出属于某个项目的端点定义。导出时，Amazon Pinpoint 会将端点定义放入您指定的 Amazon S3 存储桶。当您想要执行以下操作时，导出端点很有用：

- 查看您的客户端应用程序注册到 Amazon Pinpoint 的新的和现有的端点的最新数据。
- 将 Amazon Pinpoint 中的端点数据与您自己的客户关系管理 (CRM) 系统同步。
- 创建有关客户数据的报告或分析客户数据。



## 开始之前

要导出端点，您的 AWS 账户中需要有以下资源：

- Amazon S3 存储桶。要创建存储桶，请参阅《Amazon Simple Storage Service 用户指南》中的[创建存储桶](#)。
- AWS Identity and Access Management (IAM) 角色，用于为您的 Amazon S3 存储桶授予 Amazon Pinpoint 写入权限。要创建该角色，请参阅[用于导出端点或分段的 IAM 角色](#)。

## 示例

以下示例演示如何从 Amazon Pinpoint 项目导出端点，然后从 Amazon S3 存储桶下载这些端点。

### AWS CLI

可以通过在 AWS CLI 中运行命令来使用 Amazon Pinpoint。

Example 创建导出任务命令

要导出 Amazon Pinpoint 项目中的端点，请使用 [create-export-job](#) 命令：

```
$ aws pinpoint create-export-job \  
> --application-id application-id \  
> --export-job-request \  
> S3UrlPrefix=s3://bucket-name/prefix/\  
> RoleArn=iam-export-role-arn
```

其中：

- *application-id* 是包含端点的 Amazon Pinpoint 项目的 ID。
- *bucket-name/prefix/* 是 Amazon S3 存储桶的名称，并且可以是帮助您按层次组织存储桶中的对象的前缀。例如，`pinpoint/exports/endpoints/` 就是一个有用的前缀。
- *iam-export-role-arn* 是为 Amazon Pinpoint 授予对存储桶的写入权限的 IAM 角色的 Amazon 资源名称 (ARN)。

对此命令的响应提供了有关导出任务的详细信息：

```
{  
  "ExportJobResponse": {
```

```

    "CreationDate": "2018-06-04T22:04:20.585Z",
    "Definition": {
      "RoleArn": "iam-export-role-arn",
      "S3UrlPrefix": "s3://s3-bucket-name/prefix/"
    },
    "Id": "7390e0de8e0b462380603c5a4df90bc4",
    "JobStatus": "CREATED",
    "Type": "EXPORT"
  }
}

```

响应通过 `Id` 属性提供任务 ID。可以使用此 ID 检查导出任务的当前状态。

Example 获取导出任务命令

要检查导出任务的当前状态，请使用 [get-export-job](#) 命令：

```

$ aws pinpoint get-export-job \
> --application-id application-id \
> --job-id job-id

```

其中：

- *application-id* 是从中导出端点的 Amazon Pinpoint 项目的 ID。
- *job-id* 是您要检查的任务的 ID。

对此命令的响应提供了导出任务的当前状态：

```

{
  "ExportJobResponse": {
    "ApplicationId": "application-id",
    "CompletedPieces": 1,
    "CompletionDate": "2018-05-08T22:16:48.228Z",
    "CreationDate": "2018-05-08T22:16:44.812Z",
    "Definition": {},
    "FailedPieces": 0,
    "Id": "6c99c463f14f49caa87fa27a5798bef9",
    "JobStatus": "COMPLETED",
    "TotalFailures": 0,
    "TotalPieces": 1,
    "TotalProcessed": 215,
    "Type": "EXPORT"
  }
}

```

```
}  
}
```

响应通过 `JobStatus` 属性提供任务状态。当任务状态值为 `COMPLETED` 时，您可以从 Amazon S3 存储桶获取导出的端点。

### Example S3 CP 命令

要下载导出的端点，请使用 Amazon S3 [cp](#) 命令：

```
$ aws s3 cp s3://bucket-name/prefix/key.gz /local/directory/
```

其中：

- *bucket-name/prefix/key* 是 Amazon Pinpoint 在您导出端点时添加到您的存储桶的 `.gz` 文件的位置。此文件包含导出的端点定义。例如，在 URL `https://PINPOINT-EXAMPLE-BUCKET.s3.us-west-2.amazonaws.com/Exports/example.csv` 中，`PINPOINT-EXAMPLE-BUCKET` 是存储桶的名称，`Exports/example.csv` 是密钥。有关密钥的更多信息，请参阅《Amazon S3 用户指南》中的[密钥](#)。
- */local/directory/* 是您要将端点下载到的本地目录的文件路径。

## AWS SDK for Java

您可以通过使用 AWS SDK for Java 提供的客户端在您的 Java 应用程序中使用 Amazon Pinpoint API。

### Example 代码

要从 Amazon Pinpoint 项目中导出端点，请初始化 [CreateExportJobRequest](#) 对象。然后，将此对象传递到 AmazonPinpoint 客户端的 [createExportJob](#) 方法。

要从 Amazon Pinpoint 下载导出的端点，请使用 AmazonS3 客户端的 [getObject](#) 方法。

```
import software.amazon.awssdk.core.ResponseBytes;  
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.pinpoint.PinpointClient;  
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;  
import software.amazon.awssdk.services.pinpoint.model.PinpointException;  
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;  
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
```

```
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;
```

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
```

```
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;

/**
 * To run this code example, you need to create an AWS Identity and Access
 * Management (IAM) role with the correct policy as described in this
 * documentation:
 * https://docs.aws.amazon.com/pinpoint/latest/developerguide/audience-data-export.html
 *
 * Also, set up your development environment, including your credentials.
 *
 * For information, see this documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

public class ExportEndpoints {
    public static void main(String[] args) {
        final String usage = ""

            This program performs the following steps:

            1. Exports the endpoints to an Amazon S3 bucket.
            2. Downloads the exported endpoints files from Amazon S3.
            3. Parses the endpoints files to obtain the endpoint IDs and prints
            them.

            Usage: ExportEndpoints <applicationId> <s3BucketName>
            <iamExportRoleArn> <path>

            Where:
                applicationId - The ID of the Amazon Pinpoint application that has
            the endpoint.
                s3BucketName - The name of the Amazon S3 bucket to export the JSON
            file to.\s
                iamExportRoleArn - The ARN of an IAM role that grants Amazon
            Pinpoint write permissions to the S3 bucket. path - The path where the files
            downloaded from the Amazon S3 bucket are written (for example, C:/AWS/).
            """;

        if (args.length != 4) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
    }

    String applicationId = args[0];
    String s3BucketName = args[1];
    String iamExportRoleArn = args[2];
    String path = args[3];
    System.out.println("Deleting an application with ID: " + applicationId);

    Region region = Region.US_EAST_1;
    PinpointClient pinpoint = PinpointClient.builder()
        .region(region)
        .build();

    S3Client s3Client = S3Client.builder()
        .region(region)
        .build();

    exportAllEndpoints(pinpoint, s3Client, applicationId, s3BucketName, path,
iamExportRoleArn);
    pinpoint.close();
    s3Client.close();
}

public static void exportAllEndpoints(PinpointClient pinpoint,
    S3Client s3Client,
    String applicationId,
    String s3BucketName,
    String path,
    String iamExportRoleArn) {

    try {
        List<String> objectKeys = exportEndpointsToS3(pinpoint, s3Client,
s3BucketName, iamExportRoleArn,
            applicationId);
        List<String> endpointFileKeys = objectKeys.stream().filter(o ->
o.endsWith(".gz"))
            .collect(Collectors.toList());
        downloadFromS3(s3Client, path, s3BucketName, endpointFileKeys);

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
```

```
public static List<String> exportEndpointsToS3(PinpointClient pinpoint, S3Client
s3Client, String s3BucketName,
    String iamExportRoleArn, String applicationId) {

    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd-
HH_mm:ss.SSS_z");
    String endpointsKeyPrefix = "exports/" + applicationId + "_" +
dateFormat.format(new Date());
    String s3UrlPrefix = "s3://" + s3BucketName + "/" + endpointsKeyPrefix +
"/";
    List<String> objectKeys = new ArrayList<>();
    String key;

    try {
        // Defines the export job that Amazon Pinpoint runs.
        ExportJobRequest jobRequest = ExportJobRequest.builder()
            .roleArn(iamExportRoleArn)
            .s3UrlPrefix(s3UrlPrefix)
            .build();

        CreateExportJobRequest exportJobRequest =
CreateExportJobRequest.builder()
            .applicationId(applicationId)
            .exportJobRequest(jobRequest)
            .build();

        System.out.format("Exporting endpoints from Amazon Pinpoint application
%s to Amazon S3 " +
            "bucket %s . . .\n", applicationId, s3BucketName);

        CreateExportJobResponse exportResult =
pinpoint.createExportJob(exportJobRequest);
        String jobId = exportResult.exportJobResponse().id();
        System.out.println(jobId);
        printExportJobStatus(pinpoint, applicationId, jobId);

        ListObjectsV2Request v2Request = ListObjectsV2Request.builder()
            .bucket(s3BucketName)
            .prefix(endpointsKeyPrefix)
            .build();

        // Create a list of object keys.
        ListObjectsV2Response v2Response = s3Client.listObjectsV2(v2Request);
```

```
List<S3Object> objects = v2Response.contents();
for (S3Object object : objects) {
    key = object.key();
    objectKeys.add(key);
}

return objectKeys;

} catch (PinpointException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
return null;
}

private static void printExportJobStatus(PinpointClient pinpointClient,
    String applicationId,
    String jobId) {

    GetExportJobResponse getExportJobResult;
    String status;

    try {
        // Checks the job status until the job completes or fails.
        GetExportJobRequest exportJobRequest = GetExportJobRequest.builder()
            .jobId(jobId)
            .applicationId(applicationId)
            .build();

        do {
            getExportJobResult = pinpointClient.getExportJob(exportJobRequest);
            status =
getExportJobResult.exportJobResponse().jobStatus().toString().toUpperCase();
            System.out.format("Export job %s . . .\n", status);
            TimeUnit.SECONDS.sleep(3);

        } while (!status.equals("COMPLETED") && !status.equals("FAILED"));

        if (status.equals("COMPLETED")) {
            System.out.println("Finished exporting endpoints.");
        } else {
            System.err.println("Failed to export endpoints.");
            System.exit(1);
        }
    }
}
```



```
    } catch (PinpointException | InterruptedException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}

// Download files from an Amazon S3 bucket and write them to the path location.
public static void downloadFromS3(S3Client s3Client, String path, String
s3BucketName, List<String> objectKeys) {

    String newPath;
    try {
        for (String key : objectKeys) {
            GetObjectRequest objectRequest = GetObjectRequest.builder()
                .bucket(s3BucketName)
                .key(key)
                .build();

            ResponseBytes<GetObjectResponse> objectBytes =
s3Client.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            String fileSuffix = new
SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
            newPath = path + fileSuffix + ".gz";
            File myFile = new File(newPath);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
        }
        System.out.println("Download finished.");

    } catch (S3Exception | NullPointerException | IOException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [ExportEndpoints.java](#)。

## HTTP

可以通过直接向 REST API 发出 HTTP 请求来使用 Amazon Pinpoint。

### Example POST 导出任务请求

要导出 Amazon Pinpoint 项目中的端点，请将 POST 请求发给[导出任务](#)资源：

```
POST /v1/apps/application_id/jobs/export HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180606T001238Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180606/
us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-
control;content-length;content-type;host;postman-token;x-amz-date,
Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache

{
  "S3UrlPrefix": "s3://bucket-name/prefix",
  "RoleArn": "iam-export-role-arn"
}
```

其中：

- *application-id* 是包含端点的 Amazon Pinpoint 项目的 ID。
- *bucket-name/prefix* 是 Amazon S3 存储桶的名称，并且可以是帮助您按层次组织存储桶中的对象的前缀。例如，`pinpoint/exports/endpoints/` 就是一个有用的前缀。
- *iam-export-role-arn* 是为 Amazon Pinpoint 授予对存储桶的写入权限的 IAM 角色的 Amazon 资源名称 (ARN)。

对此请求的响应提供了有关导出任务的详细信息：

```
{
  "Id": "611bdc54c75244bfa51fe7001ddb2e36",
  "JobStatus": "CREATED",
  "CreationDate": "2018-06-06T00:12:43.271Z",
  "Type": "EXPORT",
  "Definition": {
    "S3UrlPrefix": "s3://bucket-name/prefix",
```

```
    "RoleArn": "iam-export-role-arn"
  }
}
```

响应通过 `Id` 属性提供任务 ID。可以使用此 ID 检查导出任务的当前状态。

### Example GET 导出任务请求

要检查导出任务的当前状态，请向[导出任务](#)资源发出 GET 请求：

```
GET /v1/apps/application_id/jobs/export/job_id HTTP/1.1
Content-Type: application/json
Accept: application/json
Host: pinpoint.us-east-1.amazonaws.com
X-Amz-Date: 20180606T002443Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20180606/us-east-1/mobiletargeting/aws4_request, SignedHeaders=accept;cache-control;content-type;host;postman-token;x-amz-date,
  Signature=c25cbd6bf61bd3b3667c571ae764b9bf2d8af61b875caced95d1e68d91b4170
Cache-Control: no-cache
```

其中：

- *application-id* 是从中导出端点的 Amazon Pinpoint 项目的 ID。
- *job-id* 是您要检查的任务的 ID。

对此请求的响应提供了导出任务的当前状态：

```
{
  "ApplicationId": "application_id",
  "Id": "job_id",
  "JobStatus": "COMPLETED",
  "CompletedPieces": 1,
  "FailedPieces": 0,
  "TotalPieces": 1,
  "CreationDate": "2018-06-06T00:12:43.271Z",
  "CompletionDate": "2018-06-06T00:13:01.141Z",
  "Type": "EXPORT",
  "TotalFailures": 0,
  "TotalProcessed": 217,
  "Definition": {}
}
```

```
}
```

响应通过 `JobStatus` 属性提供任务状态。当任务状态值为 `COMPLETED` 时，您可以从 Amazon S3 存储桶获取导出的端点。

## 相关信息

有关 Amazon Pinpoint API 中的导出任务资源的更多信息，包括支持的 HTTP 方法和请求参数，请参阅《Amazon Pinpoint API 参考》中的[导出任务](#)。

## 使用 Amazon Pinpoint 列出端点 ID

要更新或删除端点，需要端点 ID。因此，如果要对一个 Amazon Pinpoint 项目中的所有端点执行这些操作，第一步就是列出属于该项目的端点 ID。然后，可以遍历这些 ID 以执行诸如全局添加属性或删除项目中的所有端点之类的操作。

以下示例使用 AWS SDK for Java 并执行以下操作：

1. 调用[从 Amazon Pinpoint 导出端点](#)中的示例代码中的示例 `exportEndpointsToS3` 方法。此方法从 Amazon Pinpoint 项目导出端点定义。端点定义作为 gzip 文件添加到 Amazon S3 存储桶。
2. 下载导出的 gzip 文件。
3. 读取 gzip 文件并从每个端点的 JSON 定义获取端点 ID。
4. 将端点 ID 输出到控制台。
5. 通过删除 Amazon Pinpoint 添加到 Amazon S3 的文件来清理。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class ListEndpointIds {
    public static void main(String[] args) {
        final String usage = ""

            Usage:    <applicationId> <userId>

            Where:
                applicationId - The ID of the Amazon Pinpoint application that has
the endpoint.
                userId - The user id applicable to the endpoints""";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String applicationId = args[0];
        String userId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        listAllEndpoints(pinpoint, applicationId, userId);
        pinpoint.close();
    }

    public static void listAllEndpoints(PinpointClient pinpoint,
        String applicationId,
        String userId) {

        try {
```

```
        GetUserEndpointsRequest endpointsRequest =
        GetUserEndpointsRequest.builder()
            .userId(userId)
            .applicationId(applicationId)
            .build();

        GetUserEndpointsResponse response =
        pinpoint.getUserEndpoints(endpointsRequest);
        List<EndpointResponse> endpoints = response.endpointsResponse().item();

        // Display the results.
        for (EndpointResponse endpoint : endpoints) {
            System.out.println("The channel type is: " + endpoint.channelType());
            System.out.println("The address is " + endpoint.address());
        }

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [ListEndpoints.java](#)。

# 创建分段

用户分段表示基于共享特征（例如用户最近什么时候使用了您的应用程序或他们使用哪个设备平台）的用户子集。分段指定哪些用户接收活动传送的消息。通过定义分段，在需要邀请用户重新使用您的应用程序、提供特别优惠或以其他方式提高用户参与度和购买量时，您可以面向正确的受众。

创建分段之后，可以在一个或多个活动中使用它。活动会向分段中的用户传送定制消息。

有关更多信息，请参阅[分段](#)。

## 主题

- [构建客户细分](#)
- [导入客户细分](#)
- [使用 AWS Lambda 自定义分段](#)

## 构建客户细分

要面向活动的目标受众，请基于应用程序报告的数据构建分段。例如，要面向最近未使用您的应用程序的用户，可以为过去 30 天内未使用您应用程序的用户定义分段。

## 使用 AWS SDK for Java 构建分段

以下示例演示如何使用 AWS SDK for Java 构建分段。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
```

```
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateSegment {
    public static void main(String[] args) {
        final String usage = ""

                Usage:  <appId>

                Where:
                    appId - The application ID to create a segment for.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```



```
String appId = args[0];
PinpointClient pinpoint = PinpointClient.builder()
    .region(Region.US_EAST_1)
    .build();

SegmentResponse result = createSegment(pinpoint, appId);
System.out.println("Segment " + result.name() + " created.");
System.out.println(result.segmentType());
pinpoint.close();
}

public static SegmentResponse createSegment(PinpointClient client, String
appId) {
    try {
        Map<String, AttributeDimension> segmentAttributes = new
HashMap<>();

        segmentAttributes.put("Team", AttributeDimension.builder()
            .attributeType(AttributeType.INCLUSIVE)
            .values("Lakers")
            .build());

        RecencyDimension recencyDimension = RecencyDimension.builder()
            .duration("DAY_30")
            .recencyType("ACTIVE")
            .build();

        SegmentBehaviors segmentBehaviors = SegmentBehaviors.builder()
            .recency(recencyDimension)
            .build();

        SegmentDemographics segmentDemographics = SegmentDemographics
            .builder()
            .build();

        SegmentLocation segmentLocation = SegmentLocation
            .builder()
            .build();

        SegmentDimensions dimensions = SegmentDimensions
            .builder()
            .attributes(segmentAttributes)
            .behavior(segmentBehaviors)
            .demographic(segmentDemographics)
            .location(segmentLocation)
```

```
        .build();

        WriteSegmentRequest writeSegmentRequest =
WriteSegmentRequest.builder()
                    .name("MySegment")
                    .dimensions(dimensions)
                    .build();

        CreateSegmentRequest createSegmentRequest =
CreateSegmentRequest.builder()
                    .applicationId(appId)
                    .writeSegmentRequest(writeSegmentRequest)
                    .build();

        CreateSegmentResponse createSegmentResult =
client.createSegment(createSegmentRequest);
        System.out.println("Segment ID: " +
createSegmentResult.segmentResponse().id());
        System.out.println("Done");
        return createSegmentResult.segmentResponse();

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return null;
}
}
```

运行此示例时，IDE 的控制台窗口会显示以下内容：

```
Segment ID: 09cb2967a82b4a2fbab38fead8d1f4c4
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [CreateSegment.java](#)。

## 导入客户细分

借助 Amazon Pinpoint，可以通过导入有关属于某个用户分段的端点的信息来定义该分段。端点是单个消息发送目的地，如移动推送设备令牌、手机号码或电子邮件地址。

如果您已在 Amazon Pinpoint 外部创建了用户分段，但是需要借助 Amazon Pinpoint 活动来吸引用户，则导入分段会十分有用。

导入分段时，Amazon Pinpoint 从 Amazon Simple Storage Service (Amazon S3) 获取该分段的端点。导入之前，将端点添加到 Amazon S3，并创建一个 IAM 角色，以向 Amazon Pinpoint 授予对 Amazon S3 的访问权限。然后，向 Amazon Pinpoint 提供存储端点的 Amazon S3 位置，Amazon Pinpoint 会将每个端点添加到分段中。

要创建 IAM 角色，请参阅[用于导入端点或分段的 IAM 角色](#)。有关使用 Amazon Pinpoint 控制台导入分段的信息，请参阅《Amazon Pinpoint 用户指南》中的[导入分段](#)。

## 导入客户细分

以下示例演示如何使用 AWS SDK for Java 导入分段。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class ImportSegment {
    public static void main(String[] args) {
        final String usage = ""
```

```
Usage: <appId> <bucket> <key> <roleArn>\s
```

Where:

appId - The application ID to create a segment for.

bucket - The name of the Amazon S3 bucket that contains the segment

definitons.

key - The key of the S3 object.

roleArn - ARN of the role that allows Amazon Pinpoint to access S3.

You need to set trust management for this to work. See [https://docs.aws.amazon.com/IAM/latest/UserGuide/reference\\_policies\\_elements\\_principal.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_principal.html)

```
""";
```

```
if (args.length != 4) {  
    System.out.println(usage);  
    System.exit(1);  
}
```

```
String appId = args[0];  
String bucket = args[1];  
String key = args[2];  
String roleArn = args[3];
```

```
PinpointClient pinpoint = PinpointClient.builder()  
    .region(Region.US_EAST_1)  
    .build();
```

```
ImportJobResponse response = createImportSegment(pinpoint, appId, bucket, key,  
roleArn);  
System.out.println("Import job for " + bucket + " submitted.");  
System.out.println("See application " + response.applicationId() + " for import  
job status.");  
System.out.println("See application " + response.jobStatus() + " for import job  
status.");  
pinpoint.close();  
}
```

```
public static ImportJobResponse createImportSegment(PinpointClient client,  
    String appId,  
    String bucket,  
    String key,  
    String roleArn) {
```

```
    try {  
        ImportJobRequest importRequest = ImportJobRequest.builder()
```

```
        .defineSegment(true)
        .registerEndpoints(true)
        .roleArn(roleArn)
        .format(Format.JSON)
        .s3Url("s3://" + bucket + "/" + key)
        .build();

    CreateImportJobRequest jobRequest = CreateImportJobRequest.builder()
        .importJobRequest(importRequest)
        .applicationId(appId)
        .build();

    CreateImportJobResponse jobResponse = client.createImportJob(jobRequest);
    return jobResponse.importJobResponse();

} catch (PinpointException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
return null;
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [ImportingSegments.java](#)。

## 使用 AWS Lambda 自定义分段

这是适用于公共测试版中功能的预发行文档。本文档随时可能更改。

您可以使用 AWS Lambda 函数来定制 Amazon Pinpoint 活动与您的目标受众互动的方式。借助 AWS Lambda，您可以在 Amazon Pinpoint 发送活动消息的那一刻修改活动的分段。

AWS Lambda 是一项计算服务，您可用来运行代码而无需预配置或管理服务器。您可将代码打包并作为 Lambda 函数上传到 Lambda。当一个函数被调用（由您手动调用，或者为响应事件而自动调用）时，Lambda 会运行该函数。有关更多信息，请参阅 [AWS Lambda 开发人员指南](#)。

要将 Lambda 函数分配给活动，您可以使用 Amazon Pinpoint API 中的 [活动](#) 资源定义活动的 CampaignHook 设置。这些设置包括 Lambda 函数名称。还包括 CampaignHook 模式，用于指定 Amazon Pinpoint 是否接收函数返回值。

您分配给活动的 Lambda 函数称为 Amazon Pinpoint 扩展。

定义了 CampaignHook 设置之后，Amazon Pinpoint 会在运行活动时自动调用 Lambda 函数，然后再发送活动消息。当 Amazon Pinpoint 调用该函数时，它会提供有关消息传送的事件数据。此数据包括活动的分段，即 Amazon Pinpoint 将消息发送到的端点的列表。

如果 CampaignHook 模式设置为 FILTER，则在发送消息之前，Amazon Pinpoint 会允许该函数修改并返回分段。例如，该函数可能会使用包含来自 Amazon Pinpoint 外部的源中数据的属性更新端点定义。或者，该函数可能会根据函数代码中的条件删除某些端点，从而筛选分段。Amazon Pinpoint 从您的函数收到修改后的分段之后，它会使用活动的传送渠道，将消息发送到分段的每个端点。

通过使用 AWS Lambda 来处理您的分段，您可以更好地控制将消息发送给哪些用户以及这些消息中包含什么内容。您可以在发送活动消息的那一刻实时定制您的活动。通过筛选分段，您可以与定义更加明确的分段子集进行互动。通过添加或更新端点属性，您还可以使新数据可供消息变量使用。

### Note

您还可以使用 CampaignHook 设置来分配处理消息传输的 Lambda 函数。这种类型的函数对于通过 Amazon Pinpoint 不支持的自定义渠道（例如社交媒体平台）传送消息非常有用。有关更多信息，请参阅[在 Amazon Pinpoint 中创建自定义渠道](#)。

使用 Amazon Pinpoint 调用 Lambda 钩子时，Lambda 函数还必须与 Amazon Pinpoint 项目位于同一区域。

要使用 AWS Lambda 修改活动分段，请先创建一个函数，由其负责处理 Amazon Pinpoint 所发送的事件数据并返回修改后的分段。然后，通过分配 Lambda 函数策略来授权 Amazon Pinpoint 调用该函数。最后，通过定义 CampaignHook 设置，将该函数分配给一个或多个活动。

## 事件数据

当 Amazon Pinpoint 调用您的 Lambda 函数时，它会提供以下负载作为事件数据：

```
{
  "MessageConfiguration": {Message configuration}
  "ApplicationId": ApplicationId,
  "CampaignId": CampaignId,
  "TreatmentId": TreatmentId,
  "ActivityId": ActivityId,
  "ScheduledTime": Scheduled Time,
  "Endpoints": {
    EndpointId: {Endpoint definition}
  }
}
```

```
    . . .  
  }  
}
```

AWS Lambda 将事件数据传递到您的函数代码。事件数据提供了以下属性：

- MessageConfiguration – 具有与 Amazon Pinpoint API 中[消息](#)资源的 DirectMessageConfiguration 对象相同的结构。
- ApplicationId – 活动所属的 Amazon Pinpoint 项目的 ID。
- CampaignId – 为其调用该函数的 Amazon Pinpoint 活动的 ID。
- TreatmentId – 用于 A/B 测试的活动变体的 ID。
- ActivityId – 由活动执行的活动的 ID。
- ScheduledTime – 将传输活动消息的日期和时间，采用 ISO 8601 格式。
- Endpoints – 将端点 ID 与端点定义关联的映射。每个事件数据负载最多可包含 50 个端点。如果活动分段包含的端点数超过 50 个，则 Amazon Pinpoint 将重复调用该函数，一次最多处理 50 个端点，直至处理完所有端点。

## 创建 Lambda 函数

要了解如何创建 Lambda 函数，请参阅《AWS Lambda 开发人员指南》中的[入门](#)。当您创建函数时，请注意在以下条件下，消息传送会失败：

- Lambda 函数需要超过 15 秒才能返回修改后的分段。
- Amazon Pinpoint 无法解码该函数的返回值。
- 需要从 Amazon Pinpoint 调用 3 次以上才能成功调用该函数。

Amazon Pinpoint 只接受该函数返回值中的端点定义。该函数无法修改事件数据中的其他元素。

## 示例 Lambda 函数

您的 Lambda 函数处理 Amazon Pinpoint 发送的事件数据，并返回修改后的端点，如以下示例处理程序（使用 Node.js 编写）所示：

```
'use strict';  
  
exports.handler = (event, context, callback) => {  
  for (var key in event.Endpoints) {
```

```
    if (event.Endpoints.hasOwnProperty(key)) {
        var endpoint = event.Endpoints[key];
        var attr = endpoint.Attributes;
        if (!attr) {
            attr = {};
            endpoint.Attributes = attr;
        }
        attr["CreditScore"] = [ Math.floor(Math.random() * 200) + 650];
    }
}
console.log("Received event:", JSON.stringify(event, null, 2));
callback(null, event.Endpoints);
};
```

Lambda 将事件数据作为 `event` 参数传递给处理程序。

在此示例中，处理程序迭代 `event.Endpoints` 对象中的每个端点，并将新属性 `CreditScore` 添加到端点。`CreditScore` 属性的值只是一个随机数字。

`console.log()` 语句将事件记录在 CloudWatch Logs 中。

`callback()` 语句将修改后的端点返回到 Amazon Pinpoint。通常，`callback` 参数在 Node.js Lambda 函数中是可选的，但在此上下文中是必需的，因为该函数必须将更新后的端点返回给 Amazon Pinpoint。

函数返回端点的格式必须与事件数据提供的格式相同，该格式是一个将端点 ID 与端点定义关联起来的映射，如以下示例中所示：

```
{
  "eqmj8wpxszeqy/b3vch04sn41yw": {
    "ChannelType": "GCM",
    "Address": "4d5e6f1a2b3c4d5e6f7g8h9i0j1a2b3c",
    "EndpointStatus": "ACTIVE",
    "OptOut": "NONE",
    "Demographic": {
      "Make": "android"
    },
    "EffectiveDate": "2017-11-02T21:26:48.598Z",
    "User": {}
  },
  "idrexqqtn8sbwfex0ouscod0yto": {
    "ChannelType": "APNS",
    "Address": "1a2b3c4d5e6f7g8h9i0j1a2b3c4d5e6f",
```



```

    "EndpointStatus": "ACTIVE",
    "OptOut": "NONE",
    "Demographic": {
      "Make": "apple"
    },
    "EffectiveDate": "2017-11-02T21:26:48.598Z",
    "User": {}
  }
}

```

示例函数会修改并返回在事件数据中收到的 `event.Endpoints` 对象。

( 可选 ) 您可以在返回的端点定义中包含 `TitleOverride` 和 `BodyOverride` 属性。

### Note

当您使用此解决方案发送消息时，对于 `ChannelType` 属性值为以下之一的端点，Amazon Pinpoint 只接受 `TitleOverride` 和 `BodyOverride` 属性：ADM、APNS、APNS\_SANDBOX、APNS\_VOIP、APNS\_VOIP\_SANDBOX、BAIDU、GCM 或 SMS。

对于 `ChannelType` 属性值为 EMAIL 的端点，Amazon Pinpoint 不支持这些属性。

## 分配 Lambda 函数策略

要使用 Lambda 函数处理您的端点，您必须先授权 Amazon Pinpoint 调用您的 Lambda 函数。要授予调用权限，请为该函数分配 Lambda 函数策略。Lambda 函数策略是一种基于资源的权限策略，它指定哪些实体可以使用您的函数以及这些实体可以执行哪些操作。

有关更多信息，请参阅《AWS Lambda 开发人员指南》中的[将基于资源的策略用于 AWS Lambda](#)。

### 示例函数策略

以下策略授予 Amazon Pinpoint 服务主体对特定活动 (*campaign-id*) 使用 `lambda:InvokeFunction` 操作的权限：

```

{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
    "Service": "pinpoint.us-east-1.amazonaws.com"
  }
}

```

```
},
"Action": "lambda:InvokeFunction",
"Resource": "{arn:aws:lambda:us-east-1:account-id:function:function-name}",
"Condition": {
  "StringEquals": {
    "AWS:SourceAccount": "111122223333"
  },
  "ArnLike": {
    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:account-id:apps/application-id/campaigns/campaign-id"
  }
}
}
```

您的函数策略需要包含 `AWS:SourceArn` 密钥的 `Condition` 块。此代码声明允许哪个 Amazon Pinpoint 活动调用该函数。在本示例中，策略仅将权限授予一个活动。`Condition` 块还必须包含一个 `AWS:SourceAccount` 密钥，用于控制哪个 AWS 账户可以调用该操作。

要编写更为通用的策略，请使用多字符匹配通配符 (\*)。例如，您可以使用以下 `Condition` 块来允许特定 Amazon Pinpoint 项目 (*application-id*) 中的任何活动调用该函数：

```
...
"Condition": {
  "StringEquals": {
    "AWS:SourceAccount": "111122223333"
  },
  "ArnLike": {
    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:account-id:apps/application-id/campaigns/*"
  }
}
...
}
```

如果您希望 Lambda 函数成为项目的所有活动使用的默认函数，建议您按上述方法配置策略的 `Condition` 块。有关将 Lambda 函数设置为项目中的所有活动的默认函数的信息，请参阅[将 Lambda 函数分配给活动](#)。

## 授予 Amazon Pinpoint 调用权限

您可以使用 AWS Command Line Interface (AWS CLI) 将权限添加到分配给您的 Lambda 函数的 Lambda 函数策略。要允许 Amazon Pinpoint 为特定活动调用函数，请使用 Lambda [add-permission](#) 命令，如以下示例所示：

```
$ aws lambda add-permission \  
> --function-name function-name \  
> --statement-id sid \  
> --action lambda:InvokeFunction \  
> --principal pinpoint.us-east-1.amazonaws.com \  
> --source-account 111122223333 \  
> --source-arn arn:aws:mobiletargeting:us-east-1:account-id:apps/application-id/  
campaigns/campaign-id
```

您可以使用 AWS CLI 中的 [get-campaigns](#) 命令来查找活动 ID。您也可以使用 [get-apps](#) 命令查找您的应用程序 ID。

运行 Lambda add-permission 命令时，Lambda 返回以下输出：

```
{  
  "Statement": "{\\"Sid\\":\\"sid\\",  
    \\"Effect\\":\\"Allow\\",  
    \\"Principal\\":{\\"Service\\":\\"pinpoint.us-east-1.amazonaws.com\\"},  
    \\"Action\\":\\"lambda:InvokeFunction\\",  
    \\"Resource\\":\\"arn:aws:lambda:us-east-1:111122223333:function:function-name\\",  
    \\"Condition\\":  
      {\\"ArnLike\\":  
        {\\"AWS:SourceArn\\":  
          \\"arn:aws:mobiletargeting:us-east-1:111122223333:apps/application-id/  
campaigns/campaign-id\\"}}  
        {\\"StringEquals\\":  
          {\\"AWS:SourceAccount\\":  
            \\"111122223333\\"}}}}  
}
```

Statement 值是已添加到 Lambda 函数策略的语句的 JSON 字符串版本。

## 将 Lambda 函数分配给活动

您可以将 Lambda 函数分配给单独的 Amazon Pinpoint 活动。或者，将 Lambda 函数设置为某个项目的所有活动（除了单独分配函数的活动之外）默认使用的函数。

要将 Lambda 函数分配给单独的活动，请使用 Amazon Pinpoint API 来创建或更新 [Campaign](#) 对象，并定义其 CampaignHook 属性。要将某个 Lambda 函数设置为某个项目中所有活动的默认函数，请创建或更新该项目的 [Settings](#) 资源并定义其 CampaignHook 对象。

在两种情况下，设置以下 CampaignHook 属性：

- `LambdaFunctionName` – 在发送活动消息之前 Amazon Pinpoint 调用的 Lambda 函数的名称或 ARN。
- `Mode` – 设置为 `FILTER`。使用此模式时，Amazon Pinpoint 会调用该函数并等待它返回修改后的端点。收到端点之后，Amazon Pinpoint 将发送消息。Amazon Pinpoint 最多等待 15 秒钟，如果 15 秒后未收到返回结果，则认为消息传送失败。

借助为活动定义的 `CampaignHook` 设置，Amazon Pinpoint 将在发送活动消息之前调用指定的 Lambda 函数。Amazon Pinpoint 会等待接收该函数返回的修改后的端点。如果 Amazon Pinpoint 收到更新后的端点，它会使用更新后的端点数据继续消息传送。

# 创建活动

Amazon Pinpoint 有助于增强您的应用程序与其用户之间的互动，您可以使用它来创建和管理面向特定用户分段的推送通知活动。

例如，您的活动可以邀请近段时间没有运行您的应用程序的用户再次运行它，或者给近段时间没有进行采购的用户提供特价促销。

活动将定制消息发送给您指定的用户分段。活动可以将消息发送给分段中的所有用户，或者，您可以分配一个保留百分比，这一百分比的用户将收不到消息。

您可以设置活动计划，规定只发送一次消息，或按照某个重复频率（如每周一次）发送消息。为防止用户在不方便的时间收到消息，计划中可以包含不发送任何消息的安静时间。

要测试一下备选活动策略，请将活动设置为 A/B 测试。A/B 测试包括两种或两种以上的消息或计划处理。处理方法是消息或计划的变体。当用户响应活动时，您可以查看活动分析来比较每种处理的效果。

有关更多信息，请参阅[活动](#)。

## 创建标准活动

标准活动根据定义的计划将自定义推送通知发送给指定分段。

### 使用创建广告系列 AWS SDK for Java

以下示例演示了如何使用 AWS SDK for Java 创建活动。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
```

```
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateCampaign {
    public static void main(String[] args) {

        final String usage = ""

            Usage:  <appId> <segmentId>

            Where:
                appId - The ID of the application to create the campaign in.
                segmentId - The ID of the segment to create the campaign from.
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String segmentId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        createPinCampaign(pinpoint, appId, segmentId);
        pinpoint.close();
    }
}
```

```
public static void createPinCampaign(PinpointClient pinpoint, String appId, String
segmentId) {
    CampaignResponse result = createCampaign(pinpoint, appId, segmentId);
    System.out.println("Campaign " + result.name() + " created.");
    System.out.println(result.description());
}

public static CampaignResponse createCampaign(PinpointClient client, String appId,
String segmentID) {

    try {
        Schedule schedule = Schedule.builder()
            .startTime("IMMEDIATE")
            .build();

        Message defaultMessage = Message.builder()
            .action(Action.OPEN_APP)
            .body("My message body.")
            .title("My message title.")
            .build();

        MessageConfiguration messageConfiguration = MessageConfiguration.builder()
            .defaultMessage(defaultMessage)
            .build();

        WriteCampaignRequest request = WriteCampaignRequest.builder()
            .description("My description")
            .schedule(schedule)
            .name("MyCampaign")
            .segmentId(segmentID)
            .messageConfiguration(messageConfiguration)
            .build();

        CreateCampaignResponse result =
client.createCampaign(CreateCampaignRequest.builder()
            .applicationId(appId)
            .writeCampaignRequest(request).build());

        System.out.println("Campaign ID: " + result.campaignResponse().id());
        return result.campaignResponse();

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
    }
}
```

```
        System.exit(1);
    }

    return null;
}
}
```

运行此示例时，IDE 的控制台窗口会显示以下内容：

```
Campaign ID: b1c3de717aea4408a75bb3287a906b46
```

有关完整的 SDK 示例，请参阅上[GitHub](#)的 [CreateCampaign.java](#)。

## 创建 A/B 测试活动

A/B 测试行为与标准活动类似，但允许您为活动消息或计划定义不同的处理。

### 使用创建 A/B 测试广告系列 AWS SDK for Java

以下示例演示了如何使用 AWS SDK for Java 创建 A/B 测试活动。

```
import com.amazonaws.services.pinpoint.AmazonPinpointClient;
import com.amazonaws.services.pinpoint.model.Action;
import com.amazonaws.services.pinpoint.model.CampaignResponse;
import com.amazonaws.services.pinpoint.model.CreateCampaignRequest;
import com.amazonaws.services.pinpoint.model.CreateCampaignResult;
import com.amazonaws.services.pinpoint.model.Message;
import com.amazonaws.services.pinpoint.model.MessageConfiguration;
import com.amazonaws.services.pinpoint.model.Schedule;
import com.amazonaws.services.pinpoint.model.WriteCampaignRequest;
import com.amazonaws.services.pinpoint.model.WriteTreatmentResource;

import java.util.ArrayList;
import java.util.List;

public class PinpointCampaignSample {

    public CampaignResponse createAbCampaign(AmazonPinpointClient client, String appId,
String segmentId) {
        Schedule schedule = new Schedule()
            .withStartTime("IMMEDIATE");
```



```
// Default treatment.
Message defaultMessage = new Message()
    .withAction(Action.OPEN_APP)
    .withBody("My message body.")
    .withTitle("My message title.");

MessageConfiguration messageConfiguration = new MessageConfiguration()
    .withDefaultMessage(defaultMessage);

// Additional treatments
WriteTreatmentResource treatmentResource = new WriteTreatmentResource()
    .withMessageConfiguration(messageConfiguration)
    .withSchedule(schedule)
    .withSizePercent(40)
    .withTreatmentDescription("My treatment description.")
    .withTreatmentName("MyTreatment");

List<WriteTreatmentResource> additionalTreatments = new
ArrayList<WriteTreatmentResource>();
additionalTreatments.add(treatmentResource);

WriteCampaignRequest request = new WriteCampaignRequest()
    .withDescription("My description.")
    .withSchedule(schedule)
    .withSegmentId(segmentId)
    .withName("MyCampaign")
    .withMessageConfiguration(messageConfiguration)
    .withAdditionalTreatments(additionalTreatments)
    .withHoldoutPercent(10); // Hold out of A/B test

CreateCampaignRequest createCampaignRequest = new CreateCampaignRequest()
    .withApplicationId(appId).withWriteCampaignRequest(request);

CreateCampaignResult result = client.createCampaign(createCampaignRequest);

System.out.println("Campaign ID: " + result.getCampaignResponse().getId());

return result.getCampaignResponse();
}
}
```

运行此示例时，IDE 的控制台窗口会显示以下内容：

Campaign ID: b1c3de717aea4408a75bb3287a906b46

## 使用 Amazon Pinpoint SMS 和 Voice API 第 2 版

### Important

Amazon Pinpoint 已更新其用户指南文档。此页面可能不再包含最多的 up-to-date 信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

Amazon Pinpoint 包含一个专为发送短信和语音消息而设计的 API ( 称为 SMS 和 Voice API 第 2 版 )。Amazon Pinpoint API 侧重于通过计划和事件驱动的活动和旅程发送消息，而 SMS 和 Voice API 则提供了直接向个人收件人发送短信和语音消息的新特性和功能。您可以独立于 Amazon Pinpoint 活动和旅程功能使用 SMS 和 Voice API，也可以同时使用这两者来适应不同的使用案例。如果您已经在使用 Amazon Pinpoint 发送短信或语音消息，则说明您的账户已配置为使用此 API。

对于拥有多租户架构的用户 ( 例如独立软件供应商 (ISV) ) 来说，此 API 是一个很好的解决方案。此 API 可以更轻松地确保为不同的租户隔开事件数据、源电话号码和选择退出列表。

当您使用 SMS 和 Voice API 时，我们建议您设置配置集和事件目标。SMS 和 Voice API 不会自动为您发送的消息发出事件数据。设置事件目标可确保您捕获重要的事件数据，例如消息送达和故障事件。

此第 2 版 API 之前有第 1 版 API。第 1 版仍然可用，如果您目前在使用它，可以继续使用。不过，如果您迁移到第 2 版，将获得更多功能，例如创建电话号码池、以编程方式请求新的电话号码以及启用或禁用电话号码的某些功能等。

### Note

有些任务目前只能通过使用 Amazon Pinpoint 控制台来完成。例如，[账户处于短信沙盒中时验证要使用的电话号码](#)，或者，[注册使用 10DLC](#) 等等，这些情形必须使用 Amazon Pinpoint 控制台。

本节提供关于此 API 的信息，并包含如何使用它的示例。您还可以在 [SMS 和 Voice API 第 2 版参考](#) 中找到参考文档。

本节中的主题：

- [SMS 和 Voice API 第 2 版中的概念](#)
- [管理电话号码](#)

- [管理发件人 ID](#)
- [管理池](#)
- [管理选择退出列表](#)
- [管理配置集](#)
- [管理关键字](#)
- [管理事件目标](#)
- [使用 SMS 和 Voice API 发送消息](#)

## SMS 和 Voice API 第 2 版中的概念

SMS 和 Voice API 第 2 版包含几个与发送消息相关的新概念。本节提供这些新术语的额外信息。

### 源身份

从中发送短信或语音消息的电话号码（或短信的发件人 ID）。

有关管理源身份的更多信息，请参阅[管理电话号码](#)。

### 目标电话号码

短信或语音消息发送到的电话号码。

### 池

包含几个用于相同目的的源身份的对象。池可以具有与之关联的选择退出列表。

有关管理池的更多信息，请参阅[管理池](#)。

### 配置集

配置集是在您发送消息时应用的规则集。例如，配置集可以为与消息相关的事件指定目标。当短信事件（例如送达或失败事件）发生时，它们会被路由到与您在发送消息时指定的配置集关联的目标。

有关管理配置集的更多信息，请参阅[管理配置集](#)。

### 事件目标

事件目的地是发送短信和语音事件的地点（例如亚马逊 CloudWatch 日志组、Amazon Kinesis Data Firehose 直播或亚马逊简单通知服务主题）。要使用事件目标，请先创建目标，然后将其与配置集关联。当您发送消息时，您对该 API 的调用可以包括对配置集的引用。

有关管理事件目标的更多信息，请参阅[管理事件目标](#)。

## 选择退出列表

不应向其发送消息的目标身份列表。如果目标身份使用关键字 STOP 回复您的源号码，则他们会自动添加到选择退出列表中。如果您尝试向选择退出列表中的目标号码发送消息，并且选择退出列表与用于发送消息的池相关联，则 Amazon Pinpoint 不会尝试发送该消息。

### Note

如果您为电话号码启用了自我管理的选择退出功能，则当收件人使用关键字 STOP 回复您的消息时，他们不会自动选择退出。

有关选择退出的更多信息，请参阅[管理选择退出列表](#)。

## 关键字

关键字是一个特定的单词或短语，客户可将它发送到您的号码来引发响应，例如信息性消息、选择加入以接收更多消息、特价优惠以及其他促销和事务性消息。当您的号码收到了以某个关键字开头的消息时，Amazon Pinpoint 会使用可自定义的消息予以响应。

有关管理事件目标的更多信息，请参阅[管理关键字](#)。

## 管理电话号码

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

您可以使用 Amazon Pinpoint SMS 和 Voice API 第 2 版向 Amazon Pinpoint 账户请求和放弃电话号码。您还可以使用该 API 来查看账户中的所有电话号码的列表。

本节包含有关使用 AWS CLI 管理电话号码的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。

本节中的主题：

- [申请电话号码](#)

- [修改电话号码功能](#)
- [列出电话号码](#)
- [释放电话号码](#)

## 申请电话号码

您可以使用 [RequestPhoneNumber](#) API 向您的账户中添加新的电话号码。电话号码的可用性和支持的功能因国家/地区而异。

### 请求电话号码

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 request-phone-number \  
> --iso-country-code XX \  
> --message-type TRANSACTIONAL \  
> --number-capabilities VOICE \  
> --number-type LONG_CODE \  
> --pool-id poolId \  
> --deletion-protection-enabled true \  
> --opt-out-list-name optOutListName \  
> --registration-id C0123EX
```

### PowerShell

```
PS C:\> New-SMSVPhoneNumber \  
>> -IsoCountryCode XX \  
>> -MessageType TRANSACTIONAL \  
>> -NumberCapabilities VOICE \  
>> -NumberType LONG_CODE \  
>> -PoolId poolId \  
>> -DeletionProtectionEnabled $true \  
>> -OptOutListName optOutListName \  
>> -RegistrationId C0123EX
```

### Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 request-phone-number ^
```

```
--iso-country-code XX ^  
--message-type TRANSACTIONAL ^  
--number-capabilities VOICE ^  
--number-type LONG_CODE ^  
--pool-id poolId ^  
--deletion-protection-enabled true ^  
--opt-out-list-name optOutListName ^  
--registration-id C0123EX
```

在前面的命令中，进行以下更改：

- 将 *XX* 替换为电话号码所在国家/地区的双字母 ISO-3166 alpha-2 代码（例如，CA 代表加拿大）。
- 如果您想使用电话号码发送促销或营销相关的内容，请将 *TRANSACTIONAL* 替换为 *PROMOTIONAL*。否则，请使用 *TRANSACTIONAL*。
- 如果您想请求用于发送短信的电话号码，请将 *VOICE* 替换为 *SMS*。您还可以通过指定 *SMS VOICE* 来请求一个可用于发送短信和语音消息的电话号码。
- 将 *LONG\_CODE* 替换为您要请求的电话号码类型。可接受的值为 *LONG\_CODE*、*TOLL\_FREE* 和 *TEN\_DLC*。
- 将 *poolId* 替换为您要向其中添加电话号码的池的 ID 或 Amazon 资源名称 (ARN)。此参数为可选的。如果您不想将电话号码添加到池中，请忽略此参数。
- 如果要禁用此电话号码的删除保护，请忽略 *DeletionProtectionEnabled* 参数，或者将其值设置为 *false*。默认情况下，禁用删除保护。如果启用了删除保护，则您无法使用 [ReleasePhoneNumber](#) API 删除电话号码，除非您[更新](#)电话号码的配置以禁用此功能。
- 将 *optOutListName* 替换为要与电话号码关联的选择退出列表的名称或 ARN。此参数为可选的。如果您不想将电话号码与选择退出列表关联，请忽略此参数。
- 如果您要请求一个用于 10DLC 活动的电话号码，请将 *C0123EX* 替换为您要使用的 10DLC 活动的 ID。

#### Note

如果您计划使用 10DLC 电话号码，则必须先注册公司和活动。目前，完成这些注册过程的唯一方法是使用 Amazon Pinpoint 控制台。有关 10DLC 注册的更多信息，请参阅《Amazon Pinpoint 用户指南》中的 [10DLC](#)。

如果号码成功添加到您的账户，则您将看到类似于以下内容的输出：

```
{
  "PhoneNumberArn": "arn:aws:sms-voice:us-east-1:111122223333:phone-number/
phone-615790209ea34aea8da9b729fexample",
  "PhoneNumberId": "phone-615790209ea34aea8da9b729fexample",
  "PhoneNumber": "+12045550123",
  "Status": "PENDING",
  "IsoCountryCode": "CA",
  "MessageType": "TRANSACTIONAL",
  "NumberCapabilities": [
    "SMS"
  ],
  "NumberType": "LONG_CODE",
  "MonthlyLeasingPrice": "1.00",
  "TwoWayEnabled": false,
  "SelfManagedOptOutsEnabled": false,
  "OptOutListName": "Default",
  "DeletionProtectionEnabled": false,
  "CreatedTimestamp": 1645568542.0
}
```

### Note

首次购买电话号码时，Status 属性的值为 PENDING。当电话号码准备好使用时，Status 的值变为 ACTIVE。

如果符合您指定的参数的电话号码不可用，则请求失败并显示错误。

## 修改电话号码功能

请求电话号码后，您可以使用 [UpdatePhoneNumber](#) API 来更改该电话号码的设置或启用额外功能。您可以更改多个电话号码设置，包括与电话号码关联的池和选择退出列表以及删除保护设置。

您可以通过更新电话号码来启用的额外功能的一个例子是双向消息。是否支持双向消息功能因国家/地区而异。有关支持的国家和地区列表，请参阅[支持的国家和地区](#)。

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 update-phone-number \
> --phone-number-id phone-d2b0f5dd4fd14ebdb2a3b9128example \
```



```
> --deletion-protection-enabled true \  
> --opt-out-list-name optOutListName \  
> --self-managed-opt-outs-enabled true \  
> --two-way-enabled true \  
> --two-way-channel-arn arn:aws:sns:us-east-1:111122223333:MyTopic
```

## PowerShell

```
PS C:\> Update-SMSVPhoneNumber \  
>> -PhoneNumberId phone-d2b0f5dd4fd14ebdb2a3b9128example \  
>> -DeletionProtectionEnabled $true \  
>> -OptOutListName optOutListName \  
>> -SelfManagedOptOutsEnabled $true \  
>> -TwoWayEnabled $true \  
>> -TwoWayChannelArn arn:aws:sns:us-east-1:111122223333:MyTopic
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 update-phone-number ^  
--phone-number-id phone-d2b0f5dd4fd14ebdb2a3b9128example ^  
--deletion-protection-enabled true ^  
--opt-out-list-name optOutListName ^  
--self-managed-opt-outs-enabled true ^  
--two-way-enabled true ^  
--two-way-channel-arn arn:aws:sns:us-east-1:111122223333:MyTopic
```

在上述命令中，执行以下操作：

- 将 `phone-d2b0f5dd4fd14ebdb2a3b9128example` 替换为您要更新的电话号码的 `PhoneNumberID` 或 Amazon 资源名称 (ARN)。可以使用 [DescribePhoneNumbers](#) 操作找到这两个值。
- 将 `optOutListName` 替换为要与此电话号码关联的[选择退出列表](#)的名称。
- 如果要禁用删除保护功能，请将 `DeletionProtectionEnabled` 参数的值改为 `false`。
- 如果您想要[自我管理短信选择退出](#)功能，请将 `SelfManagedOptOutsEnabled` 参数的值改为 `false`。
- 如果要禁用此电话号码的[双向短信消息](#)，请将 `TwoWayEnabled` 参数的值改为 `false`。
- 如果您为电话号码启用双向消息功能，则必须指定 Amazon SNS 主题 ARN。将 `arn:aws:sns:us-east-1:111122223333:MyTopic` 替换为您要使用的 Amazon SNS 主题的 ARN。当您收到传入消息时，这些消息将发送到您指定的主题。

PhoneNumberId 参数是此命令唯一必需参数。如果您不想更改相应的设置，则可以忽略任何其他参数。

## 列出电话号码

您可以使用 [DescribePhoneNumbers](#) 获取有关 Amazon Pinpoint 账户中源电话号码的更多信息。

使用 AWS CLI 列出账户中的所有电话号码

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-phone-numbers
```

PowerShell

```
PS C:\> Get-SMSVPhoneNumber
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-phone-numbers
```

此命令的输出包含有关账户中所有电话号码的详细信息。您还可以通过包含 PhoneNumberId 参数来查看有关特定电话号码的信息。

使用 AWS CLI 查看有关特定电话号码的信息

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-phone-numbers \  
> --phone-number-id phone-d2b0f5dd4fd14ebdb2a3b9128example
```

PowerShell

```
PS C:\> Get-SMSVPhoneNumber \  
>> -PhoneNumberId phone-d2b0f5dd4fd14ebdb2a3b9128example
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-phone-numbers ^
--phone-number-id phone-d2b0f5dd4fd14ebdb2a3b9128example
```

在前面的示例中，将 *phone-d2b0f5dd4fd14ebdb2a3b9128example* 替换为您要查看其详细信息的电话号码的 PhoneNumberID 或 Amazon 资源名称 (ARN)。

您也可以使用 `filter` 参数根据您定义的条件筛选电话号码列表。例如，您可以按电话号码所在的国家/地区或其功能（即它是否支持短信和/或语音消息）进行筛选。

使用 AWS CLI 查看经过筛选的电话号码列表

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-phone-numbers \  
> --filters Name=number-capability,Values=SMS \  
> --filters Name=iso-country-code,Values=CA
```

## PowerShell

```
PS C:\> $filter1 = New-Object Amazon.PinpointSMSVoiceV2.Model.PhoneNumberFilter  
PS C:\> $filter2 = New-Object Amazon.PinpointSMSVoiceV2.Model.PhoneNumberFilter  
PS C:\> $filter1.Name = "number-capability"; $filter.Values = "SMS"  
PS C:\> $filter2.Name = "iso-country-code"; $filter.Values = "CA"  
  
PS C:\> Get-SMSVPhoneNumber -Filter $filter1,$filter2
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-phone-numbers ^  
--filters Name=number-capability,Values=SMS ^  
--filters Name=iso-country-code,Values=CA
```

筛选器 Name 值可以为以下任一值：

- `status` – 电话号码的当前状态，例如 `ACTIVE`。
- `iso-country-code` – 电话号码所在国家/地区的双字符 ISO-3166 alpha-2 代码。
- `message-type` – 使用电话号码发送的消息的类型。可能的值为 `TRANSACTIONAL` 或 `PROMOTIONAL`。
- `number-capability` – 电话号码支持的消息渠道。可能的值为 `SMS` 和 `VOICE`。
- `number-type` – 电话号码的类型，例如 `LONG_CODE`、`SHORT_CODE` 或 `TOLL_FREE`。
- `two-way-enabled` – 一个布尔值，表示是否启用了双向短信消息。
- `self-managed-opt-outs-enabled` – 一个布尔值，表示是否启用了自我管理的短信选择退出。
- `opt-out-list-name` – 与电话号码关联的选择退出列表的名称。
- `deletion-protection-enabled` – 一个布尔值，表示是否可以使用 `DeletePhoneNumber` 操作删除电话号码。

## 释放电话号码

您可以使用 [ReleasePhoneNumber](#) API 从您的账户中释放 源电话号码。当您释放一个电话号码时，该电话号码在您的账户中不再可用，也不会再向您收费。

### Important

释放电话号码的操作无法撤消。如果您释放了电话号码，则无法找回相同的电话号码。

## 释放电话号码

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 release-phone-number \  
> --phone-number-id phoneNumberId
```

PowerShell

```
PS C:\> Remove-SMSVPhoneNumber -PhoneNumberId phoneNumberId
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 release-phone-number ^  
--phone-number-id phoneNumberId
```

在前面的命令中，将 *phoneNumberId* 替换为电话号码的唯一 ID 或 Amazon 资源名称 (ARN)。

### Tip

通过完成 [列出电话号码](#) 中的步骤，您可以找到电话号码的 ID 和 ARN。

## 管理发件人 ID

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

您可以使用 Amazon Pinpoint SMS 和 Voice API 第 2 版为配置集指定默认发件人 ID，并将发件人 ID 添加到池中。发件人 ID 是一个字母数字标识符，当收件人收到您的消息时，此标识符会显示在其设备上。并非所有国家/地区都支持发件人 ID。有关 Amazon Pinpoint 支持发件人 ID 的国家/地区列表，请参阅《Amazon Pinpoint 用户指南》中的 [支持的国家和地区](#)。

发件人 ID 仅支持单向消息，也就是说，您的收件人无法回复您使用发件人 ID 发送的消息。在某些国家/地区（例如 [印度](#)），您必须注册您的使用案例和消息模板才能使用发件人 ID。

本节包含有关使用 AWS CLI 管理发件人 ID 的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。

本节中的主题：

- [将发件人 ID 与池关联](#)
- [为配置集设置默认发件人 ID](#)
- [取消发件人 ID 与配置集的关联](#)

## 将发件人 ID 与池关联

您可以使用 [AssociateOriginationIdentity](#) API 将发起身份（即电话号码或发件人 ID）添加到现有池中。

要完成此步骤，必须先创建一个池。有关更多信息，请参阅[创建池](#)。

使用 AWS CLI 将发件人 ID 添加到池中

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 associate-origination-identity \  
> --pool-id poolId \  
> --origination-identity SENDER \  
> --iso-country-code IN
```

PowerShell

```
PS C:\> Register-SMSVOriginationIdentity \  
>> -PoolId poolId \  
>> -OriginationIdentity SENDER \  
>> -IsoCountryCode IN
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 associate-origination-identity ^  
--pool-id poolId ^  
--origination-identity SENDER ^  
--iso-country-code IN
```

在前面的命令中，进行以下更改：

- 将 *poolId* 替换为您要向其中添加发件人 ID 的 ID 或 Amazon 资源名称 (ARN)。
- 将 *SENDER* 替换为您要添加到池中的发件人 ID。
- 将 *IN* 替换为两个字母的 ISO-3166 alpha-2 代码，代表发件人 ID 所在的国家/地区。

## 为配置集设置默认发件人 ID

您可以使用 [SetDefaultSenderId](#) API 为配置集设置默认发件人 ID。

要设置默认发件人 ID，必须先创建一个配置集。有关更多信息，请参阅[创建配置集](#)。

使用 AWS CLI 设置默认发件人 ID

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 set-default-sender-id \  
> --configuration-set-name configurationSetName \  
> --sender-id senderId
```

PowerShell

```
PS C:\> Set-SMSVDefaultSenderId \  
>> -ConfigurationSetName configurationSetName \  
>> -SenderId senderId
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 set-default-sender-id ^  
--configuration-set-name configurationSetName ^  
--sender-id senderId
```

在前面的示例中，进行以下更改：

- 将 *configurationSetName* 替换为要为其定义默认发件人 ID 的配置集的名称。
- 将 *senderId* 替换为您要使用的发件人 ID 的值。

## 取消发件人 ID 与配置集的关联

您可以使用 [DeleteDefaultSenderId](#) API 取消默认发件人 ID 与配置集的关联。

## 使用 AWS CLI 取消关联默认发件人 ID

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-default-sender-id \  
> --configuration-set-name configurationSetName
```

PowerShell

```
PS C:\> Remove-SMSVDefaultSenderId \  
>> -ConfigurationSetName configurationSetName
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-default-sender-id ^  
--configuration-set-name configurationSetName
```

在上述示例中，将 *configurationSetName* 替换为包含默认发件人 ID 的配置集的名称。

## 管理池

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

在 Amazon Pinpoint SMS 和 Voice API 第 2 版中，您可以使用池创建发起电话号码或发件人 ID 组。例如，通过使用池，您可以将已选择退出的目标电话号码列表与特定国家/地区的发起电话号码相关联。这使您可以防止向已选择不接收来自您的消息的用户发送消息。

本节包含有关在 Amazon Pinpoint SMS 和 Voice API 第 2 版中使用 AWS CLI 管理池的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。



本节中的主题：

- [创建池](#)
- [将发起身份添加到池中](#)
- [列出池](#)
- [列出池中的发起身份池](#)
- [删除池](#)

## 创建池

您可以使用 [CreatePool](#) API 来创建新的池。

创建新池时，可以向其中添加发起身份（电话号码或发件人 ID）。稍后，您可以向其中添加更多发起身份。有关将发起身份添加到现有池中的更多信息，请参阅[将发起身份添加到池中](#)。当您使用 `RequestPhoneNumber` API 购买电话号码时，您也可以将电话号码添加到池中。有关更多信息，请参阅[申请电话号码](#)。

### Note

添加到池中的每个发起身份的配置都必须与您在创建池时指定的第一个电话号码的配置一致。例如，如果您创建的池中包含一个启用了双向消息的电话号码，则添加到该池中的其他号码也必须启用双向消息。

### 使用 AWS CLI 创建池

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-pool \  
> --origination-identity originationIdentity \  
> --iso-country-code XX \  
> --message-type TRANSACTIONAL
```

### PowerShell

```
PS C:\> New-SMSVPool `
```

```
>> -OriginationIdentity originationIdentity \  
>> -IsoCountryCode XX \  
>> -MessageType TRANSACTIONAL
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-pool ^  
--origination-identity originationIdentity ^  
--iso-country-code XX ^  
--message-type TRANSACTIONAL
```

在前面的命令中，进行以下更改：

- 将 *originationIdentity* 替换为您要添加到池中的电话号码或发件人 ID 的唯一 ID 或 Amazon 资源名称 (ARN)。

### Tip

您可以使用 [DescribePhoneNumbers](#) 操作找到电话号码的 ID 和 ARN。您可以使用 [DescribeSenderIds](#) 操作找到发件人 ID 的 ID 和 ARN。

- 将 *XX* 替换为池所在国家/地区的 ISO-3166 alpha-2 标识符。
- 如果您打算使用池发送营销或促销信息，请将 *TRANSACTIONAL* 替换为 *PROMOTIONAL*。否则，请使用 *TRANSACTIONAL*。

## 将发起身份添加到池中

您可以使用 [AssociateOriginationIdentity](#) API 将发起身份（即电话号码或发件人 ID）添加到现有池中。

添加到池中的每个发起身份的配置都必须与您在创建池时指定的第一个电话号码的配置一致。例如，如果您创建的池中包含一个启用了双向消息的电话号码，则添加到该池中的其他号码也必须启用双向消息。

使用 AWS CLI 将发起号码添加到池中

- 在命令行输入以下命令：

## Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 associate-origination-identity \  
> --pool-id poolId \  
> --origination-identity originationIdentity \  
> --iso-country-code US
```

## PowerShell

```
PS C:\> Register-SMSV0riginationIdentity `\  
>> -PoolId poolId `\  
>> -OriginationIdentity originationIdentity `\  
>> -IsoCountryCode US
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 associate-origination-identity ^\  
--pool-id poolId ^\  
--origination-identity originationIdentity ^\  
--iso-country-code US
```

在前面的命令中，进行以下更改：

- 将 *poolId* 替换为您要向其中添加发起身份的池的 ID 或 Amazon 资源名称 (ARN)。
- 将 *originationIdentity* 替换为您要添加到池中的电话号码或发件人 ID 的唯一 ID 或 Amazon 资源名称 (ARN)。

### Tip

您可以使用 [DescribePhoneNumbers](#) 操作找到电话号码的 ID 和 ARN。您可以使用 [DescribeSenderIds](#) 操作找到发件人 ID 的 ID 和 ARN。

- 将 *+12065550142* 替换为要添加到池中的发起身份。此值可以是短代码、电话号码或发件人 ID。
- 将 *US* 替换为两个字母的 ISO-3166 alpha-2 代码，代表发起身份所在的国家/地区。

## 列出池

您可以使用 [DescribePools](#) API 查看有关现有池的信息。

此操作可以提供您的 Amazon Pinpoint 账户中所有池的完整列表、有关特定池的信息或根据您定义的标准筛选出来的池列表。

使用 AWS CLI 检索所有池的列表

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-pools
```

PowerShell

```
PS C:\> Get-SMSVPool
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-pools
```

要查找有关特定池的信息，请使用 PoolId 参数。

使用 AWS CLI 获取有关特定池的信息

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-pools \  
> --pool-id poolId
```

PowerShell

```
PS C:\> Get-SMSVPool -PoolId poolId
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-pools ^  
--pool-id poolId
```

要查看经过筛选的池列表，请使用 `Filters` 参数。可以使用下列筛选器值：

- `status` – 池的当前状态，例如 `ACTIVE`。
- `message-type` – 使用池发送的消息的类型。可能的值为 `TRANSACTIONAL` 或 `PROMOTIONAL`。
- `two-way-enabled` – 一个布尔值，表示是否为池中的号码启用了双向短信消息。
- `self-managed-opt-outs-enabled` – 一个布尔值，表示是否为池中的号码启用了自我管理的短信选择退出。
- `opt-out-list-name` – 与池关联的选择退出列表的名称。
- `shared-routes-enabled` – 一个布尔值，表示是否为该池启用了共享路由。
- `deletion-protection-enabled` – 一个布尔值，表示是否可以使用 `DeletePhoneNumber` 操作删除电话号码。

例如，如果要查看支持双向消息的事务性消息的池列表，请在命令行中输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-pools \  
> --filters Name=message-type,Values=TRANSACTIONAL \  
> --filters Name=two-way-enabled,Values=true
```

PowerShell

```
PS C:\> $filter1 = New-Object Amazon.PinpointSMSVoiceV2.Model.PoolFilter  
PS C:\> $filter2 = New-Object Amazon.PinpointSMSVoiceV2.Model.PoolFilter  
PS C:\> $filter1.Name = "message-type"; $filter1.Values = "Transactional"  
PS C:\> $filter2.Name = "two-way-enabled"; $filter2.Values = "True"  
  
PS C:\> Get-SMSVPool -Filter $filter1, $filter2
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-pools ^
```

```
--filters Name=message-type,Values=TRANSACTIONAL ^  
--filters Name=two-way-enabled,Values=true
```

## 列出池中的发起身份池

您可以使用 [ListPoolOriginationIdentities](#) API 查看有关已添加到特定池中的所有发起身份的信息。

使用 AWS CLI 查看池中的发起 ID 列表

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 list-pool-origination-identities \  
> --pool-id pool-78ec067f62f94d57bd3bab991example
```

PowerShell

```
PS C:\> Get-SMSVPoolOriginationIdentityList \  
>> -PoolId pool-78ec067f62f94d57bd3bab991example
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-pools ^  
--pool-id pool-78ec067f62f94d57bd3bab991example
```

## 删除池

您可以使用 [DeletePool](#) API 来删除池。

当您删除池时，Amazon Pinpoint 会首先取消所有发起身份与该池的关联，然后删除池本身。但是，原先与该池关联的发起身份仍保留在您的 Amazon Pinpoint 账户中。

使用 AWS CLI 删除池

- 要删除池，在命令行处，键入以下命令：

## Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-pool \  
> --pool-id pool-78ec067f62f94d57bd3bab991example
```

## PowerShell

```
PS C:\> Remove-SMSVPool -PoolId pool-78ec067f62f94d57bd3bab991example
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-pool ^  
--pool-id pool-78ec067f62f94d57bd3bab991example
```

在前面的命令中，将 *pool-78ec067f62f94d57bd3bab991example* 替换为池的唯一 ID 或 Amazon 资源名称 (ARN)。可以使用 [DescribePools](#) 操作找到这两个值。

## 管理选择退出列表

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

选择退出列表 是不应向其发送消息的目标身份列表。当您发送短信时，如果目标身份使用关键字 STOP 回复您的源号码，则除非您启用了自我管理的选择退出选项，否则目标身份会自动添加到选择退出列表中。如果您尝试向选择退出列表中的目标号码发送消息，并且选择退出列表与用于发送消息的池相关联，则 Amazon Pinpoint 不会尝试发送该消息。

本节包含有关在 Amazon Pinpoint SMS 和 Voice API 第 2 版中使用 AWS CLI 管理选择退出列表的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。

本节中的主题：

- [创建选择退出列表](#)
- [列出账户中的选择退出列表](#)
- [将目标号码添加到选择退出列表中](#)
- [从选择退出列表中删除目标号码](#)

## 创建选择退出列表

您可以使用 [CreateOptOutList](#) API 来创建新的选择退出列表。创建选择退出列表后，您可以[将目标身份添加到选择退出列表中](#)。

使用 AWS CLI 创建选择退出列表

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-opt-out-list \  
> --opt-out-list-name optOutListName
```

PowerShell

```
PS C:\> New-SMSV0ptOutList -OptOutListName optOutListName
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-opt-out-list ^  
--opt-out-list-name optOutListName
```

在前面的示例中，将 *optOutListName* 替换为一个便于识别选择退出列表的名称。

AWS CLI 返回有关选择退出列表的以下信息：

```
{  
  "OptOutListArn": "arn:aws:sms-voice:us-east-1:111122223333:opt-out-list/  
optOutListName",  
  "OptOutListName": "optOutListName",  
  "CreatedTimestamp": 1645568542.0
```



```
}
```

## 列出账户中的选择退出列表

您可以使用 [DescribeOptOutLists](#) API 查看有关 Amazon Pinpoint 账户中选择退出列表的信息。

使用 AWS CLI 查看有关所有选择退出列表的信息

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-opt-out-lists
```

PowerShell

```
PS C:\> Get-SMSVOptOutList
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-opt-out-lists
```

您还可以使用 `OptOutListNames` 参数查看有关特定选择退出列表的信息。

使用 AWS CLI 查看有关特定选择退出列表的信息

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-opt-out-lists \  
> --opt-out-list-names optOutListName
```

PowerShell

```
PS C:\> Get-SMSVOptOutList -OptOutListName optOutListName
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-opt-out-lists ^  
--opt-out-list-names optOutListName
```

在前面的命令中，将 *optOutListName* 替换为您要查看其详细信息的选择退出列表的名称或 Amazon 资源名称 (ARN)。您可以通过用空格分隔每个列表名称来指定多个选择退出列表。

AWS CLI 返回有关您的账户中所有选择退出列表的以下信息。

## 将目标号码添加到选择退出列表中

您可以使用 [PutOptedOutNumber](#) API 将目标电话号码添加到选择退出列表中。

使用 AWS CLI 将电话号码添加到选择退出列表中

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 put-opted-out-number \  
> --opt-out-list-name optOutListName \  
> --opted-out-number +12065550123
```

PowerShell

```
PS C:\> Set-SMSVOptedOutNumber \  
>> -OptOutListName optOutListName \  
>> -OptedOutNumber +12065550123
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 put-opted-out-number ^  
--opt-out-list-name optOutListName ^  
--opted-out-number +12065550123
```

在前面的示例中，进行以下更改：

- 将 `optOutListName` 替换为您要向其中添加目标身份的选择退出列表的名称或 Amazon 资源名称 (ARN)。
- 将 `+12065550123` 替换为要添加到选择退出列表中的电话号码。电话号码必须使用 E.164 格式。

AWS CLI 返回有关选择退出列表的以下信息：

```
{
  "OptOutListArn": "arn:aws:sms-voice:us-east-1:111122223333:opt-out-list/
optOutListName",
  "OptOutListName": "optOutListName",
  "OptedOutNumber": "+12065550123",
  "OptedOutTimestamp": 1645568542.0,
  "EndUserOptedOut": false
}
```

## 从选择退出列表中删除目标号码

您可以使用 [DeleteOptedOutNumber](#) API 从选择退出列表中删除目标电话号码。

### Note

只能每隔 30 天从选择退出列表中删除目标电话号码。

使用 AWS CLI 从选择退出列表中删除电话号码

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-opted-out-number \  
> --opt-out-list-name optOutListName \  
> --opted-out-number +12065550123
```

PowerShell

```
PS C:\> Remove-SMSVOptedOutNumber `
```

```
>> -OptOutListName optOutListName \  
>> -OptedOutNumber +12065550123
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-opted-out-number ^  
--opt-out-list-name optOutListName ^  
--opted-out-number +12065550123
```

在前面的示例中，进行以下更改：

- 将 *optOutListName* 替换为您要从中删除目标身份的选择退出列表的名称或 Amazon 资源名称 (ARN)。
- 将 *+12065550123* 替换为要从选择退出列表中删除的电话号码。电话号码必须使用 E.164 格式。

AWS CLI 返回有关选择退出列表的以下信息：

```
{  
  "OptOutListArn": "arn:aws:sms-voice:us-east-1:111122223333:opt-out-list/  
optOutListName",  
  "OptOutListName": "optOutListName",  
  "OptedOutNumber": "+12065550123",  
  "OptedOutTimestamp": 1645568542.0,  
  "EndUserOptedOut": false  
}
```

## 管理配置集

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

配置集是在您发送消息时应用的规则集。例如，配置集可以为与消息相关的事件指定目标。当短信事件（例如送达或失败事件）发生时，它们会被路由到与您在发送消息时指定的配置集关联的目标。虽然在您发送消息时不要求使用配置集，但我们建议您使用。如果您未指定带有事件目标的配置集，则该 API 不会发出事件记录。这些事件记录很有用，可用于确定您发送了多少消息、为每条消息支付了多少费用以及收件人是否收到消息，等等。

本节包含有关在 Amazon Pinpoint SMS 和 Voice API 第 2 版中使用 AWS CLI 来管理配置集的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。

本节中的主题：

- [创建配置集](#)
- [列出配置集](#)
- [删除配置集](#)

## 创建配置集

您可以使用 [CreateConfigurationSet](#) API 创建新的配置集。创建配置集后，可以将事件目标与其关联，请参阅[管理事件目标](#)。

要创建配置集，请在 AWS CLI 中运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-configuration-set \  
> --configuration-set-name configurationSet
```

PowerShell

```
PS C:\> New-SMSVConfigurationSet `\  
>> -ConfigurationSetName configurationSet
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-configuration-set ^\  
--configuration-set-name configurationSet
```

在前面的命令中，将 *configurationSet* 替换为要创建的配置集的名称。

您可以通过指定可选 `tags` 参数向配置集添加标签，如以下示例所示：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-configuration-set \  
> --configuration-set-name ConfigurationSet \  
> --tags Key=key1,Value=value1 Key=key2,Value=value2
```

PowerShell

```
PS C:\> $tag = New-Object Amazon.PinpointSMSVoiceV2.Model.Tag  
PS C:\> $tag.Key = "key1"  
PS C:\> $tag.Value = "value1"  
  
PS C:\> New-SMSVConfigurationSet `\  
>> -ConfigurationSetName configurationSet `\  
>> -Tag $tag
```

Windows Command Prompt

```
C:\>aws pinpoint-sms-voice-v2 create-configuration-set ^  
--configuration-set-name ConfigurationSet ^  
--tags Key=key1,Value=value1 Key=key2,Value=value2
```

### Note

现在，您可以将事件目标与配置集相关联，请参阅[管理事件目标](#)。

## 列出配置集

您可以使用 [DescribeConfigurationSets](#) API 来查看您的 Amazon Pinpoint 账户中配置集的信息。

使用 AWS CLI 查看您的账户中配置集的列表

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-configuration-sets
```

## PowerShell

```
PS C:\> Get-SMSVConfigurationSet
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-configuration-sets
```

如果要查看特定配置集或一组配置集的详细信息，请使用 `ConfigurationSetNames` 参数。

使用 AWS CLI 查看特定配置集的相关信息

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-configuration-sets \  
> --configuration-set-names configurationSet
```

## PowerShell

```
PS C:\> Get-SMSVConfigurationSet \  
>> -ConfigurationSetName configurationSet
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-configuration-sets ^  
--configuration-set-names configurationSet
```

在前面的命令中，将 *configurationSet* 替换为您要查找其详细信息的配置集的名称。您可以通过用空格分隔每个配置集的名称来指定多个配置集。

## 删除配置集

您可以使用 [DeleteConfigurationSet](#) API 来删除配置集。

使用 AWS CLI 删除配置集：

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-configuration-set \  
> --configuration-set-name configurationSet
```

PowerShell

```
PS C:\> Remove-SMSVConfigurationSet \  
>> -ConfigurationSetName configurationSet
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-configuration-set ^  
--configuration-set-name configurationSet
```

在前面的命令中，将 *ConfigSet* 替换为要删除的配置集的名称。

## 管理关键字

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

关键字 是一个特定的单词或短语，客户可将它发送到您的号码来引发响应，例如信息性消息、选择加入以接收更多消息、特价优惠以及其他促销和事务性消息。当您的号码收到了以某个关键字开头的消息时，Amazon Pinpoint 会使用可自定义的消息予以响应。

本节包含有关在 Amazon Pinpoint SMS 和 Voice API 第 2 版中使用 AWS CLI 管理关键字的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。



## 主题

- [创建关键字](#)
- [列出关键字](#)
- [删除关键字](#)

## 创建关键字

您可以使用 [PutKeyword](#) API 来创建新的关键字。

要创建关键字，在 AWS CLI 中运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 put-keyword \  
> --origination-identity OriginationIdentity \  
> --keyword Keyword \  
> --keyword-message KeywordMessage \  
> --keyword-action KeywordAction
```

PowerShell

```
PS C:\> Put-Keyword `\  
>> -Origination-Identity OriginationIdentity `\  
>> -keyword Keyword `\  
>> -keyword-message KeywordMessage `\  
>> -keyword-action KeywordAction
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 put-keyword ^\  
--origination-identity OriginationIdentity ^\  
--keyword Keyword ^\  
--keyword-message KeywordMessage ^\  
--keyword-action KeywordAction
```

在前面的命令中，进行以下更改：

- 将 *OriginationIdentity* 替换为您要向其添加关键字的电话号码或发件人 ID 的唯一 ID 或 Amazon 资源名称 (ARN)。

- 将 *Keyword* 替换为新的关键字。
- 将 *KeywordMessage* 替换为在响应关键字时使用的消息。
- 将 *KeywordAction* 替换为收到关键字时要执行的操作 (AUTOMATIC\_RESPONSE、OPT\_OUT、OPT\_IN)。

## 列出关键字

您可以使用 [DescribeKeywords](#) API 查看有关与源身份关联的关键字的信息。

要使用 AWS CLI 查看关键字列表，请在命令行中键入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 describe-keywords \  
> --origination-identity OriginationIdentity
```

PowerShell

```
PS C:\> Get-Keywords `\  
>> -Origination-Identity OriginationIdentity
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 describe-keywords ^\  
--origination-identity OriginationIdentity
```

将 *OriginationIdentity* 替换为您想要从中获取关键字列表的电话号码或发件人 ID 的唯一 ID 或 Amazon 资源名称 (ARN)。

## 删除关键字

您可以使用 [DeleteKeyword](#) API 来删除关键字。

在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-keyword \  
> --origination-identity OriginationIdentity \  
>
```

```
> --keyword Keyword
```

## PowerShell

```
PS C:\> Remove-Keyword `
>> -origination-identity OriginationIdentity `
>> -keyword Keyword
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-keyword ^
--origination-identity OriginationIdentity ^
--keyword Keyword
```

在前面的命令中，进行以下更改：

- 将 *OriginationIdentity* 替换为您要从中删除关键字的电话号码或发件人 ID 的唯一 ID 或 Amazon 资源名称 (ARN)。
- 将 *Keyword* 替换为要删除的关键字。

## 管理事件目标

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

在 Amazon Pinpoint SMS 和 Voice API 第 2 版中，事件目标是短信和语音事件发送到的地点，例如 CloudWatch Logs 组、Kinesis Data Firehose 流或 Amazon SNS 主题。要使用事件目标，请先创建目标，然后将其与[配置集](#)关联。您可以将最多 5 个事件目标与一个配置集相关联。当您发送消息时，您对该 API 的调用可以包括对配置集的引用。

本节包含有关在 Amazon Pinpoint SMS 和 Voice API 第 2 版中使用 AWS CLI 管理事件目标的信息。本节中的过程假定您已配置 AWS CLI。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的 [AWS CLI 入门](#)。

## 先决条件

您需要已经设置好配置集才能将事件目标与之关联，请参阅[创建配置集](#)。

本节中的主题：

- [事件类型](#)
- [创建事件目标](#)
- [更新事件目标](#)
- [删除事件目标](#)
- [创建和更新 CloudWatch Logs 事件目标](#)
- [创建和更新 Kinesis Data Firehose 事件目标](#)
- [创建和更新 Amazon SNS 事件目标](#)

## 事件类型

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

使用事件目标的最简单方法是将所有短信和语音事件发送到一个目标。但是，您可以配置事件目标，以将特定类型的事件发送到不同目标。例如，您可以将所有与送达相关的事件发送到 Amazon S3 存储桶进行存储，而将所有失败事件发送到 Amazon Pinpoint 主题，以便在发生故障时您收到通知。您还可以将短信事件和语音事件发送到不同的地点。

您可以配置事件目标以发送以下类型的事件：

- ALL – 将所有短信和语音事件发送到指定目标。
- TEXT\_ALL – 将所有短信事件发送到指定目标。
- VOICE\_ALL – 将所有语音事件发送到指定目标。
- TEXT\_DELIVERED – 将所有短信送达事件发送到指定目标。
- TEXT\_SUCCESSFUL – 将所有短信成功事件发送到指定目标。当消息被收件人的运营商接受时，就会发生成功事件。

- **TEXT\_QUEUED** – 将所有排队的短信事件发送到指定目标。当消息排队等待传送但尚未送达时，就会发生排队事件。
- **TEXT\_PENDING** – 将所有待处理短信事件发送到指定目标。当消息正在传中但尚未送达（或未能送达）时，就会发生待处理事件。
- **TEXT\_BLOCKED** – 将所有短信屏蔽事件发送到指定目标。当收件人的设备或运营商屏蔽发送给该收件人的消息时，就会发生屏蔽事件。
- **TEXT\_TTL\_EXPIRED** – 将所有短信 TTL 过期事件发送到指定目标。当传送消息所需的时间超过您在发送消息时指定的 TTL 值时，就会发生 TTL 过期事件。
- **TEXT\_CARRIER\_UNREACHABLE** – 将短信的所有运营商无法联系事件发送到指定目标。当消息收件人的运营商网络发生暂时性错误时，就会发生无法联系运营商事件。
- **TEXT\_INVALID** – 将所有短信无效事件发送到指定目标。当目标电话号码无效时，就会发生无效事件。
- **TEXT\_INVALID\_MESSAGE** – 将短信的所有无效消息事件发送到指定目标。当短信的正文无效且无法传送时，就会发生无效消息事件。
- **TEXT\_CARRIER\_BLOCKED** – 将所有运营商屏蔽短信事件发送到指定目标。当收件人的运营商阻止消息传送时，就会发生运营商屏蔽事件。当运营商将消息识别为恶意消息（例如，其中包含与钓鱼骗术相关的信息）或滥用消息（例如，消息被怀疑是未经请求的或违禁的）时，就会发生这种情况。
- **TEXT\_UNREACHABLE** – 将短信的所有无法联系事件发送到指定目标。当收件人的设备不可用时，就会发生无法联系事件。如果设备未连接到移动网络或已关机，则可能会发生这种情况。
- **TEXT\_SPAM** – 将短信的所有垃圾消息事件发送到指定目标。当收件人的运营商识别出消息中包含未经请求的商业内容并阻止传送消息时，就会发生垃圾消息事件。
- **TEXT\_UNKNOWN** – 将所有未知短信事件发送到指定目标。当消息由于其他事件类型未涵盖的原因而未能传送时，就会发生未知事件。未知错误可能是暂时的，也可能是永久性的。
- **VOICE\_COMPLETED** – 将所有已完成的语音消息事件发送到指定目标。向接收者播放音频消息时，就会发生已完成事件。此状态并不一定意味着消息已送达人类收件人。例如，它可能表示留言已传送到语音信箱系统。
- **VOICE\_ANSWERED** – 将所有已应答的语音消息事件发送到指定目标。当收件人接听电话时，就会发生已应答事件。
- **VOICE\_INITIATED** – 每当有语音消息发起时，都将事件发送到指定目标。
- **VOICE\_TTL\_EXPIRED** – 将所有语音 TTL 过期事件发送到指定目标。当传送消息所需的时间超过您在发送消息时指定的 TTL 值时，就会发生 TTL 过期事件。
- **VOICE\_BUSY** – 将语音消息的所有忙碌事件发送到指定目标。当收件人的电话线占线时，就会发生忙碌事件。

- VOICE\_NO\_ANSWER – 将语音消息的所有无应答事件发送到指定目标。拨打了电话但接收者（或其语音信箱系统）从不应答时，就会发生无应答事件。
- VOICE\_RINGING – 将语音消息的所有响铃事件发送到指定目标。拨打了电话但在接收者应答之前，就会发生响铃事件。
- VOICE\_FAILED – 将所有语音消息失败事件发送到指定目标。当消息传送失败时，就会发生失败事件。

## 创建事件目标

创建事件目标的过程因事件目标的类型而异。

- 有关创建向 CloudWatch Logs 组发送事件的事件目标的信息，请参阅[the section called “创建 CloudWatch Logs 事件目标”](#)。
- 有关创建向 Kinesis Data Firehose 流发送事件的事件目标的信息，请参阅[the section called “创建 Kinesis Data Firehose 事件目标”](#)。
- 有关创建向 Amazon SNS 主题发送事件的事件目标的信息，请参阅[the section called “创建 Amazon SNS 事件目标”](#)。

### Note

Amazon Pinpoint SMS 和 Voice API 第 2 版不支持跨账户事件目标。

## 更新事件目标

同样，更新事件目标的过程因事件目标的类型而异。

- 有关更新向 CloudWatch Logs 组发送事件的事件目标的信息，请参阅[the section called “更新 CloudWatch Logs 事件目标”](#)。
- 有关更新向 Kinesis Data Firehose 流发送事件的事件目标的信息，请参阅[the section called “更新 Kinesis Data Firehose 事件目标”](#)。
- 有关更新向 Amazon SNS 主题发送事件的事件目标的信息，请参阅[the section called “更新 Amazon SNS 事件目标”](#)。

## 删除事件目标

您可以使用 [DeleteEventDestination](#) API 来删除事件目标。

无论您要删除的事件目标的类型如何，删除事件目标的过程都是一样的。

### 删除配置集

- 在命令行输入以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 delete-event-destination \  
> --event-destination-name eventDestinationName \  
> --configuration-set-name configurationSetName
```

### PowerShell

```
PS C:\> Remove-SMSVEventDestination \  
>> -EventDestinationName eventDestinationName \  
>> -ConfigurationSetName configurationSetName
```

### Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 delete-event-destination ^  
--event-destination-name eventDestinationName ^  
--configuration-set-name configurationSetName
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为要删除的事件目标的名称或 Amazon 资源名称 (ARN)。
- 将 *configurationSetName* 替换为与事件目标关联的配置集的名称或 ARN。

## 创建和更新 CloudWatch Logs 事件目标

### ⚠ Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

Amazon CloudWatch Logs 是一项 AWS 服务，可以用来监控、存储和访问日志文件。在您创建 CloudWatch Logs 事件目标时，Amazon Pinpoint 会将您在事件目标中指定的事件类型发送到 CloudWatch Logs 组。要了解有关 CloudWatch Logs 的更多信息，请参阅 [Amazon CloudWatch Logs 用户指南](#)。

本节中的过程假定您已安装和配置 AWS Command Line Interface。有关设置 AWS CLI 的更多信息，请参阅《[AWS Command Line Interface 用户指南](#)》。

本节中的主题：

- [创建 CloudWatch Logs 事件目标](#)
- [更新 CloudWatch Logs 事件目标](#)

### 创建 CloudWatch Logs 事件目标

在创建 CloudWatch Logs 事件目标之前，必须先创建一个 CloudWatch Logs 组。有关创建日志组的更多信息，请参阅《Amazon CloudWatch Logs 用户指南》中的[使用日志组和日志流](#)。

您必须创建一个 IAM 角色来允许 SMS 和 Voice API 写入日志组。下一节包含有关该角色要求的信息。

您还需要已经设置了与事件目标关联的配置集，请参阅[创建配置集](#)。

用于 CloudWatch Logs 的 IAM 策略

使用以下示例创建用于向 CloudWatch Logs 组发送事件的策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```



```

    "Action": [
      "logs:CreateLogStream",
      "logs:DescribeLogStreams",
      "logs:PutLogEvents"
    ],
    "Resource": [
      "arn:aws:logs:us-east-1:111122223333:log-group:*"
    ]
  }
]
}

```

有关 IAM 策略的更多信息，请参阅《IAM 用户指南》中的 [IAM 中的策略与权限](#)。

下面的示例语句使用 `SourceAccount` 和 `SourceArn` 条件（可选但建议）来确保只有 Amazon Pinpoint 所有者账户才能访问配置集。在此示例中，将 `accountId` 替换为您的 AWS 账户 id，将 `region` 替换为 AWS 区域名称，将 `ConfigSetName` 替换为配置集的名称。

创建策略之后，创建一个新的 IAM 角色，然后将策略附加给该角色。创建角色时，还要向其添加以下信任策略：

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "sms-voice.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "accountId"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:sms-voice:region:accountId:configuration-  
set/ConfigSetName"
      }
    }
  }
}

```

有关创建 IAM 角色的更多信息，请参阅《IAM 用户指南》中的 [创建 IAM 角色](#)。

## 创建事件目标

创建 IAM 角色和 CloudWatch Logs 组后，您可以创建事件目标。

您可以使用 [CreateEventDestination](#) API 来创建事件目标。

### 使用创CloudWatch Logs 事件目标 AWS CLI

- 在命令行处，运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-event-destination \  
> --event-destination-name eventDestinationName \  
> --configuration-set-name configurationSet \  
> --matching-event-types ALL \  
> --cloud-watch-logs-destination  
  IamRoleArn=arn:aws:iam::111122223333:role/  
CWLSMSRole,LogGroupArn=arn:aws:logs:us-east-1:111122223333:log-  
group:MyCWLLogGroup
```

PowerShell

```
PS C:\> New-SMSVEventDestination \  
>> -EventDestinationName eventDestinationName \  
>> -ConfigurationSetName configurationSet \  
>> -MatchingEventType ALL \  
>> -CloudWatchLogsDestination_IamRoleArn  
  arn:aws:iam::111122223333:role/CWLSMSRole \  
>> -CloudWatchLogsDestination_LogGroupArn arn:aws:logs:us-  
east-1:111122223333:log-group:MyCWLLogGroup
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-event-destination ^  
  --event-destination-name eventDestinationName ^  
  --configuration-set-name configurationSet ^  
  --matching-event-types ALL ^  
  --cloud-watch-logs-destination  
  IamRoleArn=arn:aws:iam::111122223333:role/  
CWLSMSRole,LogGroupArn=arn:aws:logs:us-east-1:111122223333:log-  
group:MyCWLLogGroup
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为描述事件目标的名称。
- 将 *configurationSet* 替换为您要与事件目标关联的配置集的名称。
- 将 *ALL* 替换为 [事件类型](#) 中列出的事件类型之一。
- 将 *IamRoleArn* 的值替换为具有 [用于 CloudWatch Logs 的 IAM 策略](#) 中所述策略的 IAM 角色的 Amazon 资源名称 (ARN)。
- 将 *LogGroupArn* 的值替换为要向其发送事件的 CloudWatch Logs 组的 ARN。

## 更新 CloudWatch Logs 事件目标

您可以使用 [UpdateEventDestination](#) API 更新事件目标。

更新 CloudWatch Logs 事件目标的过程与创建事件目标的过程类似。

在 AWS CLI 中更新事件目标

- 在命令行处，运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 update-event-destination \  
> --event-destination-name eventDestinationName \  
> --configuration-set-name configurationSet \  
> --matching-event types eventTypes \  
> --cloud-watch-logs-destination  
  IamRoleArn=arn:aws:iam::111122223333:role/  
CWLSMSRole,LogGroupArn=arn:aws:logs:us-east-1:111122223333:log-  
group:MyCWLLogGroup
```

PowerShell

```
PS C:\> aws pinpoint-sms-voice-v2 update-event-destination \  
>> -EventDestinationName eventDestinationName \  
>> -ConfigurationSetName configurationSet \  
>> -MatchingEventType eventTypes \  
>> -CloudWatchLogsDestination_IamRoleArn  
  arn:aws:iam::111122223333:role/CWLSMSRole \  
>>
```

```
>> -CloudWatchLogsDestination_LogGroupArn arn:aws:logs:us-east-1:111122223333:log-group:MyCWLLogGroup
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 update-event-destination ^  
--event-destination-name eventDestinationName ^  
--configuration-set-name configurationSet ^  
--matching-event types eventTypes ^  
--cloud-watch-logs-destination  
IamRoleArn=arn:aws:iam::111122223333:role/  
CWLSMSRole,LogGroupArn=arn:aws:logs:us-east-1:111122223333:log-  
group:MyCWLLogGroup
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为要修改的事件目标名称。
- 将 *configurationSet* 替换为您要与事件目标关联的配置集的名称。您可以将事件目标与另外的配置集相关联。
- 将 *eventTypes* 替换为[事件类型](#)中列出的事件类型之一。
- 将 IamRoleArn 的值替换为具有[用于 CloudWatch Logs 的 IAM 策略](#)中所述策略的 IAM 角色的 Amazon 资源名称 (ARN)。
- 将 LogGroupArn 的值替换为要向其发送事件的 CloudWatch Logs 组的 ARN。

## 创建和更新 Kinesis Data Firehose 事件目标

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

Amazon Kinesis Data Firehose 是一项完全托管的服务，用于将实时流数据传送到多种类型的目标。Kinesis Data Firehose 是 Kinesis 流数据平台的一部分。要了解有关 Kinesis Data Firehose 的更多信息，请参阅 [Amazon Kinesis Data Firehose 开发人员指南](#)。

本节中的过程假定您已安装和配置 AWS Command Line Interface。有关设置 AWS CLI 的更多信息，请参阅《[AWS Command Line Interface 用户指南](#)》。

本节中的主题：

- [创建 Kinesis Data Firehose 事件目标](#)
- [更新 Kinesis Data Firehose 事件目标](#)

## 创建 Kinesis Data Firehose 事件目标

要创建 Kinesis Data Firehose 事件目标，必须先创建 Kinesis Data Firehose 流。有关更多信息，请参阅《Amazon Kinesis Data Firehose 开发人员指南》中的[创建 Amazon Kinesis Data Firehose 传输流](#)。

您必须创建一个 IAM 角色来允许 SMS 和 Voice API 向流发送数据。下一节包含有关该角色要求的信息。

您还需要已经设置了与事件目标关联的配置集，请参阅[创建配置集](#)。

### Kinesis Data Firehose 的 IAM 策略

使用以下示例创建用于向 Kinesis Data Firehose 流发送事件的策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "firehose:PutRecord",
      "Resource": "arn:aws:firehose:us-east-1:111122223333:deliverystream/*"
    }
  ]
}
```

有关 IAM 策略的更多信息，请参阅《IAM 用户指南》中的 [IAM 中的策略与权限](#)。

下面的示例语句使用 SourceAccount 和 SourceArn 条件（可选但建议）来确保只有 Amazon Pinpoint 所有者账户才能访问配置集。在此示例中，将 *accountId* 替换为您的 AWS 账户 id，将 *region* 替换为 AWS 区域名称，将 *ConfigSetName* 替换为配置集的名称。

创建策略之后，创建一个新的 IAM 角色，然后将策略附加给该角色。创建角色时，还要向其添加以下信任策略：

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "sms-voice.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "accountId"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:sms-voice:region:accountId:configuration-
set/ConfigSetName"
      }
    }
  }
}
```

有关创建 IAM 角色的更多信息，请参阅《IAM 用户指南》中的[创建 IAM 角色](#)。

## 创建事件目标

创建 IAM 角色和 Kinesis Data Firehose 传输流后，即可以创建事件目标。

您可以使用 [CreateEventDestination](#) API 来创建事件目标。

## 使用 AWS CLI 创建 Kinesis Data Firehose 事件目标

- 在命令行处，运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-event-destination \
> --event-destination-name eventDestinationName \
> --configuration-set-name configurationSet \
> --matching-event-types eventTypes \
> --kinesis-firehose-destination
  IamRoleArn=arn:aws:iam::111122223333:role/
AKFSMSRole,DeliveryStreamArn=arn:aws:firehose:us-
east-1:111122223333:deliverystream/MyDeliveryStream
```

## PowerShell

```
PS C:\> New-SMSVEventDestination `
>> -EventDestinationName eventDestinationName `
>> -ConfigurationSetName configurationSet `
>> -MatchingEventType eventTypes `
>> -KinesisFirehoseDestination_IamRoleArn_IamRoleArn
    arn:aws:iam::111122223333:role/AKFSMSRole `
>> -KinesisFirehoseDestination_DeliveryStreamArn arn:aws:firehose:us-
east-1:111122223333:deliverystream/MyDeliveryStream
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-event-destination ^
    --event-destination-name eventDestinationName ^
    --configuration-set-name configurationSet ^
    --matching-event-types eventTypes ^
    --kinesis-firehose-destination
    IamRoleArn=arn:aws:iam::111122223333:role/
AKFSMSRole,DeliveryStreamArn=arn:aws:firehose:us-
east-1:111122223333:deliverystream/MyDeliveryStream
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为描述事件目标的名称。
- 将 *configurationSet* 替换为您要与事件目标关联的配置集的名称。
- 将 *eventTypes* 替换为 [事件类型](#) 中列出的事件类型之一。
- 将 IamRoleArn 的值替换为具有 [用于 CloudWatch Logs 的 IAM 策略](#) 中所述策略的 IAM 角色的 Amazon 资源名称 (ARN)。
- 将 DeliveryStreamArn 的值替换为您要向其发送事件的 Kinesis Data Firehose 流的 ARN。

## 更新 Kinesis Data Firehose 事件目标

您可以使用 [UpdateEventDestination](#) API 更新事件目标。

更新 Kinesis Data Firehose 事件目标的过程与创建事件目标的过程类似。

## 使用 AWS CLI 更新 Kinesis Data Firehose 事件目标

- 在命令行处，运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-event-destination \  
> --event-destination-name eventDestinationName \  
> --configuration-set-name configurationSet \  
> --matching-event-types eventTypes \  
> --kinesis-firehose-destination  
IamRoleArn=arn:aws:iam::111122223333:role/  
AKFSMSRole,DeliveryStreamArn=arn:aws:firehose:us-  
east-1:111122223333:deliverystream/MyDeliveryStream
```

PowerShell

```
PS C:\> Update-SMSVEventDestination \  
>> -EventDestinationName eventDestinationName \  
>> -ConfigurationSetName configurationSet \  
>> -MatchingEventType eventTypes \  
>> -  
KinesisFirehoseDestination_IamRoleArn_IamRoleArn 111122223333:role/AKFSMSRole \  
>> -KinesisFirehoseDestination_DeliveryStreamArn arn:aws:firehose:us-  
east-1:111122223333:deliverystream/MyDeliveryStream
```

Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-event-destination ^  
--event-destination-name eventDestinationName ^  
--configuration-set-name configurationSet ^  
--matching-event-types eventTypes ^  
--kinesis-firehose-destination  
IamRoleArn=111122223333:role/AKFSMSRole,DeliveryStreamArn=arn:aws:firehose:us-  
east-1:111122223333:deliverystream/MyDeliveryStream
```

在上面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为要修改的事件目标名称。



- 将 `configurationSet` 替换为您要与事件目标关联的配置集的名称。您可以将事件目标与另外的配置集相关联。
- 将 `eventTypes` 替换为[事件类型](#)中列出的事件类型之一。
- 将 `IamRoleArn` 的值替换为具有[用于 CloudWatch Logs 的 IAM 策略](#)中所述策略的 IAM 角色的 Amazon 资源名称 (ARN)。
- 将 `DeliveryStreamArn` 的值替换为您要向其发送事件的 Kinesis Data Firehose 流的 ARN。

## 创建和更新 Amazon SNS 事件目标

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

Amazon Simple Notification Service (Amazon SNS) 是一项 Web 服务，它使应用程序、终端用户和设备能够立即发送和接收通知。有关 Amazon SNS 的更多信息，请参阅 [Amazon Simple Notification Service 开发人员指南](#)。

本节中的过程假定您已安装和配置 AWS Command Line Interface。有关设置 AWS CLI 的更多信息，请参阅《[AWS Command Line Interface 用户指南](#)》。

本节中的主题：

- [创建 Amazon SNS 事件目标](#)

## 创建 Amazon SNS 事件目标

在创建 Amazon SNS 事件目标之前，必须先创建 Amazon SNS 主题。有关创建 Amazon SNS 主题的更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的[创建主题](#)。

您必须创建一个 IAM 角色来允许 SMS 和 Voice API 向流发送数据。下一节包含有关该角色要求的信息。

您还需要已经设置了与事件目标关联的配置集，请参阅[创建配置集](#)。

## 用于 Amazon SNS 的 IAM 策略

使用以下示例创建将事件发送到 Amazon SNS 主题的策略。

下面的示例语句使用 `SourceAccount` 和 `SourceArn` 条件（可选但建议）来确保只有 Amazon Pinpoint 所有者账户才能访问配置集。在此示例中，将 `accountId` 替换为您的 AWS 账户 id，将 `region` 替换为 AWS 区域名称，将 `ConfigSetName` 替换为配置集的名称。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-east-1:111122223333:MyTopic",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:sms-voice:region:accountId:configuration-
set/ConfigSetName"
        }
      }
    }
  ]
}
```

有关 IAM 策略的更多信息，请参阅《IAM 用户指南》中的 [IAM 中的策略与权限](#)。

### 创建事件目标

您可以使用 [CreateEventDestination](#) API 来创建事件目标。

### 在 AWS CLI 中创建 Amazon SNS 事件目标

- 在命令行处，运行以下命令：

Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 create-event-destination \
> --event-destination-name eventDestinationName \
```

```
> --configuration-set-name configurationSet \  
> --matching-event-types eventTypes \  
> --sns-destination TopicArn=arn:aws:sns:us-east-1:111122223333:snsTopic
```

## PowerShell

```
PS C:\> New-SMSVEventDestination `\  
>> -EventDestinationName eventDestinationName `\  
>> -ConfigurationSetName configurationSet `\  
>> -MatchingEventType eventTypes `\  
>> -SnsDestination_TopicArn arn:aws:sns:us-east-1:111122223333:snsTopic
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 create-event-destination ^\  
--event-destination-name eventDestinationName ^\  
--configuration-set-name configurationSet ^\  
--matching-event types eventTypes ^\  
--sns-destination TopicArn=arn:aws:sns:us-east-1:111122223333:snsTopic
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为事件目标的描述性名称。
- 将 *configurationSet* 替换为您要与事件目标关联的配置集的名称。
- 将 *eventTypes* 替换为 [事件类型](#) 中列出的事件类型之一。
- 将 TopicArn 的值替换为您要向其发送事件的 Amazon SNS 主题的 Amazon 资源名称 (ARN)。

## 更新 Amazon SNS 事件目标

您可以使用 [UpdateEventDestination](#) API 更新事件目标。

更新 Amazon SNS 事件目标的过程与创建事件目标的过程类似。

在 AWS CLI 中更新 Amazon SNS 事件目标

- 在命令行处，运行以下命令：

## Linux, macOS, or Unix

```
$ aws pinpoint-sms-voice-v2 update-event-destination \  
> --event-destination-name eventDestinationName \  
> --configuration-set-name configurationSet \  
> --matching-event types eventTypes \  
> --sns-destination TopicArn=arn:aws:sns:us-east-1:111122223333:snsTopic
```

## PowerShell

```
PS C:\> Update-SMSVEventDestination \  
>> -EventDestinationName eventDestinationName \  
>> -ConfigurationSetName configurationSet \  
>> -MatchingEventType eventTypes \  
>> -SnsDestination_TopicArn arn:aws:sns:us-east-1:111122223333:snsTopic
```

## Windows Command Prompt

```
C:\> aws pinpoint-sms-voice-v2 update-event-destination ^  
--event-destination-name eventDestinationName ^  
--configuration-set-name configurationSet ^  
--matching-event types eventTypes ^  
--sns-destination TopicArn=arn:aws:sns:us-east-1:111122223333:snsTopic
```

在前面的命令中，进行以下更改：

- 将 *eventDestinationName* 替换为要修改的事件目标名称。
- 将 *configurationSet* 替换为您要与事件目标关联的配置集的名称。您可以将事件目标与另外的配置集相关联。
- 将 *eventTypes* 替换为[事件类型](#)中列出的事件类型之一。
- 将 TopicArn 的值替换为您要向其发送事件的 Amazon SNS 主题的 Amazon 资源名称 (ARN)。

## 使用 SMS 和 Voice API 发送消息

### Important

Amazon Pinpoint 已更新其用户指南文档。该页面可能不再具有最新的信息。要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 [Amazon Pinpoint SMS 用户指南](#)。

您可以使用 Amazon Pinpoint SMS 和 Voice API 第 2 版直接从您的应用程序发送事务性消息。事务性消息是发送给特定收件人的消息，与您通过活动或旅程发送的消息相反。

本节包括发送[短信](#)和[语音消息](#)的代码示例。

本节中的主题：

- [发送短信](#)
- [发送语音消息](#)

## 发送短信

您可以参照以下代码示例，使用 AWS SDK for Python (Boto3) 发送短信。

```
import boto3
from botocore.exceptions import ClientError

def send_sms_message(sms_voice_v2_client, configuration_set, context_keys,
                    country_parameters, destination_number, dry_run, keyword,
                    max_price, message_body, message_type, origination_number,
                    ttl):
    try:
        response = sms_voice_v2_client.send_text_message(
            ConfigurationSetName=configuration_set,
            Context=context_keys,
            DestinationCountryParameters=country_parameters,
            DestinationPhoneNumber=destination_number,
            DryRun=dry_run,
            Keyword=keyword,
            MaxPrice=max_price,
            MessageBody=message_body,
```

```
        MessageType=message_type,
        OriginationIdentity=origination_number,
        TimeToLive=ttl
    )

except ClientError as e:
    print(e.response)
else:
    return response['MessageId']

def main():
    configuration_set = "MyConfigurationSet"
    context_keys = {"key1": "value1"}
    country_parameters = {
        "IN_TEMPLATE_ID": "TEMPLATE01234",
        "IN_ENTITY_ID": "ENTITY98765"
    }
    destination_number = "+14258918757"
    dry_run = False
    keyword = "MyKeyword"
    max_price = "2.00"
    message_body = ("This is a test message sent from Amazon Pinpoint "
                    "using the AWS SDK for Python (Boto3). ")
    message_type = "TRANSACTIONAL"
    origination_number = "+18449831743"
    ttl = 120

    print(
        f"Sending text message to {destination_number}.")

    message_id = send_sms_message(
        boto3.client('pinpoint-sms-voice-v2'), configuration_set, context_keys,
        country_parameters, destination_number, dry_run, keyword, max_price,
        message_body, message_type, origination_number, ttl)

    print(f"Message sent!\nMessage ID: {message_id}")

if __name__ == '__main__':
    main()
```

在前面的示例中，对 main() 函数进行以下更改：

- 将 `configuration_set` 的值改为您要用于发送此消息的配置集的名称或 Amazon 资源名称 (ARN)。
- 将 `context_keys` 的值改为您要在发送此消息时使用的键和值。这些键出现在与此消息关联的事件记录中。
- 如果您使用已注册的发件人 ID 向印度的客户发送消息，请将 `country_parameters` 的值改为匹配您在注册发件人 ID 时收到的注册实体 ID 和模板 ID。

### Important

如果您不使用注册的发件人 ID 向印度的客户发送消息，请完全忽略此参数。同时，还必须删除 `send_sms_message` 函数中的相应行。

- 将 `destination_number` 的值改为您要向其发送消息的电话号码。
- 如果要在不发送任何消息的情况下执行此操作，请将 `dry_run` 的值改为 `True`。
- 将 `max_price` 的值改为您发送此消息的每个消息部分要花费的最大金额（以美元为单位）。一个消息部分最多包含 140 字节的信息。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的[短信字符限制](#)。
- 更改 `message_body` 的值以包括您要发送的消息。一条消息的最大长度取决于其所包含的字符。有关短信字符编码的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[短信字符限制](#)。
- 更改 `message_type` 的值以表示相应的消息类别。有效值包括 `TRANSACTIONAL`（适用于重要或对时间敏感的消息）和 `PROMOTION`（适用于不重要或对时间不敏感的消息）。
- 将 `origination_number` 的值改为您要用于发送消息的电话号码。电话号码必须使用 E.164 格式。
- 将 `ttl` 的值改为 Amazon Pinpoint 应尝试传送消息的时间量（以秒为单位）。您可以将 TTL 值设置为最多 259200 秒（72 小时）。

## 发送语音消息

您可以使用以下代码示例，通过 AWS SDK for Python (Boto3) 发送语音消息。

```
import boto3
from botocore.exceptions import ClientError

def send_voice_message(sms_voice_v2_client, configuration_set, context_keys,
                      destination_number, dry_run, max_price, message_body,
```

```
        message_type, origination_number, ttl, voice_id):
    try:
        response = sms_voice_v2_client.send_voice_message(
            ConfigurationSetName=configuration_set,
            Context=context_keys,
            DestinationPhoneNumber=destination_number,
            DryRun=dry_run,
            MaxPricePerMinute=max_price,
            MessageBody=message_body,
            MessageBodyTextType=message_type,
            OriginationIdentity=origination_number,
            TimeToLive=ttl,
            VoiceId=voice_id
        )

    except ClientError as e:
        print(e.response)
    else:
        return response['MessageId']

def main():
    configuration_set = "MyConfigurationSet"
    context_keys = {"key1": "value1"}
    destination_number = "+12065550123"
    dry_run = False
    max_price = "2.00"
    message_body = (
        "<speak>"
        "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis>"
        "using the <break strength='weak' /> AWS SDK for Python (Boto3). "
        "<amazon:effect phonation='soft'>Thank you for listening."
        "</amazon:effect>"
        "</speak>")
    message_type = "SSML"
    origination_number = "+18445550142"
    ttl = 120
    voice_id = "MATTHEW"

    print(
        f"Sending voice message from {origination_number} to {destination_number}.")

    message_id = send_voice_message(
        boto3.client('pinpoint-sms-voice-v2'), configuration_set, context_keys,
```



```
destination_number, dry_run, max_price, message_body, message_type,
origination_number, ttl, voice_id)

print(f"Message sent!\nMessage ID: {message_id}")

if __name__ == '__main__':
    main()
```

在前面的示例中，对 `main()` 函数进行以下更改：

- 将 `configuration_set` 的值改为您要用于发送此消息的配置集的名称或 Amazon 资源名称 (ARN)。
- 将 `context_keys` 的值改为您要在发送此消息时使用的键和值。这些键出现在与此消息关联的事件记录中。
- 将 `destination_number` 的值改为您要向其发送消息的电话号码。
- 将 `max_price` 的值改为发送此消息每分钟要花费的最大金额。
- 更改 `message_body` 的值以包括您要发送的消息。消息最多可包含 6,000 个字符。
- 如果要使用纯文本脚本而不是 SSML 格式的脚本，请将 `message_type` 的值改为 `TEXT`。
- 将 `origination_number` 的值改为您要用于发送消息的电话号码。电话号码必须使用 E.164 格式。
- 如果要在不发送任何消息的情况下执行此操作，请将 `dry_run` 的值改为 `True`。
- 将 `ttl` 的值改为 Amazon Pinpoint 应尝试传送消息的时间量（以秒为单位）。您可以将 TTL 值设置为最多 259200 秒（72 小时）。
- 将 `MATTHEW` 替换为要用于发送消息的 Amazon Polly 语音的名称。有关支持的语音的完整列表，请参阅《SMS 和 Voice API 第 2 版》中的 [SendVoiceMessage](#)。如果您未指定语音，则您的消息将使用“MATTHEW”语音发送。

## 发送和验证一次性密码 (OTP)

Amazon Pinpoint 包含一次性密码 (OTP) 管理功能。您可以使用此功能生成新的一次性密码，并将其作为短信发送给收件人。然后，您的应用程序可以调用 Amazon Pinpoint API 来验证这些密码。

### Note

要使用此功能，您的账户必须具有生产短信访问权限。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的[关于 Amazon Pinpoint 短信沙盒](#)。

在某些国家和地区，您必须先获得专用的电话号码或发件人 ID，然后才能发送短信。例如，当您向美国的收件人发送消息时，您必须有一个专用的免费电话号码、10DLC 号码或短代码。当您向印度的收件人发送消息时，您必须拥有注册的发件人 ID，其中包括主体实体 ID (PEID) 和模板 ID。在使用 OTP 功能时，这些要求仍然适用。

要使用此功能，您需要具有发送和验证 OTP 消息的权限，请参阅[一次性密码](#)。如果您在确定权限方面需要帮助，请参阅[Amazon Pinpoint 身份和访问管理问题排查](#)。

## 发送 OTP 消息

您可以使用 Amazon Pinpoint API 中的 `SendOtpMessages` 操作向应用程序用户发送 OTP 代码。当您使用此 API 时，Amazon Pinpoint 会生成一个随机代码并将其作为短信发送给您的用户。您的请求中必须包括以下参数：

- `Channel` – 发送 OTP 代码的通信渠道。目前，仅支持短信，因此唯一可接受的值是 `SMS`。
- `BrandName` – 与 OTP 代码关联的品牌、公司或产品的名称。该名称最多可以包含 20 个字符。

### Note


当 Amazon Pinpoint 发送 OTP 消息时，品牌名称会自动插入到以下消息模板中：

```
This is your One Time Password: {{otp}} from {{brand}}
```

因此，如果您指定 `ExampleCorp` 您的品牌名称，并且 Amazon Pinpoint 生成了一个 `123456` 的一次性密码，则它会向您的用户发送以下消息：

```
This is your One Time Password: 123456 from ExampleCorp
```

- `CodeLength` – 发送给收件人的 OTP 代码中将包含的位数。OTP 代码可以包含 5 到 8 位数字。
- `ValidityPeriod` – OTP 代码的有效时间（以分钟为单位）。有效期可以为 5 到 60 分钟。
- `AllowedAttempts` – 收件人验证 OTP 失败的次数。如果验证次数超过此值，OTP 将自动失效。最多可验证 5 次。
- `Language` – 发送消息时使用的语言，采用 IETF BCP-47 格式。可接受的值如下：
  - `de-DE` – 德语
  - `en-GB` – 英语（英国）
  - `en-US` – 英语（美国）
  - `es-419` – 西班牙语（拉丁美洲）
  - `es-ES` – 西班牙语
  - `fr-CA` – 法语（加拿大）
  - `fr-FR` – 法语
  - `it-IT` – 意大利语
  - `ja-JP` – 日语
  - `ko-KR` – 韩语
  - `pt-BR` – 巴西葡萄牙语
  - `zh-CN` – 简体中文
  - `zh-TW` – 繁体中文
- `OriginationIdentity` – 用于发送 OTP 代码的源身份（例如长代码、短代码或发件人 ID）。如果您使用长代码或免费电话号码发送 OTP，则电话号码必须采用 E.164 格式。
- `DestinationIdentity` – OTP 代码发送到的电话号码，采用 E.164 格式。
- `ReferenceId` – 请求的唯一参考 ID。该参考 ID 与您在验证 OTP 时提供的参考 ID 完全一致。该参考 ID 可以包含 1 到 48 个字符。
- `EntityId` – 在监管机构注册的实体 ID。目前仅在向印度的收件人发送消息时使用此参数。如果您不是向位于印度的收件人发送消息，则可以忽略此参数。
- `TemplateId` – 在监管机构注册的模板 ID。目前仅在向印度的收件人发送消息时使用此参数。如果您不是向位于印度的收件人发送消息，则可以忽略此参数。

 Note

有关向印度的收件人发送消息的要求的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[向印度的收件人发送短信的特殊要求](#)。

为确保正确配置您的 Amazon Pinpoint 账户以发送 OTP 消息，您可以使用 AWS CLI 发送测试消息。有关安装和配置的更多信息 AWS CLI，请参阅《[AWS Command Line Interface 用户指南](#)》。

## Linux, macOS, or Unix

要使用发送测试 OTP 消息 AWS CLI，请在终端中运行[send-otp-message](#)以下命令：

```
aws pinpoint send-otp-message \  
--application-id 7353f53e6885409fa32d07cedexample \  
--send-otp-message-request '{  
  "Channel": "SMS",  
  "BrandName": "ExampleCorp",  
  "CodeLength": 5,  
  "ValidityPeriod": 20,  
  "AllowedAttempts": 5,  
  "OriginationIdentity": "+18555550142",  
  "DestinationIdentity": "+12065550007",  
  "ReferenceId": "SampleReferenceId"  
}'
```

## PowerShell

要使用发送测试 OTP 消息 AWS CLI，[send-otp-message](#)请在以下位置运行命令：PowerShell

```
aws pinpoint send-otp-message \  
--application-id 7353f53e6885409fa32d07cedexample \  
--send-otp-message-request '{ \  
  "Channel": "SMS", \  
  "BrandName": "ExampleCorp", \  
  "CodeLength": 5, \  
  "ValidityPeriod": 20, \  
  "AllowedAttempts": 5, \  
  "OriginationIdentity": "+18555550142", \  
  "DestinationIdentity": "+12065550007", \  
  "ReferenceId": "SampleReferenceId"}'
```

## Windows command prompt

要使用发送测试 OTP 消息 AWS CLI，请在 Windows [send-otp-message](#) 命令提示符下运行命令：

```
aws pinpoint send-otp-message ^  
--application-id 7353f53e6885409fa32d07cedexample ^  
--send-otp-message-request '{ ^
```

```
"Channel": "SMS", ^
"BrandName": "ExampleCorp", ^
"CodeLength": 5, ^
"ValidityPeriod": 20, ^
"AllowedAttempts": 5, ^
"OriginationIdentity": "+18555550142", ^
"DestinationIdentity": "+12065550007", ^
"ReferenceId": "SampleReferenceId" ^
}'
```

## SendOtpMessage 响应

成功发送 OTP 消息后，您将收到与类似以下示例的响应：

```
{
  "MessageResponse": {
    "ApplicationId": "7353f53e6885409fa32d07cedexample",
    "RequestId": "255d15ea-75fe-4040-b919-096f2example",
    "Result": {
      "+12065550007": {
        "DeliveryStatus": "SUCCESSFUL",
        "MessageId": "nvrmgq9kq4en96qgp0tlqli3og1at6aexample",
        "StatusCode": 200,
        "StatusMessage": "MessageId: nvrmgq9kq4en96qgp0tlqli3og1at6aexample"
      }
    }
  }
}
```

## 验证 OTP 消息

要验证 OTP 码，请调用 `VerifyOtpMessages` API。您的请求中必须包括以下参数：

- `DestinationIdentity` – OTP 代码发送到的电话号码，采用 E.164 格式。
- `ReferenceId` – 您向收件人发送 OTP 代码时使用的参考 ID。参考 ID 必须完全匹配。
- `Otp` – 您正在验证的 OTP 代码。

您可以使用 AWS CLI 来测试验证过程。有关安装和配置的更多信息 AWS CLI，请参阅 [《AWS Command Line Interface 用户指南》](#)。

## Linux, macOS, or Unix

要使用验证 OTP AWS CLI，请在终端中运行[verify-otp-message](#)命令：

```
aws pinpoint verify-otp-message \  
--application-id 7353f53e6885409fa32d07cedexample \  
--verify-otp-message-request-parameters '{  
  "DestinationIdentity": "+12065550007",  
  "ReferenceId": "SampleReferenceId",  
  "Otp": "012345"  
}'
```

## PowerShell

要使用验证 OTP AWS CLI，[verify-otp-message](#)请在以下位置运行命令：PowerShell

```
aws pinpoint verify-otp-message \  
--application-id 7353f53e6885409fa32d07cedexample \  
--verify-otp-message-request-parameters '{\  
  "DestinationIdentity": "+12065550007",\  
  "ReferenceId": "SampleReferenceId",\  
  "Otp": "012345"}'
```

## Windows command prompt

要使用验证 OTP AWS CLI，请在 Windows [verify-otp-message](#) 命令提示符下运行以下命令：

```
aws pinpoint verify-otp-message \  
--application-id 7353f53e6885409fa32d07cedexample \  
--verify-otp-message-request-parameters '{  
  "DestinationIdentity": "+12065550007",  
  "ReferenceId": "SampleReferenceId",  
  "Otp": "012345"}'
```

## VerifyOtpMessage 响应

当您向 VerifyOTPMMessage API 发送请求时，它会返回一个 VerificationResponse 对象，其中包含单个属性 Valid。如果参考 ID、电话号码和 OTP 都与 Amazon Pinpoint 预期的值相匹配，并且 OTP 没有过期，则 Valid 的值为 true；否则为 false。以下是 OTP 验证成功响应的示例：

```
{
```

```
"VerificationResponse": {
    "Valid": true
}
```

## 代码示例

本节包含的代码示例展示了如何使用 SDK for Python (Boto3) 发送和验证 OTP 代码。

### 生成参考 ID

以下函数根据收件人的电话号码、收件人收到 OTP 的产品或品牌以及请求的来源（例如，可以是网站或应用程序中页面的名称）为每个收件人生成一个唯一的参考 ID。验证 OTP 代码时，必须传递相同的参考 ID，这样验证才能成功。发送和验证代码示例都使用此实用函数。

此函数不是必需的，但却是一种有用的方法，它可以将 OTP 发送和验证过程的范围限定为特定的事务，这样便于在验证步骤中重新提交。您可以随便使用任何参考 ID —— 这只是一个基本示例。但是，本节中的其他代码示例依赖于此函数。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0

import hashlib

def generate_ref_id(destinationNumber,brandName,source):
    refId = brandName + source + destinationNumber
    return hashlib.md5(refId.encode()).hexdigest()
```

### 发送 OTP 代码

以下代码示例展示如何使用 SDK for Python (Boto3) 发送 OTP 代码。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0

import boto3
from botocore.exceptions import ClientError
from generate_ref_id import generate_ref_id

### Some variables that are unlikely to change from request to request. ###
```

```
# The AWS Region that you want to use to send the message.
region = "us-east-1"

# The phone number or short code to send the message from.
originationNumber = "+18555550142"

# The project/application ID to use when you send the message.
appId = "7353f53e6885409fa32d07cedexample"

# The number of times the user can unsuccessfully enter the OTP code before it becomes
  invalid.
allowedAttempts = 3

# Function that sends the OTP as an SMS message.
def send_otp(destinationNumber, codeLength, validityPeriod, brandName, source, language):
    client = boto3.client('pinpoint', region_name=region)
    try:
        response = client.send_otp_message(
            ApplicationId=appId,
            SendOTPMessageRequestParameters={
                'Channel': 'SMS',
                'BrandName': brandName,
                'CodeLength': codeLength,
                'ValidityPeriod': validityPeriod,
                'AllowedAttempts': allowedAttempts,
                'Language': language,
                'OriginationIdentity': originationNumber,
                'DestinationIdentity': destinationNumber,
                'ReferenceId': generate_ref_id(destinationNumber, brandName, source)
            }
        )

    except ClientError as e:
        print(e.response)
    else:
        print(response)

# Send a message to +14255550142 that contains a 6-digit OTP that is valid for 15
  minutes. The
# message will include the brand name "ExampleCorp", and the request originated from a
  part of your
# site or application called "CreateAccount". The US English message template should be
  used to
# send the message.
```



```
send_otp("+14255550142",6,15,"ExampleCorp","CreateAccount","en-US")
```

## 验证 OTP 代码

以下代码示例展示如何使用 SDK for Python (Boto3) 验证已发送的 OTP 代码。为使验证步骤成功，您的请求中必须包含与发送消息时使用的参考 ID 完全一致的参考 ID。

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0

import boto3
from botocore.exceptions import ClientError
from generate_ref_id import generate_ref_id

# The AWS Region that you want to use to send the message.
region = "us-east-1"

# The project/application ID to use when you send the message.
appId = "7353f53e6885409fa32d07cedexample"

# Function that verifies the OTP code.
def verify_otp(destinationNumber,otp,brandName,source):
    client = boto3.client('pinpoint',region_name=region)
    try:
        response = client.verify_otp_message(
            ApplicationId=appId,
            VerifyOTPMessageRequestParameters={
                'DestinationIdentity': destinationNumber,
                'ReferenceId': generate_ref_id(destinationNumber,brandName,source),
                'Otp': otp
            }
        )

    except ClientError as e:
        print(e.response)
    else:
        print(response)

# Verify the OTP 012345, which was sent to +14255550142. The brand name ("ExampleCorp")
# and the
# source name ("CreateAccount") are used to generate the correct reference ID.
verify_otp("+14255550142","012345","ExampleCorp","CreateAccount")
```

# 在 Amazon Pinpoint 中发送和检索应用程序内消息

您可以使用应用程序内消息向应用程序的用户发送定向消息。应用程序内消息是高度可定制的。它们可以包括用于打开网站或使用户转向应用程序特定部分的按钮。您可以配置背景和文本颜色，定位文本，以及向通知中添加按钮和图像。您可以发送一条消息，或者创建最多包含五条独特消息的轮盘。有关应用程序内消息的概述，包括创建应用程序内消息模板的说明，请参阅《Amazon Pinpoint 用户》指南中的[创建应用程序内模板](#)。

您可以使用 AWS Amplify 将 Amazon Pinpoint 的应用程序内消息传递功能无缝集成到您的应用程序中。Amplify 可以自动完成获取消息、呈现消息以及向 Amazon Pinpoint 发送分析数据的过程。React Native 应用程序目前支持这种集成。有关更多信息，请参阅《Amplify Framework 文档》中的[应用程序内消息](#)。

本节提供有关为应用程序中的端点请求应用程序内消息以及解释该请求结果的信息。

## 检索端点的应用程序内消息

您的应用程序可以调用 [GetInAppMessages](#) API 来检索给定端点有权接收的所有应用程序内消息。在调用 GetInAppMessages API 时，您需要提供以下参数：

- ApplicationId – 与应用程序内消息活动关联的 Amazon Pinpoint 项目的唯一 ID。
- EndpointId – 您要为其检索消息的端点的唯一 ID。

当您使用这些值调用 API 时，它会返回一个消息列表。有关此操作生成的响应的更多信息，请参阅[了解 GetInAppMessages API 响应](#)。

您还可以使用 AWS SDK 调用 GetInAppMessages 操作。以下代码示例包括检索应用程序内消息的函数。

### JavaScript

在单独的模块中创建客户端并将其导出：

```
import { PinpointClient } from "@aws-sdk/client-pinpoint";
const REGION = "us-east-1";
const pinClient = new PinpointClient({ region: REGION });
export { pinClient };
```

检索端点的应用程序内消息：

```
// Import required AWS SDK clients and commands for Node.js
import { PinpointClient, GetInAppMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "../lib/pinClient.js";

("use strict");

//The Amazon Pinpoint application ID.
const projectId = "4c545b28d21a490cb51b0b364example";

//The ID of the endpoint to retrieve messages for.
const endpointId = "c5ac671ef67ee3ad164cf7706example";

const params = {
  ApplicationId: projectId,
  EndpointId: endpointId
};

const run = async () => {
  try {
    const data = await pinClient.send(new GetInAppMessagesCommand(params));
    console.log(JSON.stringify(data, null, 4));
    return data;
  } catch (err) {
    console.log("Error", err);
  }
};
run();
```

## Python

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def retrieve_inapp_messages(
    pinpoint_client, project_id, endpoint_id):
    """
    Retrieves the in-app messages that a given endpoint is entitled to.

    :param pinpoint_client: A Boto3 Pinpoint client.
```

```
:param project_id: An Amazon Pinpoint project ID.
:param endpoint_id: The ID of the endpoint to retrieve messages for.
:return: A JSON object that contains information about the in-app message.
"""

try:
    response = pinpoint_client.get_in_app_messages(
        ApplicationId=project_id,
        EndpointId=endpoint_id)
except ClientError:
    logger.exception("Couldn't retrieve messages.")
    raise
else:
    return response

def main():
    project_id = "4c545b28d21a490cb51b0b364example"
    endpoint_id = "c5ac671ef67ee3ad164cf7706example"
    inapp_response = retrieve_inapp_messages(
        boto3.client('pinpoint'), project_id, endpoint_id)
    print(inapp_response)

if __name__ == '__main__':
    main()
```

## 了解 GetInAppMessages API 响应

当您调用 [GetInAppMessages](#) API 操作时，它会返回指定端点有权接收的消息列表。然后，您的应用程序可以根据响应中的值呈现消息。

以下是您在调用 GetInAppMessages API 时返回的 JSON 对象的示例：

```
{
  "InAppMessagesResponse": {
    "InAppMessageCampaigns": [
      {
        "CampaignId": "inAppTestCampaign-4c545b28d21a490cb51b0b364example",
        "DailyCap": 0,
        "InAppMessage": {
          "Content": [
            {
              "BackgroundColor": "#f8e71c",
```

```
    "BodyConfig":{
      "Alignment":"CENTER",
      "Body":"This is a sample in-app message sent using Amazon Pinpoint.",
      "TextColor":"#d0021b"
    },
    "HeaderConfig":{
      "Alignment":"CENTER",
      "Header":"Sample In-App Message",
      "TextColor":"#d0021b"
    },
    "ImageUrl":"https://example.com/images/thumbnail.png",
    "PrimaryBtn":{
      "DefaultConfig":{
        "BackgroundColor":"#d0021b",
        "BorderRadius":50,
        "ButtonAction":"CLOSE",
        "Text":"Dismiss",
        "TextColor":"#f8e71c"
      }
    }
  ],
  "Layout":"MIDDLE_BANNER"
},
"Priority":3,
"Schedule":{
  "EndDate":"2021-11-06T00:08:05Z",
  "EventFilter":{
    "Dimensions":{
      "Attributes":{

      },
      "EventType":{
        "DimensionType":"INCLUSIVE",
        "Values":[
          "_session.start"
        ]
      },
      "Metrics":{

      }
    }
  }
},
},
```

```

        "SessionCap":0,
        "TotalCap":0,
        "TreatmentId":"0"
    }
]
}
}

```

以下各节提供有关此响应的组成部分的更多信息。

## InAppMessageCampaigns 对象

InAppMessageCampaigns 对象包含以下属性：

属性	描述	在哪里设置
CampaignId	一个字符串，其中包含发送消息的 Amazon Pinpoint 活动的名称和唯一活动 ID。该名称位于活动 ID 之前。这两个值用连字符 (-) 分隔。	由 Amazon Pinpoint 在您创建活动自动创建。
TreatmentId	一个整数，表示此消息的活动处理的 ID。如果活动只有一种处理，则值为 0。	
Priority	应用程序内消息的优先级，表示为 1 到 5 之间的整数，其中 1 为最高优先级，5 为最低优先级。	创建活动过程的 <a href="#">第 1 步</a> 。
InAppMessage	一个 <a href="#">InAppMessage 对象</a> ，其中包含有关如何呈现消息的信息。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Schedule	Schedule 对象，其中包含关于消息发送时间的信息。	创建活动过程的 <a href="#">第 4 步</a> （如果活动是在控制台创建的），或者是 Schedule 对象（如果

属性	描述	在哪里设置
		活动是使用 API 或 SDK 创建的 )。
DailyCap	在 24 小时内可以向用户显示一条应用程序内消息的次数，以整数表示。	继承自 <a href="#">项目级设置</a> 。如果活动包含覆盖项目设置的设置，则会改用这些设置。
SessionCap	应用程序会话期间可以向用户显示一条应用程序内消息的次数，以整数表示。	
TotalCap	每个活动可以向端点显示任何应用程序内消息的总次数，以整数表示。	

## InAppMessage 对象

InAppMessage 对象包含以下属性：

属性	描述	在哪里设置
Content	一个包含 <a href="#">InAppMessageContent</a> 对象的数组，该对象描述了消息的内容。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Layout	描述应用程序内消息在收件人设备上的显示方式的字符串。 可能的值有： <ul style="list-style-type: none"> <li>BOTTOM_BANNER – 显示为页面底部横幅的消息。</li> <li>TOP_BANNER – 显示为页面顶部横幅的消息。</li> <li>OVERLAYS – 覆盖整个屏幕的消息。</li> </ul>	

属性	描述	在哪里设置
	<ul style="list-style-type: none"> <li>MOBILE_FEED – 在页面置顶窗口中显示的消息。</li> <li>MIDDLE_BANNER – 显示为页面中间横幅的消息。</li> <li>CAROUSEL – 最多包含五条唯一消息的可滚动布局。</li> </ul>	

## HeaderConfig 对象

HeaderConfig 对象包含以下属性：

属性	描述	在哪里设置
Alignment	一个字符串，指定标题文本的文本对齐方式。可能的值为 LEFT、CENTER 和 RIGHT。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Header	消息标题文本。	
TextColor	标题文本的颜色，以十六进制颜色代码（例如，#000000 代表黑色）表示。	

## BodyConfig 对象

BodyConfig 对象包含以下属性：

属性	描述	在哪里设置
Alignment	一个字符串，指定消息正文的文本对齐方式。可能的值为 LEFT、CENTER 和 RIGHT。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Body	消息的主体文本。	



属性	描述	在哪里设置
TextColor	正文的颜色，以由十六进制颜色代码（例如，#000000 代表黑色）组成的字符串表示。	

## InAppMessageContent 对象

InAppMessageContent 对象包含以下属性：

属性	描述	在哪里设置
BackgroundColor	应用程序内消息的背景色，以十六进制颜色代码（例如，#000000 代表黑色）表示。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
BodyConfig	一个 <a href="#">BodyConfig</a> 对象，其中包含与消息正文内容相关的信息。	
HeaderConfig	一个 <a href="#">HeaderConfig</a> 对象，其中包含与消息标题相关的信息。	
ImageUrl	消息中显示的图像的 URL。	
PrimaryBtn	一个 <a href="#">InAppMessageButton</a> 对象，其中包含有关消息中主要按钮的信息。	
SecondaryBtn	一个 <a href="#">InAppMessageButton</a> 对象，其中包含有关消息中辅助按钮的信息。如果应用程序内消息模板未指定辅助按钮，则不显示。	

## Schedule 对象

Schedule 对象包含以下属性：

属性	描述	在哪里设置
EndDate	将结束活动的计划时间，采用 ISO 8601 格式。	创建活动过程的 <a href="#">第 4 步</a> （如果活动是在控制台创建的），或者是 Schedule 对象（如果活动是使用 API 或 SDK 创建的）。
EventFilter	有关导致显示应用程序内消息的事件的信息。当您生成与 Amazon Pinpoint 应用程序内活动匹配的事件时，系统会显示该消息。	

## InAppMessageButton 对象

InAppMessageButton 对象包含以下属性：

属性	描述	在哪里设置
DefaultConfig	一个 <a href="#">DefaultButtonConfig</a> 对象，其中包含应用程序内消息中按钮的默认设置的信息。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Android	一个 <a href="#">OverrideButtonConfig</a> 对象，用于指定按钮在 Android 设备上的行为方式。这将覆盖 DefaultConfig 对象中详细介绍的默认按钮配置。	
iOS	一个 <a href="#">OverrideButtonConfig</a> 对象，用于指定按钮在 iOS 设备上的行为方式。这将覆盖 DefaultConfig 对象中详细介绍的默认按钮配置。	

属性	描述	在哪里设置
Web	一个 <a href="#">OverrideButtonConfig</a> 对象，用于指定按钮在 Web 应用程序中的行为方式。这将覆盖 DefaultConfig 对象中详细介绍的默认按钮配置。	

## DefaultButtonConfig 对象

DefaultButtonConfig 对象包含以下属性：

属性	描述	在哪里设置
BackgroundColor	按钮的背景色，以十六进制颜色代码（例如，#000000 代表黑色）表示。	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
BorderRadius	按钮边框的半径（以像素为单位），以整数表示。数字越大，圆角越大。	
ButtonAction	一个字符串，描述当收件人在应用程序内消息中选择按钮时发生的操作。可能的值有： <ul style="list-style-type: none"> <li>LINK – 指向 Web 目标的链接。</li> <li>DEEP_LINK – 指向应用程序中特定页面的链接。</li> <li>CLOSE – 关闭消息。</li> </ul>	
Link	按钮的目标 URL。对于 ButtonAction 为 CLOSE 的按钮不存在。	
Text	按钮上显示的文本。	

属性	描述	在哪里设置
TextColor	按钮上文本的颜色，以十六进制颜色代码（例如，#000000 代表黑色）表示。	

## OverrideButtonConfig 对象

仅当应用程序内消息模板使用覆盖按钮时，OverrideButtonConfig 对象才会出现。覆盖按钮是针对特定设备类型（例如 iOS 设备、Android 设备或 Web 浏览器）具有特定配置的按钮。

OverrideButtonConfig 对象包含以下属性：

属性	描述	在哪里设置
ButtonAction	当收件人在应用程序内消息中选择按钮时发生的操作。可能的值有： <ul style="list-style-type: none"> <li>LINK – 指向 Web 目标的链接。</li> <li>DEEP_LINK – 指向应用程序中特定页面的链接。</li> <li>CLOSE – 关闭消息。</li> </ul>	基于为活动指定的 <a href="#">应用程序内消息模板</a> 中的内容。
Link	按钮的目标 URL。对于 ButtonAction 为 CLOSE 的按钮不存在。	
Text	按钮上显示的文本。	
TextColor	按钮上文本的颜色，以十六进制颜色代码（例如，#000000 代表黑色）表示。	

# 在 Amazon Pinpoint 中验证电话号码

Amazon Pinpoint 包含电话号码验证服务，您可以使用该服务来确定电话号码是否有效，以及获得有关电话号码本身的额外信息。例如，当您使用电话号码验证服务时，它返回以下信息：

- E.164 格式的电话号码。
- 电话号码类型（如手机、固定电话或 VoIP）。
- 电话号码所在的城市和国家。
- 与电话号码关联的服务提供商。

使用电话号码验证服务需要额外收费。有关更多信息，请参阅 [Amazon Pinpoint 定价](#)。

## Important

对于在美国和加拿大发起的电话号码，电话号码验证 API 将不再返回 City、County、Timezone 和 ZipCode 的数据。

## 电话号码验证使用案例

您可以使用电话号码验证服务来实现多种使用案例，包括：

- 验证 Web 表单上提供的电话号码 – 如果您使用基于 Web 的表单来收集客户的联系信息，请验证客户提供的电话号码，然后再提交表单。使用网站后端，通过 Amazon Pinpoint API 验证该号码。API 响应表示该号码是否无效，例如，该电话号码格式设置是否错误。如果确定客户提供的电话号码无效，则 Web 表单可提示该客户提供其他号码。
- 清理现有联系人数据库 – 如果您有一个客户电话号码数据库，则可验证每个电话号码，然后根据结果更新数据库。例如，如果发现端点的电话号码无法接收短信，可以将该端点的 ChannelType 属性从 SMS 更改为 VOICE。您可以先验证电话号码，然后按照[向 Amazon Pinpoint 中添加端点](#)中（对于单个端点）或[将端点批量添加到 Amazon Pinpoint](#)中（对于多个端点）的说明，更新新端点或现有端点的 ChannelType 属性。
- 在发送消息前选择正确的渠道 – 如果您打算发送短信，但确定目标号码是无效的，则您可以通过另外的渠道向收件人发送消息。例如，如果端点无法接收短信，则您可以改为发送语音消息。

## 使用电话号码验证服务

要验证号码，请在 Amazon Pinpoint API 中向 `/v1/phone/number/validate/` URI 发送 HTTP POST 请求。以下示例中的请求包括所需的 HTTP 标头以及简单的 JSON 正文。正文通过 `PhoneNumber` 参数指定要验证的号码：

```
POST /v1/phone/number/validate/ HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
X-Amz-Date: 20190805T031042Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20190805/us-east-1/mobiletargeting/aws4_request, SignedHeaders=content-length;content-type;host;x-amz-date, Signature=39df573629ddb283aea1fa2f7eef4106c0fb4826edf72e9934f03cf77example
Cache-Control: no-cache

{
  "PhoneNumber": "+12065550100"
}
```

有关支持的方法、参数和架构的信息，请参阅《Amazon Pinpoint API 参考》中的[电话号码验证](#)。

您也可以使用 AWS CLI 来快速验证各个电话号码。

要使用电话号码验证服务，请使用 AWS CLI

- 在命令行输入以下命令：

```
aws pinpoint phone-number-validate --number-validate-request
PhoneNumber=+442079460881
```

在前面的命令中，将 **+442079460881** 替换成要验证的电话号码。有关更多信息，请参阅[phone-number-validate](#) 《AWS CLI 命令参考》。

### Note

将电话号码提供给电话号码验证服务时，应始终包含国家/地区代码。如果不包含国家/地区代码，则该服务可能会返回位于其他国家/地区的电话号码信息。

## 电话号码验证响应

根据可用于您提供的电话号码的数据，电话号码验证服务提供的信息会略有不同。本节包含电话号码验证服务返回的响应示例。

### Note

电话号码验证服务提供的数据基于全球电信提供商和其他实体提供的信息。某些国家/地区的提供商更新这些信息的频率可能低于其他国家/地区的提供商。例如，如果您发出手机号码验证请求，并且您提供的号码从一家移动运营商转网到了另一家，则电话号码验证服务的响应可能包含原始运营商的名称，而不是当前运营商的名称。

### 有效手机号码

当向电话号码验证服务发送请求，并且该电话号码为有效的手机号码时，会返回类似于以下示例的信息：

```
{
  "NumberValidateResponse": {
    "Carrier": "ExampleCorp Mobile",
    "City": "Seattle",
    "CleansedPhoneNumberE164": "+12065550142",
    "CleansedPhoneNumberNational": "2065550142",
    "Country": "United States",
    "CountryCodeIso2": "US",
    "CountryCodeNumeric": "1",
    "OriginalPhoneNumber": "+12065550142",
    "PhoneType": "MOBILE",
    "PhoneTypeCode": 0,
    "Timezone": "America/Los_Angeles",
    "ZipCode": "98101"
  }
}
```

### 有效固定电话号码

如果请求包含有效的固定电话号码，则电话号码验证服务返回类似于以下示例的信息：

```
{
  "CountryCodeIso2": "US",
```

```
"CountryCodeNumeric": "1",
"Country": "United States",
"City": "Santa Clara",
"ZipCode": "95037",
"Timezone": "America/Los_Angeles",
"CleansedPhoneNumberNational": "4085550101",
"CleansedPhoneNumberE164": "14085550101",
"Carrier": "AnyCompany",
"PhoneTypeCode": 1,
"PhoneType": "LANDLINE",
"OriginalPhoneNumber": "+14085550101"
}
```

## 有效 VoIP 电话号码

如果请求包含有效的 IP 语音 (VoIP) 电话号码，则电话号码验证服务返回类似于以下示例的信息：

```
{
  "NumberValidateResponse": {
    "Carrier": "ExampleCorp",
    "City": "Countrywide",
    "CleansedPhoneNumberE164": "+441514960001",
    "CleansedPhoneNumberNational": "1514960001",
    "Country": "United Kingdom",
    "CountryCodeIso2": "GB",
    "CountryCodeNumeric": "44",
    "OriginalPhoneNumber": "+441514960001",
    "PhoneType": "VOIP",
    "PhoneTypeCode": 2
  }
}
```

## 无效电话号码

如果请求包含无效的电话号码，则电话号码验证服务返回类似于以下示例的信息：

```
{
  "NumberValidateResponse": {
    "CleansedPhoneNumberE164": "+44163296076",
    "CleansedPhoneNumberNational": "163296076",
    "Country": "United Kingdom",
    "CountryCodeIso2": "GB",
    "CountryCodeNumeric": "44",
```



```
    "OriginalPhoneNumber": "+440163296076",
    "PhoneType": "INVALID",
    "PhoneTypeCode": 3
  }
}
```

注意，此响应中的 PhoneType 属性指示该电话号码为 INVALID，并且它不包含有关与此电话号码相关联的运营商或位置。应避免向 PhoneType 为 INVALID 的电话号码发送短信或语音，因为这些号码不太可能属于实际收件人。

## 其他电话号码

有时，电话号码验证服务的响应包含的 PhoneType 值为 OTHER。在以下情况下，该服务可能返回此类响应：

- 电话号码为免费（免费电话）号码。
- 电话号码是为在电视节目和电影中使用而预留的，例如以 555 开头的北美电话号码。
- 电话号码包含一个当前未在使用的区号，例如北美的 999 区号。
- 电话号码是为其他某个用途预留的。

以下示例说明当请求包含虚构的北美电话号码时电话号码验证服务提供的响应：

```
{
  "NumberValidateResponse": {
    "Carrier": "Multiple OCN Listing",
    "CleansedPhoneNumberE164": "+14255550199",
    "CleansedPhoneNumberNational": "4255550199",
    "Country": "United States",
    "CountryCodeIso2": "US",
    "CountryCodeNumeric": "1",
    "OriginalPhoneNumber": "+14255550199",
    "PhoneType": "OTHER",
    "PhoneTypeCode": 4,
    "Timezone": "America/Los_Angeles"
  }
}
```

## 预付费电话号码

如果请求包含有效的预付费电话号码，则电话号码验证服务返回类似于以下示例的信息：

```
{
  "NumberValidateResponse": {
    "Carrier": "ExampleCorp",
    "City": "Countrywide",
    "CleansedPhoneNumberE164": "+14255550199",
    "CleansedPhoneNumberNational": "4255550199",
    "Country": "United States",
    "CountryCodeIso2": "US",
    "CountryCodeNumeric": "1",
    "OriginalPhoneNumber": "+14255550199",
    "PhoneType": "PREPAID",
    "PhoneTypeCode": 5
  }
}
```

有关这些响应中包含的信息的详情，请参阅《Amazon Pinpoint API 参考》中的[电话号码验证](#)。

## 从应用程序发送事务性消息

您可以使用 Amazon Pinpoint API 和 AWS SDK 直接从您的应用程序发送事务性消息。事务性消息是发送给特定收件人的消息，与发送给分段的消息截然相反。出于多种原因，您可能想要发送事务性消息而不是基于活动的消息。例如，您可以在买家下订单时通过电子邮件发送订单确认。您还可以通过短信或语音发送一次性密码，买家可使用它完成为您的服务创建账户的过程。

本部分包括多种编程语言的示例代码，您可以使用这些示例来开始发送事务性电子邮件、SMS 消息和语音消息。

本节中的主题：

- [发送事务性电子邮件消息](#)
- [发送 SMS 消息](#)
- [发送语音消息](#)
- [发送推送通知](#)

## 发送事务性电子邮件消息

本部分提供完整的代码示例，您可以使用它们来通过 Amazon Pinpoint 发送事务性电子邮件消息：

- [通过使用 Amazon Pinpoint API 中的 SendMessages 操作](#)：您可以使用 Amazon Pinpoint API 中的 SendMessages 操作在 Amazon Pinpoint 支持的所有渠道（包括推送通知、短信、语音和电子邮件渠道）中发送消息。

使用此操作的好处是，在所有渠道发送消息的请求语法都非常类似。这样，您可以更容易地重新利用现有代码。SendMessages 操作还允许您替换电子邮件消息中的内容，允许您发送电子邮件到 Amazon Pinpoint 端点 ID，而不是到特定的电子邮件地址。

本部分包括多种编程语言的示例代码，您可以使用这些示例来开始发送事务性电子邮件。

本节中的主题：

- [选择一种电子邮件发送方法](#)
- [在 Amazon Pinpoint 与 Amazon Simple Email Service \(SES\) 之间进行选择](#)
- [使用 Amazon Pinpoint API 发送电子邮件](#)

## 选择一种电子邮件发送方法

发送事务性电子邮件的最佳方法因具体使用案例而定。例如，如果您需要使用第三方应用程序发送电子邮件，或者您的编程语言没有可用的 AWS 软件开发工具包，则您可能必须使用 SMTP 接口。如果您想在 Amazon Pinpoint 支持的其他渠道中发送消息，并且想要使用一致的代码提出这些请求，则应使用 Amazon Pinpoint API 中的 `SendMessage` 操作。

## 在 Amazon Pinpoint 与 Amazon Simple Email Service (SES) 之间进行选择

如果您发送大量交易性电子邮件（如购买确认信息或密码重置消息），请考虑使用 Amazon SES。Amazon SES 具有 API 和 SMTP 接口，这两种接口都非常适合从您的应用程序或服务发送电子邮件。它还提供了其他电子邮件功能，包括电子邮件接收功能、配置集和发送授权功能。

Amazon SES 还包括一个 SMTP 接口，您可以将其与现有的第三方应用程序集成，包括客户关系管理 (CRM) 服务（如 Salesforce）。有关使用 Amazon SES 发送电子邮件的更多信息，请参阅 [Amazon Simple Email Service 开发人员指南](#) 以了解更多信息。

## 使用 Amazon Pinpoint API 发送电子邮件

本部分包含完整代码示例，您可以参照它们，使用 AWS SDK，通过 Amazon Pinpoint API 来发送电子邮件。

### C#

参照此示例，使用 [AWS SDK for .NET](#) 发送电子邮件。此示例假定您已安装和配置 AWS SDK for .NET。有关更多信息，请参阅《AWS SDK for .NET 开发人员指南》中的 [AWS SDK for .NET 入门](#)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for .NET 开发人员指南》中的 [配置 AWS 凭证](#)。

此代码示例使用 AWS SDK for .NET 版本 3.3.29.13 和 .NET Core 运行时版本 2.1.2 进行测试。

```
using System;
using System.Collections.Generic;
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;

namespace PinpointEmailSendMessageAPI
{
```

```
class MainClass
{
    // The AWS Region that you want to use to send the email. For a list of
    // AWS Regions where the Amazon Pinpoint API is available, see
    // https://docs.aws.amazon.com/pinpoint/latest/apireference/
    static string region = "us-west-2";

    // The "From" address. This address has to be verified in Amazon Pinpoint
    // in the region you're using to send email.
    static string senderAddress = "sender@example.com";

    // The address on the "To" line. If your Amazon Pinpoint account is in
    // the sandbox, this address also has to be verified.
    static string toAddress = "recipient@example.com";

    // The Amazon Pinpoint project/application ID to use when you send this
message.
    // Make sure that the SMS channel is enabled for the project or application
    // that you choose.
    static string appId = "ce796be37f32f178af652b26eexample";

    // The subject line of the email.
    static string subject = "Amazon Pinpoint Email test";

    // The body of the email for recipients whose email clients don't
    // support HTML content.
    static string textBody = @"Amazon Pinpoint Email Test (.NET)
-----
This email was sent using the Amazon Pinpoint API using the AWS SDK for .NET.";

    // The body of the email for recipients whose email clients support
    // HTML content.
    static string htmlBody = @"<html>
<head></head>
<body>
<h1>Amazon Pinpoint Email Test (AWS SDK for .NET)</h1>
<p>This email was sent using the
<a href='https://aws.amazon.com/pinpoint/'>Amazon Pinpoint</a> API
using the <a href='https://aws.amazon.com/sdk-for-net/'>
AWS SDK for .NET</a>.</p>
</body>
</html>";

    // The character encoding the you want to use for the subject line and
```

```
// message body of the email.
static string charset = "UTF-8";

public static void Main(string[] args)
{
    using (var client = new
AmazonPinpointClient(RegionEndpoint.GetBySystemName(region)))
    {
        var sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses = new Dictionary<string, AddressConfiguration>
                {
                    {
                        toAddress,
                        new AddressConfiguration
                        {
                            ChannelType = "EMAIL"
                        }
                    }
                },
                MessageConfiguration = new DirectMessageConfiguration
                {
                    EmailMessage = new EmailMessage
                    {
                        FromAddress = senderAddress,
                        SimpleEmail = new SimpleEmail
                        {
                            HtmlPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = htmlBody
                            },
                            TextPart = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = textBody
                            },
                            Subject = new SimpleEmailPart
                            {
                                Charset = charset,
                                Data = subject
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```
        }
    }
}

};
try
{
    Console.WriteLine("Sending message...");
    SendMessagesResponse response =
client.SendMessages(sendRequest);
    Console.WriteLine("Message sent!");
}
catch (Exception ex)
{
    Console.WriteLine("The message wasn't sent. Error message: " +
ex.Message);
}
}
}
}
```

## Java

参照此示例，使用 [AWS SDK for Java](#) 发送电子邮件。此示例假定您已安装和配置 AWS SDK for Java 2.x。有关更多信息，请参阅《AWS SDK for Java 2.x 开发人员指南》中的[入门](#)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Java 开发人员指南》中的[设置默认凭证和区域](#)。

此代码示例使用 AWS SDK for Java 版本 2.3.1 和 OpenJDK 版本 11.0.1 进行测试。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.SendMessageRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessageRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
```



```
public class SendMessage {

    // The character encoding the you want to use for the subject line and
    // message body of the email.
    public static String charset = "UTF-8";

    // The body of the email for recipients whose email clients support HTML
    content.
    static final String body = """"
        Amazon Pinpoint test (AWS SDK for Java 2.x)

        This email was sent through the Amazon Pinpoint Email API using the AWS SDK
    for Java 2.x

        """";

    public static void main(String[] args) {
        final String usage = """"

            Usage:    <subject> <appId> <senderAddress>
    <toAddress>

            Where:
                subject - The email subject to use.
                senderAddress - The from address. This address has to be verified in
    Amazon Pinpoint in the region you're using to send email\s
                toAddress - The to address. This address has to be verified in Amazon
    Pinpoint in the region you're using to send email\s
            """";

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String subject = args[0];
        String senderAddress = args[1];
        String toAddress = args[2];
        System.out.println("Sending a message");
        PinpointEmailClient pinpoint = PinpointEmailClient.builder()
            .region(Region.US_EAST_1)
            .build();

        sendEmail(pinpoint, subject, senderAddress, toAddress);
    }
}
```

```
        System.out.println("Email was sent");
        pinpoint.close();
    }

    public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
subject, String senderAddress, String toAddress) {
        try {
            Content content = Content.builder()
                .data(body)
                .build();

            Body messageBody = Body.builder()
                .text(content)
                .build();

            Message message = Message.builder()
                .body(messageBody)
                .subject(Content.builder().data(subject).build())
                .build();

            Destination destination = Destination.builder()
                .toAddresses(toAddress)
                .build();

            EmailContent emailContent = EmailContent.builder()
                .simple(message)
                .build();

            SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
                .fromEmailAddress(senderAddress)
                .destination(destination)
                .content(emailContent)
                .build();

            pinpointEmailClient.sendEmail(sendEmailRequest);
            System.out.println("Message Sent");

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [SendMessage.java](#)。

## JavaScript (Node.js)

参考此示例，通过使用 [AWS SDK for JavaScript \(Node.js\)](#) 来发送电子邮件。此示例假定您已安装和配置该 SDK for JavaScript (Node.js)。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[入门](#)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[设置凭证](#)。

此代码示例使用 SDK for JavaScript ( Node.js 2.388.0 版和 Node.js 11.7.0 版 ) 进行测试。

```
"use strict";

const AWS = require("aws-sdk");

// The AWS Region that you want to use to send the email. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const aws_region = "us-west-2";

// The "From" address. This address has to be verified in Amazon Pinpoint
// in the region that you use to send email.
const senderAddress = "sender@example.com";

// The address on the "To" line. If your Amazon Pinpoint account is in
// the sandbox, this address also has to be verified.
var toAddress = "recipient@example.com";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
const appId = "ce796be37f32f178af652b26eexample";

// The subject line of the email.
var subject = "Amazon Pinpoint (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
Node.js.`;
```

```
For more information, see https://aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for JavaScript in Node.js)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>the Amazon Pinpoint API</a> using the
    <a href='https://aws.amazon.com/sdk-for-node-js/'>
      AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding the you want to use for the subject line and
// message body of the email.
var charset = "UTF-8";

// Specify that you're using a shared credentials file.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: appId,
  MessageRequest: {
    Addresses: {
      [toAddress]: {
        ChannelType: "EMAIL",
      },
    },
  },
  MessageConfiguration: {
    EmailMessage: {
      FromAddress: senderAddress,
      SimpleEmail: {
        Subject: {
          Charset: charset,
          Data: subject,
```

```
    },
    HtmlPart: {
      Charset: charset,
      Data: body_html,
    },
    TextPart: {
      Charset: charset,
      Data: body_text,
    },
  },
},
},
},
};

//Try to send the email.
pinpoint.sendMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
  } else {
    console.log(
      "Email sent! Message ID: ",
      data["MessageResponse"]["Result"][toAddress]["MessageId"]
    );
  }
});
```

## Python

参照此示例，使用 [AWS SDK for Python \(Boto3\)](#) 发送电子邮件。此示例假定您已安装和配置该 SDK for Python (Boto3)。有关更多信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的 [快速入门](#)。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
```

```

def send_email_message(
    pinpoint_client,
    app_id,
    sender,
    to_addresses,
    char_set,
    subject,
    html_message,
    text_message,
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project ID to use when you send this message.
    :param sender: The "From" address. This address must be verified in
                   Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
    account
                       is in the sandbox, these addresses must be verified.
    :param char_set: The character encoding to use for the subject line and message
                     body of the email.
    :param subject: The subject line of the email.
    :param html_message: The body of the email for recipients whose email clients
    can
                       display HTML content.
    :param text_message: The body of the email for recipients whose email clients
                       don't support HTML content.
    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
    to_addresses
                },
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": sender,
                        "SimpleEmail": {
                            "Subject": {"Charset": char_set, "Data": subject},
                            "HtmlPart": {"Charset": char_set, "Data": html_message},

```

```

        "TextPart": {"Charset": char_set, "Data": text_message},
    },
}
),
except ClientError:
    logger.exception("Couldn't send email.")
    raise
else:
    return {
        to_address: message["MessageId"]
        for to_address, message in response["MessageResponse"]["Result"].items()
    }

def main():
    app_id = "ce796be37f32f178af652b26eexample"
    sender = "sender@example.com"
    to_address = "recipient@example.com"
    char_set = "UTF-8"
    subject = "Amazon Pinpoint Test (SDK for Python (Boto3))"
    text_message = """Amazon Pinpoint Test (SDK for Python)
    -----
    This email was sent with Amazon Pinpoint using the AWS SDK for Python (Boto3).
    For more information, see https://aws.amazon.com/sdk-for-python/
    """
    html_message = """<html>
    <head></head>
    <body>
    <h1>Amazon Pinpoint Test (SDK for Python (Boto3))</h1>
    <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>Amazon Pinpoint</a> using the
    <a href='https://aws.amazon.com/sdk-for-python/'>
    AWS SDK for Python (Boto3)</a>.</p>
    </body>
    </html>
    """

    print("Sending email.")
    message_ids = send_email_message(
        boto3.client("pinpoint"),
        app_id,
        sender,

```

```
        [to_address],
        char_set,
        subject,
        html_message,
        text_message,
    )
    print(f"Message sent! Message IDs: {message_ids}")

if __name__ == "__main__":
    main()
```

您也可以使用消息模板发送电子邮件消息，如以下示例所示：

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_templated_email_message(
    pinpoint_client, project_id, sender, to_addresses, template_name,
    template_version
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: The Amazon Pinpoint project ID to use when you send this
message.
    :param sender: The "From" address. This address must be verified in
        Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
        account is in the sandbox, these addresses must be
verified.
    :param template_name: The name of the email template to use when sending the
message.
    :param template_version: The version number of the message template.

    :return: A dict of to_addresses and their message IDs.
    """
    try:
```



```
response = pinpoint_client.send_messages(
    ApplicationId=project_id,
    MessageRequest={
        "Addresses": {
            to_address: {"ChannelType": "EMAIL"} for to_address in
to_addresses
        },
        "MessageConfiguration": {"EmailMessage": {"FromAddress": sender}},
        "TemplateConfiguration": {
            "EmailTemplate": {
                "Name": template_name,
                "Version": template_version,
            }
        },
    },
)
except ClientError:
    logger.exception("Couldn't send email.")
    raise
else:
    return {
        to_address: message["MessageId"]
        for to_address, message in response["MessageResponse"]["Result"].items()
    }

def main():
    project_id = "296b04b342374fceb661bf494example"
    sender = "sender@example.com"
    to_addresses = ["recipient@example.com"]
    template_name = "My_Email_Template"
    template_version = "1"

    print("Sending email.")
    message_ids = send_templated_email_message(
        boto3.client("pinpoint"),
        project_id,
        sender,
        to_addresses,
        template_name,
        template_version,
    )
    print(f"Message sent! Message IDs: {message_ids}")
```

```
if __name__ == "__main__":
    main()
```

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的[凭证](#)。

## 发送 SMS 消息

您可以使用 Amazon Pinpoint API 将短信（文本消息）发送到特定电话号码或端点 ID。本部分包含完整代码示例，您可以参照它们，使用 AWS SDK，通过 Amazon Pinpoint API 来发送短信。

### C#

参照此示例，使用 [AWS SDK for .NET](#) 发送 SMS 消息。此示例假定您已安装和配置 AWS SDK for .NET。有关更多信息，请参阅《AWS SDK for .NET 开发人员指南》中的[入门](#)。

此示例假定您正在使用共享凭证文件来指定现有 IAM 用户的访问密钥和私密访问密钥。有关更多信息，请参阅《AWS SDK for .NET 开发人员指南》中的[配置 AWS 凭证](#)。

```
using System;
using System.Collections.Generic;
using Amazon;
using Amazon.Pinpoint;
using Amazon.Pinpoint.Model;

namespace SendMessage
{
    class MainClass
    {
        // The AWS Region that you want to use to send the message. For a list of
        // AWS Regions where the Amazon Pinpoint API is available, see
        // https://docs.aws.amazon.com/pinpoint/latest/apireference/
        private static readonly string region = "us-east-1";

        // The phone number or short code to send the message from. The phone number
        // or short code that you specify has to be associated with your Amazon
        Pinpoint
        // account. For best results, specify long codes in E.164 format.
        private static readonly string originationNumber = "+12065550199";
```

```
// The recipient's phone number. For best results, you should specify the
// phone number in E.164 format.
private static readonly string destinationNumber = "+14255550142";

// The content of the SMS message.
private static readonly string message = "This message was sent through
Amazon Pinpoint"
    + "using the AWS SDK for .NET. Reply STOP to opt out.";

// The Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
private static readonly string appId = "ce796be37f32f178af652b26eexample";

// The type of SMS message that you want to send. If you plan to send
// time-sensitive content, specify TRANSACTIONAL. If you plan to send
// marketing-related content, specify PROMOTIONAL.
private static readonly string messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
private static readonly string registeredKeyword = "myKeyword";

// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
private static readonly string senderId = "mySenderId";

public static void Main(string[] args)
{
    using (AmazonPinpointClient client = new
AmazonPinpointClient(RegionEndpoint.GetBySystemName(region)))
    {
        SendMessagesRequest sendRequest = new SendMessagesRequest
        {
            ApplicationId = appId,
            MessageRequest = new MessageRequest
            {
                Addresses = new Dictionary<string, AddressConfiguration>
                {
                    {
                        destinationNumber,
                        new AddressConfiguration
                        {

```

```
        ChannelType = "SMS"
    }
}
},
MessageConfiguration = new DirectMessageConfiguration
{
    SMSMessage = new SMSMessage
    {
        Body = message,
        MessageType = messageType,
        OriginationNumber = originationNumber,
        SenderId = senderId,
        Keyword = registeredKeyword
    }
}
};
try
{
    Console.WriteLine("Sending message...");
    SendMessagesResponse response =
client.SendMessages(sendRequest);
    Console.WriteLine("Message sent!");
}
catch (Exception ex)
{
    Console.WriteLine("The message wasn't sent. Error message: " +
ex.Message);
}
}
}
```

## Java

参照此示例，使用 [AWS SDK for Java](#) 发送 SMS 消息。此示例假定您已安装和配置该 SDK for Java。有关更多信息，请参阅《AWS SDK for Java 开发人员指南》中的 [入门](#)。

此示例假定您正在使用共享凭证文件来指定现有 IAM 用户的访问密钥和私密访问密钥。有关更多信息，请参阅《AWS SDK for Java 开发人员指南》中的 [设置默认凭证和区域](#)。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendMessage {

    // The type of SMS message that you want to send. If you plan to send
    // time-sensitive content, specify TRANSACTIONAL. If you plan to send
    // marketing-related content, specify PROMOTIONAL.
    public static String messageType = "TRANSACTIONAL";
```

```
// The registered keyword associated with the originating short code.
public static String registeredKeyword = "myKeyword";

// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html
public static String senderId = "MySenderId";

public static void main(String[] args) {
    final String usage = ""

        Usage:  <message> <appId> <originationNumber>
<destinationNumber>\s

        Where:
            message - The body of the message to send.
            appId - The Amazon Pinpoint project/application ID
to use when you send this message.
            originationNumber - The phone number or short code
that you specify has to be associated with your Amazon Pinpoint account. For best
results, specify long codes in E.164 format (for example, +1-555-555-5654).
            destinationNumber - The recipient's phone number.
For best results, you should specify the phone number in E.164 format (for example,
+1-555-555-5654).\s

        """;

    if (args.length != 4) {
        System.out.println(usage);
        System.exit(1);
    }

    String message = args[0];
    String appId = args[1];
    String originationNumber = args[2];
    String destinationNumber = args[3];
    System.out.println("Sending a message");
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    sendSMSMessage(pinpoint, message, appId, originationNumber,
destinationNumber);
    pinpoint.close();
}
```

```
    }

    public static void sendSMSMessage(PinpointClient pinpoint, String message,
String appId,
        String originationNumber,
        String destinationNumber) {
    try {
        Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
        AddressConfiguration addConfig =
AddressConfiguration.builder()
            .channelType(ChannelType.SMS)
            .build();

        addressMap.put(destinationNumber, addConfig);
        SMSMessage smsMessage = SMSMessage.builder()
            .body(message)
            .messageType(messageType)
            .originationNumber(originationNumber)
            .senderId(senderId)
            .keyword(registeredKeyword)
            .build();

        // Create a DirectMessageConfiguration object.
        DirectMessageConfiguration direct =
DirectMessageConfiguration.builder()
            .smsMessage(smsMessage)
            .build();

        MessageRequest msgReq = MessageRequest.builder()
            .addresses(addressMap)
            .messageConfiguration(direct)
            .build();

        // create a SendMessagesRequest object
        SendMessagesRequest request = SendMessagesRequest.builder()
            .applicationId(appId)
            .messageRequest(msgReq)
            .build();

        SendMessagesResponse response =
pinpoint.sendMessages(request);
        MessageResponse msg1 = response.messageResponse();
        Map map1 = msg1.result();
    }
}
```

```
        // Write out the result of sendMessage.
        map1.forEach((k, v) -> System.out.println((k + ":" + v)));

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [SendMessage.java](#)。

### JavaScript (Node.js)

参考此示例，通过使用 [AWS SDK for JavaScript \(Node.js\)](#) 来发送短信。此示例假定您已安装和配置该 SDK for JavaScript (Node.js)。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[入门](#)。

此示例假定您正在使用共享凭证文件来指定现有 IAM 用户的访问密钥和私密访问密钥。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[设置凭证](#)。

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/.
var aws_region = "us-east-1";

// The phone number or short code to send the message from. The phone number
// or short code that you specify has to be associated with your Amazon Pinpoint
// account. For best results, specify long codes in E.164 format.
var originationNumber = "+12065550199";

// The recipient's phone number. For best results, you should specify the
// phone number in E.164 format.
var destinationNumber = "+14255550142";

// The content of the SMS message.
var message =
    "This message was sent through Amazon Pinpoint " +
```



```
"using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
"opt out.";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
var applicationId = "ce796be37f32f178af652b26eexample";

// The type of SMS message that you want to send. If you plan to send
// time-sensitive content, specify TRANSACTIONAL. If you plan to send
// marketing-related content, specify PROMOTIONAL.
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";

// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-countries.html
var senderId = "MySenderId";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: applicationId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
  },
  MessageConfiguration: {
    SMSMessage: {
      Body: message,
```

```
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
    },
},
};

//Try to send the message.
pinpoint.sendMessage(params, function (err, data) {
    // If something goes wrong, print an error message.
    if (err) {
        console.log(err.message);
        // Otherwise, show the unique ID for the message.
    } else {
        console.log(
            "Message sent! " +
            data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"]
        );
    }
});
```

## Python

参照此示例，使用 [AWS SDK for Python \(Boto3\)](#) 发送 SMS 消息。此示例假定您已安装和配置该 SDK for Python。有关更多信息，请参阅《AWS SDK for Python (Boto3) 入门》中的[快速入门](#)。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_sms_message(
    pinpoint_client,
    app_id,
    origination_number,
    destination_number,
    message,
```

```
message_type,
):
    """
    Sends an SMS message with Amazon Pinpoint.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project/application ID to use when you send
        this message. The SMS channel must be enabled for the project or
        application.
    :param destination_number: The recipient's phone number in E.164 format.
    :param origination_number: The phone number to send the message from. This phone
        number must be associated with your Amazon Pinpoint
        account and be in E.164 format.
    :param message: The content of the SMS message.
    :param message_type: The type of SMS message that you want to send. If you send
        time-sensitive content, specify TRANSACTIONAL. If you send
        marketing-related content, specify PROMOTIONAL.
    :return: The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "Body": message,
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
            },
        )
    except ClientError:
        logger.exception("Couldn't send message.")
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]

def main():
    app_id = "ce796be37f32f178af652b26eexample"
    origination_number = "+12065550199"
```

```
destination_number = "+14255550142"
message = (
    "This is a sample message sent from Amazon Pinpoint by using the AWS SDK for
"
    "Python (Boto 3).")
)
message_type = "TRANSACTIONAL"

print("Sending SMS message.")
message_id = send_sms_message(
    boto3.client("pinpoint"),
    app_id,
    origination_number,
    destination_number,
    message,
    message_type,
)
print(f"Message sent! Message ID: {message_id}.")

if __name__ == "__main__":
    main()
```

您也可以使用消息模板发送短信，如以下示例所示：

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_templated_sms_message(
    pinpoint_client,
    project_id,
    destination_number,
    message_type,
    origination_number,
    template_name,
    template_version,
):
    """
    Sends an SMS message to a specific phone number using a pre-defined template.
```

```
:param pinpoint_client: A Boto3 Pinpoint client.
:param project_id: An Amazon Pinpoint project (application) ID.
:param destination_number: The phone number to send the message to.
:param message_type: The type of SMS message (promotional or transactional).
:param origination_number: The phone number that the message is sent from.
:param template_name: The name of the SMS template to use when sending the
message.
:param template_version: The version number of the message template.

:return The ID of the message.
"""
try:
    response = pinpoint_client.send_messages(
        ApplicationId=project_id,
        MessageRequest={
            "Addresses": {destination_number: {"ChannelType": "SMS"}},
            "MessageConfiguration": {
                "SMSMessage": {
                    "MessageType": message_type,
                    "OriginationNumber": origination_number,
                }
            },
            "TemplateConfiguration": {
                "SMSTemplate": {"Name": template_name, "Version":
template_version}
            },
        },
    )

except ClientError:
    logger.exception("Couldn't send message.")
    raise
else:
    return response["MessageResponse"]["Result"][destination_number]
["MessageId"]

def main():
    region = "us-east-1"
    origination_number = "+18555550001"
    destination_number = "+14255550142"
    project_id = "7353f53e6885409fa32d07cedexample"
    message_type = "TRANSACTIONAL"
```

```
template_name = "My_SMS_Template"
template_version = "1"
message_id = send_templated_sms_message(
    boto3.client("pinpoint", region_name=region),
    project_id,
    destination_number,
    message_type,
    origination_number,
    template_name,
    template_version,
)
print(f"Message sent! Message ID: {message_id}.")

if __name__ == "__main__":
    main()
```

此示例假定您正在使用共享凭证文件来指定现有 IAM 用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的[凭证](#)。

## 发送语音消息

您可以使用 Amazon Pinpoint API 将语音消息发送给特定电话号码。本部分包含完整代码示例，您可以参照它们，使用 AWS SDK，通过 Amazon Pinpoint SMS 和 Voice API 来发送语音消息。

### Java

参照此示例，使用 [AWS SDK for Java](#) 发送语音消息。此示例假定您已安装和配置该 SDK for Java。有关更多信息，请参阅《AWS SDK for Java 开发人员指南》中的[入门](#)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Java 开发人员指南》中的[设置用于开发的 AWS 凭证和区域](#)。

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
```

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
```

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendVoiceMessage {

    // The Amazon Polly voice that you want to use to send the message. For a
    list
    // of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
    static final String voiceName = "Matthew";

    // The language to use when sending the message. For a list of supported
    // languages, see
    // https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
    static final String languageCode = "en-US";

    // The content of the message. This example uses SSML to customize and
    control
    // certain aspects of the message, such as by adding pauses and changing
    // phonation. The message can't contain any line breaks.
```

```
static final String ssm1Message = "<speak>This is a test message sent from "
    + "<emphasis>Amazon Pinpoint</emphasis> "
    + "using the <break strength='weak'/>AWS "
    + "SDK for Java. "
    + "<amazon:effect phonation='soft'>Thank "
    + "you for listening.</amazon:effect></speak>";

public static void main(String[] args) {

    final String usage = ""

        Usage:  <originationNumber> <destinationNumber>\s

        Where:
            originationNumber - The phone number or short code
that you specify has to be associated with your Amazon Pinpoint account. For best
results, specify long codes in E.164 format (for example, +1-555-555-5654).
            destinationNumber - The recipient's phone number.
For best results, you should specify the phone number in E.164 format (for example,
+1-555-555-5654).\s

        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String originationNumber = args[0];
    String destinationNumber = args[1];
    System.out.println("Sending a voice message");

    // Set the content type to application/json.
    List<String> listVal = new ArrayList<>();
    listVal.add("application/json");
    Map<String, List<String>> values = new HashMap<>();
    values.put("Content-Type", listVal);

    ClientOverrideConfiguration config2 =
ClientOverrideConfiguration.builder()
        .headers(values)
        .build();

    PinpointSmsVoiceClient client = PinpointSmsVoiceClient.builder()
        .overrideConfiguration(config2)
```



```
                .region(Region.US_EAST_1)
                .build();

        sendVoiceMsg(client, originationNumber, destinationNumber);
        client.close();
    }

    public static void sendVoiceMsg(PinpointSmsVoiceClient client, String
originationNumber,
        String destinationNumber) {
        try {
            SSMLMessageType ssmlMessageType = SSMLMessageType.builder()
                .languageCode(languageCode)
                .text(ssmlMessage)
                .voiceId(voiceName)
                .build();

            VoiceMessageContent content = VoiceMessageContent.builder()
                .ssmlMessage(ssmlMessageType)
                .build();

            SendVoiceMessageRequest voiceMessageRequest =
SendVoiceMessageRequest.builder()
                .destinationPhoneNumber(destinationNumber)
                .originationPhoneNumber(originationNumber)
                .content(content)
                .build();

            client.sendVoiceMessage(voiceMessageRequest);
            System.out.println("The message was sent successfully.");

        } catch (PinpointSmsVoiceException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

有关完整的 SDK 示例，请参阅 [GitHub](#) 上的 [SendVoiceMessage.java](#)。

## JavaScript (Node.js)

参考此示例，通过使用 AWS SDK for JavaScript (Node.js) 来发送语音消息。此示例假定您已安装和配置该 SDK for JavaScript (Node.js)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[设置凭证](#)。

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the voice message. For a list of
// AWS Regions where the Amazon Pinpoint SMS and Voice API is available, see
// https://docs.aws.amazon.com/pinpoint-sms-voice/latest/APIReference/
var aws_region = "us-east-1";

// The phone number that the message is sent from. The phone number that you
// specify has to be associated with your Amazon Pinpoint account. For best results,
// you
// should specify the phone number in E.164 format.
var originationNumber = "+12065550110";

// The recipient's phone number. For best results, you should specify the phone
// number in E.164 format.
var destinationNumber = "+12065550142";

// The language to use when sending the message. For a list of supported
// languages, see https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
var languageCode = "en-US";

// The Amazon Polly voice that you want to use to send the message. For a list
// of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
var voiceId = "Matthew";

// The content of the message. This example uses SSML to customize and control
// certain aspects of the message, such as the volume or the speech rate.
// The message can't contain any line breaks.
var ssmlMessage =
  "<speak>" +
  "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> " +
  "using the <break strength='weak'/>AWS SDK for JavaScript in Node.js. " +
  "<amazon:effect phonation='soft'>Thank you for listening." +
  "</amazon:effect>" +
  "</speak>";

// The phone number that you want to appear on the recipient's device. The phone
```

```
// number that you specify has to be associated with your Amazon Pinpoint account.
var callerId = "+12065550199";

// The configuration set that you want to use to send the message.
var configurationSet = "ConfigSet";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpointSMSVoice = new AWS.PinpointSMSVoice();

var params = {
  CallerId: callerId,
  ConfigurationSetName: configurationSet,
  Content: {
    SSMLMessage: {
      LanguageCode: languageCode,
      Text: ssmlMessage,
      VoiceId: voiceId,
    },
  },
  DestinationPhoneNumber: destinationNumber,
  OriginationPhoneNumber: originationNumber,
};

//Try to send the message.
pinpointSMSVoice.sendVoiceMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
  } else {
    console.log("Message sent! Message ID: " + data["MessageId"]);
  }
});
```

## Python

参照此示例，使用 AWS SDK for Python (Boto3) 发送语音消息。此示例假定您已安装和配置该 SDK for Python (Boto3)。

此示例假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的[凭证](#)。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_voice_message(
    sms_voice_client,
    origination_number,
    caller_id,
    destination_number,
    language_code,
    voice_id,
    ssml_message,
):
    """
    Sends a voice message using speech synthesis provided by Amazon Polly.

    :param sms_voice_client: A Boto3 PinpointSMSVoice client.
    :param origination_number: The phone number that the message is sent from.
                               The phone number must be associated with your Amazon
                               Pinpoint account and be in E.164 format.
    :param caller_id: The phone number that you want to appear on the recipient's
                      device. The phone number must be associated with your Amazon
                      Pinpoint account and be in E.164 format.
    :param destination_number: The recipient's phone number. Specify the phone
                               number in E.164 format.
    :param language_code: The language to use when sending the message.
    :param voice_id: The Amazon Polly voice that you want to use to send the
    message.
    :param ssml_message: The content of the message. This example uses SSML to
    control
                           certain aspects of the message, such as the volume and the
```

```
        speech rate. The message must not contain line breaks.
:return: The ID of the message.
"""
try:
    response = sms_voice_client.send_voice_message(
        DestinationPhoneNumber=destination_number,
        OriginationPhoneNumber=origination_number,
        CallerId=caller_id,
        Content={
            "SSMLMessage": {
                "LanguageCode": language_code,
                "VoiceId": voice_id,
                "Text": ssml_message,
            }
        },
    )
except ClientError:
    logger.exception(
        "Couldn't send message from %s to %s.",
        origination_number,
        destination_number,
    )
    raise
else:
    return response["MessageId"]

def main():
    origination_number = "+12065550110"
    caller_id = "+12065550199"
    destination_number = "+12065550142"
    language_code = "en-US"
    voice_id = "Matthew"
    ssml_message = (
        "<speak>"
        "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> "
        "using the <break strength='weak'/>AWS SDK for Python (Boto3). "
        "<amazon:effect phonation='soft'>Thank you for listening."
        "</amazon:effect>"
        "</speak>"
    )
    print(f"Sending voice message from {origination_number} to "
          f"{destination_number}.")
    message_id = send_voice_message(
```

```
boto3.client("pinpoint-sms-voice"),
    origination_number,
    caller_id,
    destination_number,
    language_code,
    voice_id,
    ssml_message,
)
print(f"Message sent!\nMessage ID: {message_id}")

if __name__ == "__main__":
    main()
```

## 发送推送通知

Amazon Pinpoint API 可以将事务性推送通知发送到特定设备标识符。本部分包含完整代码示例，您可以参照它们，使用 AWS SDK，通过 Amazon Pinpoint API 来发送推送通知。

您可以使用这些示例，通过 Amazon Pinpoint 支持的任何推送通知服务来发送推送通知。目前，Amazon Pinpoint 支持以下渠道：Firebase Cloud Messaging (FCM)、Apple Push Notification service (APNs)、百度云推送和 Amazon Device Messaging (ADM)。

### Note

当您通过 Firebase Cloud Messaging (FCM) 服务发送推送通知时，请在对 Amazon Pinpoint API 的调用中使用服务名称 GCM。虽然 Google 已于 2018 年 4 月 10 日停止了 Google Cloud Messaging (GCM) 服务，但 Amazon Pinpoint API 将 GCM 服务名称用于其通过 FCM 服务发送的消息，这样做有助于维护与 GCM 服务停止前编写的 API 代码的兼容性。

### JavaScript (Node.js)

参考此示例，通过使用 AWS SDK for JavaScript (Node.js) 来发送推送通知。此示例假定您已安装和配置该 SDK for JavaScript (Node.js)。

此示例还假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for JavaScript (Node.js) 开发人员指南》中的[设置凭证](#)。

```
'use strict';
```

```
const AWS = require('aws-sdk');

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const region = 'us-east-1';

// The title that appears at the top of the push notification.
var title = 'Test message sent from Amazon Pinpoint.';

// The content of the push notification.
var message = 'This is a sample message sent from Amazon Pinpoint by using the '
    + 'AWS SDK for JavaScript in Node.js';

// The Amazon Pinpoint project ID that you want to use when you send this
// message. Make sure that the push channel is enabled for the project that
// you choose.
var applicationId = 'ce796be37f32f178af652b26eexample';

// An object that contains the unique token of the device that you want to send
// the message to, and the push service that you want to use to send the message.
var recipient = {
    'token': 'a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0',
    'service': 'GCM'
};

// The action that should occur when the recipient taps the message. Possible
// values are OPEN_APP (opens the app or brings it to the foreground),
// DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
// specific URL in the device's web browser.)
var action = 'URL';

// This value is only required if you use the URL action. This variable contains
// the URL that opens in the recipient's web browser.
var url = 'https://www.example.com';

// The priority of the push notification. If the value is 'normal', then the
// delivery of the message is optimized for battery usage on the recipient's
// device, and could be delayed. If the value is 'high', then the notification is
// sent immediately, and might wake a sleeping device.
var priority = 'normal';

// The amount of time, in seconds, that the push notification service provider
```

```
// (such as FCM or APNS) should attempt to deliver the message before dropping
// it. Not all providers allow you specify a TTL value.
var ttl = 30;

// Boolean that specifies whether the notification is sent as a silent
// notification (a notification that doesn't display on the recipient's device).
var silent = false;

function CreateMessageRequest() {
  var token = recipient['token'];
  var service = recipient['service'];
  if (service == 'GCM') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'GCM'
        }
      },
      'MessageConfiguration': {
        'GCMMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
          'TimeToLive': ttl,
          'Url': url
        }
      }
    };
  } else if (service == 'APNS') {
    var messageRequest = {
      'Addresses': {
        [token]: {
          'ChannelType' : 'APNS'
        }
      },
      'MessageConfiguration': {
        'APNSMessage': {
          'Action': action,
          'Body': message,
          'Priority': priority,
          'SilentPush': silent,
          'Title': title,
```



```
        'TimeToLive': ttl,
        'Url': url
    }
}
};
} else if (service == 'BAIDU') {
var messageRequest = {
    'Addresses': {
        [token]: {
            'ChannelType' : 'BAIDU'
        }
    },
    'MessageConfiguration': {
        'BaiduMessage': {
            'Action': action,
            'Body': message,
            'SilentPush': silent,
            'Title': title,
            'TimeToLive': ttl,
            'Url': url
        }
    }
};
} else if (service == 'ADM') {
var messageRequest = {
    'Addresses': {
        [token]: {
            'ChannelType' : 'ADM'
        }
    },
    'MessageConfiguration': {
        'ADMMessage': {
            'Action': action,
            'Body': message,
            'SilentPush': silent,
            'Title': title,
            'Url': url
        }
    }
};
}

return messageRequest
}
```

```
function ShowOutput(data){
  if (data["MessageResponse"]["Result"][recipient["token"]]["DeliveryStatus"]
    == "SUCCESSFUL") {
    var status = "Message sent! Response information: ";
  } else {
    var status = "The message wasn't sent. Response information: ";
  }
  console.log(status);
  console.dir(data, { depth: null });
}
```

```
function SendMessage() {
  var token = recipient['token'];
  var service = recipient['service'];
  var messageRequest = CreateMessageRequest();

  // Specify that you're using a shared credentials file, and specify the
  // IAM profile to use.
  var credentials = new AWS.SharedIniFileCredentials({ profile: 'default' });
  AWS.config.credentials = credentials;

  // Specify the AWS Region to use.
  AWS.config.update({ region: region });

  //Create a new Pinpoint object.
  var pinpoint = new AWS.Pinpoint();
  var params = {
    "ApplicationId": applicationId,
    "MessageRequest": messageRequest
  };

  // Try to send the message.
  pinpoint.sendMessages(params, function(err, data) {
    if (err) console.log(err);
    else ShowOutput(data);
  });
}
```

```
SendMessage()
```

## Python

参考此示例，通过使用 AWS SDK for Python (Boto3) 发送推送通知。此示例假定您已安装和配置该 SDK for Python (Boto3)。

此示例还假定您正在使用共享凭证文件来指定现有用户的访问密钥和秘密访问密钥。有关更多信息，请参阅《AWS SDK for Python (Boto3) API 参考》中的[凭证](#)。

```
import json
import boto3
from botocore.exceptions import ClientError

# The AWS Region that you want to use to send the message. For a list of
# AWS Regions where the Amazon Pinpoint API is available, see
# https://docs.aws.amazon.com/pinpoint/latest/apireference/
region = "us-east-1"

# The title that appears at the top of the push notification.
title = "Test message sent from Amazon Pinpoint."

# The content of the push notification.
message = ("This is a sample message sent from Amazon Pinpoint by using the "
           "AWS SDK for Python (Boto3).")

# The Amazon Pinpoint project/application ID to use when you send this message.
# Make sure that the push channel is enabled for the project or application
# that you choose.
application_id = "ce796be37f32f178af652b26eexample"

# A dictionary that contains the unique token of the device that you want to send
# the
# message to, and the push service that you want to use to send the message.
recipient = {
    "token": "a0b1c2d3e4f5g6h7i8j9k0l1m2n3o4p5q6r7s8t9u0v1w2x3y4z5a6b7c8d8e9f0",
    "service": "GCM"
}

# The action that should occur when the recipient taps the message. Possible
# values are OPEN_APP (opens the app or brings it to the foreground),
# DEEP_LINK (opens the app to a specific page or interface), or URL (opens a
# specific URL in the device's web browser.)
action = "URL"
```

```
# This value is only required if you use the URL action. This variable contains
# the URL that opens in the recipient's web browser.
url = "https://www.example.com"

# The priority of the push notification. If the value is 'normal', then the
# delivery of the message is optimized for battery usage on the recipient's
# device, and could be delayed. If the value is 'high', then the notification is
# sent immediately, and might wake a sleeping device.
priority = "normal"

# The amount of time, in seconds, that the push notification service provider
# (such as FCM or APNS) should attempt to deliver the message before dropping
# it. Not all providers allow you specify a TTL value.
ttl = 30

# Boolean that specifies whether the notification is sent as a silent
# notification (a notification that doesn't display on the recipient's device).
silent = False

# Set the MessageType based on the values in the recipient variable.
def create_message_request():

    token = recipient["token"]
    service = recipient["service"]

    if service == "GCM":
        message_request = {
            'Addresses': {
                token: {
                    'ChannelType': 'GCM'
                }
            },
            'MessageConfiguration': {
                'GCMMessage': {
                    'Action': action,
                    'Body': message,
                    'Priority' : priority,
                    'SilentPush': silent,
                    'Title': title,
                    'TimeToLive': ttl,
                    'Url': url
                }
            }
        }
    }
```

```
elif service == "APNS":
    message_request = {
        'Addresses': {
            token: {
                'ChannelType': 'APNS'
            }
        },
        'MessageConfiguration': {
            'APNSMessage': {
                'Action': action,
                'Body': message,
                'Priority' : priority,
                'SilentPush': silent,
                'Title': title,
                'TimeToLive': ttl,
                'Url': url
            }
        }
    }
elif service == "BAIDU":
    message_request = {
        'Addresses': {
            token: {
                'ChannelType': 'BAIDU'
            }
        },
        'MessageConfiguration': {
            'BaiduMessage': {
                'Action': action,
                'Body': message,
                'SilentPush': silent,
                'Title': title,
                'TimeToLive': ttl,
                'Url': url
            }
        }
    }
elif service == "ADM":
    message_request = {
        'Addresses': {
            token: {
                'ChannelType': 'ADM'
            }
        },
    },
```

```
        'MessageConfiguration': {
            'ADMMMessage': {
                'Action': action,
                'Body': message,
                'SilentPush': silent,
                'Title': title,
                'Url': url
            }
        }
    }
else:
    message_request = None

return message_request

# Show a success or failure message, and provide the response from the API.
def show_output(response):
    if response['MessageResponse']['Result']['recipient["token"]']['DeliveryStatus']
    == "SUCCESSFUL":
        status = "Message sent! Response information:\n"
    else:
        status = "The message wasn't sent. Response information:\n"
    print(status, json.dumps(response,indent=4))

# Send the message through the appropriate channel.
def send_message():

    token = recipient["token"]
    service = recipient["service"]
    message_request = create_message_request()

    client = boto3.client('pinpoint',region_name=region)

    try:
        response = client.send_messages(
            ApplicationId=application_id,
            MessageRequest=message_request
        )
    except ClientError as e:
        print(e.response['Error']['Message'])
    else:
        show_output(response)
```

```
send_message()
```

# 在 Amazon Pinpoint 中创建自定义渠道

Amazon Pinpoint 包含对通过推送通知、电子邮件、短信和语音渠道发送消息的内置支持。还可以通过创建自定义渠道将 Amazon Pinpoint 配置为通过其他渠道发送消息。通过 Amazon Pinpoint 中的自定义渠道，您可以通过任何具有 API 的服务（包括第三方服务）发送消息。您可以通过使用 Webhook 或调用 AWS Lambda 函数来与 API 进行交互。

您向其发送自定义渠道活动的分段可以包含所有类型的端点（也即，其中 ChannelType 属性的值为 EMAIL、VOICE、SMS、CUSTOM 的端点，或各种推送通知端点类型之一）。

## 创建通过自定义渠道发送消息的活动

要将 Lambda 函数或 Webhook 分配给单独的活动，请使用 Amazon Pinpoint API 来创建或更新[活动](#)对象。

活动中的 MessageConfiguration 对象还必须包含一个 CustomMessage 对象。此对象有一个成员 Data。Data 的值是一个 JSON 字符串，其中包含要发送到自定义渠道的消息有效负载。

活动必须包含一个 CustomDeliveryConfiguration 对象。在 CustomDeliveryConfiguration 对象中，指定以下内容：

- EndpointTypes – 一个数组，其中包含自定义渠道活动应发送到的所有端点类型。它可以包含以下任何或全部渠道类型：
  - ADM
  - APNS
  - APNS\_SANDBOX
  - APNS\_VOIP
  - APNS\_VOIP\_SANDBOX
  - BAIDU
  - CUSTOM
  - EMAIL
  - GCM
  - SMS
  - VOICE
- DeliveryUri – 将端点发送到的目标。您可以只指定以下几项之一：



- 您要在活动运行时执行的 Lambda 函数的 Amazon 资源名称 (ARN)。
- 您要在活动运行时向其发送端点数据的 Webhook 的 URL。

### Note

Campaign 对象还可以包含一个 Hook 对象。此对象仅用于在执行活动时创建由 Lambda 函数自定义的分段。有关更多信息，请参阅[使用 AWS Lambda 自定义分段](#)。

## 了解 Amazon Pinpoint 发送到自定义渠道的事件数据

在创建通过自定义渠道发送消息的 Lambda 函数之前，您应该自行熟悉 Amazon Pinpoint 发出的数据。当 Amazon Pinpoint 活动通过自定义渠道发送消息时，它会向目标 Lambda 函数发送类似于以下示例的负载：

```
{
  "Message": {},
  "Data": "The payload that's provided in the CustomMessage object in
MessageConfiguration",
  "ApplicationId": "3a9b1f4e6c764ba7b031e7183example",
  "CampaignId": "13978104ce5d6017c72552257example",
  "TreatmentId": "0",
  "ActivityId": "575cb1929d5ba43e87e2478eeexample",
  "ScheduledTime": "2020-04-08T19:00:16.843Z",
  "Endpoints": {
    "1dbcd396df28ac6cf8c1c2b7fexample": {
      "ChannelType": "EMAIL",
      "Address": "mary.major@example.com",
      "EndpointStatus": "ACTIVE",
      "OptOut": "NONE",
      "Location": {
        "City": "Seattle",
        "Country": "USA"
      },
      "Demographic": {
        "Make": "OnePlus",
        "Platform": "android"
      },
      "EffectiveDate": "2020-04-01T01:05:17.267Z",
      "Attributes": {
```

```
    "CohortId": [
      "42"
    ],
    "CreationDate": "2020-04-01T01:05:17.267Z"
  }
}
```

事件数据提供了以下属性：

- **ApplicationId** – 活动所属的 Amazon Pinpoint 项目的 ID。
- **CampaignId** – 调用 Lambda 函数的 Amazon Pinpoint 活动的 ID。
- **TreatmentId** – 活动变体的 ID。如果您创建了标准活动，则此值始终为 0。如果您创建了 A/B 测试活动，则此值是介于 0 到 4 之间的整数。
- **ActivityId** – 由活动执行的操作的 ID。
- **ScheduledTime** – Amazon Pinpoint 执行活动的时间，以 ISO 8601 格式显示。
- **Endpoints** – 活动指向的端点列表。每个负载最多可包含 50 个端点。如果将活动发送到的分段包含的端点数超过 50 个，则 Amazon Pinpoint 将重复调用该函数，一次最多处理 50 个端点，直至处理完所有端点。

您可以在创建和测试自定义渠道 Lambda 函数时使用此示例数据。

## 配置 Webhook

如果您使用 Webhook 发送自定义渠道消息，则 Webhook 的 URL 必须以“https://”开头。Webhook URL 只能包含字母数字字符，加上以下符号：连字符 (-)、句点 (.)、下划线 (\_)、波浪符 (~)、问号 (?)、正斜杠或斜线分隔符 (/)、井号或哈希符号 (#) 以及分号 (:)。URL 必须符合 [RFC3986](https://www.rfc-editor.org/rfc/rfc3986)。

当您创建指定 Webhook URL 的活动时，Amazon Pinpoint 会向该 URL 发出 HTTP HEAD。对 HEAD 请求的响应必须包含名为 X-Amz-Pinpoint-AccountId 的标头。此标头的值必须等于您的 AWS 账户 ID。

## 配置 Lambda 函数

本节概述了当您创建通过自定义渠道发送消息的 Lambda 函数时需要执行的步骤。首先，创建该函数。之后，您将执行策略添加到函数中。此策略允许 Amazon Pinpoint 在活动运行时执行策略。

有关创建 Lambda 函数的介绍，请参阅《AWS Lambda 开发人员指南》中的[构建 Lambda 函数](#)。

## 示例 Lambda 函数

以下代码示例处理负载并记录 CloudWatch 中每个端点类型的端点数。

```
import boto3
import random
import pprint
import json
import time

cloudwatch = boto3.client('cloudwatch')

def lambda_handler(event, context):
    customEndpoints = 0
    smsEndpoints = 0
    pushEndpoints = 0
    emailEndpoints = 0
    voiceEndpoints = 0
    numEndpoints = len(event['Endpoints'])

    print("Payload:\n", event)
    print("Endpoints in payload: " + str(numEndpoints))

    for key in event['Endpoints'].keys():
        if event['Endpoints'][key]['ChannelType'] == "CUSTOM":
            customEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "SMS":
            smsEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "EMAIL":
            emailEndpoints += 1
        elif event['Endpoints'][key]['ChannelType'] == "VOICE":
            voiceEndpoints += 1
        else:
            pushEndpoints += 1

    response = cloudwatch.put_metric_data(
        MetricData = [
            {
                'MetricName': 'EndpointCount',
                'Dimensions': [
                    {
```

```
        'Name': 'CampaignId',
        'Value': event['CampaignId']
    },
    {
        'Name': 'ApplicationId',
        'Value': event['ApplicationId']
    }
],
'Unit': 'None',
'Value': len(event['Endpoints'])
},
{
    'MetricName': 'CustomCount',
    'Dimensions': [
        {
            'Name': 'CampaignId',
            'Value': event['CampaignId']
        },
        {
            'Name': 'ApplicationId',
            'Value': event['ApplicationId']
        }
    ],
    'Unit': 'None',
    'Value': customEndpoints
},
{
    'MetricName': 'SMSCount',
    'Dimensions': [
        {
            'Name': 'CampaignId',
            'Value': event['CampaignId']
        },
        {
            'Name': 'ApplicationId',
            'Value': event['ApplicationId']
        }
    ],
    'Unit': 'None',
    'Value': smsEndpoints
},
{
    'MetricName': 'EmailCount',
    'Dimensions': [
```

```
        {
            'Name': 'CampaignId',
            'Value': event['CampaignId']
        },
        {
            'Name': 'ApplicationId',
            'Value': event['ApplicationId']
        }
    ],
    'Unit': 'None',
    'Value': emailEndpoints
},
{
    'MetricName': 'VoiceCount',
    'Dimensions': [
        {
            'Name': 'CampaignId',
            'Value': event['CampaignId']
        },
        {
            'Name': 'ApplicationId',
            'Value': event['ApplicationId']
        }
    ],
    'Unit': 'None',
    'Value': voiceEndpoints
},
{
    'MetricName': 'PushCount',
    'Dimensions': [
        {
            'Name': 'CampaignId',
            'Value': event['CampaignId']
        },
        {
            'Name': 'ApplicationId',
            'Value': event['ApplicationId']
        }
    ],
    'Unit': 'None',
    'Value': pushEndpoints
},
{
    'MetricName': 'EndpointCount',
```

```
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': len(event['Endpoints'])
    },
    {
        'MetricName': 'CustomCount',
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': customEndpoints
    },
    {
        'MetricName': 'SMSCount',
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': smsEndpoints
    },
    {
        'MetricName': 'EmailCount',
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': emailEndpoints
    },
    {
        'MetricName': 'VoiceCount',
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': voiceEndpoints
    },
    {
        'MetricName': 'PushCount',
        'Dimensions': [
        ],
        'Unit': 'None',
        'Value': pushEndpoints
    }
    ],
    Namespace = 'PinpointCustomChannelExecution'
)
```

```
print("cloudwatchResponse:\n",response)
```

当 Amazon Pinpoint 活动执行此 Lambda 函数时，Amazon Pinpoint 会向函数发送分段成员列表。该函数计算每个 ChannelType 的端点的数量。然后，它将数据发送到 Amazon CloudWatch。您还可以在 CloudWatch 控制台的指标部分查看这些指标。这些指标可用于 PinpointCustomChannelExecution 命名空间中。

您可以修改此代码示例，以便它也连接到外部服务的 API，从而通过该服务发送消息。

## Amazon Pinpoint Lambda 函数响应格式

如果您想在自定义渠道活动之后使用旅程多变量或是/否拆分来确定端点路径，则必须将您的 Lambda 函数响应构造成 Amazon Pinpoint 可以理解的格式，然后将端点发送到正确的路径。

响应结构应采用以下格式：

```
{
  <Endpoint ID 1>:{
    EventAttributes: {
      <Key1>: <Value1>,
      <Key2>: <Value2>,
      ...
    }
  },
  <Endpoint ID 2>:{
    EventAttributes: {
      <Key1>: <Value1>,
      <Key2>: <Value2>,
      ...
    }
  },
  ...
}
```

然后，这将允许您选择要确定端点路径的键和值。

Yes/no split Info ×

Select a condition type  
Event

Choose a journey message activity and event  
lambdaFunction (Custom)

Call to function or webhook response

Response Info

Attribute	Value
<input type="text"/>	<input type="text"/>

+ Add condition

Condition evaluation  
The amount of time that Amazon Pinpoint waits before it evaluates the conditions.

Evaluate after

1 hours

Description - optional  
*Add description*

Save

## 授予 Amazon Pinpoint 权限以调用 Lambda 函数

您可以使用 AWS Command Line Interface (AWS CLI) 将权限添加到分配给您的 Lambda 函数的 Lambda 函数策略。要允许 Amazon Pinpoint 调用某个函数，请使用 Lambda [add-permission](#) 命令，如下示例所示：

```
aws lambda add-permission \
--function-name myFunction \
--statement-id sid0 \
--action lambda:InvokeFunction \
--principal pinpoint.us-east-1.amazonaws.com \
--source-arn arn:aws:mobiletargeting:us-east-1:111122223333:apps/* \
--source-account 111122223333
```

在上述命令中，执行以下操作：



- 将 *myFunction* 替换为 Lambda 函数的名称。
- 将 *us-east-1* 替换为您使用 Amazon Pinpoint 的 AWS 区域。
- 将 *111122223333* 替换为您的 AWS 账户 ID。

运行 `add-permission` 命令时，Lambda 将返回以下输出：

```
{
  "Statement": "{\"Sid\":\"sid\",
    \"Effect\":\"Allow\",
    \"Principal\":{\"Service\":\"pinpoint.us-east-1.amazonaws.com\"},
    \"Action\":\"lambda:InvokeFunction\",
    \"Resource\":\"arn:aws:lambda:us-east-1:111122223333:function:myFunction\",
    \"Condition\":
      {\"ArnLike\":
        {\"AWS:SourceArn\":
          \"arn:aws:mobiletargeting:us-east-1:111122223333:apps/*\"}},
        {\"StringEquals\":
          {\"AWS:SourceAccount\":
            \"111122223333\"}}}}
}
```

Statement 值是已添加到 Lambda 函数策略的语句的 JSON 字符串版本。

## 进一步限制执行策略

您可以通过将执行策略限制为特定 Amazon Pinpoint 项目来修改执行策略。为此，请将上述示例中的 \* 替换为项目的唯一 ID。您可以通过将策略限制为特定活动来进一步限制策略。例如，要将策略限制为在项目 ID 为 `dbaf6ec2226f0a9a8615e3ea5example` 的项目中仅允许活动 ID 为 `95fee4cd1d7f5cd67987c1436example` 的活动，请对 `source-arn` 属性使用以下值：

```
arn:aws:mobiletargeting:us-east-1:111122223333:apps/dbaf6ec2226f0a9a8615e3ea5example/campaigns/95fee4cd1d7f5cd67987c1436example
```

### Note

如果您确实将 Lambda 函数的执行限制为特定活动，则首先必须使用限制性较低的策略创建函数。接下来，您必须在 Amazon Pinpoint 中创建活动并选择此函数。最后，您必须更新执行策略以引用指定的活动。

# 将 Amazon Pinpoint 事件流式传输到 Kinesis

在 Amazon Pinpoint 中，事件是在以下情况下发生的操作：用户与您的某个应用程序进行交互，您从活动或旅程发送消息，或者您发送事务性短信或电子邮件。例如，如果您发送电子邮件，则会发生几个事件：

- 在您发送消息时，会发生 send 事件。
- 消息到达接收人的收件箱时，会发生 delivered 事件。
- 当接收人打开消息时，会发生 open 事件。

您可以配置 Amazon Pinpoint 将有关事件的信息发送到 Amazon Kinesis。Kinesis 平台提供的服务可用于实时收集、处理和分析来自 AWS 服务的数据。Amazon Pinpoint 可以将事件数据发送到 Kinesis Data Firehose，后者将这些数据流式传输 AWS 到亚马逊 S3 或 Amazon Redshift 等数据存储。Amazon Pinpoint 还可以将数据流式传输到 Kinesis Data Streams，Kinesis Data Streams 会提取和存储多个数据流，供分析应用程序处理。

Amazon Pinpoint 事件流包含有关用户与您连接到 Amazon Pinpoint 的应用程序交互的信息。它还包含有关您从活动、通过任何渠道以及从任何旅程发送的所有消息。这也可以包括您定义的任何自定义事件。最后，它包含有关您发送的所有事务性电子邮件和短信的信息。

## Note

Amazon Pinpoint 不流式传输有关事务性推送通知或语音消息的信息。

本章提供有关设置 Amazon Pinpoint 以将事件数据流式传输到 Kinesis 的信息。它还包含 Amazon Pinpoint 流式传输的事件数据的示例。

## 主题

- [设置事件流式传输](#)
- [应用程序事件](#)
- [活动事件](#)
- [旅程事件](#)
- [电子邮件事件](#)
- [短信事件](#)

## 设置事件流式传输

您可以配置 Amazon Pinpoint 以将事件数据发送到 Amazon Kinesis Streams 或 Amazon Kinesis Data Firehose 传输流。Amazon Pinpoint 可以发送活动、旅程以及事务性电子邮件和短信的事件数据。

此部分包含有关以编程方式设置事件流式传输的信息。您也可以使用 Amazon Pinpoint 控制台来设置事件流式传输。有关使用 Amazon Pinpoint 控制台设置事件流的信息，请参阅《Amazon Pinpoint 用户指南》中的[事件流设置](#)。

### 先决条件

本节中的示例需要以下输入：

- 与 Amazon Pinpoint 集成并报告事件的应用程序的 `应用程序 ID`。有关如何集成的信息，请参阅[将 Amazon Pinpoint 与您的应用程序集成](#)。
- 您账户中 Kinesis 直播或 Kinesis Data Firehose 传输流的亚马逊资源名称 (ARN)。AWS 有关创建这些资源的信息，请参阅《Amazon Kinesis Data Streams 开发人员指南》中的[创建和更新数据流](#)，或者《Amazon Kinesis Data Firehose 开发人员指南》中的[创建 Amazon Kinesis Data Firehose 传输流](#)。
- 授权 Amazon Pinpoint 向直播发送数据的 AWS Identity and Access Management (IAM) 角色的 ARN。有关创建角色的信息，请参阅[用于将事件流式传输到 Kinesis 的 IAM 角色](#)。

### AWS CLI

以下 AWS CLI 示例使用 `put-event-stream` 命令。此命令配置 Amazon Pinpoint 将事件发送到 Kinesis 流：

```
aws pinpoint put-event-stream \  
--application-id projectId \  
--write-event-stream DestinationStreamArn=streamArn,RoleArn=roleArn
```

### AWS SDK for Java

以下 Java 示例配置 Amazon Pinpoint 向 Kinesis 流发送事件：

```
public PutEventStreamResult createEventStream(AmazonPinpoint pinClient,  
    String appId, String streamArn, String roleArn) {  
  
    WriteEventStream stream = new WriteEventStream()
```

```
        .withDestinationStreamArn(streamArn)
        .withRoleArn(roleArn);

    PutEventStreamRequest request = new PutEventStreamRequest()
        .withApplicationId(appId)
        .withWriteEventStream(stream);

    return pinClient.putEventStream(request);
}
```

此示例构建了一个 [WriteEventStream](#) 对象，它存储 Kinesis 流和 IAM 角色的 ARN。WriteEventStream 对象会传递给 [PutEventStreamRequest](#) 对象，用于将 Amazon Pinpoint 配置为流式传输特定应用程序的事件。PutEventStreamRequest 对象会传递给 Amazon Pinpoint 客户端的 [putEventStream](#) 方法。

您可以将 Kinesis 流分配给多个应用程序。如果您执行此操作，Amazon Pinpoint 从每个应用程序将使用 base64 编码的事件数据发送到流中，这使您能够将数据作为集合进行分析。以下示例方法接受应用程序 ID 的列表，它使用上一个示例方法 `createEventStream` 将流分配给每个应用程序：

```
public List<PutEventStreamResult> createEventStreamFromAppList(
    AmazonPinpoint pinClient, List<String> appIDs,
    String streamArn, String roleArn) {

    return appIDs.stream()
        .map(appId -> createEventStream(pinClient, appId, streamArn,
            roleArn))
        .collect(Collectors.toList());
}
```

虽然您可以将一个流分配给多个应用程序，但不能将多个流分配给一个应用程序。

## 禁用事件流式传输

如果将 Kinesis 流分配给了一个应用程序，则可以对该应用程序禁用事件流式传输。Amazon Pinpoint 停止将事件流式传输到 Kinesis，但您可以使用 Amazon Pinpoint 控制台查看事件分析。

## AWS CLI

使用 [delete-event-stream](#) 命令：

```
aws pinpoint delete-event-stream --application-id application-id
```

## AWS SDK for Java

使用 Amazon Pinpoint 客户端 [deleteEventStream](#) 的方法：

```
pinClient.deleteEventStream(new DeleteEventStreamRequest().withApplicationId(appId));
```

## 应用程序事件

将您的应用程序与 Amazon Pinpoint 集成后，Amazon Pinpoint 可以流式传输有关用户活动、自定义事件和应用程序的消息送达的事件数据。

### 示例

应用程序事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_session.stop",
  "event_timestamp": 1487973802507,
  "arrival_timestamp": 1487973803515,
  "event_version": "3.0",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "cognito_identity_pool_id": "us-east-1:a1b2c3d4-e5f6-g7h8-i9j0-k1l2m3n4o5p6",
    "package_name": "main.page",
    "sdk": {
      "name": "aws-sdk-mobile-analytics-js",
      "version": "0.9.1:2.4.8"
    },
    "title": "title",
    "version_name": "1.0",
    "version_code": "1"
  },
  "client": {
    "client_id": "m3n4o5p6-a1b2-c3d4-e5f6-g7h8i9j0k1l2",
    "cognito_id": "us-east-1:i9j0k1l2-m3n4-o5p6-a1b2-c3d4e5f6g7h8"
  },
  "device": {
    "locale": {
      "code": "en_US",
      "country": "US",
      "language": "en"
    }
  },
}
```

```

    "make": "generic web browser",
    "model": "Unknown",
    "platform": {
      "name": "android",
      "version": "10.10"
    }
  },
  "session": {
    "session_id": "f549dea9-1090-945d-c3d1-e4967example",
    "start_timestamp": 1487973202531,
    "stop_timestamp": 1487973802507
  },
  "attributes": {},
  "metrics": {}
}

```

## 应用程序事件属性

此部分定义了应用程序事件流中包含的属性。

属性	描述
event_type	<p>事件类型。可能的值有：</p> <ul style="list-style-type: none"> <li>• <code>_session.start</code> – 端点启动了新的会话。</li> <li>• <code>_session.stop</code> – 端点结束了会话。</li> <li>• <code>_userauth.sign_in</code> – 端点登录到了您的应用程序。</li> <li>• <code>_userauth.sign_up</code> – 新端点在您的应用程序中完成了注册过程。</li> <li>• <code>_userauth.auth_fail</code> – 端点尝试登录您的应用程序，但无法完成登录。</li> <li>• <code>_monetization.purchase</code> – 端点进行了应用程序内购买。</li> <li>• <code>_session.pause</code> – 端点暂停了会话。暂停的会话可以恢复，这样您就可以继续收集指标而无需启动全新会话。</li> <li>• <code>_session.resume</code> – 节点恢复了会话。</li> </ul>

属性	描述
event_timestamp	报告事件的时间，显示为以毫秒为单位的 Unix 时间。
arrival_timestamp	Amazon Pinpoint 收到事件的时间，显示为以毫秒为单位的 Unix 时间。
event_version	事件 JSON 架构的版本。 <div> <b>Tip</b> 在事件处理应用程序中检查此版本，以便知道何时更新应用程序以响应架构更新。</div>
application	与事件关联的 Amazon Pinpoint 项目的相关信息。有关更多信息，请参阅 <a href="#">应用程序表</a> 。
client	报告事件的端点的相关信息。有关更多信息，请参阅 <a href="#">客户端表</a> 。
device	报告事件的设备的相关信息。有关更多信息，请参阅 <a href="#">设备表</a> 。
session	有关生成事件的会话的信息。有关更多信息，请参阅 <a href="#">会话表</a> 。
attributes	与事件关联的属性。对于您的应用程序报告的事件，此对象包含您定义的自定义属性。
metrics	与事件相关的指标。您可以选择配置应用程序将自定义指标发送到 Amazon Pinpoint。

## 应用程序

包括与事件关联的 Amazon Pinpoint 项目的相关信息。

属性	描述
app_id	报告事件的 Amazon Pinpoint 项目的唯一 ID。
cognito_identity_pool_id	与端点关联的 Amazon Cognito 身份池的 ID。
package_name	应用程序包的名称，例如 <code>com.example.my_app</code> 。
sdk	用于报告事件的开发工具包的相关信息。有关更多信息，请参阅 <a href="#">开发工具包表</a> 。
title	应用程序的名称。
version_name	应用程序的版本名称，例如 V2.5。
version_code	应用程序的版本号，例如 3。

## SDK

包括用于报告事件的开发工具包的相关信息。

属性	描述
name	用于报告事件的开发工具包的名称。
version	开发工具包的版本。

## 客户端

包括有关生成事件的端点的信息。

属性	描述
client_id	端点的 ID。
cognito_id	与端点关联的 Amazon Cognito ID 令牌。



## 设备

包括有关生成事件的端点设备的信息。

属性	描述
locale	包含有关端点设备的语言和区域设置的信息。有关更多信息，请参阅 <a href="#">区域设置</a> 表。
make	端点设备的制造商。
model	端点设备的型号标识符。
platform	有关端点设备上操作系统的信息。有关更多信息，请参阅 <a href="#">平台</a> 表。

## 区域设置

包含有关端点设备的语言和区域设置的信息。

属性	描述
code	与设备关联的区域设置标识符。
country	与设备区域设置关联的国家或地区。
language	与设备区域设置关联的语言。

## 平台

包含有关端点设备上操作系统的信息。

属性	描述
name	设备上操作系统的名称。
version	设备上操作系统的版本。

## 会话

包括有关生成事件的会话的信息。

属性	描述
session_id	标识会话的唯一 ID。
start_timestamp	会话开始的日期和时间，显示为以毫秒为单位的 Unix 时间。
stop_timestamp	会话结束的日期和时间，显示为以毫秒为单位的 Unix 时间。

## 活动事件

如果您使用 Amazon Pinpoint 通过任意渠道发送活动，Amazon Pinpoint 可以流式传输有关这些活动的事件数据。这包括您从活动发送的任意电子邮件或短信的事件数据。有关 Amazon Pinpoint 为这些消息类型流式传输的数据的详细信息，请参阅[the section called “电子邮件事件”](#)和[the section called “短信事件”](#)。

## 示例事件

活动事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_campaign.send",
  "event_timestamp": 1562109497426,
  "arrival_timestamp": 1562109497494,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "d8dcf7c5-e81a-48ae-8313-f540cexample"
  },
  "device": {
    "platform": {}
  }
}
```

```



},
"session": {},
"attributes": {
  "treatment_id": "0",
  "campaign_activity_id": "5473285727f04865bc673e527example",
  "delivery_type": "GCM",
  "campaign_id": "4f8d6097c2e8400fa3081d875example",
  "campaign_send_status": "SUCCESS"
},
"client_context": {
  "custom": {
    "endpoint": "{\"ChannelType\": \"GCM\", \"EndpointStatus\": \"ACTIVE\",
      #\"OptOut\": \"NONE\", \"RequestId\": \"ec229696-9d1e-11e9-8bf1-85d0aexample\",
      #\"EffectiveDate\": \"2019-07-02T23:12:54.836Z\", \"User\": {}}"
  }
},
"awsAccountId": "123456789012"
}

```

## 活动事件属性

此部分定义活动事件流中包含的属性。

属性	描述
event_type	<p>事件类型。可能的值有：</p> <ul style="list-style-type: none"> <li>_campaign.send – Amazon Pinpoint 执行了该活动。</li> <li>_campaign.opened_notification – 对于推送通知活动，此事件类型指示收件人点击并打开了通知。</li> <li>_campaign.received_foreground – 对于推送通知活动，此事件类型指示收件人以前台通知方式接收了消息。</li> <li>_campaign.received_background – 对于推送通知活动，此事件类型指示收件人以后台通知方式接收了消息。</li> </ul>

属性	描述
	<div data-bbox="862 212 1507 716" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"> <p> <b>Note</b></p> <p><code>_campaign.opened_notification</code>、<code>_campaign.received_foreground</code> 和 <code>_campaign.received_background</code> 事件仅当您使用 AWS Amplify 时才返回。有关将您的应用程序与 AWS Amplify 集成的更多信息，请参阅<a href="#">使用 AWS Amplify 将您的前端应用程序连接到 Amazon Pinpoint</a>。</p> </div>
event_timestamp	报告事件的时间，显示为以毫秒为单位的 Unix 时间。
arrival_timestamp	Amazon Pinpoint 收到事件的时间，显示为以毫秒为单位的 Unix 时间。
event_version	事件 JSON 架构的版本。 <div data-bbox="829 1119 1507 1381" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Tip</b></p> <p>在事件处理应用程序中检查此版本，以便知道何时更新应用程序以响应架构更新。</p> </div>
application	与事件关联的 Amazon Pinpoint 项目的相关信息。有关更多信息，请参阅 <a href="#">应用程序表</a> 。
client	与事件关联的端点的相关信息。有关更多信息，请参阅 <a href="#">客户端表</a> 。
device	报告事件的设备的相关信息。对于活动和事务性消息，此对象为空。

属性	描述
session	有关生成事件的会话的信息。对于活动，此对象为空。
attributes	与事件关联的属性。对于您的应用程序之一报告的事件，此对象包含由应用程序定义的自定义属性。对于在您发送活动时创建的事件，此对象包含与活动关联的属性。对于在您发送事务性电子邮件时生成的事件，此对象包含与电子邮件本身相关的信息。  有关更多信息，请参阅 <a href="#">属性表</a> 。
client_context	包含一个 custom 对象，其中包含一个 endpoint 属性。endpoint 属性包含将活动发送到的端点的端点记录内容。
awsAccountId	已用于发送电子邮件的 AWS 账户的 ID。

## 应用程序

包括与事件关联的 Amazon Pinpoint 项目的相关信息。

属性	描述
app_id	报告事件的 Amazon Pinpoint 项目的唯一 ID。
sdk	用于报告该事件的开发工具包。

## 属性

包含有关生成事件的活动的信息。

属性	描述
treatment_id	如果使用 A/B 测试活动发送了消息，则此值表示消息的处理编号。对于标准活动，此值为 0。
campaign_activity_id	发生事件时 Amazon Pinpoint 生成的唯一 ID。
delivery_type	<p>活动的交付方式。不要将此属性与 client_context 的 endpoint 属性下指定的 ChannelType 字段混淆。该 ChannelType 字段通常基于消息发送到的端点。</p> <p>对于仅支持一种端点类型的渠道，delivery_type 和 ChannelType 字段的值相同。例如，对于电子邮件渠道，delivery_type 和 ChannelType 字段的值与 EMAIL 相同。</p> <p>但是，对于支持不同端点类型的渠道（例如自定义渠道），情况并不总是如此。您可以为不同的端点使用自定义渠道，例如 EMAIL、SMS、CUSTOM 等。在这种情况下，delivery_type 标识自定义投放事件 CUSTOM，ChannelType 指定活动发送到的端点类型，例如 EMAIL、SMS、CUSTOM 等。有关创建自定义渠道的更多信息，请参阅<a href="#">创建自定义渠道</a>。</p> <p>可能的值有：</p> <ul style="list-style-type: none"><li>• EMAIL</li><li>• SMS</li><li>• ADM</li><li>• APNS</li><li>• APNS_SANDBOX</li><li>• APNS_VOIP</li><li>• APNS_VOIP_SANDBOX</li><li>• VOICE</li></ul>

属性	描述
	<ul style="list-style-type: none"><li>• GCM</li><li>• BAIDU</li><li>• PUSH</li><li>• CUSTOM</li></ul>
campaign_id	发送消息的活动的唯一 ID。

属性	描述
campaign_send_status	<p data-bbox="829 226 1458 260">指示目标端点的活动的状态。可能的值包括：</p> <ul data-bbox="829 306 1500 1814" style="list-style-type: none"><li data-bbox="829 306 1382 340">• SUCCESS – 活动已成功发送到端点。</li><li data-bbox="829 363 1297 396">• FAILURE – 活动未发送到端点。</li><li data-bbox="829 420 1479 499">• DAILY_CAP – 没有将活动发送到端点，因为已将最大数量的每日消息发送到端点。</li><li data-bbox="829 522 1500 648">• EXPIRED – 没有将活动发送到端点，因为发送该活动将会超出活动的最大持续时间或发送速率设置。</li><li data-bbox="829 672 1500 756">• QUIET_TIME – 由于安静时间限制，没有将活动发送到端点。</li><li data-bbox="829 779 1500 863">• HOLDOUT – 没有将活动发送到端点，因为端点是保留组的成员。</li><li data-bbox="829 886 1500 970">• DUPLICATE_ADDRESS – 分段中有重复的端点地址。该活动已发送到端点地址一次。</li><li data-bbox="829 993 1500 1077">• QUIET_TIME – 由于安静时间限制，没有将活动发送到端点。</li><li data-bbox="829 1100 1479 1226">• CAMPAIGN_CAP – 没有将活动发送到端点，因为已从该活动将最大数量的消息发送到端点。</li><li data-bbox="829 1249 1500 1333">• FAILURE_PERMANENT – 发送到端点时发生永久故障。</li><li data-bbox="829 1356 1479 1440">• TRANSIENT_FAILURE – 发送到端点时发生暂时性故障。</li><li data-bbox="829 1463 1289 1497">• THROTTLED – 发送已被限制。</li><li data-bbox="829 1520 1203 1554">• UNKNOWN – 未知故障。</li><li data-bbox="829 1577 1344 1610">• HOOK_FAILURE – 活动挂钩失败。</li><li data-bbox="829 1633 1484 1717">• CUSTOM_DELIVERY_FAILURE – 自定义交付失败。</li><li data-bbox="829 1740 1451 1824">• RECOMMENDATION_FAILURE – 推荐失败。</li></ul>



属性	描述
	<ul style="list-style-type: none"> <li>UNSUPPORTED_CHANNEL – 不支持渠道。</li> </ul>

## 客户端

包括活动所定向到的端点的相关信息。

属性	描述
client_id	活动发送到的端点的 ID。

## 旅程事件

如果您发布旅程，Amazon Pinpoint 可以流式传输有关旅程的事件数据。这包括您从旅程中发送的任何电子邮件、短信、推送或自定义消息的事件数据。

有关 Amazon Pinpoint 流式传输的数据的信息，请参阅以下内容：

- 有关电子邮件，请参阅[the section called “电子邮件事件”](#)。
- 有关短信，请参阅[短信事件](#)。

## 示例事件

旅程事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_journey.send",
  "event_timestamp": 1572989078843,
  "arrival_timestamp": 1572989078843,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {

    }
  },
  "client": {
```

```

    "client_id":"d8dcf7c5-e81a-48ae-8313-f540cexample"
  },
  "device":{
    "platform":{

    }
  },
  "session":{


  },
  "attributes":{
    "journey_run_id":"edc9a0b577164d1daf72ebd15example",
    "journey_send_status":"SUCCESS",
    "journey_id":"546401670c5547b08811ac6a9example",
    "journey_activity_id":"0yKexample",
    "journey_activity_type": "EMAIL"
  },
  "client_context":{
    "custom":{
      "endpoint":{"ChannelType\\":"EMAIL\\","EndpointStatus\\":"ACTIVE\\","OptOut\\":"NONE\\","Demographic\\":{"Timezone\\":"America/Los_Angeles\\"}}
    }
  },
  "awsAccountId":"123456789012"
}

```

## 旅程事件属性

此部分定义 Amazon Pinpoint 为旅程生成的事件流数据中包含的属性。

属性	描述
event_type	事件类型。对于旅程事件，此属性的值始终为 <code>_journey.send</code> ，这表示 Amazon Pinpoint 已执行旅程。
event_timestamp	报告事件的时间，显示为以毫秒为单位的 Unix 时间。
arrival_timestamp	Amazon Pinpoint 收到事件的时间，显示为以毫秒为单位的 Unix 时间。

属性	描述
event_version	<p>事件 JSON 架构的版本。</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Tip</b> 在事件处理应用程序中检查此版本，以便知道何时更新应用程序以响应架构更新。</p> </div>
application	与事件关联的 Amazon Pinpoint 项目的相关信息。有关更多信息，请参阅 <a href="#">应用程序表</a> 。
client	与事件关联的端点的相关信息。有关更多信息，请参阅 <a href="#">客户端表</a> 。
device	报告事件的设备的相关信息。对于历程，此对象为空。
session	有关生成事件的会话的信息。对于历程，此对象为空。
attributes	与生成事件的旅程和旅程活动关联的属性。有关更多信息，请参阅 <a href="#">属性表</a> 。
client_context	包含一个 custom 对象，其中包含一个 endpoint 属性。endpoint 属性包含与事件关联的端点的端点记录内容。
awsAccountId	用于执行旅程的 AWS 账户的 ID。

## 应用程序

包括与事件关联的 Amazon Pinpoint 项目的相关信息。

属性	描述
app_id	报告事件的 Amazon Pinpoint 项目的唯一 ID。

属性	描述
sdk	用于报告该事件的开发工具包。

## 客户端

包括与事件关联的端点的相关信息。


属性	描述
client_id	端点的 ID。

## Attributes

包括有关生成事件的旅程的信息。

属性	描述
journey_run_id	生成事件的旅程的唯一 ID。Amazon Pinpoint 会自动为旅程的每一个新运行生成并分配此 ID。
journey_send_status	指示与事件关联的消息的传输状态。可能的值包括： <ul style="list-style-type: none"> <li>• SUCCESS – 消息已成功发送到端点。</li> <li>• FAILURE – 由于出错，消息未发送到端点。</li> <li>• DAILY_CAP – 由于发送消息将超过旅程或项目在 24 小时内可以向单个端点发送的最大消息数，消息未发送到端点。</li> <li>• QUIET_TIME – 由于旅程或项目的安静时间限制，未发送消息。</li> <li>• QUIET_TIME_MISSING_TIMEZONE – 消息未发送，因为时区估计无法估计端点的时区，并且已启用安静时间。</li> </ul>
journey_id	生成事件的旅程的唯一 ID。

属性	描述
journey_activity_id	生成事件的旅程活动的唯一 ID。
journey_activity_type	事件的旅程活动类型。可以是 EMAIL、SMS、PUSH、CONTACT_CENTER 或 CUSTOM。

 **Note**  
VOICE 不是支持的旅程活动类型。

## 电子邮件事件

当您发送电子邮件时，Amazon Pinpoint 会流式传输数据，提供这些消息的下列事件类型的附加信息：

- 发送数
- 已送达数
- 退回数
- 投诉数
- 打开次数
- 点击次数
- 拒绝数
- 取消订阅数
- 呈现失败数

[电子邮件事件属性](#)中详细说明了上述列表中的事件类型。

根据您用于发送电子邮件的 API 和设置，您可能会看到其他事件类型或不同的数据。例如，如果您使用将事件数据发布到 Amazon Kinesis 的配置集（如 Amazon Simple Email Service (Amazon SES) 提供的配置集）发送消息，则这些数据还可能包括模板渲染失败的事件。有关该数据的信息，请参阅《Amazon Simple Email Service 开发人员指南》中的[使用 Amazon SES 事件发布进行监控](#)。

## 示例事件

### 电子邮件发送

email send 事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_email.send",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618622025,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "9a311b17-6f8e-4093-be61-4d0bbexample"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "received"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rnbd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
```

```
        "value": "Amazon Pinpoint Test"
      }, {
        "name": "MIME-Version",
        "value": "1.0"
      }, {
        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"-----=_Part_314159_271828\""
      }
    ],
    "common_headers": {
      "from": "sender@example.com",
      "to": ["recipient@example.com"],
      "subject": "Amazon Pinpoint Test"
    }
  },
  "send": {}
}
}
```

## 电子邮件送达

email delivered 事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_email.delivered",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618622690,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "e9a3000d-daa2-40dc-ac47-1cd34example"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "delivered"
  },
  "awsAccountId": "123456789012",
```

```
"facets": {
  "email_channel": {
    "mail_event": {
      "mail": {
        "message_id": "0200000073rnbnmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
        "message_send_timestamp": 1564618621380,
        "from_address": "sender@example.com",
        "destination": ["recipient@example.com"],
        "headers_truncated": false,
        "headers": [{
          "name": "From",
          "value": "sender@example.com"
        }, {
          "name": "To",
          "value": "recipient@example.com"
        }, {
          "name": "Subject",
          "value": "Amazon Pinpoint Test"
        }, {
          "name": "MIME-Version",
          "value": "1.0"
        }, {
          "name": "Content-Type",
          "value": "multipart/alternative; boundary=\"-----_Part_314159_271828\""
        }
      ],
      "common_headers": {
        "from": "sender@example.com",
        "to": ["recipient@example.com"],
        "subject": "Amazon Pinpoint Test"
      }
    },
    "delivery": {
      "smtp_response": "250 ok: Message 82080542 accepted",
      "reporting_mta": "a8-53.smtp-out.amazonses.com",
      "recipients": ["recipient@example.com"],
      "processing_time_millis": 1310
    }
  }
}
}
```

## 电子邮件点击



email click 事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_email.click",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618713751,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "49c1413e-a69c-46dc-b1c4-6470eexample"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "feedback": "https://aws.amazon.com/pinpoint/"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rn bmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
```

```

        "name": "Content-Type",
        "value": "multipart/alternative; boundary=\"-----=_Part_314159_271828\""
    }, {
        "name": "Message-ID",
        "value": "null"
    }
  ]],
  "common_headers": {
    "from": "sender@example.com",
    "to": ["recipient@example.com"],
    "subject": "Amazon Pinpoint Test"
  }
},
"click": {
  "ip_address": "72.21.198.67",
  "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.1.2 Safari/605.1.15",
  "link": "https://aws.amazon.com/pinpoint/"
}
}
}
}
}
}

```

## 电子邮件打开

email open 事件的 JSON 对象包含以下示例中显示的数据。

```

{
  "event_type": "_email.open",
  "event_timestamp": 1564618621380,
  "arrival_timestamp": 1564618712316,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "8dc1f651-b3ec-46fc-9b67-2a050example"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {

```

```
    "feedback": "opened"
  },
  "awsAccountId": "123456789012",
  "facets": {
    "email_channel": {
      "mail_event": {
        "mail": {
          "message_id": "0200000073rn bmd1-mbvdg3uo-q8ia-m3ku-ibd3-ms77kexample-000000",
          "message_send_timestamp": 1564618621380,
          "from_address": "sender@example.com",
          "destination": ["recipient@example.com"],
          "headers_truncated": false,
          "headers": [{
            "name": "From",
            "value": "sender@example.com"
          }, {
            "name": "To",
            "value": "recipient@example.com"
          }, {
            "name": "Subject",
            "value": "Amazon Pinpoint Test"
          }, {
            "name": "MIME-Version",
            "value": "1.0"
          }, {
            "name": "Content-Type",
            "value": "multipart/alternative; boundary=\"-----=_Part_314159_271828\""
          }, {
            "name": "Message-ID",
            "value": "null"
          }
        ],
        "common_headers": {
          "from": "sender@example.com",
          "to": ["recipient@example.com"],
          "subject": "Amazon Pinpoint Test"
        }
      }
    },
    "open": {
      "ip_address": "72.21.198.67",
      "user_agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6)
AppleWebKit/605.1.15 (KHTML, like Gecko)"
    }
  }
}
```

```
}  
}
```

## 电子邮件事件属性

此部分定义在您发送电子邮件时 Amazon Pinpoint 生成的事件流数据中包含的属性。

属性	描述
event_type	<p>事件类型。可能的值有：</p> <ul style="list-style-type: none"><li>• <code>_email.send</code> – Amazon Pinpoint 接受邮件并尝试将其传送给收件人。</li><li>• <code>_email.delivered</code> – 邮件已传送给收件人。</li><li>• <code>_email.rejected</code> – Amazon Pinpoint 确定该邮件包含恶意软件，因此没有尝试发送。</li><li>• <code>_email.hardbounce</code> – 一个永久性问题阻碍了 Amazon Pinpoint 发送邮件。Amazon Pinpoint 不会再次尝试传送邮件。</li><li>• <code>_email.softbounce</code> – 一个临时性问题阻碍了 Amazon Pinpoint 传送邮件。Amazon Pinpoint 将经过一定时间后尝试再次传送邮件。如果仍无法传送邮件，则不会再尝试重试。电子邮件的最终状态将是 <code>SOFTBOUNCE</code>。</li><li>• <code>_email.complaint</code> – 收件人收到了邮件，然后向其电子邮件提供商举报该邮件为垃圾邮件（例如，使用电子邮件客户端的“报告垃圾邮件”功能）。</li><li>• <code>_email.open</code> – 收件人收到并打开了邮件。</li><li>• <code>_email.click</code> – 收件人已收到邮件并点击了邮件中的链接。</li><li>• <code>_email.unsubscribe</code> – 收件人已收到邮件并点击了邮件中的取消订阅链接。</li></ul>

属性	描述
	<ul style="list-style-type: none"> <li>• <code>_email.rendering_failure</code> – 由于渲染失败，邮件未发送。当模板数据丢失或模板参数与数据不匹配时，可能会发生此事件类型。</li> </ul>
<code>event_timestamp</code>	发送邮件的时间，显示为以毫秒为单位的 Unix 时间。对于为一条邮件生成的所有事件，该值通常都相同。
<code>arrival_timestamp</code>	Amazon Pinpoint 收到事件的时间，显示为以毫秒为单位的 Unix 时间。
<code>event_version</code>	事件 JSON 架构的版本。 <div data-bbox="829 768 1507 1037" style="border: 1px solid #ccc; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Tip</b> 在事件处理应用程序中检查此版本，以便知道何时更新应用程序以响应架构更新。</p> </div>
<code>application</code>	与事件关联的 Amazon Pinpoint 项目的相关信息。有关更多信息，请参阅应用程序表。
<code>client</code>	包括安装在设备上用于报告事件的应用程序客户端的相关信息。有关更多信息，请参阅客户端表。
<code>device</code>	报告事件的设备的相关信息。有关更多信息，请参阅设备表。  对于电子邮件事件，此对象为空。
<code>session</code>	对于电子邮件事件，此对象为空。

属性	描述
attributes	<p>与事件关联的属性。有关更多信息，请参阅属性表。</p> <p>对于您的应用程序之一报告的事件，此对象包含由应用程序定义的自定义属性。对于在您从活动或旅程发送电子邮件时创建的事件，此对象包含与活动或旅程关联的属性。对于在您发送事务性电子邮件时生成的事件，此对象包含与电子邮件本身相关的信息。</p>
client_context	<p>对于电子邮件事件，此对象包含名为 custom 的对象，该对象包含 legacy_identifier 属性。legacy_identifier 属性的值是发送电子邮件的项目的 ID。</p>
facets	<p>有关电子邮件的其他信息，例如电子邮件标题，请参阅分面表。</p>
awsAccountId	<p>已用于发送电子邮件的 AWS 账户的 ID。</p>

## 应用程序

包括与事件关联的 Amazon Pinpoint 项目的相关信息。

属性	描述
app_id	<p>报告事件的 Amazon Pinpoint 项目的唯一 ID。</p>
sdk	<p>用于报告该事件的开发工具包。如果您通过直接调用 Amazon Pinpoint API 或者使用 Amazon Pinpoint 控制台来发送事务性电子邮件，则此对象为空。</p>

## 属性

包含有关生成事件的活动或旅程的信息。

### 活动

包含有关生成事件的活动的信息。

属性	描述
feedback	对于 <code>_email.click</code> 事件，此属性的值是收件人点击邮件以生成事件的链接的 URL。对于其他事件，此值表示事件类型（例如 <code>received</code> 、 <code>opened</code> 或 <code>clicked</code> ）。
treatment_id	如果使用 A/B 测试活动发送了消息，则此值表示消息的处理编号。对于标准活动和事务性电子邮件，该值为 0。
campaign_activity_id	发生事件时 Amazon Pinpoint 生成的唯一 ID。
campaign_id	发送邮件的活动的唯一 ID。

### 旅程

包含有关生成事件的旅程的信息。

属性	描述
journey_run_id	发送邮件的旅程运行的唯一 ID。Amazon Pinpoint 会自动为旅程的每一个新运行生成并分配此 ID。
feedback	对于 <code>_email.click</code> 事件，此属性的值是收件人点击邮件以生成事件的链接的 URL。对于其他事件，此值表示事件类型（例如 <code>received</code> 、 <code>delivered</code> 或 <code>opened</code> ）。
journey_id	发送邮件的旅程的唯一 ID。

属性	描述
journey_activity_id	发送邮件的旅程活动的唯一 ID。

## 客户端

活动或旅程所针对的客户端的唯一标识符。

属性	描述
client_id	事件 ID 该值是活动和旅程的端点 ID，对于事务性发送，它是 UUID。

## 分面

包括有关邮件和事件类型的信息。

属性	描述
email_channel	包含一个 mail_event 对象，其中包含两个对象：mail 以及一个与事件类型对应的对象。

## 邮件

包含有关电子邮件内容的信息，以及与邮件本身相关的元数据。

属性	描述
message_id	邮件的 ID。Amazon Pinpoint 在接受邮件时会自动生成此编号。
message_send_timestamp	发送邮件的日期和时间，按 <a href="#">RFC 822</a> 中指定的格式显示。
from_address	发送邮件的电子邮件地址。



属性	描述
destination	包含邮件发送到的电子邮件地址的数组。
headers_truncated	一个布尔值，用于指示是否截断电子邮件标头。
headers	<p>一个对象，其中包含与电子邮件中标头对应的多个名称/值对。此对象通常包含有关以下标头的信息：</p> <ul style="list-style-type: none"> <li>• From – 发件人的电子邮件地址。</li> <li>• To – 收件人的电子邮件地址。</li> <li>• Subject – 电子邮件的主题行。</li> </ul> <div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> <b>Tip</b> campaign_email.send 事件不包含主题标题。</p> </div> <ul style="list-style-type: none"> <li>• MIME-Version – 指示邮件为 MIME 格式。如果此标头存在，则值始终为 1.0。</li> <li>• Content-Type – 邮件内容的 MIME 媒体类型。</li> </ul>
common_headers	包含有关电子邮件的几个常用标头的信息。这些信息可以包括邮件的发送日期，以及邮件的收件人、发件人和主题行。

## 短信事件

如果为项目启用了短信渠道，Amazon Pinpoint 可以流式传输有关项目的短信传送事件数据。运营商生成的短信事件最多可能需要 72 小时才能接收，因此不应将其用于判断出站消息传送是否存在延迟。72 小时后，如果 Amazon Pinpoint 仍未收到运营商的最终事件，则该服务将自动返回 UNKNOWN record\_status，因为我们不知道该消息发生了什么情况。

## 示例

短信事件的 JSON 对象包含以下示例中显示的数据。

```
{
  "event_type": "_SMS.SUCCESS",
  "event_timestamp": 1553104954322,
  "arrival_timestamp": 1553104954064,
  "event_version": "3.1",
  "application": {
    "app_id": "a1b2c3d4e5f6g7h8i9j0k1l2m3n4o5p6",
    "sdk": {}
  },
  "client": {
    "client_id": "123456789012"
  },
  "device": {
    "platform": {}
  },
  "session": {},
  "attributes": {
    "sender_request_id": "565d4425-4b3a-11e9-b0a5-example",
    "campaign_activity_id": "cbcfc3c5e3bd48a8ae2b9cb41example",
    "origination_phone_number": "+12065550142",
    "destination_phone_number": "+14255550199",
    "record_status": "DELIVERED",
    "iso_country_code": "US",
    "treatment_id": "0",
    "number_of_message_parts": "1",
    "message_id": "1111-2222-3333",
    "message_type": "Transactional",
    "campaign_id": "52dc44b35c4742c98c5935269example"
  },
  "metrics": {
    "price_in_millicents_usd": 645.0
  },
  "awsAccountId": "123456789012"
}
```

## 短信事件属性

此部分定义在您发送短信时 Amazon Pinpoint 生成的事件流数据中包含的属性。

## 事件

属性	描述
event_type	<p>事件类型。可能的值有：</p> <ul style="list-style-type: none"> <li>_SMS.BUFFERED – 消息仍在传送给接收人的过程中。</li> <li>_SMS.SUCCESS – 消息已成功被运营商接收/传送给接收人。</li> <li>_SMS.FAILURE – Amazon Pinpoint 无法将消息传送给接收人。要了解有关阻止消息传送的错误的更多信息，请参阅 <code>attribute s.record_status</code>。</li> <li>_SMS.OPTOUT – 客户收到消息并通过发送退订关键字（通常为“STOP”）来回复。</li> </ul>
event_timestamp	报告事件的时间，显示为以毫秒为单位的 Unix 时间。
arrival_timestamp	Amazon Pinpoint 收到事件的时间，显示为以毫秒为单位的 Unix 时间。
event_version	<p>事件 JSON 架构的版本。</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Tip</b></p> <p>在事件处理应用程序中检查此版本，以便知道何时更新应用程序以响应架构更新。</p> </div>
application	与事件关联的 Amazon Pinpoint 项目的相关信息。有关更多信息，请参阅 <a href="#">应用程序表</a> 。
client	安装在设备上用于报告事件的应用程序客户端的相关信息。有关更多信息，请参阅 <a href="#">客户端表</a> 。

属性	描述
device	报告事件的设备的相关信息。有关更多信息，请参阅 <a href="#">设备表</a> 。  对于短信事件，此对象为空。
session	对于短信事件，此对象为空。
attributes	与事件关联的属性。对于您的应用程序之一报告的事件，此对象包含由应用程序定义的自定义属性。对于在您发送活动时创建的事件，此对象包含与活动关联的属性。对于在您发送事务性电子邮件时生成的事件，此对象包含与电子邮件本身相关的信息。  有关更多信息，请参阅 <a href="#">属性表</a> 。
metrics	与事件关联的其他指标。有关更多信息，请参阅 <a href="#">指标表</a> 。
awsAccountId	已用于发送电子邮件的 AWS 账户的 ID。

## 应用程序

包括有关与事件关联的 Amazon Pinpoint 项目和（如果适用）用于报告事件的开发工具包的信息。

属性	描述
app_id	报告事件的 Amazon Pinpoint 项目的唯一 ID。
sdk	用于报告该事件的开发工具包。如果您通过直接调用 Amazon Pinpoint API 或使用 Amazon Pinpoint 控制台来发送事务性短信，则此对象为空。

## 属性

包括与事件关联的属性的相关信息。

属性	描述
sender_request_id	与发送短信的请求关联的唯一 ID。
campaign_activity_id	活动内活动的唯一 ID。
origination_phone_number	用于发送消息的电话号码。
destination_phone_number	尝试将消息发送到的电话号码。
record_status	有关信息状态的其他消息。可能的值包括： <ul style="list-style-type: none"><li>• SUCCESSFUL/DELIVERED – 消息已成功传送。</li><li>• PENDING – 消息尚未传送到接收人的设备。</li><li>• INVALID – 目标电话号码无效。</li><li>• UNREACHABLE – 接收人的设备当前无法访问或者不可用。例如，设备可能已关闭，或者可能断开与网络的连接。您可以稍后再次尝试发送消息。</li><li>• UNKNOWN – 出现错误，阻止了消息的传送。此错误通常是临时的，您可以稍后再次尝试发送消息。</li><li>• BLOCKED – 接收人的设备阻止了来自发送号码的短信。</li><li>• CARRIER_UNREACHABLE – 接收人的移动网络出现问题，阻止了消息的传送。此错误通常是临时的，您可以稍后再次尝试发送消息。</li><li>• SPAM – 接收人的移动运营商将消息内容标识为垃圾内容并阻止了消息的传送。</li><li>• INVALID_MESSAGE – 短信的正文无效，无法传送。</li></ul>

属性	描述
	<ul style="list-style-type: none"> <li>• CARRIER_BLOCKED – 接收人的运营商阻止了此消息的传送。当运营商确定消息的内容是未经请求内容或恶意内容时，通常会出现这种情况。</li> <li>• TTL_EXPIRED – 短信无法在特定时间范围内传送。此错误通常是临时的，您可以稍后再次尝试发送消息。</li> <li>• MAX_PRICE_EXCEEDED – 发送消息可能会产生超过您账户的每月短信支出限额的费用。您可以通过完成《Amazon Pinpoint 用户指南》中的<a href="#">请求增加每月短信支出限额</a>中的步骤来申请增加此限额。</li> <li>• OPTED_OUT – 由于收件人选择不接收您的消息，因此未发送短信。</li> <li>• NO_QUOTA_LEFT_ON_ACCOUNT – 您的账户上剩余的支出限额不足，无法发送消息。您可以通过完成《Amazon Pinpoint 用户指南》中的<a href="#">请求增加每月短信支出限额</a>中的步骤来申请增加此限额。</li> <li>• NO_ORIGINATION_IDENTITY_AVAILABLE_TO_SEND – 您的账户中没有可用于向目的地发送消息的电话号码。</li> <li>• DESTINATION_COUNTRY_NOT_SUPPORTED – 目的地国家/地区已被屏蔽。有关所有受支持的国家/地区，请参阅<a href="#">支持的国家 and 地区 (短信渠道)</a>。</li> <li>• ACCOUNT_IN_SANDBOX – 您的账户位于沙盒中，只能发送到经过验证的目的地号码。您可以在 Amazon Pinpoint 控制台中验证目标号码，也可以开始将账户移出沙盒的过程，请参阅<a href="#">从 Amazon Pinpoint 短信沙盒移出到生产环境</a>。</li> </ul>

属性	描述
	<ul style="list-style-type: none"> <li>• RATE_EXCEEDED – 您试图发送消息的速度太快并受到限制。您需要降低调用率。有关我们的限制的详细信息，请参阅<a href="#">每秒消息部分数 (MPS) 限制</a>。</li> <li>• INVALID_ORIGINATION_IDENTITY – 提供的源身份无效。</li> <li>• ORIGINATION_IDENTITY_DOES_NOT_EXIST – 提供的源身份不存在。</li> <li>• INVALID_DLT_PARAMETERS – 提供了无效的 DLT 参数 ( 为印度的目的地所必需 )。</li> <li>• INVALID_PARAMETERS – 提供的参数无效。</li> <li>• ACCESS_DENIED – 您的账户被禁止发送消息。请联系客户支持以找出原因并解决问题。</li> <li>• INVALID_KEYWORD – 提供的关键字无效。关键字的格式可能不正确或未在您的账户中设置。</li> <li>• INVALID_SENDER_ID – 提供的发件人 ID 无效。发件人 ID 的格式或长度可能不正确。</li> <li>• INVALID_POOL_ID – 提供的池 ID 无效。池 ID 的格式可能不正确或不属于您的账户。</li> <li>• SENDER_ID_NOT_SUPPORTED_FOR_DESTINATION – 目的地国家/地区不支持发件人 ID。您必须使用电话号码或其他源身份进行发送。</li> <li>• INVALID_PHONE_NUMBER – 提供的源电话号码无效。电话号码的格式或长度可能不正确。</li> </ul>
iso_country_code	与接收人的电话号码关联的国家，按 ISO 3166-1 alpha-2 格式显示。
treatment_id	在 A/B 活动中发送消息时，消息处理的 ID。

属性	描述
treatment_id	如果使用 A/B 测试活动发送了消息，则此值表示消息的处理编号。对于事务性短信，此值为 0。
number_of_message_parts	Amazon Pinpoint 为了发送消息而创建的消息部分数量。  通常，短信只能包含 160 个 GSM-7 字符或 67 个非 GSM 字符，但这些限制会因国家而异。如果您发送的消息超出了这些限制，Amazon Pinpoint 会自动将消息拆分为较小的部分。我们根据您发送的消息部分数量收取费用。
message_id	Amazon Pinpoint 在接受消息时生成的唯一 ID。
message_type	消息类型。可能的值为 Promotional 和 Transactional。您可以在创建活动，或在 Amazon Pinpoint API 中使用 <a href="#">SendMessages</a> 操作来发送事务性消息时指定此值。
campaign_id	发送消息的 Amazon Pinpoint 活动的唯一 ID。

## 客户端

包括安装在设备上用于报告事件的应用程序客户端的相关信息。

属性	描述
client_id	对于应用程序生成的事件，此值是安装在设备上的应用程序客户端的唯一 ID。此 ID 由 AWS Mobile SDK for iOS 和 AWS Mobile SDK for Android 自动生成。  对于在您发送活动和事务性消息时生成的事件，此值等于您将消息发送到的端点的 ID。



属性	描述
cognito_id	在应用程序使用的 Amazon Cognito 身份池中分配给应用程序客户端的唯一 ID。

## 设备

包括报告事件的设备的相关信息。

属性	描述
locale	设备区域设置。
make	设备制造商，如 Apple 或 Samsung。
model	设备型号，如 iPhone。
platform	设备平台，如 ios 或 android。

## 指标

包括与事件关联的指标的相关信息。

属性	描述
price_in_millicents_usd	<p>我们向您收取的发送消息的费用。此价格以千分之一美分显示。例如，如果此属性的值为 645，则我们收取的消息发送单价是 0.645¢ (645 / 1000 = 0.645¢ = \$0.00645)。</p> <div data-bbox="829 1562 1508 1780" style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px;"> <p> <b>Note</b></p> <p>对于 event_type 为 _SMS.BUFFERED 的消息，不显示此属性。</p> </div>

## 查询 Amazon Pinpoint 分析数据

除了使用 Amazon Pinpoint 控制台上的分析页面之外，您还可以使用 Amazon Pinpoint Analytics API 查询一部分标准指标的分析数据，从而了解与用户参与度、活动推广等相关的趋势。这些指标也称为关键绩效指标 (KPI)，它们是一些可测量的值，能帮助您监控和评估项目、活动和旅程的绩效。

如果您使用 API 查询分析数据，则可以使用所选的报告工具来分析数据，而无需登录 Amazon Pinpoint 控制台，或者分析来自诸如 Amazon Kinesis Streams 等源的原始事件数据。例如，您可以构建一个自定义控制面板，来显示每周活动成果或提供活动送达率的深度分析数据。

您可以使用 Amazon Pinpoint REST API、AWS Command Line Interface (AWS CLI) 或软件开发工具包查询数据。AWS 要查询数据，请向 Amazon Pinpoint API 发送请求，并使用支持的参数指定您所需的数据以及要应用的任何筛选器。提交查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。然后，您可以将这些结果传递到其他服务或应用程序，以便进行更深入的分析、存储或报告。

## 受支持的指标

Amazon Pinpoint 提供对于以下几种标准指标的分析数据的编程访问：

- 应用程序指标 – 通过这些指标可了解与项目相关的所有活动和事务性消息的趋势。例如，您可以使用应用程序指标了解为项目的各个相关活动发送的消息中，具体有多少被收件人打开。要访问应用程序指标的数据，请使用 Amazon Pinpoint API 的[应用程序指标](#)资源。
- 活动指标 – 通过这些指标可了解单个活动的绩效。例如，您可以使用活动指标来确定将活动消息发送到了多少个端点。要访问活动指标的数据，请使用 Amazon Pinpoint API 的[活动指标](#)资源。
- 旅程参与指标 – 通过这些指标可以了解各个旅程的绩效。例如，您可以使用旅程参与指标获取在旅程的每个活动中，参与者打开的消息数量的明细。要访问旅程参与指标的相关数据，请使用 Amazon Pinpoint API 的[旅程参与指标](#)资源。
- 旅程执行指标 – 通过这些指标可以了解各个旅程的参与趋势。例如，您可以使用旅程执行指标来确定有多少个参与者在进行旅程中的活动。要访问旅程执行指标的数据，请使用 Amazon Pinpoint API 的[旅程执行指标](#)资源。
- 旅程活动执行指标 – 通过这些指标可以了解旅程中的各个活动的参与趋势。例如，您可以使用旅程活动执行指标确定有多少个参与者完成了活动。要访问旅程活动执行指标的数据，请使用 Amazon Pinpoint API 的[旅程活动执行指标](#)资源。

有关可通过编程方式查询的标准指标的完整列表，请参阅[标准指标](#)。

Amazon Pinpoint 自动为所有项目、活动和旅程收集和聚合所有受支持指标的数据。此外，由于这些数据不断更新，导致数据延迟，但延迟时间限于约两小时内。但要注意，某些指标可能具有更长时间的数据延迟。这是因为，某些指标的数据基于我们从收件人的电子邮件提供商那里收到的信息。一些提供商会立即向我们发送此类信息，而另一些提供商可能不会这么快地向我们发送信息。

Amazon Pinpoint 将这些数据存储 90 天。要想将数据存储 90 天以上或实时访问原始分析数据，您可以配置 Amazon Pinpoint 项目以将事件数据流式流传输到 Amazon Kinesis Data Streams 或 Amazon Kinesis Data Firehose。有关配置事件流的信息，请参阅[将 Amazon Pinpoint 事件流式传输到 Kinesis](#)。

## 查询基础知识

要查询指标的数据，您可以将 get 请求发送到 Amazon Pinpoint API 的相应指标资源。在您的请求中，您可以使用以下查询组件的受支持参数来定义查询：

- 项目 – 通过提供项目 ID 作为 `application-id` 参数的值来指定项目。此参数是所有指标的必需参数。
- 活动 – 通过提供活动 ID 作为 `campaign-id` 参数的值来指定活动。此参数仅是活动指标的必需参数。
- 旅程 – 通过提供旅程 ID 作为 `journey-id` 参数的值来指定旅程。仅旅程参与和执行指标以及旅程活动执行指标需要使用该参数。
- 旅程活动 – 通过提供旅程活动 ID 作为 `journey-activity-id` 参数的值来指定旅程活动。仅旅程活动执行指标需要使用该参数。
- 日期范围 ( 可选 ) – 要按日期范围筛选数据，请使用支持的开始和结束时间参数来提供日期范围的起始和截止日期和时间。这些值应采用扩展的 ISO 8601 格式，并使用协调世界时 (UTC)，例如，`2019-07-19T20:00:00Z` 表示协调世界时 2019 年 7 月 19 日晚上 8 点。

日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始日期和时间必须距离当前日期不到 90 天。如果未指定日期范围，则 Amazon Pinpoint 返回前 31 个日历日期间的数据。除了旅程执行指标和旅程活动执行指标以外，所有其他指标均支持日期范围参数。

- 指标 – 通过提供指标名称作为 `kpi-name` 参数的值来指定指标。此值描述了关联的指标并包含两个或两个以上的术语，这些术语由小写字母数字字符组成并由连字符分隔。示例包括 `email-open-rate` 和 `successful-delivery-rate`。除了旅程执行指标和旅程活动执行指标以外，所有其他指标均需要使用该参数。有关受支持的指标及其所用的 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

发送查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。在响应中，结果的结构因您查询的指标而异。

某些指标仅提供一个值，例如，某个活动送达的消息数。某些指标则提供多个值，并且这些值通常按相关字段进行分组，例如，对于某个活动的每次运行，按活动运行来分组送达的消息数量。如果指标提供多个值并进行分组，则 JSON 响应包括一个字段，以指示使用哪个字段对数据进行分组。要了解有关查询结果结构的更多信息，请参阅[使用查询结果](#)。

## 有关查询 Amazon Pinpoint 分析数据的 IAM 策略

通过使用 Amazon Pinpoint API，可以查询分析数据来获取标准指标的子集，这些标准指标也称为适用于 Amazon Pinpoint 项目、活动和旅程的关键绩效指标 (KPI)。这些指标可以帮助您监控和评估项目、活动和旅程的绩效。

要管理对这些数据的访问，您可以创建 AWS Identity and Access Management (IAM) 策略，来为获准访问数据的 IAM 角色或用户定义权限。为帮助精细控制对这些数据的访问，Amazon Pinpoint 提供了几种您可以在 IAM 策略中指定的不同操作。一种操作是在 Amazon Pinpoint 控制台上查看分析数据 (mobiletargeting:GetReports)，其他操作还包括使用 Amazon Pinpoint API 以编程方式访问分析数据等。

要创建管理对分析数据的访问的 IAM 策略，您可以使用 AWS Management Console、AWS CLI 或 IAM API。请注意，AWS Management Console 上的可视化编辑器选项卡目前不包含查看或查询 Amazon Pinpoint 分析数据的操作。不过，您可以使用控制台上的 JSON 选项卡手动向 IAM 策略添加必要操作。

例如，以下策略允许对所有 AWS 区域中的所有项目、活动和旅程的所有分析数据进行编程访问：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "QueryAllAnalytics",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:GetApplicationDateRangeKpi",
        "mobiletargeting:GetCampaignDateRangeKpi",
        "mobiletargeting:GetJourneyDateRangeKpi",
        "mobiletargeting:GetJourneyExecutionMetrics",
        "mobiletargeting:GetJourneyExecutionActivityMetrics"
      ]
    }
  ],
}
```

```

    "Resource": [
      "arn:aws:mobiletargeting:*:accountId:apps/*/kpis/*",
      "arn:aws:mobiletargeting:*:accountId:apps/*/campaigns/*/kpis/*",
      "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/kpis/*",
      "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/execution-
metrics",
      "arn:aws:mobiletargeting:*:accountId:apps/*/journeys/*/activities/*/
execution-metrics"
    ]
  }
]
}

```

其中，*accountId* 是您的 AWS 账户 ID。

但作为最佳实践，您应创建遵循最低权限原则的策略。换句话说，您应创建仅包含执行特定任务所需的权限的策略。为了支持此实践并实施更精细的控制，您可以将对分析数据的编程访问权限限于特定 AWS 区域的特定项目，例如：

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "QueryProjectAnalytics",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:GetApplicationDateRangeKpi",
        "mobiletargeting:GetCampaignDateRangeKpi",
        "mobiletargeting:GetJourneyDateRangeKpi",
        "mobiletargeting:GetJourneyExecutionMetrics",
        "mobiletargeting:GetJourneyExecutionActivityMetrics"
      ],
      "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/kpis/*",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/campaigns/*/
kpis/*",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
kpis/*",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
execution-metrics",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/*/
activities/*/execution-metrics"
      ]
    }
  ]
}

```

```
    }  
  ]  
}
```

其中：

- *region* 是托管项目的 AWS 区域的名称。
- *accountId* 是您的 AWS 账户 ID。
- *projectId* 是您要提供其访问权限的项目的标识符。

同样，以下策略示例仅允许编程访问特定活动的分析数据：

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "QueryCampaignAnalytics",  
      "Effect": "Allow",  
      "Action": "mobiletargeting:GetCampaignDateRangeKpi",  
      "Resource": "arn:aws:mobiletargeting:region:accountId:apps/projectId/  
campaigns/campaignId/kpis/*"  
    }  
  ]  
}
```

其中：

- *region* 是托管项目的 AWS 区域的名称。
- *accountId* 是您的 AWS 账户 ID。
- *projectId* 是与活动关联的项目的标识符。
- *campaignId* 是您要提供其访问权限的活动的标识符。

以下策略示例允许编程访问特定旅程和构成该旅程的活动的的所有分析数据，包括参与度和执行数据：

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "QueryJourneyAnalytics",
```

```
    "Effect": "Allow",
    "Action": [
        "mobiletargeting:GetJourneyDateRangeKpi",
        "mobiletargeting:GetJourneyExecutionMetrics",
        "mobiletargeting:GetJourneyExecutionActivityMetrics"
    ],
    "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/
        journeys/journeyId/kpis/*",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/
        journeys/journeyId/execution-metrics",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/
        journeys/journeyId/activities/*/execution-metrics"
    ]
  }
]
```

其中：

- *region* 是托管项目的 AWS 区域的名称。
- *accountId* 是您的 AWS 账户 ID。
- *projectId* 是与旅程关联的项目的标识符。
- *journeyId* 是您要提供其访问权限的旅程的标识符。

有关您可以在 IAM 策略中使用的 Amazon Pinpoint API 操作的完整列表，请参阅[用于 IAM 策略的 Amazon Pinpoint 操作](#)。有关创建和管理 IAM 策略的详细信息，请参阅[IAM 用户指南](#)。

## Amazon Pinpoint 标准分析指标

您可以使用 Amazon Pinpoint Analytics API 查询一部分适用于 Amazon Pinpoint 项目、活动和旅程的标准指标的分析数据。这些指标也称为关键绩效指标 (KPI)，它们是一些可测量的值，能帮助您监控和评估项目、活动和旅程的绩效。

Amazon Pinpoint 提供对于以下几种标准指标的分析数据的编程访问：

- 应用程序指标 – 通过这些指标可以了解与项目（也称为应用程序）相关的所有活动和事务性消息的趋势。例如，您可以使用应用程序指标了解为项目的各个相关活动发送的消息中，具体有多少被收件人打开。

- **活动指标** – 通过这些指标可了解单个活动的绩效。例如，您可以使用活动指标来确定已向多少个端点发送活动消息，或者其中有多少消息送达端点。
- **旅程参与指标** – 通过这些指标可以了解各个旅程的绩效。例如，您可以使用旅程参与指标获取在旅程的每个活动中，参与者打开消息的数量的明细。
- **旅程执行指标** – 通过这些指标可以了解各个旅程的参与趋势。例如，您可以使用旅程执行指标确定有多少个参与者启动了旅程。
- **旅程活动执行指标** – 通过这些指标可以了解旅程中的各个活动的参与趋势。例如，您可以使用旅程活动执行指标确定有多少个参与者启动了活动，以及有多少参与者完成了活动中的每个路径。

此部分中的主题列出并描述了每种指标类型可查询的各个指标。

## 主题

- [活动的应用程序指标](#)
- [事务性电子邮件消息的应用程序指标](#)
- [事务性短信的应用程序指标](#)
- [活动指标](#)
- [旅程参与指标](#)
- [旅程执行指标](#)
- [旅程活动执行指标](#)
- [旅程和活动执行指标](#)

## 活动的应用程序指标

下表列出并描述了有关活动的标准应用程序指标，您可以查询这些指标来评估与 Amazon Pinpoint 项目相关的所有活动的绩效。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[应用程序指标](#)资源。该表中的 kpi-name 列表示查询中的 kpi-name 参数所用的值。

指标	Kpi-name	描述
送达率	successful-delivery-rate	<p>对于项目相关的所有活动，已送达收件人的消息的百分比。</p> <p>此指标的计算方式为：项目的所有活动已发送并且已送达</p>



指标	Kpi-name	描述
		收件人的消息的数量，除以所有这些活动已发送的消息的数量。
按日期分组的送达率	successful-delivery-rate-grouped-by-date	<p>对于项目相关的所有活动，在指定日期范围内的每一天已送达收件人的消息的百分比。</p> <p>此指标的计算方式为：在指定日期范围内的每一天，项目的所有活动发送的并且已送达收件人的消息的数量，除以所有这些活动发送的消息的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
电子邮件打开率	email-open-rate	<p>对于项目相关的所有活动，收件人打开的电子邮件的百分比。</p> <p>此指标的计算方式为：项目的所有活动发送的并且收件人打开的电子邮件的数量，除以所有这些活动发送的并且已送达收件人的电子邮件的数量。</p>

指标	Kpi-name	描述
按活动分组的电子邮件打开率	email-open-rate-grouped-by-campaign	<p>对于项目相关的每个活动，收件人打开的电子邮件的百分比。</p> <p>此指标的计算方式为：某个活动发送的并且收件人打开的电子邮件的数量，除以该活动发送的并且已送达收件人的电子邮件的数量。</p> <p>此指标的查询结果按活动 ID (CampaignId) 分组，这是一个可唯一识别活动的字符串。</p>
端点送达数	unique-deliveries	对于项目相关的所有活动，消息送达的唯一端点数量。
按活动分组的端点送达数	unique-deliveries-grouped-by-campaign	<p>对于项目相关的每个活动，消息送达的唯一端点数量。</p> <p>此指标的查询结果按活动 ID (CampaignId) 分组，这是一个可唯一识别活动的字符串。</p>
按日期分组的端点送达数	unique-deliveries-grouped-by-date	<p>对于项目相关的所有活动，在指定日期范围内的每一天消息送达的唯一端点数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>

指标	Kpi-name	描述
按活动分组的已送达消息数	successful-deliveries-grouped-by-campaign	<p>对于项目相关的每个活动，送达收件人的消息的数量。</p> <p>此指标的计算方式为：某个活动发送的消息数量，减去此活动发送的但因硬退信而无法送达收件人的消息数量。</p> <p>此指标的查询结果按活动 ID (CampaignId ) 分组，这是一个可唯一识别活动的字符串。</p>
Push open rate (推送通知打开率)	push-open-rate	<p>对于项目相关的所有活动，收件人打开的推送通知百分比。</p> <p>此指标的计算方式为：项目的所有活动发送的并且收件人打开的推送通知数量，除以所有这些活动发送并送达到收件人的推送通知数量。</p>
按活动分组的推送通知打开率	push-open-rate-grouped-by-campaign	<p>对于项目相关的每个活动，收件人打开的推送通知百分比。</p> <p>此指标的计算方式为：某个活动发送的并且收件人打开的推送通知数量，除以此活动发送并送达到收件人的推送通知数量。</p> <p>此指标的查询结果按活动 ID (CampaignId ) 分组，这是一个可唯一识别活动的字符串。</p>

## 事务性电子邮件消息的应用程序指标

下表列出并描述了有关事务性电子邮件消息的标准应用程序指标，您可以查询这些指标来监控与 Amazon Pinpoint 项目相关的所有事务性电子邮件消息的趋势。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[应用程序指标](#)资源。该表中的 kpi-name 列表示查询中的 kpi-name 参数所用的值。

请注意，这些指标不提供关于活动所发送的电子邮件消息的数据。它们仅提供有关事务性电子邮件消息的数据。要查询一个或多个活动所发送的消息的数据，请使用[活动指标](#)或[活动应用程序指标](#)。

指标	Kpi-name	描述
点击次数	txn-emails-clicked	收件人点击消息中的链接的次数。如果一个收件人单击了一封邮件中的多个链接，或多次单击同一链接，则每次单击均包含在该计数中。
按日期分组的点击次数	txn-emails-clicked-grouped-by-date	在指定日期范围内的每一天，收件人点击消息中链接的次数。如果一个收件人单击了一封邮件中的多个链接，或多次单击同一链接，则每次单击均包含在该计数中。  此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。
投诉率	txn-emails-complaint-rate	收件人报告的未经请求或不需要的电子邮件所占的百分比。  该指标的计算方法为：收件人报告的未经请求或不需要的电子邮件的数量除以已发送邮件的数量。
按日期分组的投诉率	txn-emails-complaint-rate-grouped-by-date	在指定日期范围内的每一天，收件人报告的未经请求或不需要的电子邮件所占的百分比。

指标	Kpi-name	描述
		<p>该指标的计算方法为：在指定日期范围内的每一天，收件人报告的未经请求或不需要的电子邮件的数量除以已发送邮件的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
投诉数	txn-emails-with-complaints	收件人报告的未经请求或不需要的电子邮件的数量。
按日期分组的投诉数	txn-emails-with-complaints-grouped-by-date	<p>在指定日期范围内的每一天，收件人报告的未经请求或不需要的电子邮件的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
已传送数	txn-emails-delivered	<p>已送达收件人的邮件的数量。</p> <p>该指标的计算方法是：发送的消息数减去因软或硬退信或者被拒绝而无法传送的消息数。如果 Amazon Pinpoint 确定消息中包含恶意软件，则该消息将被拒绝。Amazon Pinpoint 不会尝试发送被拒绝的消息。</p>

指标	Kpi-name	描述
按日期分组的已传送数	txn-emails-delivered-grouped-by-date	<p>在指定日期范围内的每一天，已送达收件人的邮件的数量。</p> <p>此指标的计算方法为：在指定日期范围内的每一天，已发送消息的数量减去因软或硬退信或者被拒绝而无法传送的消息数。如果 Amazon Pinpoint 确定消息中包含恶意软件，则该消息将被拒绝。Amazon Pinpoint 不会尝试发送被拒绝的消息。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
送达率	txn-emails-delivery-rate	<p>已送达收件人的消息所占的百分比。</p> <p>此指标的计算方法为：已发送并且已送达收件人的消息的数量除以已发送消息的数量。</p>
按日期分组的送达率	txn-emails-delivery-rate-grouped-by-date	<p>在指定日期范围内的每一天，已送达收件人的消息所占的百分比。</p> <p>此指标的计算方法为：在指定日期范围内的每一天，已发送并且已送达收件人的消息的数量除以已发送消息的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>

指标	Kpi-name	描述
硬退信数	txn-emails-hard-bounced	因硬退信而导致无法送达收件人的消息的数量。如果因持久性问题（例如，收件人的电子邮件地址不存在）导致邮件无法送达，会造成硬退信。
按日期分组的硬退信数	txn-emails-hard-bounced-grouped-by-date	<p>在指定日期范围内的每一天，因硬退信而导致无法送达收件人的消息的数量。如果因持久性问题（例如，收件人的电子邮件地址不存在）导致邮件无法送达，会造成硬退信。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
打开	txn-emails-opened	收件人打开的消息的数量。
按日期分组的打开次数	txn-emails-opened-grouped-by-date	<p>在指定日期范围内的每一天，收件人打开的消息的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
发送	txn-emails-sent	发送的消息的数量。
按日期分组的发送数	txn-emails-sent-grouped-by-date	<p>在指定日期范围内的每一天发送的邮件的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>

指标	Kpi-name	描述
软退信数	txn-emails-soft-bounced	因软退信导致无法送达收件人的消息的数量。如果因临时性问题（例如收件人的收件箱已满或接收服务器暂时不可用）导致邮件无法送达，会造成软退信。
按日期分组的软退信数	txn-emails-soft-bounced-grouped-by-date	<p>在指定日期范围内的每一天，因软退信导致无法送达收件人的消息的数量。如果因临时性问题（例如收件人的收件箱已满或接收服务器暂时不可用）导致邮件无法送达，会造成软退信。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
唯一用户点击事件	txn-emails-unique-clicks	<p>点击消息中链接的唯一收件人（端点）的数量。</p> <p>与点击次数指标不同，该指标报告的是点击链接的唯一收件人的数量，而不是发生的点击事件的次数。例如，如果一个收件人单击了同一邮件中的多个链接，或多次单击同一个链接，该指标报告为该收件人只有一个单击事件。</p>



指标	Kpi-name	描述
按日期分组的唯一用户点击事件	txn-emails-unique-clicks-grouped-by-date	<p>在指定日期范围内的每一天，点击消息中链接的唯一收件人（端点）的数量。</p> <p>与按日期分组的点击次数指标不同，该指标报告的是点击链接的唯一收件人的数量，而不是发生的点击事件的次数。例如，如果一个收件人单击了同一邮件中的多个链接，或多次单击同一个链接，该指标报告为该收件人只有一个单击事件。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
唯一用户打开事件	txn-emails-unique-opens	<p>打开消息的唯一收件人（端点）的数量。</p> <p>与打开次数指标不同，该指标报告的是打开消息的唯一收件人的数量，而不是发生的打开事件的次数。例如，如果一个收件人多次打开同一封邮件，该指标报告为该收件人只有一个打开事件。</p>

指标	Kpi-name	描述
按日期分组的唯一用户打开事件	txn-emails-unique-opens-grouped-by-date	<p>在指定日期范围内的每一天，打开消息的唯一收件人（端点）的数量。</p> <p>与按日期分组的打开次数指标不同，该指标报告的是打开消息的唯一收件人的数量，而不是发生的打开事件的次数。例如，如果一个收件人多次打开同一封邮件，该指标报告为该收件人只有一个打开事件。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>

## 事务性短信的应用程序指标

下表列出并描述了有关事务性短信的标准应用程序指标，您可以查询这些指标来了解与 Amazon Pinpoint 项目相关的所有事务性短信的趋势。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[应用程序指标](#)资源。该表中的 kpi-name 列表示查询中的 kpi-name 参数所用的值。

请注意，这些指标不提供有关活动发送的短信的数据。它们仅提供有关事务性短信的数据。要查询一个或多个活动所发送的消息的数据，请使用[活动指标](#)或[活动应用程序指标](#)。

指标	Kpi-name	描述
按国家/地区分组的每消息平均价格	txn-sms-average-price-grouped-by-country	<p>对于消息发往的每个国家或地区，发送每个消息的平均成本。价格单位为千分之一美分。例如，如果此属性的值为 645，则我们收取的消息发送单价是 0.645¢ (645 / 1000 = 0.645¢ = \$0.00645)。</p>

指标	Kpi-name	描述
		<p>此指标的计算方式是：发送给某个国家或地区的收件人的所有消息的总成本除以发送给该国家或地区的收件人的消息数。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
按国家/地区分组的每消息部分的平均价格	txn-sms-average-price-by-parts-grouped-by-country	<p>对于消息发往的每个国家或地区，发送每个消息部分的平均成本。消息部分是短信的一部分。价格单位为千分之一美分。例如，如果此属性的值为 645，则我们收取的消息发送单价是 0.645¢ (<math>645 / 1000 = 0.645¢ = \\$0.00645</math>)。</p> <p>此指标的计算方式是：发送给某个国家或地区的收件人的所有消息部分的总成本除以发送给该国家或地区的收件人的消息部分数。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
已传送数	txn-sms-delivered	已送达收件人的邮件的数量。

指标	Kpi-name	描述
按国家/地区分组的已送达数	txn-sms-delivered-grouped-by-country	<p>对于消息发往的每个国家或地区，已送达收件人的消息数量。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
按日期分组的已传送数	txn-sms-delivered-grouped-by-date	<p>在指定日期范围内的每一天，已送达收件人的邮件的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
传输错误数	txn-sms-error-distribution	<p>传输消息时发生的各种错误的次数。</p> <p>此指标的查询结果针对发生的每种类型的错误按错误代码来分组。</p>
送达率	txn-sms-delivery-rate	<p>已送达收件人的消息所占的百分比。</p> <p>此指标的计算方法为：已发送并且已送达收件人的消息的数量除以已发送消息的数量。</p>

指标	Kpi-name	描述
按日期分组的送达率	txn-sms-delivery-rate-grouped-by-date	<p>在指定日期范围内的每一天，已送达收件人的消息所占的百分比。</p> <p>此指标的计算方法为：在指定日期范围内的每一天，已发送并且已送达收件人的消息的数量除以已发送消息的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
消息部分已送达数	txn-sms-delivered-by-parts	<p>已送达的消息部分数。消息部分是短信的一部分。如果短信包含的字符数超过短信协议允许的字符数，Amazon Pinpoint 会根据需要将消息拆分为若干消息部分，以便将消息发送给收件人。</p>
按国家/地区分组的消息部分已送达数	txn-sms-delivered-by-parts-grouped-by-country	<p>对于消息发往的每个国家或地区，已送达收件人的消息部分数量。消息部分是短信的一部分。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
消息部分已发送数	txn-sms-sent-by-parts	<p>发送的消息部分的数量。消息部分是短信的一部分。如果短信包含的字符数超过短信协议允许的字符数，Amazon Pinpoint 会根据需要将消息拆分为若干消息部分，以便将消息发送给收件人。</p>

指标	Kpi-name	描述
按国家/地区分组的已发送的消息部分数	txn-sms-sent-by-parts-grouped-by-country	<p>对于消息发往的每个国家或地区，已发送的消息部分数量。消息部分是短信的一部分。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
发送的消息数	txn-sms-sent	发送的消息的数量。
按国家/地区分组的已发送的消息数	txn-sms-sent-grouped-by-country	<p>对于消息发往的每个国家或地区，已发送的消息数量。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>
按日期分组的已发送的消息数	txn-sms-sent-grouped-by-date	<p>在指定日期范围内的每一天发送的邮件的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
按国家/地区分组的总价格	txn-sms-total-price-grouped-by-country	<p>对于消息发往的每个国家或地区，发送的消息的总成本。价格单位为千分之一美分。例如，如果此属性的值为 645，则我们收取的消息发送单价是 0.645¢ (645 / 1000 = 0.645¢ = \$0.00645)。</p> <p>此指标的查询结果按国家或地区分组，采用 ISO 3166-1 alpha-2 格式。</p>

## 活动指标

下表列出并描述了标准活动指标，您可以查询这些指标来评估单个活动的绩效。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[活动指标](#)资源。该表中的 kpi-name 列表示查询中的 kpi-name 参数所用的值。

指标	Kpi-name	描述
退回邮件率	hard-bounce-rate	<p>对于所有活动运行，无法送达收件人的电子邮件的百分比。此指标仅用于衡量硬退信，即，邮件的收件人电子邮件地址存在永久性的问题，使得邮件无法送达。</p> <p>此指标的计算方式为：所有活动运行发送的但被退回的电子邮件的数量，除以所有这些活动运行发送的电子邮件的数量。</p>
按活动运行分组的退回邮件率	hard-bounce-rate-grouped-by-campaign-activity	<p>对于每个活动运行，无法送达收件人的电子邮件的百分比。此指标仅用于衡量硬退信，即，邮件的收件人电子邮件地址存在永久性的问题，使得邮件无法送达。</p> <p>此指标的计算方式为：某个活动运行发送的但被退回的电子邮件的数量，除以该活动运行发送的电子邮件的数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>

指标	Kpi-name	描述
送达率	successful-delivery-rate	<p>对于所有活动运行，已送达收件人的消息的百分比。</p> <p>此指标的计算方式为：所有活动运行发送的并且已送达收件人的消息的数量，除以所有这些活动运行发送的消息的数量。</p>
按活动运行分组的送达率	successful-delivery-rate-grouped-by-campaign-activity	<p>对于每个活动运行，送达收件人的消息的百分比。</p> <p>此指标的计算方式为：某个活动运行发送的并且已送达收件人的消息的数量，除以此活动运行发送的消息的数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
按日期分组的送达率	successful-delivery-rate-grouped-by-date	<p>对于所有活动运行，在指定日期范围内的每一天已送达收件人的消息的百分比。</p> <p>此指标的计算方式为：在指定日期范围内的每一天，所有活动运行发送的并且已送达收件人的消息的数量，除以所有这些活动运行发送的消息的数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>



指标	Kpi-name	描述
电子邮件打开率	email-open-rate	<p>对于所有活动运行，收件人已打开的电子邮件的百分比。</p> <p>此指标的计算方式为：所有活动运行发送的并且收件人打开的电子邮件的数量，除以所有这些活动运行发送的并且已送达收件人的电子邮件的数量。</p>
按活动运行分组的电子邮件打开率	email-open-rate-grouped-by-campaign-activity	<p>对于每个活动运行，收件人已打开的电子邮件的百分比。</p> <p>此指标的计算方式为：某个活动运行发送的并且收件人打开的电子邮件的数量，除以此活动运行发送的并且已送达收件人的电子邮件的数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
按活动运行分组的已打开电子邮件数	direct-email-opens-grouped-by-campaign-activity	<p>对于每个活动运行，收件人已打开的电子邮件的数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
端点送达数	unique-deliveries	对于所有活动运行，消息送达到的唯一端点数量。

指标	Kpi-name	描述
按活动运行分组的端点送达数	unique-deliveries-grouped-by-campaign-activity	<p>对于每个活动运行，消息送达到的唯一端点数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
按日期分组的端点送达数	unique-deliveries-grouped-by-date	<p>对于所有活动运行，在指定日期范围内的每一天消息送达到的唯一端点数量。</p> <p>此指标的查询结果以扩展 ISO 8601 格式按日历日进行分组。</p>
按活动运行分组的已点击链接数	clicks-grouped-by-campaign-activity	<p>对于每个活动运行，收件人点击电子邮件中的链接的次数。如果一个收件人点击了邮件中的多个链接，或多次点击同一链接，则每次点击均计数。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>

指标	Kpi-name	描述
按活动运行分组的已送达消息数	successful-deliveries-grouped-by-campaign-activity	<p>对于每个活动运行，已送达收件人的消息的数量。</p> <p>此指标的计算方式为：某个活动运行发送的消息数量，减去因硬退信而无法送达该运行的收件人的消息数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
按活动运行分组的已发送消息数	attempted-deliveries-grouped-by-campaign-activity	<p>对于每个活动运行，已发送的消息的数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId ) 分组，这是一个可唯一识别活动运行的字符串。</p>
Push open rate (推送通知打开率)	push-open-rate	<p>对于所有活动运行，收件人已打开的推送通知百分比。</p> <p>此指标的计算方式为：所有活动运行发送的并且收件人打开的推送通知数量，除以所有这些活动运行发送并送达到收件人的推送通知数量。</p>

指标	Kpi-name	描述
按活动运行分组的推送通知打开率	push-open-rate-grouped-by-campaign-activity	<p>对于每个活动运行，收件人已打开的推送通知百分比。</p> <p>此指标的计算方式为：某个活动运行发送的并且收件人打开的推送通知数量，除以该活动运行发送并送达到收件人的推送通知数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId) 分组，这是一个可唯一识别活动运行的字符串。</p>
按活动运行分组的已打开推送通知总数	direct-push-opens-grouped-by-campaign-activity	<p>对于每个活动运行，收件人已打开的推送通知数量。</p> <p>此指标的查询结果按活动 ID (CampaignActivityId) 分组，这是一个可唯一识别活动运行的字符串。</p>
短信总支出	sms-spend	对于所有活动，发送短信的支出总金额（以毫美分为单位）。

## 旅程参与指标

下表列出并描述了标准旅程参与指标，您可以查询这些指标以监控 Amazon Pinpoint 旅程发送的所有电子邮件的趋势。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[旅程参与指标](#)资源。该表中的 kpi-name 列表示查询中的 kpi-name 参数所用的值。

指标	Kpi-name	描述
点击次数	journey-emails-clicked	参与者点击消息中的链接的次数。如果一个参与者点击了一

指标	Kpi-name	描述
		个消息中的多个链接，或多次点击同一链接，则每次点击均计数。
按活动分组的点击次数	emails-clicked-grouped-by-journey-activity	<p>对于旅程中的每个活动，参与者点击消息中的链接的次数。如果一个参与者点击了一个消息中的多个链接，或多次点击同一链接，则每次点击均计数。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
投诉	journey-emails-complained	参与者报告的未经请求或不需要的电子邮件的数量。
按活动分组的投诉数	emails-complained-grouped-by-journey-activity	<p>对于旅程中的每个活动，参与者报告的未经请求或不需要的电子邮件的数量。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
已传送数	journey-emails-delivered	<p>已送达参与者的消息数。</p> <p>该指标的计算方法是：发送的消息数减去由于软或硬退信或者被拒绝而无法传送的消息数。</p>

指标	Kpi-name	描述
按活动分组的已送达数	emails-delivered-grouped-by-journey-activity	<p>对于旅程中的每个活动，送达参与者的消息数。</p> <p>该指标的计算方式是：对于旅程中的每个活动，发送的消息数减去由于软或硬退信或者被拒绝而无法传送的消息数。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
硬退信数	journey-emails-hardbounced	<p>由于硬退信而无法传送到参与者的消息数。如果因持久性问题（例如，参与者的电子邮件地址不存在）导致邮件无法送达，会造成硬退信。</p>
按活动分组的硬退信数	emails-hardbounced-grouped-by-journey-activity	<p>对于旅程中的每个活动，由于硬退信而无法传送到参与者的消息数。如果因持久性问题（例如，参与者的电子邮件地址不存在）导致邮件无法送达，会造成硬退信。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
打开	journey-emails-opened	<p>参与者打开的消息数。</p>

指标	Kpi-name	描述
按活动分组的打开次数	emails-opened-grouped-by-journey-activity	<p>对于旅程中的每个活动，参与者打开的消息数。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
拒绝数	journey-emails-rejected	<p>由于被拒绝而未发送到参与者的消息数。如果 Amazon Pinpoint 确定消息中包含恶意软件，则该消息将被拒绝。Amazon Pinpoint 不会尝试发送被拒绝的消息。</p>
按活动分组的拒绝数	emails-rejected-grouped-by-journey-activity	<p>对于旅程中的每个活动，由于被拒绝而未发送到参与者的消息数。如果 Amazon Pinpoint 确定消息中包含恶意软件，则该消息将被拒绝。Amazon Pinpoint 不会尝试发送被拒绝的消息。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
发送	journey-emails-sent	发送的消息的数量。

指标	Kpi-name	描述
按活动分组的发送数	emails-sent-grouped-by-journey-activity	<p>对于旅程中的每个活动，发送的消息数。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
软退信数	journey-emails-softbounced	<p>由于软退信而无法传送到参与者的消息数。如果因临时性问题（例如参与者的收件箱已满或接收服务器暂时不可用）导致邮件无法送达，会造成软退信。</p>
按活动分组的软退信数	emails-softbounced-grouped-by-journey-activity	<p>对于旅程中的每个活动，由于软退信而无法传送到参与者的消息数。如果因临时性问题（例如参与者的收件箱已满或接收服务器暂时不可用）导致邮件无法送达，会造成软退信。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>
取消订阅数	journey-emails-unsubscribed	<p>参与者点击消息中的取消订阅链接的次数。如果一个参与者多次点击相同的取消订阅链接，则每次点击均计数。</p>



指标	Kpi-name	描述
按活动分组的取消订阅数	emails-unsubscribed-grouped-by-journey-activity	<p>对于旅程中的每个活动，参与者点击消息中的取消订链接的次数。如果一个参与者多次点击相同的取消订链接，则每次点击均计数。</p> <p>该指标的查询结果按活动 ID (JourneyActivityId ) 进行分组，该 ID 是唯一地标识活动的字符串。</p>

## 旅程执行指标

下表列出并描述了有关旅程的标准执行指标，您可以查询这些指标以评估 Amazon Pinpoint 旅程中的参与者的状态。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[旅程执行指标](#)资源。该表中的字段列指定在每个指标的查询结果中显示的字段的名称。

指标	字段	描述
积极参与者数	ENDPOINT_ACTIVE	<p>积极推进旅程活动的参与者数。</p> <p>该指标的计算方法是：启动旅程的参与者数，减去离开旅程以及从旅程中删除的参与者数。</p>
参与者取消数	CANCELLED	因旅程被取消而未完成旅程的参与者数量。
参与者离开数	ENDPOINT_LEFT	离开旅程的参与者数。
参与者进入数	ENDPOINT_ENTERED	启动旅程的参与者数。

指标	字段	描述
参与者例外数 ( 超过重新进入限制 )	REENTRY_CAP_EXCEEDED	由于超过了单个参与者可以重新进入旅程的最大次数而未完成旅程的参与者数。
参与者例外数 ( 被拒绝 )	ACTIVE_ENDPOINT_REJECTED	<p>由于已经是旅程的积极参与者而无法启动旅程的参与者数量。</p> <p>如果某个参与者启动了一个旅程，而您随后因故更新了其端点定义，影响到其在某个分段 ( 基于分段标准 ) 或该旅程 ( 基于活动条件 ) 中的包含性，则该参与者被拒绝。</p>

## 旅程活动执行指标

下表列出并描述了有关旅程活动的标准执行指标，您可以查询这些指标以评估 Amazon Pinpoint 旅程的每种类型的单独活动中的参与者的状态。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[旅程活动执行指标](#)资源。该表中的指标列列出在每种类型的活动的查询结果中显示的字段。它还提供了每个字段的简要描述。

活动类型	指标
是/否拆分 (CONDITIONAL_SPLIT )	<p>这些指标是：</p> <ul style="list-style-type: none"> <li>Branch_FALSE – 不符合活动的条件并按“否”路径继续活动的参与者数量。</li> <li>Branch_TRUE – 符合活动的条件并按“是”路径继续活动的参与者数量。</li> </ul> <p>活动中每个路径可以使用其他指标。有关这些指标的信息，请参阅该表中与该类型的活动对应的行。</p>

活动类型	指标
保留 (HOLDOUT)	<p>这些指标是：</p> <ul style="list-style-type: none"><li>• HOLDOUT – 从旅程中删除的作为活动的保留百分比一部分的参与者数。</li><li>• PASSED – 继续进入旅程中下一个活动的参与者数。</li></ul>
电子邮件 (MESSAGE)	<p>这些指标是：</p> <ul style="list-style-type: none"><li>• DAILY_CAP_EXCEEDED – 由于超过了单个参与者在 24 小时内可接收的最多邮件数而未发送的邮件数。</li><li>• FAILURE_PERMANENT – 由于永久性问题而未发送的邮件数。</li><li>• QUIET_TIME – 由于送达时间处于参与者时区的安静时间而未发送的邮件数。</li><li>• SERVICE_FAILURE – 由于 Amazon Pinpoint 存在问题而未发送的邮件数。</li><li>• SUCCESS – 成功送达参与者的邮件数。</li><li>• THROTTLED – 由于将会超过您的 Amazon Pinpoint 账户的发送限额而未发送的邮件数。</li><li>• TRANSIENT_FAILURE – 由于临时问题而未发送的邮件数。</li><li>• UNKNOWN – 由于未知问题而未发送的邮件数。</li></ul>

活动类型	指标
多元拆分 (MULTI_CONDITIONAL_SPLIT )	<p>对于活动的每个路径，在该路径上继续开展活动的参与者数量。</p> <p>该指标的查询结果按路径 Branch_# 分组，其中 # 是路径的数字标识符，例如 Branch_1 表示活动的第一个路径。</p> <p>活动中每个路径可以使用其他指标。有关这些指标的信息，请参阅该表中与该类型的活动对应的行。</p>
随机拆分 (RANDOM_SPLIT )	<p>对于活动的每个路径，在该路径上继续开展活动的参与者数量。</p> <p>该指标的查询结果按路径 Branch_# 分组，其中 # 是路径的数字标识符，例如 Branch_1 表示活动的第一个路径。</p> <p>活动中每个路径可以使用其他指标。有关这些指标的信息，请参阅该表中与该类型的活动对应的行。</p>
等待 (WAIT)	<p>这些指标是：</p> <ul style="list-style-type: none"> <li>• WAIT_FINISHED – 完成等待指定时间的参与者数。</li> <li>• WAIT_SKIPPED – 未等待指定时间的参与者数，原因通常是他们在计划的活动结束后启动了活动或旅程。</li> <li>• WAIT_STARTED – 开始等待并且未跳过或完成等待指定时间的参与者数。</li> </ul>

活动类型	指标
联系中心 (CONTACT_CENTER )	<p>这些指标是：</p> <ul style="list-style-type: none"> <li>• CALL_QUEUED – 已拨入并排队等候 Amazon Connect 服务的参与者数。包括重拨尝试。</li> <li>• CONTINUE_WAITING – 继续等待拨号尝试的参与者数。</li> <li>• DIAL_FAILURE – 拨号尝试失败的参与者数。</li> <li>• DROPPED – 发送时不再符合先前旅程活动中定义的条件条件的参与者数。</li> <li>• TIMEOUT – 多次尝试拨号后未收到 Amazon Connect 处置代码的参与者数。</li> <li>• WAIT_FINISHED – 完成等待指定时间的参与者数。</li> <li>• WAIT_FOR_QUIET_HOURS – 等待安静时间结束以便向渠道投放内容的参与者数。</li> <li>• WAIT_STARTED – 开始等待并且未跳过或完成等待指定时间的参与者数。</li> </ul>

## 旅程和活动执行指标

您可以查询这些标准执行指标以评估 Amazon Pinpoint 旅程或活动的每种类型的单独活动中参与者的状态。要查询这些指标的数据，请使用 Amazon Pinpoint API 的[旅程运行活动执行指标](#)或[活动指标](#)资源。下表列出在每种类型的活动的查询结果中显示的字段。

指标名称	适用于旅程和/或活动	描述
ENDPOINT_PRODUCED	二者	在进行任何筛选之前，最初从分段或事件中生成的端点数量。
ENDPOINTS_FROM_USER	二者	如果客户具有仅用户 ID 分段，则将添加这些用户的所有端

指标名称	适用于旅程和/或活动	描述
		点。该指标衡量以此方式添加的端点的数量。
ENDPOINT_OPT_OUT	二者	端点已选择退出且未进入活动或旅程。
ENDPOINT_INACTIVE	二者	端点处于非活动状态且未进入活动或旅程。
FILTERED_OUT_BY_SEGMENT	二者	端点与分段筛选条件不匹配且未进入活动或旅程。
ENDPOINT_MISSING_ADDRESS	二者	端点缺少地址且未进入活动或旅程。
ENDPOINT_MISSING_CHANNEL	二者	端点缺少渠道且未进入活动或旅程。
ENDPOINT_MISSING_TIMEZONE	二者	端点缺少时区值，因此已被过滤掉。只有在需要时区值时才会发生这种情况。
ENDPOINT_TIMEZONE_MISMATCH	二者	端点所在的时区当时未包含在执行中。
ENDPOINT_CHANNEL_MISMATCH	市场活动	活动没有为此端点的渠道类型配置消息。
DUPLICATE_ENDPOINT	二者	发现了重复的端点并进行了去重。
DUPLICATE_USER	二者	发现了重复的用户并从仅用户 ID 的分段进行了去重。如果重复用户具有相同的用户 ID，则会发出一个 1 的指标。
PAUSED	旅程	由于旅程已暂停，所以已从执行中移除。

指标名称	适用于旅程和/或活动	描述
ENDED	旅程	由于旅程已结束，所以已从执行中移除。
TREATMENT_HOLDOUT	市场活动	这是在 A/B 活动中发出的指标，适用于其群组与当前处理方式不匹配的端点。例如，在 50/50 A/B 拆分中，对于每次处理，50% 的终点节点将发出此指标
ENDPOINT_ESTIMATED_TIMEZONE	旅程	时区估计能够估计端点的时区。

## 查询活动的 Amazon Pinpoint 分析数据

除了使用 Amazon Pinpoint 控制台上的分析页面之外，您还可以使用 Amazon Pinpoint Analytics API 查询一部分标准指标的分析数据，从而了解活动的送达和参与度趋势。

每个指标即是一个可测量的值，也称为关键绩效指标 (KPI)，它们可以帮助您监控和评估一个或多个活动的绩效。例如，您可以使用指标来了解活动消息发送给了多少个端点，或者，其中有多少消息送达预期端点。

Amazon Pinpoint 自动为您的所有活动收集并聚合这些数据。它可以将数据存储在 90 天。如果您已使用 AWS Mobile SDK 将移动应用程序与 Amazon Pinpoint 集成，则 Amazon Pinpoint 会扩展此支持以包括其他指标，例如被收件人打开的推送通知的百分比。有关集成移动应用程序的信息，请参阅[将 Amazon Pinpoint 与您的应用程序集成](#)。

如果使用 Amazon Pinpoint Analytics API 查询数据，则可以选择用于定义查询的范围、数据、分组和筛选器的各种选项。除了要应用的任何基于日期的筛选器之外，还可使用参数指定要查询的项目、活动和指标来执行此操作。

本主题提供了有关如何选择这些选项和查询一个或多个活动的数据的示例，并对这些示例进行了说明。

### 先决条件

在查询一个或多个活动的分析数据之前，收集以下信息很有用，可以用它们来定义查询：

- 项目 ID – 与活动关联的项目的唯一标识符。在 Amazon Pinpoint API 中，此值存储在 `application-id` 属性中。在 Amazon Pinpoint 控制台上，此值在所有项目页面上显示为项目 ID。
- 活动 ID – 活动的唯一标识符（如果仅查询一个活动的数据）。在 Amazon Pinpoint API 中，此值存储在 `campaign-id` 属性中。此值不会显示在控制台上。
- 日期范围（可选）– 设置查询数据的日期范围的起始和截止日期和时间。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始时间必须距离当前日期不到 90 天。如果未指定日期范围，Amazon Pinpoint 将自动查询前 31 个日历日期间的数据。
- 指标类型 – 要查询的指标的类型。有两种类型，即应用程序指标 和 活动指标。应用程序指标 提供与项目（也称为应用程序）关联的所有活动的数据。活动指标 仅提供一个活动的数据。
- 指标 – 要查询的指标的名称，更具体地说，即指标的 `kpi-name` 值。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

它还可帮助确定是否要按相关字段对数据进行分组。如果您要分组，则可以选择一个指标来自动分组数据，从而简化分析和报告。例如，Amazon Pinpoint 提供了多个标准指标来报告已送达活动收件人的邮件的百分比。其中一个指标自动按日期对数据进行分组 (`successful-delivery-rate-grouped-by-date`)。另一个指标自动按活动运行对数据进行分组 (`successful-delivery-rate-grouped-by-campaign-activity`)。还有一个指标仅返回单个值，即，对于所有活动运行，送达收件人的消息的百分比 (`successful-delivery-rate`)。

如果找不到能以您需要的方式来分组数据的标准指标，您可以开发一系列查询来返回您需要的数据。然后，您可以手动细分或者合并查询结果到您设计的自定义分组中。

最后，请务必确认您有权访问要查询的数据。有关更多信息，请参阅[有关查询 Amazon Pinpoint 分析数据的 IAM 策略](#)。

## 查询一个活动的数据

要查询一个活动的数据，您可以使用[活动指标](#) API 并指定以下所需参数的值：

- `application-id` – 项目 ID，它是与活动关联的项目的唯一标识符。在 Amazon Pinpoint 中，项目和应用程序 具有相同的含义。
- `campaign-id` – 市场活动的唯一标识符。
- `kpi-name` – 要查询的指标的名称。此值描述了关联的指标并包含两个或两个以上的术语，这些术语由小写字母数字字符组成并由连字符分隔。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。



您也可以应用筛选器来查询特定日期范围的数据。如果未指定日期范围，则 Amazon Pinpoint 返回前 31 个日历日期间的数据。要按不同的日期筛选数据，请使用支持的日期范围参数指定日期范围的起始和截止日期和时间。这些值应采用扩展的 ISO 8601 格式，并使用协调世界时 (UTC)，例如，2019-07-19T20:00:00Z 表示协调世界时 2019 年 7 月 19 日晚上 8 点。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始日期和时间必须距离当前日期不到 90 天。

以下示例演示了如何使用 Amazon Pinpoint REST API、AWS CLI 和 AWS SDK for Java 查询活动的分析数据。您可以使用任何受支持的 AWS SDK 查询活动的分析数据。AWS CLI 示例的格式适用于 Microsoft Windows。对于 Unix、Linux 和 macOS，请将插入符号 (^) 行继续符替换为反斜杠 (\)。

## REST API

要使用 Amazon Pinpoint REST API 查询活动的分析数据，请向[活动指标](#) URI 发送 HTTP(S) GET 请求。在此 URI 中，为所需的路径参数指定适当的值：

```
https://endpoint/v1/apps/application-id/campaigns/campaign-id/kpis/daterange/kpi-name
```

其中：

- *endpoint* 是托管与活动关联的项目的 AWS 区域的 Amazon Pinpoint 端点。
- *application-id* 是与活动关联的项目的唯一标识符。
- *campaign-id* 是活动的唯一标识符。
- *kpi-name* 是要查询的指标的 kpi-name 值。

所有参数都应是 URL 编码的。

要应用一个筛选器来查询特定日期范围的数据，请将 start-time 和 end-time 查询参数和值附加到 URI。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。使用 & 符号分隔参数。

例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个活动的所有运行，消息送达的唯一端点的数量：

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/1234567890123456789012345example/campaigns/80b8efd84042ff8d9c96ce2f8example/kpis/daterange/unique-deliveries?start-time=2019-07-19T00:00:00Z&end-time=2019-07-26T23:59:59Z
```

其中：

- `pinpoint.us-east-1.amazonaws.com` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `1234567890123456789012345example` 是与活动关联的项目的唯一标识符。
- `80b8efd84042ff8d9c96ce2f8example` 是活动的唯一标识符。
- `unique-deliveries` 是 `endpoint deliveries` 活动指标的 `kpi-name` 值，该指标用于报告对于某个活动的所有运行，消息送达到的唯一端点数量。
- `2019-07-19T00:00:00Z` 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- `2019-07-26T23:59:59Z` 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## AWS CLI

要使用 AWS CLI 查询活动的分析数据，请使用 `get-campaign-date-range-kpi` 命令并为所需参数指定适当的值：

```
C:\> aws pinpoint get-campaign-date-range-kpi ^
--application-id application-id ^
--campaign-id campaign-id ^
--kpi-name kpi-name
```

其中：

- *application-id* 是与活动关联的项目的唯一标识符。
- *campaign-id* 是活动的唯一标识符。
- *kpi-name* 是要查询的指标的 `kpi-name` 值。

要应用一个筛选器来查询特定日期范围的数据，请将 `start-time` 和 `end-time` 参数和值添加到您的查询。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个活动的所有运行，消息送达的唯一端点的数量：

```
C:\> aws pinpoint get-campaign-date-range-kpi ^
--application-id 1234567890123456789012345example ^
--campaign-id 80b8efd84042ff8d9c96ce2f8example ^
--kpi-name unique-deliveries ^
--start-time 2019-07-19T00:00:00Z ^
--end-time 2019-07-26T23:59:59Z
```

其中：

- 1234567890123456789012345example 是与活动关联的项目的唯一标识符。
- 80b8efd84042ff8d9c96ce2f8example 是活动的唯一标识符。
- unique-deliveries 是 endpoint deliveries 活动指标的 kpi-name 值，该指标用于报告对于某个活动的所有运行，消息送达到的唯一端点数量。
- 2019-07-19T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-07-26T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## SDK for Java

要使用 AWS SDK for Java 查询活动的分析数据，请使用[活动指标](#) API 的 `GetCampaignDateRangeKpiRequest` 方法。为所需的参数指定相应的值：

```
GetCampaignDateRangeKpiRequest request = new GetCampaignDateRangeKpiRequest()
    .withApplicationId("applicationId")
    .withCampaignId("campaignId")
    .withKpiName("kpiName")
```

其中：

- *applicationId* 是与活动关联的项目的唯一标识符。
- *campaignId* 是活动的唯一标识符。
- *kpiName* 是要查询的指标的 kpi-name 值。

要应用一个筛选器来查询特定日期范围的数据，请将 `startTime` 和 `endTime` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个活动的所有运行，消息送达的唯一端点的数量：

```
GetCampaignDateRangeKpiRequest request = new GetCampaignDateRangeKpiRequest()
    .withApplicationId("1234567890123456789012345example")
    .withCampaignId("80b8efd84042ff8d9c96ce2f8example")
    .withKpiName("unique-deliveries")
    .withStartTime(Date.from(Instant.parse("2019-07-19T00:00:00Z")))
    .withEndTime(Date.from(Instant.parse("2019-07-26T23:59:59Z")));
```

其中：

- 1234567890123456789012345example 是与活动关联的项目的唯一标识符。
- 80b8efd84042ff8d9c96ce2f8example 是活动的唯一标识符。
- unique-deliveries 是 endpoint deliveries 活动指标的 kpi-name 值，该指标用于报告对于某个活动的所有运行，消息送达到唯一端点数量。
- 2019-07-19T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-07-26T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

发送查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。结果的结构因您查询的指标而异。一些指标仅返回一个值。例如，上述示例中使用的 endpoint deliveries (unique-deliveries) 活动指标只返回一个值，即，对于某个活动的所有运行，消息送达到唯一端点数。在这种情况下，JSON 响应如下所示：

```
{
  "CampaignDateRangeKpiResponse":{
    "ApplicationId":"1234567890123456789012345example",
    "CampaignId":"80b8efd84042ff8d9c96ce2f8example",
    "EndTime":"2019-07-26T23:59:59Z",
    "KpiName":"unique-deliveries",
    "KpiResult":{
      "Rows":[
        {
          "Values":[
            {
              "Key":"UniqueDeliveries",
              "Type":"Double",
              "Value":"123.0"
            }
          ]
        }
      ]
    },
    "StartTime":"2019-07-19T00:00:00Z"
  }
}
```

另一些指标返回多个值，并按相关字段对这些值进行分组。如果指标返回多个值，则 JSON 响应将包含一个字段，该字段指示对数据进行分组时所用的字段。

要了解有关查询结果结构的更多信息，请参阅[使用查询结果](#)。

## 查询多个活动的数据

可通过两种方式查询多个活动的数据。哪种方式最佳，这取决于您是否要查询均与同一项目关联的活动的的数据。如果您要查询，则最佳方式还取决于您是要查询所有这些活动的的数据，还是仅查询一部分活动的的数据。

要查询与不同项目关联的活动的的数据，或仅查询与同一项目关联的部分活动的的数据，最佳方式是创建并运行一系列单独的查询，并且一个查询对应一个活动。上一部分说明了如何仅查询一个活动的的数据。

要查询与同一项目关联的所有活动的的数据，可以使用[应用程序指标](#) API。为以下所需的参数指定值：

- `application-id` – 项目 ID，它是项目的唯一标识符。在 Amazon Pinpoint 中，项目和应用程序具有相同的含义。
- `kpi-name` – 要查询的指标的名称。此值描述了关联的指标并包含两个或两个以上的术语，这些术语由小写字母数字字符组成并由连字符分隔。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

您也可以按日期范围筛选数据。如果未指定日期范围，则 Amazon Pinpoint 返回前 31 个日历日期间的的数据。要按不同的日期筛选数据，请使用支持的日期范围参数指定日期范围的起始和截止日期和时间。这些值应采用扩展的 ISO 8601 格式，并使用协调世界时 (UTC)，例如，`2019-07-19T20:00:00Z` 表示协调世界时 2019 年 7 月 19 日晚上 8 点。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始日期和时间必须距离当前日期不到 90 天。

以下示例演示了如何使用 Amazon Pinpoint REST API、AWS CLI 和 AWS SDK for Java 查询活动的分析数据。您可以使用任何受支持的 AWS SDK 查询活动的分析数据。AWS CLI 示例的格式适用于 Microsoft Windows。对于 Unix、Linux 和 macOS，请将插入符号 (^) 行继续符替换为反斜杠 (\)。

### REST API

要使用 Amazon Pinpoint REST API 查询多个活动的分析数据，请向[应用程序指标](#) URI 发送 HTTP(S) GET 请求。在此 URI 中，为所需的路径参数指定适当的值：

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

其中：

- `endpoint` 是托管与活动关联的的项目的 AWS 区域的 Amazon Pinpoint 端点。
- `application-id` 是与活动关联的的项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

所有参数都应是 URL 编码的。

要应用一个筛选器来检索特定日期范围的数据，请将 `start-time` 和 `end-time` 查询参数和值附加到 URI。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。使用 `&` 符号分隔参数。

例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个项目的每个活动，消息送达到唯一端点的数量：

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/1234567890123456789012345example/kpis/daterange/unique-deliveries-grouped-by-campaign?start-time=2019-07-19T00:00:00Z&end-time=2019-07-26T23:59:59Z
```

其中：

- `pinpoint.us-east-1.amazonaws.com` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `1234567890123456789012345example` 是与活动关联的项目的唯一标识符。
- `unique-deliveries-grouped-by-campaign` 是 endpoint deliveries, grouped by campaign 应用程序指标的 `kpi-name` 值，该指标按照每个活动返回消息送达到唯一端点的数量。
- `2019-07-19T00:00:00Z` 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- `2019-07-26T23:59:59Z` 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## AWS CLI

要使用 AWS CLI 查询多个活动的分析数据，请使用 `get-application-date-range-kpi` 命令并为所需参数指定适当的值：

```
C:\> aws pinpoint get-application-date-range-kpi ^  
  --application-id application-id ^  
  --kpi-name kpi-name
```

其中：

- `application-id` 是与活动关联的项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

要应用一个筛选器来检索特定日期范围的数据，请将 `start-time` 和 `end-time` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范

围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个项目的每个活动，消息送达到的唯一端点的数量：

```
C:\> aws pinpoint get-application-date-range-kpi ^
--application-id 1234567890123456789012345example ^
--kpi-name unique-deliveries-grouped-by-campaign ^
--start-time 2019-07-19T00:00:00Z ^
--end-time 2019-07-26T23:59:59Z
```

其中：

- 1234567890123456789012345example 是与活动关联的项目的唯一标识符。
- unique-deliveries-grouped-by-campaign 是 endpoint deliveries, grouped by campaign 应用程序指标的 kpi-name 值，该指标按照每个活动返回消息送达到的唯一端点的数量。
- 2019-07-19T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-07-26T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## SDK for Java

要使用 AWS SDK for Java 查询多个活动的分析数据，请使用[应用程序指标](#) API 的 `GetApplicationDateRangeKpiRequest` 方法。为所需的参数指定相应的值：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("applicationId")
    .withKpiName("kpiName")
```

其中：

- *applicationId* 是与活动关联的项目的唯一标识符。
- *kpiName* 是要查询的指标的 kpi-name 值。

要应用一个筛选器来检索特定日期范围的数据，请将 `startTime` 和 `endTime` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 7 月 19 日至 2019 年 7 月 26 日期间，对于某个项目的每个活动，消息送达到的唯一端点的数量：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("1234567890123456789012345example")
```

```
.withKpiName("unique-deliveries-grouped-by-campaign")
.withStartTime(Date.from(Instant.parse("2019-07-19T00:00:00Z")))
.withEndTime(Date.from(Instant.parse("2019-07-26T23:59:59Z")));
```

其中：

- 1234567890123456789012345example 是与活动关联的项目的唯一标识符。
- unique-deliveries-grouped-by-campaign 是 endpoint deliveries, grouped by campaign 应用程序指标的 kpi-name 值，该指标按照每个活动返回消息送达到的唯一端点的数量。
- 2019-07-19T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-07-26T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

发送查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。结果的结构因您查询的指标而异。一些指标仅返回一个值。另一些指标返回多个值，并按相关字段对这些值进行分组。如果指标返回多个值，则 JSON 响应将包含一个字段，该字段指示对数据进行分组时所用的字段。

例如，上述示例中使用的 endpoint deliveries, grouped by campaign (unique-deliveries-grouped-by-campaign) 应用程序指标返回多个值，即，对于某个项目的每个关联活动，消息送达到的唯一端点的数量。在这种情况下，JSON 响应如下所示：

```
{
  "ApplicationDateRangeKpiResponse":{
    "ApplicationId":"1234567890123456789012345example",
    "EndTime":"2019-07-26T23:59:59Z",
    "KpiName":"unique-deliveries-grouped-by-campaign",
    "KpiResult":{
      "Rows":[
        {
          "GroupedBy":[
            {
              "Key":"CampaignId",
              "Type":"String",
              "Value":"80b8efd84042ff8d9c96ce2f8example"
            }
          ],
          "Values":[
            {
              "Key":"UniqueDeliveries",
              "Type":"Double",
              "Value":"123.0"
            }
          ]
        }
      ]
    }
  }
}
```



```
    }
  ]
},
{
  "GroupedBy": [
    {
      "Key": "CampaignId",
      "Type": "String",
      "Value": "810c7aab86d42fb2b56c8c966example"
    }
  ],
  "Values": [
    {
      "Key": "UniqueDeliveries",
      "Type": "Double",
      "Value": "456.0"
    }
  ]
},
{
  "GroupedBy": [
    {
      "Key": "CampaignId",
      "Type": "String",
      "Value": "42d8c7eb0990a57ba1d5476a3example"
    }
  ],
  "Values": [
    {
      "Key": "UniqueDeliveries",
      "Type": "Double",
      "Value": "789.0"
    }
  ]
}
],
"StartTime": "2019-07-19T00:00:00Z"
}
}
```

在此情况下，GroupedBy 字段指示按活动 ID 对值进行分组 (CampaignId)。

要了解有关查询结果结构的更多信息，请参阅[使用查询结果](#)。

## 查询事务性消息的 Amazon Pinpoint 分析数据

除了使用 Amazon Pinpoint 控制台上的分析页面之外，您还可以使用 Amazon Pinpoint Analytics API 查询一部分标准指标的分析数据，从而了解为项目发送的事务性消息的送达和参与趋势。

其中的每个指标是一个可测量的值，也称为关键绩效指标 (KPI)，它们可以帮助您监控和评估事务性消息的绩效。例如，您可以使用指标来了解您发送了多少事务性电子邮件消息或短信，或者其中有多少消息送达收件人。Amazon Pinpoint 会自动收集和汇总您为项目发送的所有事务性电子邮件消息和短信的这类数据。它可以将数据存储 90 天。

如果使用 Amazon Pinpoint Analytics API 查询数据，则可以选择用于定义查询的范围、数据、分组和筛选器的各种选项。为此，除了应用任何基于日期的筛选器之外，您还应使用参数来指定要查询的项目和指标。

本主题提供了有关如何选择这些选项和查询项目的事务性消息数据的示例，并对这些示例进行了说明。

### 先决条件

在查询事务性消息的分析数据之前，收集以下信息很有用，可以用它们来定义查询：

- 项目 ID – 从中发送消息的项目的唯一标识符。在 Amazon Pinpoint API 中，此值存储在 `application-id` 属性中。在 Amazon Pinpoint 控制台上，此值在所有项目页面上显示为项目 ID。
- 日期范围 (可选) – 设置查询数据的日期范围的起始和截止日期和时间。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始时间必须距离当前日期不到 90 天。如果未指定日期范围，Amazon Pinpoint 将自动查询前 31 个日历日期间的数据。
- 指标 – 要查询的指标的名称，更具体地说，即指标的 `kpi-name` 值。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

它还可帮助确定是否要按相关字段对数据进行分组。如果您要分组，则可以选择一个指标来自动分组数据，从而简化分析和报告。例如，Amazon Pinpoint 提供了多个标准指标来报告送达收件人的事务性短信的数量。其中一个指标自动按日期对数据进行分组 (`txn-sms-delivered-grouped-by-date`)。另一个指标自动按国家或地区对数据进行分组 (`txn-sms-delivered-grouped-by-country`)。还有一个指标仅返回单个值—送达收件人的消息数 (`txn-sms-delivered`)。如果找不到能以您需要的方式来分组数据的标准指标，您可以开发一系列查询来返回您需要的数据。然后，您可以手动细分或者合并查询结果到您设计的自定义分组中。

最后，请务必确认您有权访问要查询的数据。有关更多信息，请参阅[有关查询 Amazon Pinpoint 分析数据的 IAM 策略](#)。

## 查询事务性电子邮件消息的数据

要查询为项目发送的事务性电子邮件消息的数据，请使用[应用程序指标](#) API 并指定以下所需参数的值：

- `application-id` – 项目 ID，它是项目的唯一标识符。在 Amazon Pinpoint 中，项目和应用程序具有相同的含义。
- `kpi-name` – 要查询的指标的名称。此值描述了关联的指标并包含两个或两个以上的术语，这些术语由小写字母数字字符组成并由连字符分隔。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

您也可以应用筛选器来查询特定日期范围的数据。如果未指定日期范围，则 Amazon Pinpoint 返回前 31 个日历日期间的数据。要按不同的日期筛选数据，请使用支持的日期范围参数指定日期范围的起始和截止日期和时间。这些值应采用扩展的 ISO 8601 格式，并使用协调世界时 (UTC)，例如，`2019-09-06T20:00:00Z` 表示协调世界时 2019 年 9 月 6 日晚上 8 点。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始日期和时间必须距离当前日期不到 90 天。

以下示例演示了如何使用 Amazon Pinpoint REST API、AWS CLI 和 AWS SDK for Java 查询事务性电子邮件消息的分析数据。您可以使用任何受支持的 AWS SDK 查询事务性消息的分析数据。AWS CLI 示例的格式适用于 Microsoft Windows。对于 Unix、Linux 和 macOS，请将插入符号 (^) 行继续符替换为反斜杠 (\)。

### REST API

要使用 Amazon Pinpoint REST API 查询事务性电子邮件消息的分析数据，请向[应用程序指标](#) URI 发送 HTTP(S) GET 请求。在此 URI 中，为所需的路径参数指定适当的值：

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

其中：

- `endpoint` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `application-id` 是项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

所有参数都应是 URL 编码的。

要应用一个筛选器来查询特定日期范围的数据，请将 `start-time` 和 `end-time` 查询参数和值附加到 URI。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。使用 `&` 符号分隔参数。

例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 13 日期间为项目发送的事务性电子邮件消息的数量：

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/1234567890123456789012345example/kpis/daterange/txn-emails-sent?start-time=2019-09-06T00:00:00Z&end-time=2019-09-13T23:59:59Z
```

其中：

- `pinpoint.us-east-1.amazonaws.com` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `1234567890123456789012345example` 是项目的唯一标识符。
- `txn-emails-sent` 是发送数应用程序指标的 `kpi-name` 值，该指标用于报告为项目发送的事务性电子邮件消息的数量。
- `2019-09-06T00:00:00Z` 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- `2019-09-13T23:59:59Z` 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## AWS CLI

要使用 AWS CLI 查询事务性电子邮件消息的分析数据，请使用 `get-application-date-range-kpi` 命令并为所需的参数指定相应的值：

```
C:\> aws pinpoint get-application-date-range-kpi ^  
  --application-id application-id ^  
  --kpi-name kpi-name
```

其中：

- `application-id` 是项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

要应用一个筛选器来查询特定日期范围的数据，请将 `start-time` 和 `end-time` 参数和值添加到您的查询。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围

的起始和截止日期和时间。例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 13 日期间为项目发送的事务性电子邮件消息的数量：

```
C:\> aws pinpoint get-application-date-range-kpi ^
--application-id 1234567890123456789012345example ^
--kpi-name txn-emails-sent ^
--start-time 2019-09-06T00:00:00Z ^
--end-time 2019-09-13T23:59:59Z
```

其中：

- 1234567890123456789012345example 是项目的唯一标识符。
- txn-emails-sent 是发送数应用程序指标的 kpi-name 值，该指标用于报告为项目发送的事务性电子邮件消息的数量。
- 2019-09-06T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-09-13T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## SDK for Java

要使用 AWS SDK for Java 查询事务性电子邮件消息的分析数据，请使用[应用程序指标](#) API 的 `GetApplicationDateRangeKpiRequest` 方法。为所需的参数指定相应的值：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("applicationId")
    .withKpiName("kpiName")
```

其中：

- *applicationId* 是项目的唯一标识符。
- *kpiName* 是要查询的指标的 kpi-name 值。

要应用一个筛选器来查询特定日期范围的数据，请将 `startTime` 和 `endTime` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 13 日期间为项目发送的事务性电子邮件消息的数量：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("1234567890123456789012345example")
```

```
.withKpiName("txn-emails-sent")
.withStartTime(Date.from(Instant.parse("2019-09-06T00:00:00Z")))
.withEndTime(Date.from(Instant.parse("2019-09-13T23:59:59Z")));
```

其中：

- 1234567890123456789012345example 是项目的唯一标识符。
- txn-emails-sent 是发送数应用程序指标的 kpi-name 值，该指标用于报告为项目发送的事务性电子邮件消息的数量。
- 2019-09-06T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-09-13T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

发送查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。结果的结构因您查询的指标而异。一些指标仅返回一个值。例如，上述示例中使用的发送数 (txn-emails-sent) 应用程序指标返回一个值，即从项目发送的事务性电子邮件消息的数量。在这种情况下，JSON 响应如下所示：

```
{
  "ApplicationDateRangeKpiResponse":{
    "ApplicationId":"1234567890123456789012345example",
    "EndTime":"2019-09-13T23:59:59Z",
    "KpiName":"txn-emails-sent",
    "KpiResult":{
      "Rows":[
        {
          "Values":[
            {
              "Key":"TxnEmailsSent",
              "Type":"Double",
              "Value":"62.0"
            }
          ]
        }
      ]
    },
    "StartTime":"2019-09-06T00:00:00Z"
  }
}
```

另一些指标返回多个值，并按相关字段对这些值进行分组。如果指标返回多个值，则 JSON 响应将包含一个字段，该字段指示对数据进行分组时所用的字段。

要了解有关查询结果结构的更多信息，请参阅[使用查询结果](#)。

## 查询事务性短信的数据

要查询已为项目发送的事务性短信的数据，请使用[应用程序指标](#) API 并指定以下所需参数的值：

- `application-id` – 项目 ID，它是项目的唯一标识符。在 Amazon Pinpoint 中，项目和应用程序具有相同的含义。
- `kpi-name` – 要查询的指标的名称。此值描述了关联的指标并包含两个或两个以上的术语，这些术语由小写字母数字字符组成并由连字符分隔。有关受支持的指标及其 `kpi-name` 值的完整列表，请参阅[标准指标](#)。

您也可以应用筛选器来查询特定日期范围的数据。如果未指定日期范围，则 Amazon Pinpoint 返回前 31 个日历日期间的数据。要按不同的日期筛选数据，请使用支持的日期范围参数以指定日期范围的起始和截止日期和时间。这些值应采用扩展的 ISO 8601 格式，并使用协调世界时 (UTC)，例如，`2019-09-06T20:00:00Z` 表示协调世界时 2019 年 9 月 6 日晚上 8 点。日期范围是包含性的，必须限制为不超过 31 个日历天。此外，起始日期和时间必须距离当前日期不到 90 天。

以下示例演示了如何使用 Amazon Pinpoint REST API、AWS CLI 和 AWS SDK for Java 查询事务性短信的分析数据。您可以使用任何受支持的 AWS SDK 查询事务性消息的分析数据。AWS CLI 示例的格式适用于 Microsoft Windows。对于 Unix、Linux 和 macOS，请将插入符号 (^) 行继续符替换为反斜杠 (\)。

### REST API

要使用 Amazon Pinpoint REST API 查询事务性短信的分析数据，请向[应用程序指标](#) URI 发送 HTTP(S) GET 请求。在此 URI 中，为所需的路径参数指定适当的值：

```
https://endpoint/v1/apps/application-id/kpis/daterange/kpi-name
```

其中：

- `endpoint` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `application-id` 是项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

所有参数都应是 URL 编码的。

要应用一个筛选器来检索特定日期范围的数据，请将 `start-time` 和 `end-time` 查询参数和值附加到 URI。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。使用 `&` 符号分隔参数。

例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 8 日期间的每一天发送的事务性短信的数量：

```
https://pinpoint.us-east-1.amazonaws.com/v1/apps/1234567890123456789012345example/kpis/daterange/txn-sms-sent-grouped-by-date?start-time=2019-09-06T00:00:00Z&end-time=2019-09-08T23:59:59Z
```

其中：

- `pinpoint.us-east-1.amazonaws.com` 是托管项目的 AWS 区域的 Amazon Pinpoint 端点。
- `1234567890123456789012345example` 是项目的唯一标识符。
- `txn-sms-sent-grouped-by-date` 是按日期分组的发送数 应用程序指标的 `kpi-name` 值，该指标返回日期范围内每天发送的事务性短信的数量。
- `2019-09-06T00:00:00Z` 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- `2019-09-08T23:59:59Z` 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## AWS CLI

要使用 AWS CLI 查询事务性短信的分析数据，请使用 `get-application-date-range-kpi` 命令并为所需的参数指定相应的值：

```
C:\> aws pinpoint get-application-date-range-kpi ^  
  --application-id application-id ^  
  --kpi-name kpi-name
```

其中：

- `application-id` 是项目的唯一标识符。
- `kpi-name` 是要查询的指标的 `kpi-name` 值。

要应用一个筛选器来检索特定日期范围的数据，请将 `start-time` 和 `end-time` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 8 日期间的每一天发送的事务性短信的数量：



```
C:\> aws pinpoint get-application-date-range-kpi ^
--application-id 1234567890123456789012345example ^
--kpi-name txn-sms-sent-grouped-by-date ^
--start-time 2019-09-06T00:00:00Z ^
--end-time 2019-09-08T23:59:59Z
```

其中：

- 1234567890123456789012345example 是项目的唯一标识符。
- txn-sms-sent-grouped-by-date 是按日期分组的发送数 应用程序指标的 kpi-name 值，该指标返回日期范围内每天发送的事务性短信的数量。
- 2019-09-06T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-09-08T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

## SDK for Java

要使用 AWS SDK for Java 查询事务性短信的分析数据，请使用[应用程序指标](#) API 的 `GetApplicationDateRangeKpiRequest` 方法，并为所需的参数指定相应的值：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("applicationId")
    .withKpiName("kpiName")
```

其中：

- *applicationId* 是项目的唯一标识符。
- *kpiName* 是要查询的指标的 kpi-name 值。

要应用一个筛选器来检索特定日期范围的数据，请将 `startTime` 和 `endTime` 参数和值包含在您的查询中。通过使用这些参数，您可采用扩展的 ISO 8601 格式，指定检索数据的包含性日期范围的起始和截止日期和时间。例如，以下请求会检索在 2019 年 9 月 6 日到 2019 年 9 月 8 日期间的每一天发送的事务性短信的数量：

```
GetApplicationDateRangeKpiRequest request = new GetApplicationDateRangeKpiRequest()
    .withApplicationId("1234567890123456789012345example")
    .withKpiName("txn-sms-sent-grouped-by-date")
    .withStartTime(Date.from(Instant.parse("2019-09-06T00:00:00Z")))
```

```
.withEndTime(Date.from(Instant.parse("2019-09-08T23:59:59Z"))));
```

其中：

- 1234567890123456789012345example 是项目的唯一标识符。
- txn-sms-sent-grouped-by-date 是按日期分组的发送数 应用程序指标的 kpi-name 值，该指标返回日期范围内每天发送的事务性短信的数量。
- 2019-09-06T00:00:00Z 是检索数据的起始日期和时间，是包含性日期范围的一部分。
- 2019-09-08T23:59:59Z 是检索数据的截止日期和时间，也是包含性日期范围的一部分。

发送查询后，Amazon Pinpoint 在 JSON 响应中返回查询结果。结果的结构因您查询的指标而异。一些指标仅返回一个值。另一些指标返回多个值，并按相关字段对这些值进行分组。如果指标返回多个值，则 JSON 响应将包含一个字段，该字段指示对数据进行分组时所用的字段。

例如，上述示例中使用的按日期分组的发送数 (txn-sms-sent-grouped-by-date) 应用程序指标将返回多个值，即指定日期范围内的每一天发送的事务性短信的数量。在这种情况下，JSON 响应如下所示：

```
{
  "ApplicationDateRangeKpiResponse":{
    "ApplicationId":"1234567890123456789012345example",
    "EndTime":"2019-09-08T23:59:59Z",
    "KpiName":"txn-sms-sent-grouped-by-date",
    "KpiResult":{
      "Rows":[
        {
          "GroupedBy":[
            {
              "Key":"Date",
              "Type":"String",
              "Value":"2019-09-06"
            }
          ],
          "Values":[
            {
              "Key":"TxnSmsSent",
              "Type":"Double",
              "Value":"29.0"
            }
          ]
        }
      ]
    }
  }
}
```

```
    },
    {
      "GroupedBy": [
        {
          "Key": "Date",
          "Type": "String",
          "Value": "2019-09-07"
        }
      ],
      "Values": [
        {
          "Key": "TxnSmsSent",
          "Type": "Double",
          "Value": "35.0"
        }
      ]
    },
    {
      "GroupedBy": [
        {
          "Key": "Date",
          "Type": "String",
          "Value": "2019-09-08"
        }
      ],
      "Values": [
        {
          "Key": "TxnSmsSent",
          "Type": "Double",
          "Value": "10.0"
        }
      ]
    }
  ],
  "StartTime": "2019-09-06T00:00:00Z"
}
```

在此情况下，GroupedBy 字段指示按日历日对值进行分组 (Date)。这意味着：

- 在 2019 年 9 月 6 日发送了 29 条消息。
- 在 2019 年 9 月 7 日发送了 35 条消息。

- 在 2019 年 9 月 8 日发送了 10 条消息。

要了解有关查询结果结构的更多信息，请参阅[使用查询结果](#)。

## 使用 Amazon Pinpoint 分析查询结果

在使用 Amazon Pinpoint Analytics API 查询分析数据时，Amazon Pinpoint 会在 JSON 响应中返回结果。对于应用程序指标、活动指标和旅程参与指标，响应中的数据采用标准 JSON 架构来报告 Amazon Pinpoint 分析数据。

这意味着，您可以使用所选的编程语言或工具实施自定义解决方案，以查询一个或多个指标的数据，捕获每个查询的结果，然后将结果写入到表、对象或其他位置。随后，您可以使用其他服务或应用程序在该位置处理查询结果。

例如，您可以：

- 使用首选的数据可视化框架构建一个自定义控制面板来定期查询一组指标并显示结果。
- 通过查询适当的指标并在图表或您设计的其他类型的报告中显示结果来创建跟踪参与率的报告。
- 解析分析数据并将其写入特定的存储格式，然后将结果移植到长期存储解决方案。

请注意，Amazon Pinpoint Analytics API 不是用来创建或存储任何您随后可在 Amazon Pinpoint 项目或 Amazon Pinpoint 账户中读取或使用的持久对象的。相反，它们旨在帮助您检索分析数据并将这些数据传输到其他服务和应用程序以进行进一步的分析、存储或报告。为此，对于可通过编程方式为应用程序指标、活动指标和旅程参与指标查询的所有分析数据，它们会使用相同的 JSON 响应结构和架构。

本主题介绍了应用程序指标、活动指标或旅程参与指标查询的 JSON 响应中的结构、对象和字段。有关旅程执行指标或旅程活动执行指标查询的 JSON 响应中的字段的信息，请参阅[Amazon Pinpoint 标准分析指标](#)。

### JSON 结构

为了帮助您解析和使用查询结果，Amazon Pinpoint Analytics API 对可通过编程方式为应用程序指标、活动指标和旅程参与指标查询的所有 Amazon Pinpoint 分析数据使用相同的 JSON 响应结构。每个 JSON 响应指定定义查询的值，例如项目 ID (ApplicationId)。响应还包括一个 (并且只有一个) KpiResult 对象。KpiResult 对象包含查询的整个结果集。

每个 KpiResult 对象包含一个 Rows 对象。这是一个对象数组，其中包含查询结果以及有关这些结果中的值的相关元数据。Rows 对象的结构和内容具有以下一般特性：

- 每行查询结果均为 Rows 对象中的一个名为 Values 的单独的 JSON 对象。例如，如果查询返回三个值，则 Rows 对象包含三个 Values 对象。每个 Values 对象包含单独的查询结果。
- 每列查询结果均为其应用于的 Values 对象的属性。列的名称存储在 Values 对象的 Key 字段中。
- 对于分组的查询结果，每个 Values 对象均有一个关联的 GroupedBy 对象。GroupedBy 对象指示使用哪个字段对结果进行分组。它还提供了关联的 Values 对象的分组值。
- 如果指标的查询结果为 null，则 Rows 对象为空。

除了这些一般特性之外，Rows 对象的结构和内容因指标而异。这是因为 Amazon Pinpoint 支持两种指标，即单值指标和多值指标。

单值指标 仅提供一个累积值。例如，对于某个活动的所有运行，送达收件人的消息的百分比。多值指标 提供多个值，并按相关字段对这些值进行分组。例如，按活动运行分组，对于活动的每个运行，送达收件人的消息的百分比。

可从指标名称快速判断出指标是单值指标还是多值指标。如果指标名称不包含 grouped-by，则是单值指标。如果包含，则是多值指标。有关可通过编程方式查询的指标的完整列表，请参阅[Amazon Pinpoint 标准分析指标](#)。

## 单值指标

对于单值指标，Rows 对象包含一个 Values 对象，用于：

- 指定已查询的指标的友好名称。
- 提供已查询的指标的值。
- 标识已返回的值的数据类型。

例如，以下 JSON 响应包含一个单值指标的查询结果。该指标报告从 2019 年 8 月 1 日到 2019 年 8 月 31 日，对于与某个项目关联的所有活动，消息送达到唯一端点数量：

```
{
  "ApplicationDateRangeKpiResponse": {
    "ApplicationId": "1234567890123456789012345example",
    "EndTime": "2019-08-31T23:59:59Z",
    "KpiName": "unique-deliveries",
    "KpiResult": {
```

```
    "Rows":[
      {
        "Values":[
          {
            "Key":"UniqueDeliveries",
            "Type":"Double",
            "Value":"1368.0"
          }
        ]
      }
    ],
    "StartTime":"2019-08-01T00:00:00Z"
  }
}
```

在该示例中，响应显示从 2019 年 8 月 1 日到 2019 年 8 月 31 日，该项目的所有活动向 1,368 个唯一端点送达了消息，其中：

- Key 是其值在 Value 字段中指定的指标的友好名称（此处为 UniqueDeliveries）。
- Type 是在 Value 字段中指定的值的数据类型（此处为 Double）。
- Value 是所查询指标的实际值，包括应用的任何筛选器（此处为 1368.0）。

如果单值指标的查询结果为 null（不大于或等于零），则 Rows 对象为空。如果某个指标没有任何数据可返回，则 Amazon Pinpoint 对于该指标返回 null 值。例如：

```
{
  "ApplicationDateRangeKpiResponse":{
    "ApplicationId":"2345678901234567890123456example",
    "EndTime":"2019-08-31T23:59:59Z",
    "KpiName":"unique-deliveries",
    "KpiResult":{
      "Rows":[

      ]
    },
    "StartTime":"2019-08-01T00:00:00Z"
  }
}
```

## 多值指标

多值指标的 Rows 对象的结构和内容与单值指标的大致相同。多值指标的 Rows 对象还包含一个 Values 对象。Values 对象指定查询的指标的友好名称，提供该指标的值，并指定该值的数据类型。

不过，多值指标的 Rows 对象还包含一个或多个 GroupedBy 对象。在查询结果中，每个 Values 对象均有一个对应的 GroupedBy 对象。GroupedBy 对象指示使用哪个字段对结果中的数据进行分组以及该字段的数据类型。它还指示该字段的分组值（对于关联的 Values 对象）。

例如，以下 JSON 响应包含一个多值指标的查询结果，该指标报告从 2019 年 8 月 1 日至 2019 年 8 月 31 日，对于与某个项目关联的每个活动，消息送达的唯一端点的数量：

```
{
  "ApplicationDateRangeKpiResponse":{
    "ApplicationId":"1234567890123456789012345example",
    "EndTime":"2019-08-31T23:59:59Z",
    "KpiName":"unique-deliveries-grouped-by-campaign",
    "KpiResult":{
      "Rows":[
        {
          "GroupedBy":[
            {
              "Key":"CampaignId",
              "Type":"String",
              "Value":"80b8efd84042ff8d9c96ce2f8example"
            }
          ],
          "Values":[
            {
              "Key":"UniqueDeliveries",
              "Type":"Double",
              "Value":"123.0"
            }
          ]
        },
        {
          "GroupedBy":[
            {
              "Key":"CampaignId",
              "Type":"String",
              "Value":"810c7aab86d42fb2b56c8c966example"
            }
          ],

```

```
        "Values":[
            {
                "Key":"UniqueDeliveries",
                "Type":"Double",
                "Value":"456.0"
            }
        ],
    },
    {
        "GroupedBy":[
            {
                "Key":"CampaignId",
                "Type":"String",
                "Value":"42d8c7eb0990a57ba1d5476a3example"
            }
        ],
        "Values":[
            {
                "Key":"UniqueDeliveries",
                "Type":"Double",
                "Value":"789.0"
            }
        ]
    }
],
"StartTime":"2019-08-01T00:00:00Z"
}
}
```

在此示例中，响应显示从 2019 年 8 月 1 日至 2019 年 8 月 31 日，有三个项目活动向唯一端点送达了消息。对于其中的每个活动，送达计数具体为：

- 活动 80b8efd84042ff8d9c96ce2f8example 将消息送达 123 个唯一端点。
- 活动 810c7aab86d42fb2b56c8c966example 将消息送达 456 个唯一端点。
- 活动 42d8c7eb0990a57ba1d5476a3example 将消息送达 789 个唯一端点。

其中，对象和字段的一般结构为：

- GroupedBy.Key – 存储在 GroupedBy.Value 字段中指定的分组值的属性或字段的名称（此处为 CampaignId）。



- `GroupedBy.Type` – 在 `GroupedBy.Value` 字段中指定的值的数据类型 ( 此处为 `String` )。
- `GroupedBy.Value` – 用于分组数据的字段的实际值，如 `GroupedBy.Key` 字段中指定的 ( 活动 ID )。
- `Values.Key` – 其值在 `Values.Value` 字段中指定的指标的友好名称 ( 此处为 `UniqueDeliveries` )。
- `Values.Type` – 在 `Values.Value` 字段中指定的值的数据类型 ( 此处为 `Double` )。
- `Values.Value` – 所查询指标的实际值，包括应用的任何筛选器。

如果特定项目、活动或其他资源的多值指标的查询结果为 `null` ( 不大于或等于零 )，则 Amazon Pinpoint 不会为该资源返回任何对象或字段。如果对于所有资源，某个多值指标的查询结果为 `null`，则 Amazon Pinpoint 返回一个空 `Rows` 对象。

## JSON 对象和字段

除了指定定义查询的值 ( 例如项目 ID (`ApplicationId`) ) 以外，应用程序指标、活动指标或旅程参与指标查询的每个 JSON 响应还包含一个 `KpiResult` 对象。此对象包含查询的整个结果集，可以分析该结果集以将分析数据发送到其他服务或应用程序。根据指标，每个 `KpiResult` 对象均包含以下部分或全部标准对象和字段。

对象或字段	描述
<code>Rows</code>	包含查询的结果集的对象数组。
<code>Rows.GroupedBy</code>	对于多值指标，为一个字段数组，用于定义已用于对查询结果中的数据进行分组的字段和值。
<code>Rows.GroupedBy.Key</code>	对于多值指标，为用于存储 <code>GroupedBy.Value</code> 字段中指定的值的属性或字段的名称。
<code>Rows.GroupedBy.Type</code>	对于多值指标，为 <code>GroupedBy.Value</code> 字段中指定的值的数据类型。
<code>Rows.GroupedBy.Value</code>	对于多值指标，为已用于对查询结果中的数据进行分组的字段的实际值。此值与关联的 <code>Values</code> 对象相关。

对象或字段	描述
Rows.Values	一个包含查询结果的字段数组。
Rows.Values.Key	查询的指标的友好名称。指标的值是在 Values.Value 字段中指定的。
Rows.Values.Type	Values.Value 字段中指定的值的数据类型。
Rows.Values.Value	所查询指标的实际值，包括应用的任何筛选器。

有关旅程执行指标或旅程活动执行指标查询的 JSON 响应中的字段的信息，请参阅[Amazon Pinpoint 标准分析指标](#)。

# 使用 AWS CloudTrail 记录 Amazon Pinpoint API 调用

Amazon Pinpoint 与 AWS CloudTrail 服务集成，该服务提供 Amazon Pinpoint 中用户、角色或 AWS 服务所采取的操作的记录。CloudTrail 将 Amazon Pinpoint 的 API 调用作为事件捕获。捕获的调用包括来自 Amazon Pinpoint 控制台的调用和对 Amazon Pinpoint API 操作的代码调用。

如果您创建跟踪记录，则可以使 CloudTrail 事件持续传送到 Amazon Simple Storage Service (Amazon S3) 存储桶（包括 Amazon Pinpoint 事件）。如果您不配置跟踪，则仍可在 CloudTrail 控制台上使用事件历史记录查看最新事件。使用 CloudTrail 收集的信息，您可以确定向 Amazon Pinpoint 发出了什么请求、发出请求的 IP 地址、何人发出的请求、请求的发出时间以及其它详细信息。

要了解有关 CloudTrail 的更多信息（包括如何对其进行配置和启用），请参阅 [AWS CloudTrail 用户指南](#)。

## CloudTrail 中的 Amazon Pinpoint 信息

在您创建 AWS 账户时，将在该账户上启用 CloudTrail。当 Amazon Pinpoint 中发生受支持的事件活动时，该活动将记录在 CloudTrail 事件中，并与其他 AWS 服务事件一同保存在事件历史记录中。您可以在 AWS 账户中查看、搜索和下载最新事件。有关更多信息，请参阅 [使用 CloudTrail 事件历史记录查看事件](#)。

要持续记录 AWS 账户中的事件（包括 Amazon Pinpoint 的事件），请创建跟踪记录。通过跟踪记录，CloudTrail 可将日志文件传送到 Amazon S3 存储桶。默认情况下，在控制台中创建跟踪记录时，此跟踪记录应用于所有 AWS 区域。此跟踪记录在 AWS 分区中记录所有区域中的事件，并将日志文件传送到您指定的 Amazon S3 存储桶。此外，您可以配置其他 AWS 服务，进一步分析在 CloudTrail 日志中收集的事件数据并采取行动。有关更多信息，请参阅下列内容：

- [创建跟踪概览](#)
- [CloudTrail 支持的服务和集成](#)
- [为 CloudTrail 配置 Amazon SNS 通知](#)
- [从多个区域接收 CloudTrail 日志文件和从多个账户接收 CloudTrail 日志文件](#)

每个事件或日志条目都包含有关生成请求的人员信息。身份信息可以帮助您确定：

- 请求是使用根用户凭证还是 AWS Identity and Access Management 用户凭证发出的。
- 请求是使用角色还是联合身份用户的临时安全凭证发出的。

- 请求是否由其它 AWS 服务发出。

有关更多信息，请参阅 [CloudTrail userIdentity 元素](#)。

您可以创建跟踪并将日志文件存储在 Amazon S3 存储桶中任意长时间。您也可以定义 Amazon S3 生命周期规则以自动存档或删除日志文件。默认情况下，将使用 Amazon S3 服务器端加密 (SSE) 对日志文件进行加密。

要在传送日志文件时获得通知，请将 CloudTrail 配置为在传送新日志文件时发布 Amazon SNS 通知。有关更多信息，请参阅 [CloudTrail 配置 Amazon Pinpoint 通知](#)。

您也可以将多个 AWS 区域和多个 AWS 账户的 Amazon Pinpoint 日志文件聚合到单个 Amazon S3 存储桶中。有关更多信息，请参阅 [接收来自多个区域的 CloudTrail 日志文件](#) 和 [从多个账户中接收 CloudTrail 日志文件](#)。

您可以使用 CloudTrail 记录以下 Amazon Pinpoint API 的操作：

- [Amazon Pinpoint API](#)
- [Amazon Pinpoint SMS 和 Voice API](#)

## CloudTrail 可以记录的 Amazon Pinpoint API 操作

Amazon Pinpoint API 支持将以下操作记录为 CloudTrail 日志文件中的事件：

- [CreateApp](#)
- [CreateCampaign](#)
- [CreateEmailTemplate](#)
- [CreateExportJob](#)
- [CreateImportJob](#)
- [CreateJourney](#)
- [CreatePushTemplate](#)
- [CreateRecommenderConfiguration](#)
- [CreateSegment](#)
- [CreateSmsTemplate](#)
- [CreateVoiceTemplate](#)

- [DeleteAdmChannel](#)
- [DeleteApnsChannel](#)
- [DeleteApnsSandboxChannel](#)
- [DeleteApnsVoipChannel](#)
- [DeleteApnsVoipSandboxChannel](#)
- [DeleteApp](#)
- [DeleteBaiduChannel](#)
- [DeleteCampaign](#)
- [DeleteEmailChannel](#)
- [DeleteEmailTemplate](#)
- [DeleteEndpoint](#)
- [DeleteEventStream](#)
- [DeleteGcmChannel](#)
- [DeleteJourney](#)
- [DeletePushTemplate](#)
- [DeleteRecommenderConfiguration](#)
- [DeleteSegment](#)
- [DeleteSmsChannel](#)
- [DeleteSmsTemplate](#)
- [DeleteUserEndpoints](#)
- [DeleteVoiceChannel](#)
- [DeleteVoiceTemplate](#)
- [GetAdmChannel](#)
- [GetApnsChannel](#)
- [GetApnsSandboxChannel](#)
- [GetApnsVoipChannel](#)
- [GetApnsVoipSandboxChannel](#)
- [GetApp](#)
- [GetApplicationDateRangeKpi](#)

- [GetApplicationSettings](#)
- [GetApps](#)
- [GetBaiduChannel](#)
- [GetCampaign](#)
- [GetCampaignActivities](#)
- [GetCampaignDateRangeKpi](#)
- [GetCampaignVersion](#)
- [GetCampaignVersions](#)
- [GetCampaigns](#)
- [GetChannels](#)
- [GetEmailChannel](#)
- [GetEmailTemplate](#)
- [GetEndpoint](#)
- [GetEventStream](#)
- [GetExportJob](#)
- [GetExportJobs](#)
- [GetGcmChannel](#)
- [GetImportJob](#)
- [GetImportJobs](#)
- [GetJourney](#)
- [GetJourneyDateRangeKpi](#)
- [GetJourneyExecutionActivityMetrics](#)
- [GetJourneyExecutionMetrics](#)
- [GetPushTemplate](#)
- [GetRecommenderConfiguration](#)
- [GetRecommenderConfigurations](#)
- [GetSegment](#)
- [GetSegmentExportJobs](#)
- [GetSegmentImportJobs](#)

- [GetSegmentVersion](#)
- [GetSegmentVersions](#)
- [GetSegments](#)
- [GetSmsChannel](#)
- [GetSmsTemplate](#)
- [GetUserEndpoints](#)
- [GetVoiceChannel](#)
- [GetVoiceTemplate](#)
- [ListJourneys](#)
- [ListTagsForResource](#)
- [ListTemplates](#)
- [ListTemplateVersions](#)
- [PhoneNumberValidate](#)
- [PutEventStream](#)
- [RemoveAttributes](#)
- [TagResource](#)
- [UntagResource](#)
- [UpdateAdmChannel](#)
- [UpdateApnsChannel](#)
- [UpdateApnsSandboxChannel](#)
- [UpdateApnsVoipChannel](#)
- [UpdateApnsVoipSandboxChannel](#)
- [UpdateApplicationSettings](#)
- [UpdateBaiduChannel](#)
- [UpdateCampaign](#)
- [UpdateEmailChannel](#)
- [UpdateEmailTemplate](#)
- [UpdateEndpoint](#)
- [UpdateEndpointsBatch](#)

- [UpdateGcmChannel](#)
- [UpdateJourney](#)
- [UpdateJourneyState](#)
- [UpdatePushTemplate](#)
- [UpdateRecommenderConfiguration](#)
- [UpdateSegment](#)
- [UpdateSmsChannel](#)
- [UpdateSmsTemplate](#)
- [UpdateTemplateActiveVersion](#)
- [UpdateVoiceChannel](#)
- [UpdateVoiceTemplate](#)

CloudTrail 中未记录以下 Amazon Pinpoint API 操作：

- PutEvents
- SendMessages
- SendUsersMessages

## CloudTrail 可以记录的 Amazon Pinpoint Email API 操作

Amazon Pinpoint Email API 支持将以下操作记录为 CloudTrail 日志文件中的事件：

- [CreateConfigurationSet](#)
- [CreateConfigurationSetEventDestination](#)
- [CreateDedicatedIpPool](#)
- [CreateEmailIdentity](#)
- [DeleteConfigurationSet](#)
- [DeleteConfigurationSetEventDestination](#)
- [DeleteDedicatedIpPool](#)
- [DeleteEmailIdentity](#)
- [GetAccount](#)



- [GetConfigurationSet](#)
- [GetConfigurationSetEventDestinations](#)
- [GetDedicatedIp](#)
- [GetDedicatedIps](#)
- [GetEmailIdentity](#)
- [ListConfigurationSets](#)
- [ListDedicatedIpPools](#)
- [ListEmailIdentities](#)
- [PutAccountDedicatedIpWarmupAttributes](#)
- [PutAccountSendingAttributes](#)
- [PutConfigurationSetDeliveryOptions](#)
- [PutConfigurationSetReputationOptions](#)
- [PutConfigurationSetSendingOptions](#)
- [PutConfigurationSetTrackingOptions](#)
- [PutDedicatedIpInPool](#)
- [PutDedicatedIpWarmupAttributes](#)
- [PutEmailIdentityDkimAttributes](#)
- [PutEmailIdentityFeedbackAttributes](#)
- [PutEmailIdentityMailFromAttributes](#)
- [UpdateConfigurationSetEventDestination](#)

CloudTrail 中未记录以下 Amazon Pinpoint Email API 操作：

- [SendEmail](#)

## CloudTrail 可以记录的 Amazon Pinpoint SMS 和 Voice API 第 1 版操作

Amazon Pinpoint SMS 和 Voice API 第 1 版支持将以下操作记录为 CloudTrail 日志文件中的事件：

- [CreateConfigurationSet](#)

- [CreateConfigurationSetEventDestination](#)
- [DeleteConfigurationSet](#)
- [DeleteConfigurationSetEventDestination](#)
- [GetConfigurationSetEventDestinations](#)
- [UpdateConfigurationSetEventDestination](#)

CloudTrail 中未记录以下 Amazon Pinpoint SMS 和 Voice API 第 1 版操作：

- [SendVoiceMessage](#)

## CloudTrail 可以记录的 Amazon Pinpoint SMS 和 Voice API 第 2 版操作

Amazon Pinpoint SMS 和 Voice API 第 2 版支持将以下操作记录为 CloudTrail 日志文件中的事件：

- [AssociateOriginationIdentity](#)
- [CreateConfigurationSet](#)
- [CreateEventDestination](#)
- [CreateOptOutList](#)
- [CreatePool](#)
- [DeleteConfigurationSet](#)
- [DeleteDefaultMessageType](#)
- [DeleteDefaultSenderId](#)
- [DeleteEventDestination](#)
- [DeleteKeyword](#)
- [DeleteOptedOutNumber](#)
- [DeleteOptOutList](#)
- [DeletePool](#)
- [DeleteTextMessageSpendLimitOverride](#)
- [DeleteVoiceMessageSpendLimitOverride](#)
- [DescribeAccountAttributes](#)

- [DescribeAccountLimits](#)
- [DescribeConfigurationSets](#)
- [DescribeKeywords](#)
- [DescribeOptedOutNumbers](#)
- [DescribeOptOutLists](#)
- [DescribePhoneNumbers](#)
- [DescribePools](#)
- [DescribeSenderIds](#)
- [DescribeSpendLimits](#)
- [DisassociateOriginationIdentity](#)
- [ListPoolOriginationIdentities](#)
- [ListTagsForResource](#)
- [PutKeyword](#)
- [PutOptedOutNumber](#)
- [ReleasePhoneNumber](#)
- [RequestPhoneNumber](#)
- [SetDefaultMessageType](#)
- [SetDefaultSenderId](#)
- [SetTextMessageSpendLimitOverride](#)
- [SetVoiceMessageSpendLimitOverride](#)
- [TagResource](#)
- [UntagResource](#)
- [UpdateEventDestination](#)
- [UpdatePhoneNumber](#)
- [UpdatePool](#)

CloudTrail 中未记录以下 Amazon Pinpoint SMS 和 Voice API 第 2 版操作：

- [SendTextMessage](#)
- [SendVoiceMessage](#)

## 示例：Amazon Pinpoint 日志文件条目

跟踪是一种配置，可用于将事件作为日志文件传送到您指定的 Amazon S3 存储桶。CloudTrail 日志文件包含一个或多个日志条目。事件表示来自任何源的单个请求。它包括有关所请求操作的信息、操作的日期和时间、请求参数等。CloudTrail 日志文件不是公用 API 调用的有序堆栈跟踪，因此它们不会按任何特定顺序显示。

下例显示示范 Amazon Pinpoint API 的 GetCampaigns 和 CreateCampaign 操作的 CloudTrail 日志条目。

```
{
  "Records": [
    {
      "awsRegion": "us-east-1",
      "eventID": "example0-09a3-47d6-a810-c5f9fd2534fe",
      "eventName": "GetCampaigns",
      "eventSource": "pinpoint.amazonaws.com",
      "eventTime": "2018-02-03T00:56:48Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.05",
      "readOnly": true,
      "recipientAccountId": "123456789012",
      "requestID": "example1-b9bb-50fa-abdb-80f274981d60",
      "requestParameters": {
        "application-id": "example71dfa4c1aab66332a5839798f",
        "page-size": "1000"
      },
      "responseElements": null,
      "sourceIPAddress": "192.0.2.0",
      "userAgent": "Jersey/${project.version} (URLConnection 1.8.0_144)",
      "userIdentity": {
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "accountId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "principalId": "123456789012",
        "sessionContext": {
          "attributes": {
            "creationDate": "2018-02-02T16:55:29Z",
            "mfaAuthenticated": "false"
          }
        }
      },
      "type": "Root"
    }
  ]
}
```

```
}
},
{
  "awsRegion": "us-east-1",
  "eventID": "example0-09a3-47d6-a810-c5f9fd2534fe",
  "eventName": "CreateCampaign",
  "eventSource": "pinpoint.amazonaws.com",
  "eventTime": "2018-02-03T01:05:16Z",
  "eventType": "AwsApiCall",
  "eventVersion": "1.05",
  "readOnly": false,
  "recipientAccountId": "123456789012",
  "requestID": "example1-b9bb-50fa-abdb-80f274981d60",
  "requestParameters": {
    "Description": "****",
    "HoldoutPercent": 0,
    "IsPaused": false,
    "MessageConfiguration": "****",
    "Name": "****",
    "Schedule": {
      "Frequency": "ONCE",
      "IsLocalTime": true,
      "StartTime": "2018-02-03T00:00:00-08:00",
      "Timezone": "utc-08"
    }
  },
  "SegmentId": "exampleda204adf991a80281aa0e591",
  "SegmentVersion": 1,
  "application-id": "example71dfa4c1aab66332a5839798f"
},
"responseElements": {
  "ApplicationId": "example71dfa4c1aab66332a5839798f",
  "CreationDate": "2018-02-03T01:05:16.425Z",
  "Description": "****",
  "HoldoutPercent": 0,
  "Id": "example54a654f80948680cbba240ede",
  "IsPaused": false,
  "LastModifiedDate": "2018-02-03T01:05:16.425Z",
  "MessageConfiguration": "****",
  "Name": "****",
  "Schedule": {
    "Frequency": "ONCE",
    "IsLocalTime": true,
    "StartTime": "2018-02-03T00:00:00-08:00",
    "Timezone": "utc-08"
  }
}
```

```

    },
    "SegmentId": "example4da204adf991a80281example",
    "SegmentVersion": 1,
    "State": {
      "CampaignStatus": "SCHEDULED"
    },
    "Version": 1
  },
  "sourceIPAddress": "192.0.2.0",
  "userAgent": "aws-cli/1.14.9 Python/3.4.3 Linux/3.4.0+ botocore/1.8.34",
  "userIdentity": {
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "accountId": "123456789012",
    "arn": "arn:aws:iam::123456789012:user/userName",
    "principalId": "AIDAIHTRCDA62EXAMPLE",
    "type": "IAMUser",
    "userName": "userName"
  }
}
]
}

```

下例显示示范 Amazon Pinpoint SMS 和 Voice API 的 CreateConfigurationSet 和 CreateConfigurationSetEventDestination 操作的 CloudTrail 日志条目。

```

{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIHTRCDA62EXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/SampleUser",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "SampleUser"
      },
      "eventTime": "2018-11-06T21:45:55Z",
      "eventSource": "sms-voice.amazonaws.com",
      "eventName": "CreateConfigurationSet",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "192.0.0.1",
      "userAgent": "PostmanRuntime/7.3.0",

```

```

    "requestParameters":{
      "ConfigurationSetName":"MyConfigurationSet"
    },
    "responseElements":null,
    "requestID":"56dcc091-e20d-11e8-87d2-9994aexample",
    "eventID":"725843fc-8846-41f4-871a-7c52dexample",
    "readOnly":false,
    "eventType":"AwsApiCall",
    "recipientAccountId":"123456789012"
  },
  {
    "eventVersion":"1.05",
    "userIdentity":{
      "type":"IAMUser",
      "principalId":"AIDAIHTRCDA62EXAMPLE",
      "arn":"arn:aws:iam::111122223333:user/SampleUser",
      "accountId":"111122223333",
      "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
      "userName":"SampleUser"
    },
    "eventTime":"2018-11-06T21:47:08Z",
    "eventSource":"sms-voice.amazonaws.com",
    "eventName":"CreateConfigurationSetEventDestination",
    "awsRegion":"us-east-1",
    "sourceIPAddress":"192.0.0.1",
    "userAgent":"PostmanRuntime/7.3.0",
    "requestParameters":{
      "EventDestinationName":"CloudWatchEventDestination",
      "ConfigurationSetName":"MyConfigurationSet",
      "EventDestination":{
        "Enabled":true,
        "MatchingEventTypes":[
          "INITIATED_CALL",
          "INITIATED_CALL"
        ],
        "CloudWatchLogsDestination":{
          "IamRoleArn":"arn:aws:iam::111122223333:role/iamrole-01",
          "LogGroupArn":"arn:aws:logs:us-east-1:111122223333:log-
group:clientloggroup-01"
        }
      }
    },
    "responseElements":null,
    "requestID":"81de1e73-e20d-11e8-b158-d5536example",
  }
}

```

```
    "eventID": "fcafc21f-7c93-4a3f-9e72-fca2dexample",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
  }
]
```



# 为 Amazon Pinpoint 资源添加标签

标签是可用来定义和关联 AWS 资源（包括某些类型的 Amazon Pinpoint 资源）的一种标记。标签可帮助您以不同方式（例如按用途、所有者、环境或其他标准）对资源进行分类和管理。例如，您可以使用标签来应用策略或自动化，或用于标识要满足某些合规性要求的资源。您可以向以下类型的 Amazon Pinpoint 资源添加标签：

- 活动
- 消息模板
- 项目（应用程序）
- 客户细分

一个资源最多可以有 50 个标签。

## 管理标签

每个标签都包含您定义的一个标签键和一个可选的标签值。标签键是一种常见的标签，充当更具体的标签值的类别。标签值充当标签键的描述符。

一个标签键可以包含多达 128 个字符。一个标签值可以包含多达 256 个字符。字符可以是 Unicode 字母、数字、空格或以下任一个符号：`_ . : / = + -`。以下附加限制适用于标签：

- 标签键和值区分大小写。
- 对于每个关联的资源，每个标签键都必须是唯一的，并且只能有一个值。
- `aws`：前缀保留给 AWS 使用；您不能在您定义的任何标签键或值中使用它。此外，您无法编辑或删除使用此前缀的标签键或值。使用此前缀的标签不计入每个资源 50 个标签的限额。
- 您无法仅根据其标签更新或删除资源。您还必须指定 Amazon 资源名称 (ARN) 或资源 ID，具体取决于您使用的操作。
- 您可以将标签与公共资源或共享资源相关联。但是，标签仅适用于您的 AWS 账户，不适用于共享资源的任何其他账户。此外，标签仅适用于位于您的 AWS 账户的指定 AWS 区域中的资源。

要在 Amazon Pinpoint 资源中添加、显示、更新和删除标签键和值，您可以使用 AWS Command Line Interface (AWS CLI)、Amazon Pinpoint API、AWS Resource Groups Tagging API 或 AWS SDK。要管理位于您的 AWS 账户的特定 AWS 区域中的所有 AWS 资源（包括 Amazon Pinpoint 资源）的标签键和值，请使用 [AWS Resource Groups Tagging API](#)。

## 在 IAM 策略中使用标签

开始实施标签后，您可以对 AWS Identity and Access Management (IAM) 策略和 API 操作应用基于标签的资源级权限。这包括支持在创建资源时为资源添加标签的操作。通过这种方式使用标签，您可以更全面地控制 AWS 账户中的哪些组 and 用户拥有创建和标记资源的权限，以及哪些组 and 用户拥有创建、更新和删除标签的权限。

例如，您可以创建一个策略，允许用户只要其名称是 Amazon Pinpoint 资源的 Owner 标签中的值，就可以完全访问这些资源：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ModifyResourceIfOwner",
      "Effect": "Allow",
      "Action": "mobiletargeting:*",
      "Resource": "*",
      "Condition": {
        "StringEqualsIgnoreCase": {
          "aws:ResourceTag/Owner": "${aws:username}"
        }
      }
    }
  ]
}
```

如果您定义基于标签的资源级权限，该权限立即生效。这意味着，您的资源一经创建就很安全，而且您可以快速开始将标签用于新资源。您还可以使用资源级权限来控制哪些标签键和值可以与新的和现有资源关联。有关更多信息，请参阅《AWS IAM 用户指南》中的[使用标签控制访问](#)。

## 向资源添加标签

以下示例展示了如何使用 [AWS CLI](#) 和 [Amazon Pinpoint REST API](#) 向 Amazon Pinpoint 资源添加标签。您也可以使用任何支持的 AWS SDK 为资源添加标签。

要在单次操作中将一个标签添加到多个 Amazon Pinpoint 资源，请使用 AWS CLI 或 [AWS Resource Groups Tagging API](#) 的资源组标记操作。

## 使用 API 添加标签

要使用 Amazon Pinpoint REST API 创建新资源并为它添加标签，请向相应的资源 URI 发送 POST 请求。在请求的正文中，请包括 `tags` 参数和值。以下示例显示如何在创建新项目时指定标签。

```
POST /v1/apps HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/x-www-form-urlencoded
Accept: application/json
Cache-Control: no-cache

{
  "Name": "MyProject",
  "tags": {
    "key1": "value1"
  }
}
```

要向现有资源添加标签，请向[标签](#) URI 发送 POST 请求。在 URI 中包含资源的 Amazon 资源名称 (ARN)。ARN 应采用 URL 编码。在请求的正文中，请包含 `tags` 参数和值，如以下示例所示。

```
POST /v1/tags/resource-arn HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache

{
  "tags": {
    "key1": "value1"
  }
}
```

## 使用 AWS CLI 添加标签

要使用 AWS CLI 创建新资源并为它添加标签，请使用资源的相应 `create` 命令。包括 `tags` 参数和值。以下示例显示在创建新项目时如何指定标签。

Linux, macOS, or Unix

```
$ aws pinpoint create-app \
```

```
--create-application-request '{
  "Name":"MyProject",
  "tags": {
    "key1":"value1",
    "key2":"value2"
  }
}'
```

## Windows Command prompt

```
C:\> aws pinpoint create-app ^
  --create-application-request Name=MyProject,tags={key1=value1,key2=value2}
```

在上述示例中，执行以下操作：

- 将 *MyProject* 替换为要赋予项目的名称。
- 将 *key1* 和 *key2* 替换为要添加到资源中的标签的键。
- 将 *value1* 和 *value2* 替换为要为相应键添加的标签的值。

有关可用来创建 Amazon Pinpoint 资源的命令的信息，请参阅 [AWS CLI 命令参考](#)。

要向现有资源添加标签，请使用 `tag-resource` 命令并为必需参数指定相应的值：

## Linux, macOS, or Unix

```
$ aws pinpoint tag-resource \
  --resource-arn resource-arn \
  --tags-model '{
  "tags": {
    "key1":"value1",
    "key2":"value2"
  }
}'
```

## Windows Command Prompt

```
C:\> aws pinpoint tag-resource ^
  --resource-arn resource-arn ^
  --tags-model tags={key1=value1,key2=value2}
```

在上述示例中，执行以下操作：

- 将 *resource-arn* 替换为您要向其中添加标签的资源的 Amazon 资源名称 (ARN)。
- 将 *key1* 和 *key2* 替换为要添加到资源中的标签的键。
- 将 *value1* 和 *value2* 替换为要为相应键添加的标签的值。

## 显示资源的标签

以下示例演示如何使用 [AWS CLI](#) 和 [Amazon Pinpoint REST API](#) 显示与 Amazon Pinpoint 资源关联的所有标签（键和值）的列表。您也可以使用任何支持的 AWS SDK 来显示与资源关联的标签。

### 使用 API 显示标签

要使用 Amazon Pinpoint REST API 显示与特定资源关联的所有标签，请将 GET 请求发送到[标签](#) URI，并且在 URI 中包括资源的 Amazon 资源名称 (ARN)。ARN 应采用 URL 编码。例如，以下请求会检索与指定活动 (*resource-arn*) 关联的所有标签：

```
GET /v1/tags/resource-arn HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

该请求的 JSON 响应包括一个 tags 对象。tags 对象列出与活动关联的所有标签键和值。

要显示与同一类型的多个资源关联的所有标签，请将 GET 请求发送到该类型资源的相应 URI。例如，以下请求检索有关指定项目 (*application-id*) 中的所有活动的信息：

```
GET /v1/apps/application-id/campaigns HTTP/1.1
Host: pinpoint.us-east-1.amazonaws.com
Content-Type: application/json
Accept: application/json
Cache-Control: no-cache
```

该请求的 JSON 响应列出项目中的所有活动。每个活动的 tags 对象列出与活动关联的所有标签键和值。

## 使用 AWS CLI 显示标签

要使用 AWS CLI 显示与特定资源关联的标签列表，请运行 `list-tags-for-resource` 命令并为 `resource-arn` 参数指定资源的 Amazon 资源名称 (ARN，如以下示例所示：

Linux, macOS, or Unix

```
$ aws pinpoint list-tags-for-resource \  
  --resource-arn resource-arn
```

Windows Command Prompt

```
C:\> aws pinpoint list-tags-for-resource ^  
  --resource-arn resource-arn
```

要显示具有标签的所有 Amazon Pinpoint 资源以及与那些资源关联的所有标签的列表，请使用 AWS Resource Groups Tagging API 的 [get-resources](#) 命令。将 `resource-type-filters` 参数设置为 `mobiletargeting`，如以下示例所示。

Linux, macOS, or Unix

```
$ aws resourcegroupstaggingapi get-resources \  
  --resource-type-filters "mobiletargeting"
```

Windows Command Prompt

```
C:\> aws resourcegroupstaggingapi get-resources ^  
  --resource-type-filters "mobiletargeting"
```

该命令的输出是具有标签的所有 Amazon Pinpoint 资源的 ARN 列表。该列表包括与每个资源关联的所有标签键和值。

## 更新资源的标签

有几种方法可以更新（覆盖）Amazon Pinpoint 资源的标签。更新标签的最佳方式取决于以下情况：

- 您要为其更新标签的资源类型。

- 您是要更新一个资源的标签还是同时更新多个资源的标签。
- 您是要更新标签键、标签值还是两者。

要同时更新一个 Amazon Pinpoint 项目或多个资源的标签，可以使用 AWS CLI 或 [AWS Resource Groups Tagging API](#) 的资源组标记操作。Amazon Pinpoint API 目前不为这两项任务提供直接支持。

要更新一个资源的一个标签，您可以使用 Amazon Pinpoint API [删除当前标签](#) 并 [添加新标签](#)。

## 从资源中删除标签

以下示例展示了如何使用 [AWS CLI](#) 和 [Amazon Pinpoint REST API](#) 从 Amazon Pinpoint 资源中删除标签（包括键和值）。您也可以使用任何支持的 AWS SDK 从资源删除标签。

要在单次操作中从多个 Amazon Pinpoint 资源删除标签，请使用 AWS CLI 或 [AWS Resource Groups Tagging API](#) 的资源组标记操作。要仅从资源中删除特定标签值（而不是标签键），您可以 [更新资源的标签](#)。

### 使用 API 删除标签

要使用 Amazon Pinpoint REST API 从资源中删除标签，请向 [标签](#) URI 发送 DELETE 请求。在此 URI 中，包含要从中删除标签的资源的 Amazon 资源名称 (ARN)，后跟 `tagKeys` 参数和要删除的标签。例如：

```
https://endpoint/v1/tags/resource-arn?tagKeys=key
```

其中：

- *endpoint* 是托管资源的 AWS 区域的 Amazon Pinpoint 端点。
- *resource-arn* 是您要从中删除标签的资源的 ARN。
- *key* 是您要从资源中删除的标签。

所有参数都应是 URL 编码的。

要从一个资源中删除多个标签键及其关联值，请为每个要删除的附加标签附上 `tagKeys` 参数，并用和号 (&) 分隔它们。例如：

```
https://endpoint/v1/tags/resource-arn?tagKeys=key1&tagKeys=key2
```

所有参数都应是 URL 编码的。

## 使用 AWS CLI 删除标签

要使用 AWS CLI 从资源中删除标签，请运行 `untag-resource` 命令。命令中包含 `tag-keys` 参数，如以下示例所示。

Linux, macOS, or Unix

```
$ aws pinpoint untag-resource \  
  --resource-arn resource-arn \  
  --tag-keys key1 key2
```

Windows Command Prompt

```
C:\> aws pinpoint untag-resource ^  
  --resource-arn resource-arn ^  
  --tag-keys key1 key2
```

在前面的示例中，进行以下更改：

- 将 *resource-arn* 替换为要从中移除标签的资源的 ARN。
- 将 *key1* 和 *key2* 替换为要从资源中删除的标签的键。

## 相关信息

有关您可用来管理 Amazon Pinpoint 资源的 CLI 命令的更多信息，请参阅 [AWS CLI 命令参考](#) 的 Amazon Pinpoint 部分。

有关 Amazon Pinpoint API 中的资源的更多信息，包括受支持的 HTTP(S) 方法、参数和架构，请参阅 [Amazon Pinpoint API 参考](#)。



# 使用 AWS Lambda 自定义建议

在 Amazon Pinpoint 中，您可以从推荐器模型中检索个性化建议，并将其添加到您从活动和旅程发送的消息中。推荐器模型是一种机器学习 (ML) 模型，它在数据中查找模式，并根据找到的模式生成预测和建议。它从一组给定的产品或项目中预测特定用户喜欢的内容，并以建议集的形式为用户提供该信息。

通过将推荐器模型与 Amazon Pinpoint 一起使用，您可以根据每个消息接收人的属性和行为向接收人发送个性化建议。通过使用 AWS Lambda，您还可以自定义和改进这些建议。例如，您可以将建议从单个文本值（如产品名称或 ID）动态转换为更复杂的内容（如产品名称、描述和图像）。您可以在 Amazon Pinpoint 发送消息时实时执行该操作。

此功能在以下 AWS 区域提供：美国东部（弗吉尼亚州北部）、美国西部（俄勒冈州）、亚太地区（孟买）、亚太地区（悉尼）和欧洲地区（爱尔兰）。

## 主题

- [在消息中使用建议](#)
- [创建 Lambda 函数](#)
- [分配 Lambda 函数策略](#)
- [授权 Amazon Pinpoint 调用该函数](#)
- [配置推荐器模型](#)

## 在消息中使用建议

要将推荐器模型与 Amazon Pinpoint 一起使用，请先创建一个 Amazon Personalize 解决方案并将该解决方案部署为 Amazon Personalize 活动。然后，在 Amazon Pinpoint 中为推荐器模型创建一个配置。在该配置中，您可以指定一些设置以确定如何从 Amazon Personalize 活动中检索和处理建议数据。这包括是否调用 AWS Lambda 函数以对检索的数据执行其他处理。

Amazon Personalize 是一项旨在帮助您创建机器学习模型的 AWS 服务，以便为使用您的应用程序的客户提供实时、个性化的建议。Amazon Personalize 将指导您完成创建和训练机器学习模型的过程，然后准备和部署该模型作为 Amazon Personalize 活动。接下来，您可以从活动中检索实时的个性化建议。要了解有关 Amazon Personalize 的更多信息，请参阅 [Amazon Personalize 开发人员指南](#)。

AWS Lambda 是一项计算服务，您可用来运行代码而无需预配置或管理服务器。您可将代码打包并上传到 AWS Lambda 作为 Lambda 函数。然后在调用该函数时，AWS Lambda 会运行该函数。函数可

被手动调用、自动调用以响应事件或者响应来自应用程序或服务（包括 Amazon Pinpoint）的请求。有关创建和调用 Lambda 函数的更多信息，请参阅 [AWS Lambda 开发人员指南](#)。

在为推荐器模型创建 Amazon Pinpoint 配置后，您可以将模型中的建议添加到从活动和旅程发送的消息中。您可以使用包含建议属性的消息变量的消息模板以做到这一点。建议的属性是一个旨在存储建议数据的动态端点或用户属性。在为推荐器模型创建配置时，您可以定义这些属性。

您可以在以下类型的消息模板中使用建议属性的变量：

- 电子邮件模板，用于您从活动或旅程中发送的电子邮件。
- 推送通知模板，用于您从活动中发送的推送通知。
- 短信模板，用于您从活动中发送的短信文本消息。

有关将推荐器模型与 Amazon Pinpoint 一起使用的更多信息，请参阅《Amazon Pinpoint 用户指南》中的 [机器学习模型](#)。

如果配置 Amazon Pinpoint 以调用处理建议数据的 Lambda 函数，每次在活动或旅程的消息中发送个性化建议时，Amazon Pinpoint 都会执行以下常规任务：

1. 评估及处理消息和消息模板的配置设置和内容。
2. 确定消息模板已连接到推荐器模型。
3. 评估用于连接到和使用模型的配置设置。这些设置是由模型的 [推荐器模型](#) 资源定义的。
4. 检测模型配置设置定义的建议属性的一个或多个消息变量。
5. 从模型配置设置中指定的 Amazon Personalize 活动检索建议数据。它使用 Amazon Personalize Runtime API 的 [GetRecommendations](#) 操作来执行此任务。
6. 将相应的建议数据添加到每个消息接收人的动态建议属性 (RecommendationItems) 中。
7. 调用 Lambda 函数，并将每个接收人的建议数据发送到该函数以进行处理。

数据将作为 JSON 对象发送，其中包含每个接收人的端点定义。每个端点定义包含一个 RecommendationItems 字段，其中包含由 1–5 个值组成的有序数组。数组中的值数量取决于模型的配置设置。

8. 等待 Lambda 函数处理数据并返回结果。

结果是一个 JSON 对象，其中包含每个接收人的更新的端点定义。每个更新的端点定义包含一个新 Recommendations 对象。该对象包含 1–10 个字段，在模型配置设置中定义的每个自定义建议属性各一个字段。其中的每个字段存储端点的改进建议数据。

9. 使用每个接收人的更新的端点定义，将每个消息变量替换为该接收人对应的值。

10 发送包含每个消息接收人的个性化建议的消息版本。

要以这种方式自定义和改进建议，请先创建一个 Lambda 函数以处理 Amazon Pinpoint 发送的端点定义，然后返回更新的端点定义。接下来，分配一个 Lambda 函数策略并授权 Amazon Pinpoint 调用该函数。然后，在 Amazon Pinpoint 中配置推荐器模型。在配置模型时，指定要调用的函数并定义要使用的建议属性。

## 创建 Lambda 函数

要了解如何创建 Lambda 函数，请参阅《AWS Lambda 开发人员指南》中的[入门](#)。在设计和开发函数时，请牢记以下要求和准则。

### 输入事件数据

在 Amazon Pinpoint 为推荐器模型调用 Lambda 函数时，它发送一个负载，其中包含发送消息的活动或旅程的配置和其他设置。负载包含一个 Endpoints 对象，该对象是将端点 ID 与消息接收人的端点定义关联的映射。

端点定义使用由 Amazon Pinpoint API 的[端点](#)资源定义的结构。不过，它们还包含一个名为 RecommendationItems 的动态建议属性字段。RecommendationItems 字段包含从 Amazon Personalize 活动返回的一个或多个端点建议项目。该字段的值是由 1–5 个建议的项目（作为字符串）组成的有序数组。数组中的项目数取决于您配置 Amazon Pinpoint 以便为每个端点或用户检索的建议项目数。

例如：

```
"Endpoints": {
  "endpointIDexample-1":{
    "ChannelType":"EMAIL",
    "Address":"sofiam@example.com",
    "EndpointStatus":"ACTIVE",
    "OptOut":"NONE",
    "EffectiveDate":"2020-02-26T18:56:24.875Z",
    "Attributes":{
      "AddressType":[
        "primary"
      ]
    },
    "User":{
      "UserId":"SofiaMartínez",
```

```
    "UserAttributes":{
      "LastName":[
        "Martínez"
      ],
      "FirstName":[
        "Sofia"
      ],
      "Neighborhood":[
        "East Bay"
      ]
    }
  },
  "RecommendationItems":[
    "1815",
    "2009",
    "1527"
  ],
  "CreationDate":"2020-02-26T18:56:24.875Z"
},
"endpointIDexample-2":{
  "ChannelType":"EMAIL",
  "Address":"alejandr@example.com",
  "EndpointStatus":"ACTIVE",
  "OptOut":"NONE",
  "EffectiveDate":"2020-02-26T18:56:24.897Z",
  "Attributes":{
    "AddressType":[
      "primary"
    ]
  }
},
"User":{
  "UserId":"AlejandroRosalez",
  "UserAttributes":{
    "LastName ":[
      "Rosalez"
    ],
    "FirstName":[
      "Alejandro"
    ],
    "Neighborhood":[
      "West Bay"
    ]
  }
},
```

```
    "RecommendationItems": [
      "1210",
      "6542",
      "4582"
    ],
    "CreationDate": "2020-02-26T18:56:24.897Z"
  }
}
```

在前面的示例中，相关的 Amazon Pinpoint 设置为：

- 配置推荐器模型，以便为每个端点或用户检索三个建议的项目。（`RecommendationsPerMessage` 属性的值设置为 3。）在使用该设置时，Amazon Pinpoint 为每个端点或用户检索并仅添加第一、第二和第三个建议的项目。
- 配置项目以使用自定义用户属性，这些属性存储每个用户的名字、姓氏以及他们居住的社区。（`UserAttributes` 对象包含这些属性的值。）
- 配置项目以使用自定义端点属性 (`AddressType`)，该属性指示端点是否为用户从项目中接收消息的首选地址（渠道）。（`Attributes` 对象包含该属性的值。）

在 Amazon Pinpoint 调用 Lambda 函数并将该负载作为事件数据发送时，AWS Lambda 将该数据传递给 Lambda 函数以进行处理。

每个负载最多可以包含 50 个端点的数据。如果分段包含超过 50 个端点，则 Amazon Pinpoint 反复调用该函数（每次最多处理 50 个端点），直到该函数处理了所有数据。

## 响应数据和要求

在设计和开发 Lambda 函数时，请牢记[机器学习模型的限额](#)。如果函数不满足这些限额定义的条件，则 Amazon Pinpoint 无法处理和发送消息。

还要牢记以下要求：

- 函数必须使用输入事件数据提供的相同格式返回更新的端点定义。
- 每个更新的端点定义可以包含端点或用户的 1–10 个自定义建议属性。这些属性的名称必须与在 Amazon Pinpoint 中配置推荐器模型时指定的属性名称匹配。
- 必须在单个 `Recommendations` 对象中为每个端点或用户返回所有自定义建议属性。该要求有助于确保不会发生命名冲突。您可以将 `Recommendations` 对象添加到端点定义中的任何位置。

- 每个自定义建议属性的值必须是一个字符串（单个值）或字符串数组（多个值）。如果值是字符串数组，我们建议您保持 Amazon Personalize 返回的建议项目的顺序，如 RecommendationItems 字段中所示。否则，您的内容可能无法反映模型为端点或用户提供的预测。
- 函数不应修改事件数据中的其他元素，包括端点或用户的其他属性值。它应该只添加和返回自定义推荐属性的值。Amazon Pinpoint 不接受对函数响应中任何其他值的更新。
- 函数必须托管在与调用函数的 Amazon Pinpoint 项目相同的 AWS 区域中。如果函数和项目不在同一个区域中，则 Amazon Pinpoint 无法将事件数据发送到函数。

如果不满足任何上述要求，则 Amazon Pinpoint 无法处理消息并将其发送到一个或多个端点。这可能会导致活动或旅程活动失败。

最后，我们建议您为函数保留 256 个并发执行。

总体而言，Lambda 函数应处理 Amazon Pinpoint 发送的事件数据，并返回修改的端点定义。它可以遍历 Endpoints 对象中的每个端点，并为每个端点创建和设置要使用的自定义建议属性值以做到这一点。以下示例处理程序（使用 Python 编写并继续前面的输入事件数据示例）说明了这一点：

```
import json
import string

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event))
    print("Received context: " + str(context))
    segment_endpoints = event["Endpoints"]
    new_segment = dict()
    for endpoint_id in segment_endpoints.keys():
        endpoint = segment_endpoints[endpoint_id]
        if supported_endpoint(endpoint):
            new_segment[endpoint_id] = add_recommendation(endpoint)

    print("Returning endpoints: " + json.dumps(new_segment))
    return new_segment

def supported_endpoint(endpoint):
    return True

def add_recommendation(endpoint):
    endpoint["Recommendations"] = dict()

    customTitleList = list()
    customGenreList = list()
```

```
for i,item in enumerate(endpoint["RecommendationItems"]):
    item = int(item)
    if item == 1210:
        customTitleList.insert(i, "Hanna")
        customGenreList.insert(i, "Action")
    elif item == 1527:
        customTitleList.insert(i, "Catastrophe")
        customGenreList.insert(i, "Comedy")
    elif item == 1815:
        customTitleList.insert(i, "Fleabag")
        customGenreList.insert(i, "Comedy")
    elif item == 2009:
        customTitleList.insert(i, "Late Night")
        customGenreList.insert(i, "Drama")
    elif item == 4582:
        customTitleList.insert(i, "Agatha Christie\'s The ABC Murders")
        customGenreList.insert(i, "Crime")
    elif item == 6542:
        customTitleList.insert(i, "Hunters")
        customGenreList.insert(i, "Drama")

endpoint["Recommendations"]["Title"] = customTitleList
endpoint["Recommendations"]["Genre"] = customGenreList

return endpoint
```

在前面的示例中，AWS Lambda 将事件数据作为 `event` 参数传递给处理程序。处理程序遍历 `Endpoints` 对象中的每个端点，并设置名为 `Recommendations.Title` 和 `Recommendations.Genre` 的自定义建议属性的值。`return` 语句将每个更新的端点定义返回到 Amazon Pinpoint。

继续前面的输入事件数据示例，更新的端点定义为：

```
"Endpoints":{
  "endpointIDexample-1":{
    "ChannelType":"EMAIL",
    "Address":"sofiam@example.com",
    "EndpointStatus":"ACTIVE",
    "OptOut":"NONE",
    "EffectiveDate":"2020-02-26T18:56:24.875Z",
    "Attributes":{
      "AddressType":[
        "primary"
```

```
    ],
  },
  "User":{
    "UserId":"SofiaMartínez",
    "UserAttributes":{
      "LastName":[
        "Martínez"
      ],
      "FirstName":[
        "Sofia"
      ],
      "Neighborhood":[
        "East Bay"
      ]
    }
  },
  "RecommendationItems":[
    "1815",
    "2009",
    "1527"
  ],
  "CreationDate":"2020-02-26T18:56:24.875Z",
  "Recommendations":{
    "Title":[
      "Fleabag",
      "Late Night",
      "Catastrophe"
    ],
    "Genre":[
      "Comedy",
      "Comedy",
      "Comedy"
    ]
  }
},
"endpointIDexample-2":{
  "ChannelType":"EMAIL",
  "Address":"alejandr@example.com",
  "EndpointStatus":"ACTIVE",
  "OptOut":"NONE",
  "EffectiveDate":"2020-02-26T18:56:24.897Z",
  "Attributes":{
    "AddressType":[
      "primary"
    ]
  }
}
```



```
    ],
  },
  "User":{
    "UserId":"AlejandroRosalez",
    "UserAttributes":{
      "LastName ":[
        "Rosalez"
      ],
      "FirstName":[
        "Alejandro"
      ],
      "Neighborhood":[
        "West Bay"
      ]
    }
  },
  "RecommendationItems":[
    "1210",
    "6542",
    "4582"
  ],
  "CreationDate":"2020-02-26T18:56:24.897Z",
  "Recommendations":{
    "Title":[
      "Hanna",
      "Hunters",
      "Agatha Christie\'s The ABC Murders"
    ],
    "Genre":[
      "Action",
      "Drama",
      "Crime"
    ]
  }
}
```

在前面的示例中，函数修改了它收到的 Endpoints 对象并返回了结果。现在，每个端点的 Endpoint 对象包含一个新 Recommendations 对象，其中包含 Title 和 Genre 字段。其中的每个字段存储由三个值（作为字符串）组成的有序数组，其中的每个值为 RecommendationItems 字段中的相应建议项目提供改进的内容。

## 分配 Lambda 函数策略

您必须先授权 Amazon Pinpoint 调用 Lambda 函数，然后才能使用该函数处理建议数据。要授予调用权限，请为该函数分配 Lambda 函数策略。Lambda 函数策略 是一个基于资源的权限策略，它指定哪些实体可以使用函数以及这些实体可以执行哪些操作。有关更多信息，请参阅《AWS Lambda 开发人员指南》中的[将基于资源的策略用于 AWS Lambda](#)。

以下示例策略允许 Amazon Pinpoint 服务主体将 `lambda:InvokeFunction` 操作用于特定 Amazon Pinpoint 项目 (*projectId*) 中的特定 Amazon Pinpoint 活动 (*campaignId*)：

```
{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
    "Service": "pinpoint.us-east-1.amazonaws.com"
  },
  "Action": "lambda:InvokeFunction",
  "Resource": "{arn:aws:lambda:us-east-1:accountId:function:function-name}",
  "Condition": {
    "ArnLike": {
      "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
    }
  }
}
```

该函数策略需要使用一个包含 `AWS:SourceArn` 键的 `Condition` 块。该键指定允许哪个资源调用函数。在前面的示例中，该策略允许一个特定活动调用函数。

您还可以编写一个策略，以允许 Amazon Pinpoint 服务主体将 `lambda:InvokeFunction` 操作用于特定 Amazon Pinpoint 项目 (*projectId*) 中的所有活动和旅程。以下示例策略说明了这一点：

```
{
  "Sid": "sid",
  "Effect": "Allow",
  "Principal": {
    "Service": "pinpoint.us-east-1.amazonaws.com"
  },
  "Action": "lambda:InvokeFunction",
  "Resource": "{arn:aws:lambda:us-east-1:accountId:function:function-name}",
  "Condition": {
    "ArnLike": {
```

```

    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
  }
}
}

```

与第一个示例不同，该示例的 Condition 块中的 AWS:SourceArn 键允许一个特定项目调用函数。该权限适用于项目中的所有活动和旅程。

要编写更通用的策略，您可以使用多字符匹配通配符 (\*)。例如，您可以使用以下 Condition 块以允许任何 Amazon Pinpoint 项目调用函数：

```

"Condition": {
  "ArnLike": {
    "AWS:SourceArn": "arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*"
  }
}

```

如果要将 Lambda 函数与您的 Amazon Pinpoint 账户的所有项目一起使用，我们建议您按前面的方式配置策略的 Condition 块。不过，作为最佳实践，您创建的策略只应包含对特定资源执行特定操作所需的权限。

## 授权 Amazon Pinpoint 调用该函数

在将 Lambda 函数策略分配给函数后，您可以添加权限以允许 Amazon Pinpoint 为特定项目、活动或旅程调用该函数。您可以使用 AWS Command Line Interface (AWS CLI) 和 Lambda [add-permission](#) 命令以执行该操作。以下示例说明了如何为特定项目 (*projectId*) 执行该操作：

```

$ aws lambda add-permission \
--function-name function-name \
--statement-id sid \
--action lambda:InvokeFunction \
--principal pinpoint.us-east-1.amazonaws.com \
--source-arn arn:aws:mobiletargeting:us-east-1:accountId:recommenders/*

```

前面的示例针对 Unix、Linux 和 macOS 进行了格式设置。对于 Microsoft Windows，请将反斜杠 (\) 行继续符替换为插入符号 (^)。

如果命令成功运行，则您将看到类似于以下内容的输出：

```
{
```

```
"Statement": "{ \"Sid\": \"sid\",
  \"Effect\": \"Allow\",
  \"Principal\": { \"Service\": \"pinpoint.us-east-1.amazonaws.com\" },
  \"Action\": \"lambda:InvokeFunction\",
  \"Resource\": \"arn:aws:lambda:us-east-1:111122223333:function:function-name\",
  \"Condition\":
    { \"ArnLike\":
      { \"AWS:SourceArn\":
        \"arn:aws:mobiletargeting:us-east-1:111122223333:recommenders/*\" } } } }
```

Statement 值是已添加到 Lambda 函数策略的语句的 JSON 字符串版本。

## 配置推荐器模型

要配置 Amazon Pinpoint 以便为推荐器模型调用 Lambda 函数，请为模型指定以下 Lambda 特定的配置设置：

- RecommendationTransformerUri – 该属性指定 Lambda 函数的名称或 Amazon 资源名称 (ARN)。
- Attributes – 该对象是一个映射，它定义了函数添加到每个端点定义的自定义建议属性。可以将其中的每个属性作为消息模板中的消息变量。

当您为模型创建配置或更新模型的配置时，可以使用 Amazon Pinpoint API 的[推荐器模型](https://docs.aws.amazon.com/pinpoint/latest/apireference/recommenders-recommender-id.html)资源 <https://docs.aws.amazon.com/pinpoint/latest/apireference/recommenders-recommender-id.html> 来指定这些设置。您也可以使用 Amazon Pinpoint 控制台定义这些设置。

有关将推荐器模型与 Amazon Pinpoint 一起使用的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[机器学习模型](#)。

# 从 Amazon Pinpoint 中删除数据

根据您使用 Amazon Pinpoint 的方式，它可能会存储某些被视为个人信息的数据。例如，Amazon Pinpoint 中的端点包含最终用户的联系信息，例如此人员的电子邮件地址或手机号码。

您可以使用控制台或 Amazon Pinpoint API 永久删除个人数据。本主题包含用于删除可能被视为个人数据的数据的过程。

## 删除端点

一个端点表示联系您的某个客户的单个方法。每个端点都可以引用客户的电子邮件地址、移动设备标识符、电话号码或您可以向其发送消息的其他目标类型。在许多司法管辖区内，此类信息可能被视为个人数据。

要删除特定端点的所有数据，您可以使用 Amazon Pinpoint API 来删除端点。以下过程演示了如何通过使用 AWS CLI 与 Amazon Pinpoint API 交互来删除端点。此过程假定您已安装和配置 AWS CLI。有关更多信息，请参阅 AWS Command Line Interface 用户指南 中的 [安装 AWS CLI](#)。

要删除终端节点，请使用 AWS CLI

- 在命令行输入以下命令：

```
aws pinpoint delete-endpoint --application-id 810c7aab86d42fb2b56c8c966example --  
endpoint-id ad015a3bf4f1b2b0b82example
```

在前面的命令中，将 *810c7aab86d42fb2b56c8c966example* 替换为与端点关联的项目的 ID。此外，将 *ad015a3bf4f1b2b0b82example* 替换为端点本身的唯一 ID。

要查找特定端点的端点 ID，请确定端点所属的分段，然后从 Amazon Pinpoint 中导出该分段。导出的数据包括每个端点的端点 ID。您可以使用 Amazon Pinpoint 控制台将分段导出到文件。要了解如何操作，请参阅《Amazon Pinpoint 用户指南》中的 [导出分段](#)。您也可以使用 Amazon Pinpoint API 将分段导出到 Amazon Simple Storage Service (Amazon S3) 存储桶。要了解如何导出，请参阅本指南中的 [导出端点](#)。

## 删除存储在 Amazon S3 中的分段和端点数据

您可以使用 Amazon Pinpoint 控制台或 API 从存储在 Amazon S3 存储桶中的文件导入分段，也可以将应用程序、分段或端点数据从 Amazon Pinpoint 导出到 Amazon S3 存储桶。导入的文件和导出的文件

都可包含个人数据，包括电子邮件地址、手机号码以及有关端点的物理位置的信息。您可以从 Amazon S3 中删除这些文件。

发送到 Amazon S3 存储桶的内容可能包含客户内容。有关删除敏感数据的更多信息，请参阅[如何清空 S3 存储桶？](#)或[如何删除 S3 存储桶？](#)。

## 删除所有项目数据

可以永久删除您为 Amazon Pinpoint 项目存储的所有数据。您可以通过删除项目来完成此操作。

### Warning

如果您删除某个项目，Amazon Pinpoint 将删除所有特定于该项目的设置和项目的数据。这些信息无法恢复。

当您删除一个项目时，Amazon Pinpoint 会删除特定于该项目的设置，包括推送通知、双向短信收发渠道、所有客户细分、活动、旅程设置，以及存储在 Amazon Pinpoint 中的分析数据，例如以下内容：


- 客户细分 – 所有客户细分设置和数据。对于动态客户细分，这包括您定义的客户细分组以及筛选条件。对于导入的客户细分，这包括端点、用户 ID 以及您导入的任何其他数据以及所应用的任何筛选条件。
- 活动 – 所有消息、消息处理和变量、分析数据、计划和其他设置。
- 旅程 – 所有活动、分析数据、计划和其他设置。
- 分析 – 所有互动指标的数据，例如为活动和旅程发送并送达的消息数量，以及所有旅程执行指标。对于移动和网络应用程序，所有未流式传输到其他 AWS 服务（例如 Amazon Kinesis）的事件数据、所有渠道以及应用程序使用情况、收入和人口统计指标的数据。在删除项目之前，建议您将此数据导出到其他位置。

您可以使用 Amazon Pinpoint 控制台删除项目。要了解更多信息，请参阅《Amazon Pinpoint 用户指南》中的[删除项目](#)。您也可以使用 Amazon Pinpoint API 的[应用程序](#)资源以编程方式删除项目。

## 通过关闭 AWS 账户删除所有 AWS 数据

此外，可以通过关闭您的 AWS 账户来删除存储在 Amazon Pinpoint 中的所有数据。但是，此操作还会删除您在所有其他服务中存储的所有其他数据（个人或非个人数据）。AWS

当您关闭 AWS 账户时，我们会将您的 AWS 账户中的数据保留 90 天。此保留期结束时，我们将永久删除此数据，并且此操作不可撤销。


 Warning

以下过程将完全删除所有 AWS 服务和 AWS 地区中存储在您 AWS 账户中的所有数据。

您可以使用关闭您的 AWS 账户 [AWS Management Console](#)。

关闭您的 AWS 账户

1. 打开 [网址为 AWS Management Console https://console.aws.amazon.com](https://console.aws.amazon.com)。
2. 转到账户设置页，网址为 <https://console.aws.amazon.com/billing/home?#/account>。

 Warning

以下步骤将永久删除您在所有 AWS 区域的所有 AWS 服务中存储的所有数据。

3. 在“关闭账户”下，阅读描述关闭 AWS 账户后果的免责声明。如果您同意这些条款，请选中复选框，然后选择关闭账户。
4. 在确认对话框中，选择关闭账户。

# 将 Amazon Pinpoint 与 AWS SDK 结合使用的代码示例

以下代码示例演示了如何将 Amazon Pinpoint 与 AWS 软件开发工具包 (SDK) 结合使用。

有关 AWS SDK 开发人员指南和代码示例的完整列表，请参阅 [通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 代码示例

- [使用软件开发工具包的 Amazon Pinpoint 的代码示例 AWS](#)
  - [使用软件开发工具包对 Amazon Pinpoint 执行的操作 AWS](#)
    - [创建 Amazon Pinpoint 活动](#)
    - [创建 Amazon Pinpoint 分段](#)
    - [创建 Amazon Pinpoint 应用程序](#)
    - [删除 Amazon Pinpoint 应用程序](#)
    - [删除 Amazon Pinpoint 端点](#)
    - [导出 Amazon Pinpoint 端点](#)
    - [显示有关现有 Amazon Pinpoint 端点的信息](#)
    - [导入 Amazon Pinpoint 分段](#)
    - [列出与特定用户 ID 关联的 Amazon Pinpoint 端点](#)
    - [列出应用程序中的 Amazon Pinpoint 分段](#)
    - [使用软件开发工具包通过 Amazon Pinpoint 发送电子邮件和短信 AWS](#)
    - [使用软件开发工具包通过 Amazon Pinpoint 发送模板化电子邮件和短信 AWS](#)
    - [更新 Amazon Pinpoint 端点](#)
    - [更新 Amazon Pinpoint 渠道](#)
  - [使用软件开发工具包的 Amazon Pinpoint 短信和语音 API 的代码示例 AWS](#)
    - [使用软件开发工具包对 Amazon Pinpoint 短信和语音 API 执行的操作 AWS](#)
      - [使用软件开发工具包通过 Amazon Pinpoint 短信和语音 API 发送语音消息 AWS](#)

## 使用软件开发工具包的 Amazon Pinpoint 的代码示例 AWS

以下代码示例展示了如何将 Amazon Pinpoint 与 AWS 软件开发套件 (SDK) 配合使用。



操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景和跨服务示例的上下文查看操作。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 代码示例

- [使用软件开发工具包对 Amazon Pinpoint 执行的操作 AWS](#)
  - [创建 Amazon Pinpoint 活动](#)
  - [创建 Amazon Pinpoint 分段](#)
  - [创建 Amazon Pinpoint 应用程序](#)
  - [删除 Amazon Pinpoint 应用程序](#)
  - [删除 Amazon Pinpoint 端点](#)
  - [导出 Amazon Pinpoint 端点](#)
  - [显示有关现有 Amazon Pinpoint 端点的信息](#)
  - [导入 Amazon Pinpoint 分段](#)
  - [列出与特定用户 ID 关联的 Amazon Pinpoint 端点](#)
  - [列出应用程序中的 Amazon Pinpoint 分段](#)
  - [使用软件开发工具包通过 Amazon Pinpoint 发送电子邮件和短信 AWS](#)
  - [使用软件开发工具包通过 Amazon Pinpoint 发送模板化电子邮件和短信 AWS](#)
  - [更新 Amazon Pinpoint 端点](#)
  - [更新 Amazon Pinpoint 渠道](#)

## 使用软件开发工具包对 Amazon Pinpoint 执行的操作 AWS

以下代码示例演示了如何使用软件开发工具包执行单个 Amazon Pinpoint AWS 操作。这些代码节选调用了 Amazon Pinpoint API，是必须在上下文中运行的较大型程序的代码节选。每个示例都包含一个指向的链接 GitHub，您可以在其中找到有关设置和运行代码的说明。

以下示例仅包括最常用的操作。有关完整列表，请参阅 [Amazon Pinpoint API 参考](#)。

### 示例

- [创建 Amazon Pinpoint 活动](#)
- [创建 Amazon Pinpoint 分段](#)

- [创建 Amazon Pinpoint 应用程序](#)
- [删除 Amazon Pinpoint 应用程序](#)
- [删除 Amazon Pinpoint 端点](#)
- [导出 Amazon Pinpoint 端点](#)
- [显示有关现有 Amazon Pinpoint 端点的信息](#)
- [导入 Amazon Pinpoint 分段](#)
- [列出与特定用户 ID 关联的 Amazon Pinpoint 端点](#)
- [列出应用程序中的 Amazon Pinpoint 分段](#)
- [使用软件开发工具包通过 Amazon Pinpoint 发送电子邮件和短信 AWS](#)
- [使用软件开发工具包通过 Amazon Pinpoint 发送模板化电子邮件和短信 AWS](#)
- [更新 Amazon Pinpoint 端点](#)
- [更新 Amazon Pinpoint 渠道](#)

## 创建 Amazon Pinpoint 活动

以下代码示例演示了如何创建一个活动。

### Java

适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

创建市场活动。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.Message;
import software.amazon.awssdk.services.pinpoint.model.Schedule;
import software.amazon.awssdk.services.pinpoint.model.Action;
import software.amazon.awssdk.services.pinpoint.model.MessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.WriteCampaignRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateCampaignRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateCampaign {
    public static void main(String[] args) {

        final String usage = ""

            Usage:  <appId> <segmentId>

            Where:
                appId - The ID of the application to create the campaign in.
                segmentId - The ID of the segment to create the campaign from.
            """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        String segmentId = args[1];
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        createPinCampaign(pinpoint, appId, segmentId);
        pinpoint.close();
    }

    public static void createPinCampaign(PinpointClient pinpoint, String appId,
        String segmentId) {
        CampaignResponse result = createCampaign(pinpoint, appId, segmentId);
        System.out.println("Campaign " + result.name() + " created.");
    }
}
```

```
        System.out.println(result.description());
    }

    public static CampaignResponse createCampaign(PinpointClient client, String
appID, String segmentID) {

        try {
            Schedule schedule = Schedule.builder()
                .startTime("IMMEDIATE")
                .build();

            Message defaultMessage = Message.builder()
                .action(Action.OPEN_APP)
                .body("My message body.")
                .title("My message title.")
                .build();

            MessageConfiguration messageConfiguration =
MessageConfiguration.builder()
                .defaultMessage(defaultMessage)
                .build();

            WriteCampaignRequest request = WriteCampaignRequest.builder()
                .description("My description")
                .schedule(schedule)
                .name("MyCampaign")
                .segmentId(segmentID)
                .messageConfiguration(messageConfiguration)
                .build();

            CreateCampaignResponse result =
client.createCampaign(CreateCampaignRequest.builder()
                .applicationId(appID)
                .writeCampaignRequest(request).build());

            System.out.println("Campaign ID: " + result.campaignResponse().id());
            return result.campaignResponse();

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return null;
    }
}
```

```
}  
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[CreateCampaign](#)中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun createPinCampaign(appId: String, segmentIdVal: String) {  
  
    val schedule0b = Schedule {  
        startTime = "IMMEDIATE"  
    }  
  
    val defaultMessage0b = Message {  
        action = Action.OpenApp  
        body = "My message body"  
        title = "My message title"  
    }  
  
    val messageConfiguration0b = MessageConfiguration {  
        defaultMessage = defaultMessage0b  
    }  
  
    val writeCampaign = WriteCampaignRequest {  
        description = "My description"  
        schedule = schedule0b  
        name = "MyCampaign"  
        segmentId = segmentIdVal  
        messageConfiguration = messageConfiguration0b  
    }  
  
    PinpointClient { region = "us-west-2" }.use { pinpoint ->  
        val result: CreateCampaignResponse = pinpoint.createCampaign(  

```

```
        CreateCampaignRequest {
            applicationId = appId
            writeCampaignRequest = writeCampaign
        }
    )
    println("Campaign ID is ${result.campaignResponse?.id}")
}
}
```

- 有关 API 的详细信息，请参阅适用 [CreateCampaign](#) 于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅 [通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 创建 Amazon Pinpoint 分段

以下代码示例演示了如何创建分段。

### Java

#### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息在 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AttributeDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.AttributeType;
import software.amazon.awssdk.services.pinpoint.model.RecencyDimension;
import software.amazon.awssdk.services.pinpoint.model.SegmentBehaviors;
import software.amazon.awssdk.services.pinpoint.model.SegmentDemographics;
import software.amazon.awssdk.services.pinpoint.model.SegmentLocation;
import software.amazon.awssdk.services.pinpoint.model.SegmentDimensions;
import software.amazon.awssdk.services.pinpoint.model.WriteSegmentRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentRequest;
```

```
import software.amazon.awssdk.services.pinpoint.model.CreateSegmentResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class CreateSegment {
    public static void main(String[] args) {
        final String usage = ""

                Usage:  <appId>

                Where:
                    appId - The application ID to create a segment
for.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        SegmentResponse result = createSegment(pinpoint, appId);
        System.out.println("Segment " + result.name() + " created.");
        System.out.println(result.segmentType());
        pinpoint.close();
    }

    public static SegmentResponse createSegment(PinpointClient client, String
appId) {
```

```
        try {
            Map<String, AttributeDimension> segmentAttributes = new
HashMap<>();
            segmentAttributes.put("Team",
AttributeDimension.builder()
                .attributeType(AttributeType.INCLUSIVE)
                .values("Lakers")
                .build());

            RecencyDimension recencyDimension =
RecencyDimension.builder()
                .duration("DAY_30")
                .recencyType("ACTIVE")
                .build();

            SegmentBehaviors segmentBehaviors =
SegmentBehaviors.builder()
                .recency(recencyDimension)
                .build();

            SegmentDemographics segmentDemographics =
SegmentDemographics
                .builder()
                .build();

            SegmentLocation segmentLocation = SegmentLocation
                .builder()
                .build();

            SegmentDimensions dimensions = SegmentDimensions
                .builder()
                .attributes(segmentAttributes)
                .behavior(segmentBehaviors)
                .demographic(segmentDemographics)
                .location(segmentLocation)
                .build();

            WriteSegmentRequest writeSegmentRequest =
WriteSegmentRequest.builder()
                .name("MySegment")
                .dimensions(dimensions)
                .build();
        }
```



```
        CreateSegmentRequest createSegmentRequest =
CreateSegmentRequest.builder()
                        .applicationId(appId)
                        .writeSegmentRequest(writeSegmentRequest)
                        .build();

        CreateSegmentResponse createSegmentResult =
client.createSegment(createSegmentRequest);
        System.out.println("Segment ID: " +
createSegmentResult.segmentResponse().id());
        System.out.println("Done");
        return createSegmentResult.segmentResponse();

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return null;
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[CreateSegment](#)中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun createPinpointSegment(applicationIdVal: String?): String? {

    val segmentAttributes = mutableMapOf<String, AttributeDimension>()
    val myList = mutableListOf<String>()
    myList.add("Lakers")

    val atts = AttributeDimension {
        attributeType = AttributeType.Inclusive
    }
}
```

```
        values = myList
    }

    segmentAttributes["Team"] = atts
    val recencyDimension = RecencyDimension {
        duration = Duration.fromValue("DAY_30")
        recencyType = RecencyType.fromValue("ACTIVE")
    }

    val segmentBehaviors = SegmentBehaviors {
        recency = recencyDimension
    }

    val segmentLocation = SegmentLocation {}
    val dimensions0b = SegmentDimensions {
        attributes = segmentAttributes
        behavior = segmentBehaviors
        demographic = SegmentDemographics {}
        location = segmentLocation
    }

    val writeSegmentRequest0b = WriteSegmentRequest {
        name = "MySegment101"
        dimensions = dimensions0b
    }

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val createSegmentResult: CreateSegmentResponse = pinpoint.createSegment(
            CreateSegmentRequest {
                applicationId = applicationIdVal
                writeSegmentRequest = writeSegmentRequest0b
            }
        )
        println("Segment ID is ${createSegmentResult.segmentResponse?.id}")
        return createSegmentResult.segmentResponse?.id
    }
}
```

- 有关 API 的详细信息，请参阅适用[CreateSegment](#)于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 创建 Amazon Pinpoint 应用程序

以下代码示例演示了如何创建一个应用程序。

### CLI

#### AWS CLI

##### 示例 1：创建应用程序

以下 `create-app` 示例创建一个新的应用程序（项目）。

```
aws pinpoint create-app \  
  --create-application-request Name=ExampleCorp
```

输出：

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",  
    "Name": "ExampleCorp",  
    "tags": {}  
  }  
}
```

##### 示例 2：创建带有标签的应用程序

以下 `create-app` 示例创建一个新的应用程序（项目），并将标签（键和值）与该应用程序关联。

```
aws pinpoint create-app \  
  --create-application-request Name=ExampleCorp,tags={"Stack"="Test"}
```

输出：

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",
```

```
        "Name": "ExampleCorp",
        "tags": {
            "Stack": "Test"
        }
    }
}
```

- 有关 API 的详细信息，请参阅AWS CLI 命令参考[CreateApp](#)中的。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateAppRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateAppResponse;
import software.amazon.awssdk.services.pinpoint.model.CreateApplicationRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class CreateApp {
    public static void main(String[] args) {
        final String usage = ""

            Usage: <appName>

            Where:
```

```
        appName - The name of the application to create.

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }
    String appName = args[0];
    System.out.println("Creating an application with name: " + appName);

    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    String appID = createApplication(pinpoint, appName);
    System.out.println("App ID is: " + appID);
    pinpoint.close();
}

public static String createApplication(PinpointClient pinpoint, String
appName) {
    try {
        CreateApplicationRequest appRequest =
CreateApplicationRequest.builder()
            .name(appName)
            .build();

        CreateAppRequest request = CreateAppRequest.builder()
            .createApplicationRequest(appRequest)
            .build();

        CreateAppResponse result = pinpoint.createApp(request);
        return result.applicationResponse().id();

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return "";
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考 [CreateApp](#) 中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun createApplication(applicationName: String?): String? {  
  
    val createApplicationRequestObj = CreateApplicationRequest {  
        name = applicationName  
    }  
  
    PinpointClient { region = "us-west-2" }.use { pinpoint ->  
        val result = pinpoint.createApp(  
            CreateAppRequest {  
                createApplicationRequest = createApplicationRequestObj  
            }  
        )  
        return result.applicationResponse?.id  
    }  
}
```

- 有关 API 的详细信息，请参阅适用 [CreateApp](#) 于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅 [通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 删除 Amazon Pinpoint 应用程序

以下代码示例演示了如何删除应用程序。

## CLI

### AWS CLI

#### 删除应用程序

以下 `delete-app` 示例删除一个应用程序（项目）。

```
aws pinpoint delete-app \  
  --application-id 810c7aab86d42fb2b56c8c966example
```

输出：

```
{  
  "ApplicationResponse": {  
    "Arn": "arn:aws:mobiletargeting:us-  
west-2:AIDACKCEVSQ6C2EXAMPLE:apps/810c7aab86d42fb2b56c8c966example",  
    "Id": "810c7aab86d42fb2b56c8c966example",  
    "Name": "ExampleCorp",  
    "tags": {}  
  }  
}
```

- 有关 API 的详细信息，请参阅AWS CLI 命令参考[DeleteApp](#)中的。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

删除应用程序。

```
import software.amazon.awssdk.regions.Region;  
import software.amazon.awssdk.services.pinpoint.PinpointClient;  
import software.amazon.awssdk.services.pinpoint.model.DeleteAppRequest;  
import software.amazon.awssdk.services.pinpoint.model.DeleteAppResponse;
```

```
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class DeleteApp {
    public static void main(String[] args) {
        final String usage = ""

            Usage: <appId>

            Where:
            appId - The ID of the application to delete.

            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String appId = args[0];
        System.out.println("Deleting an application with ID: " + appId);
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        deletePinApp(pinpoint, appId);
        System.out.println("Done");
        pinpoint.close();
    }

    public static void deletePinApp(PinpointClient pinpoint, String appId) {
        try {
            DeleteAppRequest appRequest = DeleteAppRequest.builder()
                .applicationId(appId)
                .build();
        }
    }
}
```



```
        DeleteAppResponse result = pinpoint.deleteApp(appRequest);
        String appName = result.applicationResponse().name();
        System.out.println("Application " + appName + " has been deleted.");

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考 [DeleteApp](#) 中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun deletePinApp(appId: String?) {

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.deleteApp(
            DeleteAppRequest {
                applicationId = appId
            }
        )
        val appName = result.applicationResponse?.name
        println("Application $appName has been deleted.")
    }
}
```

- 有关 API 的详细信息，请参阅适用 [DeleteApp](#) 于 Kotlin 的 AWS SDK API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 删除 Amazon Pinpoint 端点

以下代码示例演示了如何删除端点。

Java

适用于 Java 2.x 的 SDK

### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

删除端点。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.DeleteEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DeleteEndpoint {
    public static void main(String[] args) {
        final String usage = ""

                Usage:  <appName> <endpointId >

                Where:
                    appId - The id of the application to delete.
    }
}
```

```
        endpointId - The id of the endpoint to delete.
        """);

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String appId = args[0];
    String endpointId = args[1];
    System.out.println("Deleting an endpoint with id: " + endpointId);
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    deletePinEndpoint(pinpoint, appId, endpointId);
    pinpoint.close();
}

public static void deletePinEndpoint(PinpointClient pinpoint, String appId,
String endpointId) {
    try {
        DeleteEndpointRequest appRequest = DeleteEndpointRequest.builder()
            .applicationId(appId)
            .endpointId(endpointId)
            .build();

        DeleteEndpointResponse result = pinpoint.deleteEndpoint(appRequest);
        String id = result.endpointResponse().id();
        System.out.println("The deleted endpoint id " + id);

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    System.out.println("Done");
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考 [DeleteEndpoint](#) 中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun deletePinEndpoint(appIdVal: String?, endpointIdVal: String?) {  
  
    val deleteEndpointRequest = DeleteEndpointRequest {  
        applicationId = appIdVal  
        endpointId = endpointIdVal  
    }  
  
    PinpointClient { region = "us-west-2" }.use { pinpoint ->  
        val result = pinpoint.deleteEndpoint(deleteEndpointRequest)  
        val id = result.endpointResponse?.id  
        println("The deleted endpoint is $id")  
    }  
}
```

- 有关 API 的详细信息，请参阅适用[DeleteEndpoint](#)于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

### 导出 Amazon Pinpoint 端点

以下代码示例展示了如何导出端点。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

导出端点。

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.ExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.CreateExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.GetExportJobRequest;
import software.amazon.awssdk.services.s3.S3Client;
import software.amazon.awssdk.services.s3.model.GetObjectRequest;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Request;
import software.amazon.awssdk.services.s3.model.ListObjectsV2Response;
import software.amazon.awssdk.services.s3.model.S3Object;
import software.amazon.awssdk.services.s3.model.GetObjectResponse;
import software.amazon.awssdk.services.s3.model.S3Exception;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.util.stream.Collectors;

/**
 * To run this code example, you need to create an AWS Identity and Access
 * Management (IAM) role with the correct policy as described in this
 * documentation:
```

```
* https://docs.aws.amazon.com/pinpoint/latest/developerguide/audience-data-export.html
*
* Also, set up your development environment, including your credentials.
*
* For information, see this documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
```

```
public class ExportEndpoints {
    public static void main(String[] args) {
        final String usage = ""
```

```
        This program performs the following steps:
```

1. Exports the endpoints to an Amazon S3 bucket.
2. Downloads the exported endpoints files from Amazon S3.
3. Parses the endpoints files to obtain the endpoint IDs and

```
prints them.
```

```
        Usage: ExportEndpoints <applicationId> <s3BucketName>
<iamExportRoleArn> <path>
```

```
        Where:
```

```
        applicationId - The ID of the Amazon Pinpoint application that
has the endpoint.
```

```
        s3BucketName - The name of the Amazon S3 bucket to export the
JSON file to.\s
```

```
        iamExportRoleArn - The ARN of an IAM role that grants Amazon
Pinpoint write permissions to the S3 bucket. path - The path where the files
downloaded from the Amazon S3 bucket are written (for example, C:/AWS/).
```

```
        """;
```

```
    if (args.length != 4) {
        System.out.println(usage);
        System.exit(1);
    }
```

```
    String applicationId = args[0];
    String s3BucketName = args[1];
    String iamExportRoleArn = args[2];
    String path = args[3];
    System.out.println("Deleting an application with ID: " + applicationId);
```

```
Region region = Region.US_EAST_1;
PinpointClient pinpoint = PinpointClient.builder()
    .region(region)
    .build();

S3Client s3Client = S3Client.builder()
    .region(region)
    .build();

exportAllEndpoints(pinpoint, s3Client, applicationId, s3BucketName, path,
iamExportRoleArn);
pinpoint.close();
s3Client.close();
}

public static void exportAllEndpoints(PinpointClient pinpoint,
    S3Client s3Client,
    String applicationId,
    String s3BucketName,
    String path,
    String iamExportRoleArn) {

    try {
        List<String> objectKeys = exportEndpointsToS3(pinpoint, s3Client,
s3BucketName, iamExportRoleArn,
            applicationId);
        List<String> endpointFileKeys = objectKeys.stream().filter(o ->
o.endsWith(".gz"))
            .collect(Collectors.toList());
        downloadFromS3(s3Client, path, s3BucketName, endpointFileKeys);

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}

public static List<String> exportEndpointsToS3(PinpointClient pinpoint,
S3Client s3Client, String s3BucketName,
    String iamExportRoleArn, String applicationId) {

    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd-
HH_mm:ss.SSS_z");
```

```
String endpointsKeyPrefix = "exports/" + applicationId + "_" +
dateFormat.format(new Date());
String s3UrlPrefix = "s3://" + s3BucketName + "/" + endpointsKeyPrefix +
"/";
List<String> objectKeys = new ArrayList<>();
String key;

try {
    // Defines the export job that Amazon Pinpoint runs.
    ExportJobRequest jobRequest = ExportJobRequest.builder()
        .roleArn(iamExportRoleArn)
        .s3UrlPrefix(s3UrlPrefix)
        .build();

    CreateExportJobRequest exportJobRequest =
CreateExportJobRequest.builder()
        .applicationId(applicationId)
        .exportJobRequest(jobRequest)
        .build();

    System.out.format("Exporting endpoints from Amazon Pinpoint
application %s to Amazon S3 " +
        "bucket %s . . .\n", applicationId, s3BucketName);

    CreateExportJobResponse exportResult =
pinpoint.createExportJob(exportJobRequest);
    String jobId = exportResult.exportJobResponse().id();
    System.out.println(jobId);
    printExportJobStatus(pinpoint, applicationId, jobId);

    ListObjectsV2Request v2Request = ListObjectsV2Request.builder()
        .bucket(s3BucketName)
        .prefix(endpointsKeyPrefix)
        .build();

    // Create a list of object keys.
    ListObjectsV2Response v2Response = s3Client.listObjectsV2(v2Request);
    List<S3Object> objects = v2Response.contents();
    for (S3Object object : objects) {
        key = object.key();
        objectKeys.add(key);
    }

    return objectKeys;
}
```



```
    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return null;
}

private static void printExportJobStatus(PinpointClient pinpointClient,
    String applicationId,
    String jobId) {

    GetExportJobResponse getExportJobResult;
    String status;

    try {
        // Checks the job status until the job completes or fails.
        GetExportJobRequest exportJobRequest = GetExportJobRequest.builder()
            .jobId(jobId)
            .applicationId(applicationId)
            .build();

        do {
            getExportJobResult =
pinpointClient.getExportJob(exportJobRequest);
            status =
getExportJobResult.exportJobResponse().jobStatus().toString().toUpperCase();
            System.out.format("Export job %s . . .\n", status);
            TimeUnit.SECONDS.sleep(3);

        } while (!status.equals("COMPLETED") && !status.equals("FAILED"));

        if (status.equals("COMPLETED")) {
            System.out.println("Finished exporting endpoints.");
        } else {
            System.err.println("Failed to export endpoints.");
            System.exit(1);
        }

    } catch (PinpointException | InterruptedException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
```

```
// Download files from an Amazon S3 bucket and write them to the path
location.
public static void downloadFromS3(S3Client s3Client, String path, String
s3BucketName, List<String> objectKeys) {

    String newPath;
    try {
        for (String key : objectKeys) {
            GetObjectRequest objectRequest = GetObjectRequest.builder()
                .bucket(s3BucketName)
                .key(key)
                .build();

            ResponseBytes<GetObjectResponse> objectBytes =
s3Client.getObjectAsBytes(objectRequest);
            byte[] data = objectBytes.asByteArray();

            // Write the data to a local file.
            String fileSuffix = new
SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
            newPath = path + fileSuffix + ".gz";
            File myFile = new File(newPath);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
        }
        System.out.println("Download finished.");
    } catch (S3Exception | NullPointerException | IOException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[CreateExportJob](#)中的。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 显示有关现有 Amazon Pinpoint 端点的信息

以下代码示例演示了如何获取端点。

### CLI

#### AWS CLI

检索有关应用程序特定端点的设置和属性的信息

以下 `get-endpoint` 示例检索有关应用程序特定端点的设置和属性的信息。

```
aws pinpoint get-endpoint \  
  --application-id 611e3e3cdd47474c9c1399a505665b91 \  
  --endpoint-id testendpoint \  
  --region us-east-1
```

输出：

```
{  
  "EndpointResponse": {  
    "Address": "+11234567890",  
    "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",  
    "Attributes": {},  
    "ChannelType": "SMS",  
    "CohortId": "63",  
    "CreationDate": "2019-01-28T23:55:11.534Z",  
    "EffectiveDate": "2021-08-06T00:04:51.763Z",  
    "EndpointStatus": "ACTIVE",  
    "Id": "testendpoint",  
    "Location": {  
      "Country": "USA"  
    },  
    "Metrics": {  
      "SmsDelivered": 1.0  
    },  
    "OptOut": "ALL",  
    "RequestId": "a204b1f2-7e26-48a7-9c80-b49a2143489d",  
    "User": {  
      "UserAttributes": {  
        "Age": [  
          "24"  
        ]  
      }  
    }  
  }  
}
```

```
    },
    "UserId": "testuser"
  }
}
```

- 有关 API 的详细信息，请参阅AWS CLI 命令参考[GetEndpoint](#)中的。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import com.google.gson.FieldNamingPolicy;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class LookUpEndpoint {
    public static void main(String[] args) {
        final String usage = ""

                Usage:  <appId> <endpoint>
```

```
        Where:
            appId - The ID of the application to delete.
            endpoint - The ID of the endpoint.\s
            """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String appId = args[0];
    String endpoint = args[1];
    System.out.println("Looking up an endpoint point with ID: " + endpoint);
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    lookupPinpointEndpoint(pinpoint, appId, endpoint);
    pinpoint.close();
}

public static void lookupPinpointEndpoint(PinpointClient pinpoint, String
appId, String endpoint) {
    try {
        GetEndpointRequest appRequest = GetEndpointRequest.builder()
            .applicationId(appId)
            .endpointId(endpoint)
            .build();

        GetEndpointResponse result = pinpoint.getEndpoint(appRequest);
        EndpointResponse endResponse = result.endpointResponse();

        // Uses the Google Gson library to pretty print the endpoint JSON.
        Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .setPrettyPrinting()
            .create();

        String endpointJson = gson.toJson(endResponse);
        System.out.println(endpointJson);

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
    }
}
```

```
        System.exit(1);
    }
    System.out.println("Done");
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[GetEndpoint](#)中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun lookupPinpointEndpoint(appId: String?, endpoint: String?) {

    PinpointClient { region = "us-west-2" }.use { pinpoint ->
        val result = pinpoint.getEndpoint(
            GetEndpointRequest {
                applicationId = appId
                endpointId = endpoint
            }
        )
        val endResponse = result.endpointResponse

        // Uses the Google Gson library to pretty print the endpoint JSON.
        val gson: com.google.gson.Gson = GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .setPrettyPrinting()
            .create()

        val endpointJson: String = gson.toJson(endResponse)
        println(endpointJson)
    }
}
```

- 有关 API 的详细信息，请参阅适用[GetEndpoint](#)于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 导入 Amazon Pinpoint 分段

以下代码示例演示了如何导入分段。

### Java

#### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息在 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

导入分段。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.ImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.ImportJobRequest;
import software.amazon.awssdk.services.pinpoint.model.Format;
import software.amazon.awssdk.services.pinpoint.model.CreateImportJobResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ImportSegment {
    public static void main(String[] args) {
        final String usage = ""
```

```
Usage:  <appId> <bucket> <key> <roleArn>\s

Where:
  appId - The application ID to create a segment for.
  bucket - The name of the Amazon S3 bucket that contains the
segment definitons.
  key - The key of the S3 object.
  roleArn - ARN of the role that allows Amazon
Pinpoint to access S3. You need to set trust management for this
to work. See https://docs.aws.amazon.com/IAM/latest/UserGuide/
reference\_policies\_elements\_principal.html
      """;

if (args.length != 4) {
    System.out.println(usage);
    System.exit(1);
}

String appId = args[0];
String bucket = args[1];
String key = args[2];
String roleArn = args[3];

PinpointClient pinpoint = PinpointClient.builder()
    .region(Region.US_EAST_1)
    .build();

ImportJobResponse response = createImportSegment(pinpoint, appId, bucket,
key, roleArn);
System.out.println("Import job for " + bucket + " submitted.");
System.out.println("See application " + response.applicationId() + " for
import job status.");
System.out.println("See application " + response.jobStatus() + " for
import job status.");
pinpoint.close();
}

public static ImportJobResponse createImportSegment(PinpointClient client,
    String appId,
    String bucket,
    String key,
    String roleArn) {
```



```
try {
    ImportJobRequest importRequest = ImportJobRequest.builder()
        .defineSegment(true)
        .registerEndpoints(true)
        .roleArn(roleArn)
        .format(Format.JSON)
        .s3Url("s3://" + bucket + "/" + key)
        .build();

    CreateImportJobRequest jobRequest = CreateImportJobRequest.builder()
        .importJobRequest(importRequest)
        .applicationId(appId)
        .build();

    CreateImportJobResponse jobResponse =
client.createImportJob(jobRequest);
    return jobResponse.importJobResponse();

} catch (PinpointException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
return null;
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[CreateImportJob](#)中的。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 列出与特定用户 ID 关联的 Amazon Pinpoint 端点

以下代码示例演示了如何列出端点。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetUserEndpointsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListEndpointIds {
    public static void main(String[] args) {
        final String usage = ""

                Usage:    <applicationId> <userId>

                Where:
                    applicationId - The ID of the Amazon Pinpoint application that
                    has the endpoint.
                    userId - The user id applicable to the endpoints""";

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
String applicationId = args[0];
String userId = args[1];
PinpointClient pinpoint = PinpointClient.builder()
    .region(Region.US_EAST_1)
    .build();

listAllEndpoints(pinpoint, applicationId, userId);
pinpoint.close();
}

public static void listAllEndpoints(PinpointClient pinpoint,
    String applicationId,
    String userId) {

    try {
        GetUserEndpointsRequest endpointsRequest =
        GetUserEndpointsRequest.builder()
            .userId(userId)
            .applicationId(applicationId)
            .build();

        GetUserEndpointsResponse response =
        pinpoint.getUserEndpoints(endpointsRequest);
        List<EndpointResponse> endpoints =
        response.endpointsResponse().item();

        // Display the results.
        for (EndpointResponse endpoint : endpoints) {
            System.out.println("The channel type is: " +
            endpoint.channelType());
            System.out.println("The address is " + endpoint.address());
        }

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考 [GetUserEndpoints](#) 中的。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 列出应用程序中的 Amazon Pinpoint 分段

以下代码示例演示了如何列出分段。

### Java

#### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

列出分段。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.GetSegmentsRequest;
import software.amazon.awssdk.services.pinpoint.model.GetSegmentsResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.SegmentResponse;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListSegments {
    public static void main(String[] args) {
        final String usage = ""

            Usage:  <appId>
```

```
        Where:
            appId - The ID of the application that contains a segment.

        """;

    if (args.length != 1) {
        System.out.println(usage);
        System.exit(1);
    }

    String appId = args[0];
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    listSegs(pinpoint, appId);
    pinpoint.close();
}

public static void listSegs(PinpointClient pinpoint, String appId) {
    try {
        GetSegmentsRequest request = GetSegmentsRequest.builder()
            .applicationId(appId)
            .build();

        GetSegmentsResponse response = pinpoint.getSegments(request);
        List<SegmentResponse> segments = response.segmentsResponse().item();
        for (SegmentResponse segment : segments) {
            System.out
                .println("Segment " + segment.id() + " " +
                    segment.name() + " " + segment.lastModifiedDate());
        }

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[GetSegments](#)中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
suspend fun listSegs(appId: String?) {  
  
    PinpointClient { region = "us-west-2" }.use { pinpoint ->  
  
        val response = pinpoint.getSegments(  
            GetSegmentsRequest {  
                applicationId = appId  
            }  
        )  
        response.segmentsResponse?.item?.forEach { segment ->  
            println("Segment id is ${segment.id}")  
        }  
    }  
}
```

- 有关 API 的详细信息，请参阅适用 [GetSegments](#) 于 Kotlin 的 AWS SDK API 参考。

有关 AWS SDK 开发者指南和代码示例的完整列表，请参阅 [通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

### 使用软件开发工具包通过 Amazon Pinpoint 发送电子邮件和短信 AWS

以下代码示例演示了如何通过 Amazon Pinpoint 发送电子邮件和文本消息。

#### CLI

##### AWS CLI

使用应用程序的端点发送短信

以下 `send-messages` 示例通过端点为应用程序发送直接消息。

```
aws pinpoint send-messages \  
  --application-id 611e3e3cdd47474c9c1399a505665b91 \  
  --message-request file://myfile.json \  
  --region us-west-2
```

myfile.json 的内容：

```
{  
  "MessageConfiguration": {  
    "SMSMessage": {  
      "Body": "hello, how are you?"  
    }  
  },  
  "Endpoints": {  
    "testendpoint": {}  
  }  
}
```

输出：

```
{  
  "MessageResponse": {  
    "ApplicationId": "611e3e3cdd47474c9c1399a505665b91",  
    "EndpointResult": {  
      "testendpoint": {  
        "Address": "+12345678900",  
        "DeliveryStatus": "SUCCESSFUL",  
        "MessageId": "itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0",  
        "StatusCode": 200,  
        "StatusMessage": "MessageId:  
itnuqhai5alf1n6ahv3udc05n7hhddr6gb3lq6g0"  
      }  
    },  
    "RequestId": "c7e23264-04b2-4a46-b800-d24923f74753"  
  }  
}
```

有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 [Amazon Pinpoint SMS 渠道](#)。

- 有关 API 的详细信息，请参阅AWS CLI 命令参考[SendMessages](#)中的。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

发送电子邮件。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmailPart;
import software.amazon.awssdk.services.pinpoint.model.SimpleEmail;
import software.amazon.awssdk.services.pinpoint.model.EmailMessage;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;

import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
```



```
public class SendEmailMessage {

    // The character encoding the you want to use for the subject line and
    // message body of the email.
    public static String charset = "UTF-8";

    // The body of the email for recipients whose email clients support HTML
    content.
    static final String body = """"
        Amazon Pinpoint test (AWS SDK for Java 2.x)

        This email was sent through the Amazon Pinpoint Email API using the AWS
        SDK for Java 2.x

        """";

    public static void main(String[] args) {
        final String usage = """"

            Usage:    <subject> <appId> <senderAddress>
<toAddress>

            Where:
                subject - The email subject to use.
                senderAddress - The from address. This address has to be verified
in Amazon Pinpoint in the region you're using to send email\s
                toAddress - The to address. This address has to be verified in
Amazon Pinpoint in the region you're using to send email\s
            """";

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String subject = args[0];
        String senderAddress = args[1];
        String toAddress = args[2];
        System.out.println("Sending a message");
        PinpointEmailClient pinpoint = PinpointEmailClient.builder()
            .region(Region.US_EAST_1)
            .build();

        sendEmail(pinpoint, subject, senderAddress, toAddress);
    }
}
```

```
        System.out.println("Email was sent");
        pinpoint.close();
    }

    public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
subject, String senderAddress, String toAddress) {
        try {
            Content content = Content.builder()
                .data(body)
                .build();

            Body messageBody = Body.builder()
                .text(content)
                .build();

            Message message = Message.builder()
                .body(messageBody)
                .subject(Content.builder().data(subject).build())
                .build();

            Destination destination = Destination.builder()
                .toAddresses(toAddress)
                .build();

            EmailContent emailContent = EmailContent.builder()
                .simple(message)
                .build();

            SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
                .fromEmailAddress(senderAddress)
                .destination(destination)
                .content(emailContent)
                .build();

            pinpointEmailClient.sendEmail(sendEmailRequest);
            System.out.println("Message Sent");

        } catch (PinpointException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

使用 CC 值发送电子邮件。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpointemail.PinpointEmailClient;
import software.amazon.awssdk.services.pinpointemail.model.Body;
import software.amazon.awssdk.services.pinpointemail.model.Content;
import software.amazon.awssdk.services.pinpointemail.model.Destination;
import software.amazon.awssdk.services.pinpointemail.model.EmailContent;
import software.amazon.awssdk.services.pinpointemail.model.Message;
import software.amazon.awssdk.services.pinpointemail.model.SendEmailRequest;
import java.util.ArrayList;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendEmailMessageCC {

    // The body of the email.
    static final String body = """
        Amazon Pinpoint test (AWS SDK for Java 2.x)

        This email was sent through the Amazon Pinpoint Email API using the AWS
        SDK for Java 2.x

        """;

    public static void main(String[] args) {
        final String usage = """

            Usage:    <subject> <senderAddress> <toAddress> <ccAddress>

            Where:
                subject - The email subject to use.
                senderAddress - The from address. This address has to be verified
                in Amazon Pinpoint in the region you're using to send email\s
        """;
    }
}
```

```
        toAddress - The to address. This address has to be verified in
        Amazon Pinpoint in the region you're using to send email\s
        ccAddress - The CC address.
        """;

    if (args.length != 4) {
        System.out.println(usage);
        System.exit(1);
    }

    String subject = args[0];
    String senderAddress = args[1];
    String toAddress = args[2];
    String ccAddress = args[3];

    System.out.println("Sending a message");
    PinpointEmailClient pinpoint = PinpointEmailClient.builder()
        .region(Region.US_EAST_1)
        .build();

    ArrayList<String> ccList = new ArrayList<>();
    ccList.add(ccAddress);
    sendEmail(pinpoint, subject, senderAddress, toAddress, ccList);
    pinpoint.close();
}

public static void sendEmail(PinpointEmailClient pinpointEmailClient, String
subject, String senderAddress, String toAddress, ArrayList<String> ccAddresses)
{
    try {
        Content content = Content.builder()
            .data(body)
            .build();

        Body messageBody = Body.builder()
            .text(content)
            .build();

        Message message = Message.builder()
            .body(messageBody)
            .subject(Content.builder().data(subject).build())
            .build();

        Destination destination = Destination.builder()
```

```
        .toAddresses(toAddress)
        .ccAddresses(ccAddresses)
        .build();

    EmailContent emailContent = EmailContent.builder()
        .simple(message)
        .build();

    SendEmailRequest sendEmailRequest = SendEmailRequest.builder()
        .fromEmailAddress(senderAddress)
        .destination(destination)
        .content(emailContent)
        .build();

    pinpointEmailClient.sendEmail(sendEmailRequest);
    System.out.println("Message Sent");

} catch (PinpointException e) {
    // Handle exception
    e.printStackTrace();
}
}
```

发送短信。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessageResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
```

```

* environment, including your credentials.
*
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class SendMessage {

    // The type of SMS message that you want to send. If you plan to send
    // time-sensitive content, specify TRANSACTIONAL. If you plan to send
    // marketing-related content, specify PROMOTIONAL.
    public static String messageType = "TRANSACTIONAL";

    // The registered keyword associated with the originating short code.
    public static String registeredKeyword = "myKeyword";

    // The sender ID to use when sending the message. Support for sender ID
    // varies by country or region. For more information, see
    // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-countries.html
    public static String senderId = "MySenderId";

    public static void main(String[] args) {
        final String usage = ""

            Usage:  <message> <appId> <originationNumber>
<destinationNumber>\s

            Where:
                message - The body of the message to send.
                appId - The Amazon Pinpoint project/application
ID to use when you send this message.
                originationNumber - The phone number or
short code that you specify has to be associated with your Amazon Pinpoint
account. For best results, specify long codes in E.164 format (for example,
+1-555-555-5654).
                destinationNumber - The recipient's phone
number. For best results, you should specify the phone number in E.164 format
(for example, +1-555-555-5654).\s
            """;

        if (args.length != 4) {
            System.out.println(usage);

```

```
        System.exit(1);
    }

    String message = args[0];
    String appId = args[1];
    String originationNumber = args[2];
    String destinationNumber = args[3];
    System.out.println("Sending a message");
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    sendSMSMessage(pinpoint, message, appId, originationNumber,
destinationNumber);
    pinpoint.close();
}

public static void sendSMSMessage(PinpointClient pinpoint, String
message, String appId,
    String originationNumber,
    String destinationNumber) {
    try {
        Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
        AddressConfiguration addConfig =
AddressConfiguration.builder()
            .channelType(ChannelType.SMS)
            .build();

        addressMap.put(destinationNumber, addConfig);
        SMSMessage smsMessage = SMSMessage.builder()
            .body(message)
            .messageType(messageType)
            .originationNumber(originationNumber)
            .senderId(senderId)
            .keyword(registeredKeyword)
            .build();

        // Create a DirectMessageConfiguration object.
        DirectMessageConfiguration direct =
DirectMessageConfiguration.builder()
            .smsMessage(smsMessage)
            .build();
```

```
        MessageRequest msgReq = MessageRequest.builder()
            .addresses(addressMap)
            .messageConfiguration(direct)
            .build();

        // create a SendMessagesRequest object
        SendMessagesRequest request =
SendMessagesRequest.builder()
            .applicationId(appId)
            .messageRequest(msgReq)
            .build();

        SendMessagesResponse response =
pinpoint.sendMessage(request);
        MessageResponse msg1 = response.getMessageResponse();
        Map map1 = msg1.getResult();

        // Write out the result of sendMessage.
        map1.forEach((k, v) -> System.out.println((k + ":" +
v)));

    } catch (PinpointException e) {
        System.err.println(e.getAwsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

发送批处理短信。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.DirectMessageConfiguration;
import software.amazon.awssdk.services.pinpoint.model.SMSMessage;
import software.amazon.awssdk.services.pinpoint.model.AddressConfiguration;
import software.amazon.awssdk.services.pinpoint.model.ChannelType;
import software.amazon.awssdk.services.pinpoint.model.MessageRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesRequest;
import software.amazon.awssdk.services.pinpoint.model.SendMessagesResponse;
import software.amazon.awssdk.services.pinpoint.model.MessageResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import java.util.HashMap;
```



```
import java.util.Map;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SendMessageBatch {

    // The type of SMS message that you want to send. If you plan to send
    // time-sensitive content, specify TRANSACTIONAL. If you plan to send
    // marketing-related content, specify PROMOTIONAL.
    public static String messageType = "TRANSACTIONAL";

    // The registered keyword associated with the originating short code.
    public static String registeredKeyword = "myKeyword";

    // The sender ID to use when sending the message. Support for sender ID
    // varies by country or region. For more information, see
    // https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-countries.html
    public static String senderId = "MySenderId";

    public static void main(String[] args) {
        final String usage = ""

            Usage:  <message> <appId> <originationNumber>
<destinationNumber> <destinationNumber1>\s

            Where:
                message - The body of the message to send.
                appId - The Amazon Pinpoint project/application
ID to use when you send this message.
                originationNumber - The phone number or
short code that you specify has to be associated with your Amazon Pinpoint
account. For best results, specify long codes in E.164 format (for example,
+1-555-555-5654).
                destinationNumber - The recipient's phone
number. For best results, you should specify the phone number in E.164 format
(for example, +1-555-555-5654).
```

```
        destinationNumber1 - The second recipient's
phone number. For best results, you should specify the phone number in E.164
format (for example, +1-555-555-5654).\s
        """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String message = args[0];
        String appId = args[1];
        String originationNumber = args[2];
        String destinationNumber = args[3];
        String destinationNumber1 = args[4];
        System.out.println("Sending a message");
        PinpointClient pinpoint = PinpointClient.builder()
            .region(Region.US_EAST_1)
            .build();

        sendSMSMessage(pinpoint, message, appId, originationNumber,
            destinationNumber, destinationNumber1);
        pinpoint.close();
    }

    public static void sendSMSMessage(PinpointClient pinpoint, String
message, String appId,
        String originationNumber,
        String destinationNumber, String destinationNumber1) {
        try {
            Map<String, AddressConfiguration> addressMap = new
HashMap<String, AddressConfiguration>();
            AddressConfiguration addConfig =
AddressConfiguration.builder()
                .channelType(ChannelType.SMS)
                .build();

            // Add an entry to the Map object for each number to whom
you want to send a
            // message.
            addressMap.put(destinationNumber, addConfig);
            addressMap.put(destinationNumber1, addConfig);
            SMSMessage smsMessage = SMSMessage.builder()
                .body(message)
```

```
                .messageType(messageType)
                .originationNumber(originationNumber)
                .senderId(senderId)
                .keyword(registeredKeyword)
                .build();

        // Create a DirectMessageConfiguration object.
        DirectMessageConfiguration direct =
DirectMessageConfiguration.builder()
                .smsMessage(smsMessage)
                .build();

        MessageRequest msgReq = MessageRequest.builder()
                .addresses(addressMap)
                .messageConfiguration(direct)
                .build();

        // Create a SendMessagesRequest object.
        SendMessagesRequest request =
SendMessagesRequest.builder()
                .applicationId(appId)
                .messageRequest(msgReq)
                .build();

        SendMessagesResponse response =
pinpoint.sendMessage(request);
        MessageResponse msg1 = response.getMessageResponse();
        Map map1 = msg1.getResult();

        // Write out the result of sendMessage.
        map1.forEach((k, v) -> System.out.println((k + ":" +
v)));

        } catch (PinpointException e) {
            System.err.println(e.getAwsErrorDetails().getErrorMessage());
            System.exit(1);
        }
    }
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考 [SendMessages](#) 中的。

## JavaScript

### 适用于 JavaScript (v3) 的软件开发工具包

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

在单独的模块中创建客户端并将其导出。

```
import { PinpointClient } from "@aws-sdk/client-pinpoint";
// Set the AWS Region.
const REGION = "us-east-1";
//Set the MediaConvert Service Object
const pinClient = new PinpointClient({ region: REGION });
export { pinClient };
```

发送电子邮件。

```
// Import required AWS SDK clients and commands for Node.js
import { SendMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "./libs/pinClient.js";

// The FromAddress must be verified in SES.
const fromAddress = "FROM_ADDRESS";
const toAddress = "TO_ADDRESS";
const projectId = "PINPOINT_PROJECT_ID";

// The subject line of the email.
var subject = "Amazon Pinpoint Test (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
Node.js.
For more information, see https://aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
```

```
var body_html = `
```

```
const run = async () => {
  try {
    const data = await pinClient.send(new SendMessagesCommand(params));

    const {
      MessageResponse: { Result },
    } = data;

    const recipientResult = Result[toAddress];

    if (recipientResult.StatusCode !== 200) {
      throw new Error(recipientResult.StatusMessage);
    } else {
      console.log(recipientResult.MessageId);
    }
  } catch (err) {
    console.log(err.message);
  }
};

run();
```

发送短信。

```
// Import required AWS SDK clients and commands for Node.js
import { SendMessagesCommand } from "@aws-sdk/client-pinpoint";
import { pinClient } from "../libs/pinClient.js";

("use strict");

/* The phone number or short code to send the message from. The phone number
   or short code that you specify has to be associated with your Amazon Pinpoint
   account. For best results, specify long codes in E.164 format. */
const originationNumber = "SENDER_NUMBER"; //e.g., +1XXXXXXXXXX

// The recipient's phone number. For best results, you should specify the phone
   number in E.164 format.
const destinationNumber = "RECEIVER_NUMBER"; //e.g., +1XXXXXXXXXX

// The content of the SMS message.
```

```
const message =
  "This message was sent through Amazon Pinpoint " +
  "using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
  "opt out.";

/*The Amazon Pinpoint project/application ID to use when you send this message.
Make sure that the SMS channel is enabled for the project or application
that you choose.*/
const projectId = "PINPOINT_PROJECT_ID"; //e.g., XXXXXXXX66e4e9986478cXXXXXXXXXX

/* The type of SMS message that you want to send. If you plan to send
time-sensitive content, specify TRANSACTIONAL. If you plan to send
marketing-related content, specify PROMOTIONAL.*/
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";

/* The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
countries.html.*/

var senderId = "MySenderId";

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: projectId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
    MessageConfiguration: {
      SMSMessage: {
        Body: message,
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
      },
    },
  },
},
```

```
};

const run = async () => {
  try {
    const data = await pinClient.send(new SendMessagesCommand(params));
    return data; // For unit tests.
    console.log(
      "Message sent! " +
      data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"]
    );
  } catch (err) {
    console.log(err);
  }
};
run();
```

- 有关 API 的详细信息，请参阅 AWS SDK for JavaScript API 参考 [SendMessages](#) 中的。适用于 JavaScript (v2) 的软件开发工具包

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

发送电子邮件。

```
"use strict";

const AWS = require("aws-sdk");

// The AWS Region that you want to use to send the email. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/
const aws_region = "us-west-2";

// The "From" address. This address has to be verified in Amazon Pinpoint
// in the region that you use to send email.
const senderAddress = "sender@example.com";
```



```
// The address on the "To" line. If your Amazon Pinpoint account is in
// the sandbox, this address also has to be verified.
var toAddress = "recipient@example.com";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
const appId = "ce796be37f32f178af652b26eexample";

// The subject line of the email.
var subject = "Amazon Pinpoint (AWS SDK for JavaScript in Node.js)";

// The email body for recipients with non-HTML email clients.
var body_text = `Amazon Pinpoint Test (SDK for JavaScript in Node.js)
-----
This email was sent with Amazon Pinpoint using the AWS SDK for JavaScript in
Node.js.
For more information, see https://aws.amazon.com/sdk-for-node-js/`;

// The body of the email for recipients whose email clients support HTML content.
var body_html = `
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for JavaScript in Node.js)</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>the Amazon Pinpoint API</a> using
    the
    <a href='https://aws.amazon.com/sdk-for-node-js/'>
      AWS SDK for JavaScript in Node.js</a>.</p>
</body>
</html>`;

// The character encoding the you want to use for the subject line and
// message body of the email.
var charset = "UTF-8";

// Specify that you're using a shared credentials file.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
```

```
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: appId,
  MessageRequest: {
    Addresses: {
      [toAddress]: {
        ChannelType: "EMAIL",
      },
    },
    MessageConfiguration: {
      EmailMessage: {
        FromAddress: senderAddress,
        SimpleEmail: {
          Subject: {
            Charset: charset,
            Data: subject,
          },
          HtmlPart: {
            Charset: charset,
            Data: body_html,
          },
          TextPart: {
            Charset: charset,
            Data: body_text,
          },
        },
      },
    },
  },
};

//Try to send the email.
pinpoint.sendMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
  } else {
    console.log(
      "Email sent! Message ID: ",
      data["MessageResponse"]["Result"][toAddress]["MessageId"]
    );
  }
}
```

```
});
```

发送短信。

```
"use strict";

var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the message. For a list of
// AWS Regions where the Amazon Pinpoint API is available, see
// https://docs.aws.amazon.com/pinpoint/latest/apireference/.
var aws_region = "us-east-1";

// The phone number or short code to send the message from. The phone number
// or short code that you specify has to be associated with your Amazon Pinpoint
// account. For best results, specify long codes in E.164 format.
var originationNumber = "+12065550199";

// The recipient's phone number. For best results, you should specify the
// phone number in E.164 format.
var destinationNumber = "+14255550142";

// The content of the SMS message.
var message =
  "This message was sent through Amazon Pinpoint " +
  "using the AWS SDK for JavaScript in Node.js. Reply STOP to " +
  "opt out.";

// The Amazon Pinpoint project/application ID to use when you send this message.
// Make sure that the SMS channel is enabled for the project or application
// that you choose.
var applicationId = "ce796be37f32f178af652b26eexample";

// The type of SMS message that you want to send. If you plan to send
// time-sensitive content, specify TRANSACTIONAL. If you plan to send
// marketing-related content, specify PROMOTIONAL.
var messageType = "TRANSACTIONAL";

// The registered keyword associated with the originating short code.
var registeredKeyword = "myKeyword";
```

```
// The sender ID to use when sending the message. Support for sender ID
// varies by country or region. For more information, see
// https://docs.aws.amazon.com/pinpoint/latest/userguide/channels-sms-
// countries.html
var senderId = "MySenderId";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpoint = new AWS.Pinpoint();

// Specify the parameters to pass to the API.
var params = {
  ApplicationId: applicationId,
  MessageRequest: {
    Addresses: {
      [destinationNumber]: {
        ChannelType: "SMS",
      },
    },
    MessageConfiguration: {
      SMSMessage: {
        Body: message,
        Keyword: registeredKeyword,
        MessageType: messageType,
        OriginationNumber: originationNumber,
        SenderId: senderId,
      },
    },
  },
};

//Try to send the message.
pinpoint.sendMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
  }
});
```

```
// Otherwise, show the unique ID for the message.
} else {
  console.log(
    "Message sent! " +
      data["MessageResponse"]["Result"][destinationNumber]["StatusMessage"]
  );
}
});
```

- 有关 API 的详细信息，请参阅 AWS SDK for JavaScript API 参考 [SendMessages](#) 中的。

## Kotlin

### 适用于 Kotlin 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
/**
Before running this Kotlin code example, set up your development environment,
including your credentials.

For more information, see the following documentation topic:
https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
*/

val body: String = """
    Amazon Pinpoint test (AWS SDK for Kotlin)

    This email was sent through the Amazon Pinpoint Email API using the AWS
    SDK for Kotlin.

    """.trimIndent()

suspend fun main(args: Array<String>) {
    val usage = """
    Usage:
```

```
<subject> <appId> <senderAddress> <toAddress>
```

Where:

subject - The email subject to use.

senderAddress - The from address. This address has to be verified in Amazon Pinpoint in the region you're using to send email

toAddress - The to address. This address has to be verified in Amazon Pinpoint in the region you're using to send email

```
""
```

```
if (args.size != 3) {  
    println(usage)  
    exitProcess(0)  
}
```

```
val subject = args[0]  
val senderAddress = args[1]  
val toAddress = args[2]  
sendEmail(subject, senderAddress, toAddress)
```

```
}
```

```
suspend fun sendEmail(subjectVal: String?, senderAddress: String, toAddressVal:  
String) {
```

```
    var content = Content {  
        data = body  
    }
```

```
    val messageBody = Body {  
        text = content  
    }
```

```
    val subContent = Content {  
        data = subjectVal  
    }
```

```
    val message = Message {  
        body = messageBody  
        subject = subContent  
    }
```

```
    val destination0b = Destination {  
        toAddresses = listOf(toAddressVal)  
    }
```

```
val emailContent = EmailContent {
    simple = message
}

val sendEmailRequest = SendEmailRequest {
    fromEmailAddress = senderAddress
    destination = destinationObj
    this.content = emailContent
}

PinpointEmailClient { region = "us-east-1" }.use { pinpointemail ->
    pinpointemail.sendEmail(sendEmailRequest)
    println("Message Sent")
}
}
```

- 有关 API 的详细信息，请参阅适用[SendMessages](#)于 Kotlin 的 AWS SDK API 参考。

## Python

### SDK for Python (Boto3)

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

发送电子邮件。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_email_message(
    pinpoint_client,
    app_id,
```

```

sender,
to_addresses,
char_set,
subject,
html_message,
text_message,
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project ID to use when you send this
message.
    :param sender: The "From" address. This address must be verified in
        Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
account
        is in the sandbox, these addresses must be verified.
    :param char_set: The character encoding to use for the subject line and
message
        body of the email.
    :param subject: The subject line of the email.
    :param html_message: The body of the email for recipients whose email clients
can
        display HTML content.
    :param text_message: The body of the email for recipients whose email clients
        don't support HTML content.
    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=app_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
to_addresses
                },
                "MessageConfiguration": {
                    "EmailMessage": {
                        "FromAddress": sender,
                        "SimpleEmail": {
                            "Subject": {"Charset": char_set, "Data": subject},
                            "HtmlPart": {"Charset": char_set, "Data":
html_message},

```



```

        "TextPart": {"Charset": char_set, "Data":
text_message},
    },
    },
    },
)
except ClientError:
    logger.exception("Couldn't send email.")
    raise
else:
    return {
        to_address: message["MessageId"]
        for to_address, message in response["MessageResponse"]
["Result"].items()
    }

def main():
    app_id = "ce796be37f32f178af652b26eexample"
    sender = "sender@example.com"
    to_address = "recipient@example.com"
    char_set = "UTF-8"
    subject = "Amazon Pinpoint Test (SDK for Python (Boto3))"
    text_message = """Amazon Pinpoint Test (SDK for Python)
-----
This email was sent with Amazon Pinpoint using the AWS SDK for Python
(Boto3).
For more information, see https://aws.amazon.com/sdk-for-python/
"""
    html_message = """<html>
<head></head>
<body>
  <h1>Amazon Pinpoint Test (SDK for Python (Boto3))</h1>
  <p>This email was sent with
    <a href='https://aws.amazon.com/pinpoint/'>Amazon Pinpoint</a> using the
    <a href='https://aws.amazon.com/sdk-for-python/'>
      AWS SDK for Python (Boto3)</a>.</p>
</body>
</html>
"""

    print("Sending email.")
    message_ids = send_email_message(

```

```
        boto3.client("pinpoint"),
        app_id,
        sender,
        [to_address],
        char_set,
        subject,
        html_message,
        text_message,
    )
    print(f"Message sent! Message IDs: {message_ids}")

if __name__ == "__main__":
    main()
```

发送短信。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_sms_message(
    pinpoint_client,
    app_id,
    origination_number,
    destination_number,
    message,
    message_type,
):
    """
    Sends an SMS message with Amazon Pinpoint.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param app_id: The Amazon Pinpoint project/application ID to use when you
    send
                       this message. The SMS channel must be enabled for the project
    or
                       application.
```

```

:param destination_number: The recipient's phone number in E.164 format.
:param origination_number: The phone number to send the message from. This
phone
                                number must be associated with your Amazon
Pinpoint
                                account and be in E.164 format.
:param message: The content of the SMS message.
:param message_type: The type of SMS message that you want to send. If you
send
                                time-sensitive content, specify TRANSACTIONAL. If you
send
                                marketing-related content, specify PROMOTIONAL.
:return: The ID of the message.
"""
try:
    response = pinpoint_client.send_messages(
        ApplicationId=app_id,
        MessageRequest={
            "Addresses": {destination_number: {"ChannelType": "SMS"}},
            "MessageConfiguration": {
                "SMSMessage": {
                    "Body": message,
                    "MessageType": message_type,
                    "OriginationNumber": origination_number,
                }
            },
        },
    )
except ClientError:
    logger.exception("Couldn't send message.")
    raise
else:
    return response["MessageResponse"]["Result"][destination_number]
["MessageId"]

def main():
    app_id = "ce796be37f32f178af652b26eexample"
    origination_number = "+12065550199"
    destination_number = "+14255550142"
    message = (
        "This is a sample message sent from Amazon Pinpoint by using the AWS SDK
for "
        "Python (Boto 3).")

```

```
)
message_type = "TRANSACTIONAL"

print("Sending SMS message.")
message_id = send_sms_message(
    boto3.client("pinpoint"),
    app_id,
    origination_number,
    destination_number,
    message,
    message_type,
)
print(f"Message sent! Message ID: {message_id}.")

if __name__ == "__main__":
    main()
```

- 有关 API 的详细信息，请参阅适用[SendMessages](#)于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 使用软件开发工具包通过 Amazon Pinpoint 发送模板化电子邮件和短信 AWS

以下代码示例演示了如何通过 Amazon Pinpoint 发送模板电子邮件和文本消息。

### Python

#### SDK for Python (Boto3)

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

使用现有电子邮件模板发送电子邮件消息。

```
import logging
```

```
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_templated_email_message(
    pinpoint_client, project_id, sender, to_addresses, template_name,
    template_version
):
    """
    Sends an email message with HTML and plain text versions.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: The Amazon Pinpoint project ID to use when you send this
    message.
    :param sender: The "From" address. This address must be verified in
    Amazon Pinpoint in the AWS Region you're using to send email.
    :param to_addresses: The addresses on the "To" line. If your Amazon Pinpoint
    account is in the sandbox, these addresses must be
    verified.
    :param template_name: The name of the email template to use when sending the
    message.
    :param template_version: The version number of the message template.

    :return: A dict of to_addresses and their message IDs.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {
                    to_address: {"ChannelType": "EMAIL"} for to_address in
                    to_addresses
                },
                "MessageConfiguration": {"EmailMessage": {"FromAddress":
                    sender}},
                "TemplateConfiguration": {
                    "EmailTemplate": {
                        "Name": template_name,
                        "Version": template_version,
                    }
                },
            },
        ),
```

```
    )
except ClientError:
    logger.exception("Couldn't send email.")
    raise
else:
    return {
        to_address: message["MessageId"]
        for to_address, message in response["MessageResponse"]
["Result"].items()
    }

def main():
    project_id = "296b04b342374fceb661bf494example"
    sender = "sender@example.com"
    to_addresses = ["recipient@example.com"]
    template_name = "My_Email_Template"
    template_version = "1"

    print("Sending email.")
    message_ids = send_templated_email_message(
        boto3.client("pinpoint"),
        project_id,
        sender,
        to_addresses,
        template_name,
        template_version,
    )
    print(f"Message sent! Message IDs: {message_ids}")

if __name__ == "__main__":
    main()
```

使用现有短信模板发送短信。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)
```

```
def send_templated_sms_message(
    pinpoint_client,
    project_id,
    destination_number,
    message_type,
    origination_number,
    template_name,
    template_version,
):
    """
    Sends an SMS message to a specific phone number using a pre-defined template.

    :param pinpoint_client: A Boto3 Pinpoint client.
    :param project_id: An Amazon Pinpoint project (application) ID.
    :param destination_number: The phone number to send the message to.
    :param message_type: The type of SMS message (promotional or transactional).
    :param origination_number: The phone number that the message is sent from.
    :param template_name: The name of the SMS template to use when sending the
    message.
    :param template_version: The version number of the message template.

    :return The ID of the message.
    """
    try:
        response = pinpoint_client.send_messages(
            ApplicationId=project_id,
            MessageRequest={
                "Addresses": {destination_number: {"ChannelType": "SMS"}},
                "MessageConfiguration": {
                    "SMSMessage": {
                        "MessageType": message_type,
                        "OriginationNumber": origination_number,
                    }
                },
                "TemplateConfiguration": {
                    "SMSTemplate": {"Name": template_name, "Version":
template_version}
                },
            },
        )

    except ClientError:
        logger.exception("Couldn't send message.")
```

```
        raise
    else:
        return response["MessageResponse"]["Result"][destination_number]
["MessageId"]

def main():
    region = "us-east-1"
    origination_number = "+18555550001"
    destination_number = "+14255550142"
    project_id = "7353f53e6885409fa32d07cedexample"
    message_type = "TRANSACTIONAL"
    template_name = "My_SMS_Template"
    template_version = "1"
    message_id = send_templated_sms_message(
        boto3.client("pinpoint", region_name=region),
        project_id,
        destination_number,
        message_type,
        origination_number,
        template_name,
        template_version,
    )
    print(f"Message sent! Message ID: {message_id}.")

if __name__ == "__main__":
    main()
```

- 有关 API 的详细信息，请参阅适用[SendMessages](#)于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。


## 更新 Amazon Pinpoint 端点

以下代码示例演示了如何更新端点。



## Java

## 适用于 Java 2.x 的 SDK

 Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.EndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.EndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointRequest;
import software.amazon.awssdk.services.pinpoint.model.GetEndpointResponse;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.EndpointDemographic;
import software.amazon.awssdk.services.pinpoint.model.EndpointLocation;
import software.amazon.awssdk.services.pinpoint.model.EndpointUser;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.List;
import java.util.UUID;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Date;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class UpdateEndpoint {
    public static void main(String[] args) {
```

```
final String usage = ""

    Usage: <appId>

    Where:
        appId - The ID of the application to create an endpoint for.

    """;

if (args.length != 1) {
    System.out.println(usage);
    System.exit(1);
}

String appId = args[0];
PinpointClient pinpoint = PinpointClient.builder()
    .region(Region.US_EAST_1)
    .build();

EndpointResponse response = createEndpoint(pinpoint, appId);
System.out.println("Got Endpoint: " + response.id());
pinpoint.close();
}

public static EndpointResponse createEndpoint(PinpointClient client, String
appId) {
    String endpointId = UUID.randomUUID().toString();
    System.out.println("Endpoint ID: " + endpointId);

    try {
        EndpointRequest endpointRequest = createEndpointRequestData();
        UpdateEndpointRequest updateEndpointRequest =
UpdateEndpointRequest.builder()
            .applicationId(appId)
            .endpointId(endpointId)
            .endpointRequest(endpointRequest)
            .build();

        UpdateEndpointResponse updateEndpointResponse =
client.updateEndpoint(updateEndpointRequest);
        System.out.println("Update Endpoint Response: " +
updateEndpointResponse.messageBody());

        GetEndpointRequest getEndpointRequest = GetEndpointRequest.builder()
```

```
        .applicationId(appId)
        .endpointId(endpointId)
        .build();

        GetEndpointResponse getEndpointResponse =
client.getEndpoint(getEndpointRequest);
        System.out.println(getEndpointResponse.endpointResponse().address());

System.out.println(getEndpointResponse.endpointResponse().channelType());

System.out.println(getEndpointResponse.endpointResponse().applicationId());

System.out.println(getEndpointResponse.endpointResponse().endpointStatus());

System.out.println(getEndpointResponse.endpointResponse().requestId());
        System.out.println(getEndpointResponse.endpointResponse().user());

        return getEndpointResponse.endpointResponse();

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return null;
}

private static EndpointRequest createEndpointRequestData() {
    try {
        List<String> favoriteTeams = new ArrayList<>();
        favoriteTeams.add("Lakers");
        favoriteTeams.add("Warriors");
        HashMap<String, List<String>> customAttributes = new HashMap<>();
        customAttributes.put("team", favoriteTeams);

        EndpointDemographic demographic = EndpointDemographic.builder()
            .appVersion("1.0")
            .make("apple")
            .model("iPhone")
            .modelVersion("7")
            .platform("ios")
            .platformVersion("10.1.1")
            .timezone("America/Los_Angeles")
            .build();
    }
}
```

```
EndpointLocation location = EndpointLocation.builder()
    .city("Los Angeles")
    .country("US")
    .latitude(34.0)
    .longitude(-118.2)
    .postalCode("90068")
    .region("CA")
    .build();

Map<String, Double> metrics = new HashMap<>();
metrics.put("health", 100.00);
metrics.put("luck", 75.00);

EndpointUser user = EndpointUser.builder()
    .userId(UUID.randomUUID().toString())
    .build();

DateFormat df = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm'Z'"); //
Quoted "Z" to indicate UTC, no timezone                                     //
offset                                                                    //

String nowAsISO = df.format(new Date());

return EndpointRequest.builder()
    .address(UUID.randomUUID().toString())
    .attributes(customAttributes)
    .channelType("APNS")
    .demographic(demographic)
    .effectiveDate(nowAsISO)
    .location(location)
    .metrics(metrics)
    .optOut("NONE")
    .requestId(UUID.randomUUID().toString())
    .user(user)
    .build();

} catch (PinpointException e) {
    System.err.println(e.awsErrorDetails().errorMessage());
    System.exit(1);
}
return null;
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[UpdateEndpoint](#)中的。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 更新 Amazon Pinpoint 渠道

以下代码示例演示了如何更新渠道。

### CLI

#### AWS CLI

检索有关应用程序的短信渠道的状态和设置的信息

以下 `get-sms-channel` 示例检索应用程序的短信渠道的状态和设置。

```
aws pinpoint get-sms-channel \  
  --application-id 6e0b7591a90841d2b5d93fa11143e5a7 \  
  --region us-east-1
```

输出：

```
{  
  "SMSChannelResponse": {  
    "ApplicationId": "6e0b7591a90841d2b5d93fa11143e5a7",  
    "CreationDate": "2019-10-08T18:39:18.511Z",  
    "Enabled": true,  
    "Id": "sms",  
    "IsArchived": false,  
    "LastModifiedDate": "2019-10-08T18:39:18.511Z",  
    "Platform": "SMS",  
    "PromotionalMessagesPerSecond": 20,  
    "TransactionalMessagesPerSecond": 20,  
    "Version": 1  
  }  
}
```

- 有关 API 的详细信息，请参阅 AWS CLI 命令参考[GetSmsChannel](#)中的。

## Java

### 适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpoint.PinpointClient;
import software.amazon.awssdk.services.pinpoint.model.SMSChannelResponse;
import software.amazon.awssdk.services.pinpoint.model.GetSmsChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.PinpointException;
import software.amazon.awssdk.services.pinpoint.model.SMSChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateSmsChannelRequest;
import software.amazon.awssdk.services.pinpoint.model.UpdateSmsChannelResponse;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class UpdateChannel {
    public static void main(String[] args) {
        final String usage = ""

                Usage: CreateChannel <appId>

                Where:
                appId - The name of the application whose channel is updated.

                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
    }
}
```

```
    }

    String appId = args[0];
    PinpointClient pinpoint = PinpointClient.builder()
        .region(Region.US_EAST_1)
        .build();

    SMSChannelResponse getResponse = getSMSChannel(pinpoint, appId);
    toggleSmsChannel(pinpoint, appId, getResponse);
    pinpoint.close();
}

private static SMSChannelResponse getSMSChannel(PinpointClient client, String
appId) {
    try {
        GetSmsChannelRequest request = GetSmsChannelRequest.builder()
            .applicationId(appId)
            .build();

        SMSChannelResponse response =
client.getSmsChannel(request).smsChannelResponse();
        System.out.println("Channel state is " + response.enabled());
        return response;

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
    return null;
}

private static void toggleSmsChannel(PinpointClient client, String appId,
SMSChannelResponse getResponse) {
    boolean enabled = !getResponse.enabled();
    try {
        SMSChannelRequest request = SMSChannelRequest.builder()
            .enabled(enabled)
            .build();

        UpdateSmsChannelRequest updateRequest =
UpdateSmsChannelRequest.builder()
            .smsChannelRequest(request)
            .applicationId(appId)
            .build();
```

```
        UpdateSmsChannelResponse result =
client.updateSmsChannel(updateRequest);
        System.out.println("Channel state: " +
result.smsChannelResponse().enabled());

    } catch (PinpointException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[GetSmsChannel](#)中的。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

## 使用软件开发工具包的 Amazon Pinpoint 短信和语音 API 的代码示例 AWS

以下代码示例展示了如何将 Amazon Pinpoint 短信和语音 API 与 AWS 软件开发套件 (SDK) 一起使用。

操作是大型程序的代码摘录，必须在上下文中运行。您可以通过操作了解如何调用单个服务函数，还可以通过函数相关场景和跨服务示例的上下文查看操作。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

### 代码示例

- [使用软件开发工具包对 Amazon Pinpoint 短信和语音 API 执行的操作 AWS](#)
- [使用软件开发工具包通过 Amazon Pinpoint 短信和语音 API 发送语音消息 AWS](#)



## 使用软件开发工具包对 Amazon Pinpoint 短信和语音 API 执行的操作 AWS

以下代码示例演示了如何使用软件开发工具包执行单个 Amazon Pinpoint 短信和语音 API 操作。AWS 这些代码节选调用了 Amazon Pinpoint SMS 和 Voice API，是必须在上下文中运行的较大型程序的代码节选。每个示例都包含一个指向的链接 GitHub，您可以在其中找到有关设置和运行代码的说明。

以下示例仅包括最常用的操作。有关完整列表，请参阅 [Amazon Pinpoint SMS 和 Voice API 参考](#)。

### 示例

- [使用软件开发工具包通过 Amazon Pinpoint 短信和语音 API 发送语音消息 AWS](#)

## 使用软件开发工具包通过 Amazon Pinpoint 短信和语音 API 发送语音消息 AWS

以下代码示例演示了如何通过 Amazon Pinpoint 短信和语音 API 发送语音消息。

### Java

适用于 Java 2.x 的 SDK

#### Note

还有更多相关信息 GitHub。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import software.amazon.awssdk.core.client.config.ClientOverrideConfiguration;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.pinpointsmsvoice.PinpointSmsVoiceClient;
import software.amazon.awssdk.services.pinpointsmsvoice.model.SSMLMessageType;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.VoiceMessageContent;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.SendVoiceMessageRequest;
import
    software.amazon.awssdk.services.pinpointsmsvoice.model.PinpointSmsVoiceException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
```

```
* Before running this Java V2 code example, set up your development
* environment, including your credentials.
*
* For more information, see the following documentation topic:
*
* https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
*/
public class SendVoiceMessage {

    // The Amazon Polly voice that you want to use to send the message. For a
    list
    // of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
    static final String voiceName = "Matthew";

    // The language to use when sending the message. For a list of supported
    // languages, see
    // https://docs.aws.amazon.com/polly/latest/dg/SupportedLanguage.html
    static final String languageCode = "en-US";

    // The content of the message. This example uses SSML to customize and
    control
    // certain aspects of the message, such as by adding pauses and changing
    // phonation. The message can't contain any line breaks.
    static final String ssmlMessage = "<speak>This is a test message sent
    from "

        + "<emphasis>Amazon Pinpoint</emphasis> "
        + "using the <break strength='weak'>AWS "
        + "SDK for Java. "
        + "<amazon:effect phonation='soft'>Thank "
        + "you for listening.</amazon:effect></speak>";

    public static void main(String[] args) {

        final String usage = ""

            Usage:  <originationNumber> <destinationNumber>

\s

        Where:
            originationNumber - The phone number or
            short code that you specify has to be associated with your Amazon Pinpoint
```

account. For best results, specify long codes in E.164 format (for example, +1-555-555-5654).

destinationNumber - The recipient's phone number. For best results, you should specify the phone number in E.164 format (for example, +1-555-555-5654).\s

```

        """;

    if (args.length != 2) {
        System.out.println(usage);
        System.exit(1);
    }

    String originationNumber = args[0];
    String destinationNumber = args[1];
    System.out.println("Sending a voice message");

    // Set the content type to application/json.
    List<String> listVal = new ArrayList<>();
    listVal.add("application/json");
    Map<String, List<String>> values = new HashMap<>();
    values.put("Content-Type", listVal);

    ClientOverrideConfiguration config2 =
ClientOverrideConfiguration.builder()
        .headers(values)
        .build();

    PinpointSmsVoiceClient client = PinpointSmsVoiceClient.builder()
        .overrideConfiguration(config2)
        .region(Region.US_EAST_1)
        .build();

    sendVoiceMsg(client, originationNumber, destinationNumber);
    client.close();
}

public static void sendVoiceMsg(PinpointSmsVoiceClient client, String
originationNumber,
    String destinationNumber) {
    try {
        SSMLMessageType ssmlMessageType =
SSMLMessageType.builder()
            .languageCode(languageCode)
            .text(ssmlMessage)

```

```
                .voiceId(voiceName)
                .build();

        VoiceMessageContent content =
VoiceMessageContent.builder()
                .ssmlMessage(ssmlMessageType)
                .build();

        SendVoiceMessageRequest voiceMessageRequest =
SendVoiceMessageRequest.builder()

        .destinationPhoneNumber(destinationNumber)

        .originationPhoneNumber(originationNumber)
                .content(content)
                .build();

        client.sendVoiceMessage(voiceMessageRequest);
        System.out.println("The message was sent successfully.");

    } catch (PinpointSmsVoiceException e) {
        System.err.println(e.awsErrorDetails().errorMessage());
        System.exit(1);
    }
}
}
```

- 有关 API 的详细信息，请参阅 AWS SDK for Java 2.x API 参考[SendVoiceMessage](#)中的。

## JavaScript

### 适用于 JavaScript (v2) 的软件开发工具包

#### Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
"use strict";
```

```
var AWS = require("aws-sdk");

// The AWS Region that you want to use to send the voice message. For a list of
// AWS Regions where the Amazon Pinpoint SMS and Voice API is available, see
// https://docs.aws.amazon.com/pinpoint-sms-voice/latest/APIReference/
var aws_region = "us-east-1";

// The phone number that the message is sent from. The phone number that you
// specify has to be associated with your Amazon Pinpoint account. For best
// results, you
// should specify the phone number in E.164 format.
var originationNumber = "+12065550110";

// The recipient's phone number. For best results, you should specify the phone
// number in E.164 format.
var destinationNumber = "+12065550142";

// The language to use when sending the message. For a list of supported
// languages, see https://docs.aws.amazon.com/polly/latest/dg/
// SupportedLanguage.html
var languageCode = "en-US";

// The Amazon Polly voice that you want to use to send the message. For a list
// of voices, see https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
var voiceId = "Matthew";

// The content of the message. This example uses SSML to customize and control
// certain aspects of the message, such as the volume or the speech rate.
// The message can't contain any line breaks.
var ssmlMessage =
  "<speak>" +
  "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> " +
  "using the <break strength='weak'>AWS SDK for JavaScript in Node.js. " +
  "<amazon:effect phonation='soft'>Thank you for listening." +
  "</amazon:effect>" +
  "</speak>";

// The phone number that you want to appear on the recipient's device. The phone
// number that you specify has to be associated with your Amazon Pinpoint
// account.
var callerId = "+12065550199";

// The configuration set that you want to use to send the message.
```

```
var configurationSet = "ConfigSet";

// Specify that you're using a shared credentials file, and optionally specify
// the profile that you want to use.
var credentials = new AWS.SharedIniFileCredentials({ profile: "default" });
AWS.config.credentials = credentials;

// Specify the region.
AWS.config.update({ region: aws_region });

//Create a new Pinpoint object.
var pinpointSMSVoice = new AWS.PinpointSMSVoice();


var params = {
  CallerId: callerId,
  ConfigurationSetName: configurationSet,
  Content: {
    SSMLMessage: {
      LanguageCode: languageCode,
      Text: ssmlMessage,
      VoiceId: voiceId,
    },
  },
  DestinationPhoneNumber: destinationNumber,
  OriginationPhoneNumber: originationNumber,
};

//Try to send the message.
pinpointSMSVoice.sendVoiceMessage(params, function (err, data) {
  // If something goes wrong, print an error message.
  if (err) {
    console.log(err.message);
    // Otherwise, show the unique ID for the message.
  } else {
    console.log("Message sent! Message ID: " + data["MessageId"]);
  }
});
```

- 有关 API 的详细信息，请参阅 AWS SDK for JavaScript API 参考 [SendVoiceMessage](#) 中的。

## Python

## SDK for Python (Boto3)

 Note

还有更多相关信息 [GitHub](#)。在 [AWS 代码示例存储库](#) 中查找完整示例，了解如何进行设置和运行。

```
import logging
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)

def send_voice_message(
    sms_voice_client,
    origination_number,
    caller_id,
    destination_number,
    language_code,
    voice_id,
    ssml_message,
):
    """
    Sends a voice message using speech synthesis provided by Amazon Polly.

    :param sms_voice_client: A Boto3 PinpointSMSVoice client.
    :param origination_number: The phone number that the message is sent from.
        The phone number must be associated with your
    Amazon
        Pinpoint account and be in E.164 format.
    :param caller_id: The phone number that you want to appear on the recipient's
        device. The phone number must be associated with your
    Amazon
        Pinpoint account and be in E.164 format.
    :param destination_number: The recipient's phone number. Specify the phone
        number in E.164 format.
    :param language_code: The language to use when sending the message.
```

```
:param voice_id: The Amazon Polly voice that you want to use to send the
message.
:param ssml_message: The content of the message. This example uses SSML to
control
                    certain aspects of the message, such as the volume and
the
                    speech rate. The message must not contain line breaks.
:return: The ID of the message.
"""
try:
    response = sms_voice_client.send_voice_message(
        DestinationPhoneNumber=destination_number,
        OriginationPhoneNumber=origination_number,
        CallerId=caller_id,
        Content={
            "SSMLMessage": {
                "LanguageCode": language_code,
                "VoiceId": voice_id,
                "Text": ssml_message,
            }
        },
    )
except ClientError:
    logger.exception(
        "Couldn't send message from %s to %s.",
        origination_number,
        destination_number,
    )
    raise
else:
    return response["MessageId"]

def main():
    origination_number = "+12065550110"
    caller_id = "+12065550199"
    destination_number = "+12065550142"
    language_code = "en-US"
    voice_id = "Matthew"
    ssml_message = (
        "<speak>"
        "This is a test message sent from <emphasis>Amazon Pinpoint</emphasis> "
        "using the <break strength='weak'/>AWS SDK for Python (Boto3). "
        "<amazon:effect phonation='soft'>Thank you for listening."
    )
```



```
        "</amazon:effect>"
        "</speak>"
    )
    print(f"Sending voice message from {origination_number} to
{destination_number}.")
    message_id = send_voice_message(
        boto3.client("pinpoint-sms-voice"),
        origination_number,
        caller_id,
        destination_number,
        language_code,
        voice_id,
        ssml_message,
    )
    print(f"Message sent!\nMessage ID: {message_id}")

if __name__ == "__main__":
    main()
```

- 有关 API 的详细信息，请参阅适用[SendVoiceMessage](#)于 Python 的AWS SDK (Boto3) API 参考。

有关 S AWS DK 开发者指南和代码示例的完整列表，请参阅[通过 AWS SDK 使用 Amazon Pinpoint](#)。本主题还包括有关入门的信息以及有关先前的 SDK 版本的详细信息。

# Amazon Pinpoint 中的安全性

云安全 AWS 是重中之重。作为 AWS 客户，您可以受益于专为满足大多数安全敏感型组织的要求而构建的数据中心和网络架构。

安全是双方 AWS 的共同责任。[责任共担模式](#)将其描述为云的安全性和云中的安全性：

- 云安全 — AWS 负责保护在 AWS 云中运行 AWS 服务的基础架构。AWS 还为您提供可以安全使用的服务。作为[AWS 合规计划](#)的一部分，第三方审计师定期测试和验证我们安全的有效性。要了解适用于 Amazon Pinpoint 的合规计划，请参阅按合规计划提供的[范围内的AWS 服务按合规计划](#)。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您的公司的要求以及适用的法律法规。

该文档帮助您了解如何在使用 Amazon Pinpoint 时应用责任共担模式。以下主题介绍如何配置 Amazon Pinpoint 以满足您的安全性和合规性目标。您还将学习如何使用其他 AWS 服务来帮助您监控和保护您的 Amazon Pinpoint 资源。

有关参考架构的更多信息，请参阅 [Amazon Pinpoint 弹性架构指南](#)。

## 主题

- [Amazon Pinpoint 中的数据保护](#)
- [Amazon Pinpoint 的身份和访问管理](#)
- [Amazon Pinpoint 中的日志记录和监控](#)
- [Amazon Pinpoint 的合规性验证](#)
- [Amazon Pinpoint 中的故障恢复能力](#)
- [Amazon Pinpoint 中的基础设施安全性](#)
- [Amazon Pinpoint 中的配置和漏洞分析](#)
- [Amazon Pinpoint 的安全最佳实践](#)

## Amazon Pinpoint 中的数据保护

AWS [分担责任模型](#)适用于 Amazon Pinpoint 中的数据保护。如本模型所述 AWS ，负责保护运行所有内容的全球基础架构 AWS Cloud。您负责维护对托管在此基础设施上的内容的控制。您

还负责您所使用的 AWS 服务的安全配置和管理任务。有关数据隐私的更多信息，请参阅[数据隐私常见问题](#)。有关欧洲数据保护的信息，请参阅 AWS 安全性博客上的[AWS 责任共担模式和 GDPR](#) 博客文章。

出于数据保护目的，我们建议您保护 AWS 账户凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。我们还建议您通过以下方式保护数据：

- 对每个账户使用 multi-factor authentication ( MFA )。
- 使用 SSL/TLS 与资源通信。AWS 我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用设置 API 和用户活动日志 AWS CloudTrail。
- 使用 AWS 加密解决方案以及其中的所有默认安全控件 AWS 服务。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果您在 AWS 通过命令行界面或 API 进行访问时需要经过 FIPS 140-2 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅[《美国联邦信息处理标准 \( FIPS \) 第 140-2 版》](#)。

我们强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您使用控制台、API 或软件开发工具包 AWS 服务使用 Amazon Pinpoint 或其他软件开发工具包 AWS CLI 的情况。AWS 在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

根据您的配置和使用服务的方式，Amazon Pinpoint 可能会为您或您的客户存储以下类型的个人数据：

### 配置数据

这包括项目配置数据，如用于定义 Amazon Pinpoint 如何以及何时通过受支持的渠道发送消息的凭证和设置，以及向其发送消息的用户分段。要发送消息，此数据可以包括用于电子邮件消息的专用 IP 地址、用于短信文本消息的短代码和发件人 ID，以及用于与推送通知服务（如 Apple Push Notification service (APNs) 和 Firebase Cloud Messaging (FCM)）进行通信的凭证。

### 用户和端点数据

这包括用于存储和管理有关 Amazon Pinpoint 项目的用户和端点数据的标准属性和自定义属性。属性可以存储有关特定用户的信息（例如用户名）或用户的特定端点的信息（例如用户的电子邮件地址、手机号码或移动设备令牌）。此数据还可以包括将 Amazon Pinpoint 项目的用户与外部系

统（如客户关系管理系统）中的用户关联的外部用户 ID。有关这些数据可能包含的内容的更多信息，请参阅《Amazon Pinpoint API 参考》中的[用户](#)和[端点](#)架构。

## 分析数据

这包括指标（也称为关键绩效指标 (KPI)）数据，可通过这类数据深入了解 Amazon Pinpoint 项目在用户参与度和购买活动等领域的绩效。另外还包括可帮助深入了解项目的用户人口统计信息的指标的数据。数据可来自用户和端点的标准和自定义属性，例如用户居住的城市，还可来自事件，例如，您为项目发送的电子邮件消息的打开和点击事件。

## 导入的数据

这包括从外部源添加或导入并在 Amazon Pinpoint 中使用的任何用户、细分和分析数据。例如，您导入到 Amazon Pinpoint 中以构建静态分段的 JSON 文件（直接通过控制台或从 Amazon S3 存储桶导入）。又如，您以编程方式添加的用以构建动态分段的端点数据、您向其发送直接消息的端点地址以及您配置应用程序以向 Amazon Pinpoint 报告的事件，等等。

## 主题

- [数据加密](#)
- [互连网络流量隐私保护](#)
- [为 Amazon Pinpoint 创建接口 VPC 端点](#)

## 数据加密

Amazon Pinpoint 数据在传输和静态时加密。当您向 Amazon Pinpoint 提交数据时，它会在接收和存储时加密数据。当您从 Amazon Pinpoint 检索数据时，它会使用当前的安全协议将数据传输给您。

### 静态加密

Amazon Pinpoint 会加密为您存储的所有数据。这包括配置数据、用户和端点数据、分析数据以及您添加或导入 Amazon Pinpoint 的任何数据。为了加密您的数据，Amazon Pinpoint 使用该服务代表您拥有和维护的内部 AWS Key Management Service (AWS KMS) 密钥。我们会定期轮换这些密钥。有关的信息 AWS KMS，请参阅《[AWS Key Management Service 开发人员指南](#)》。

### 传输中加密

Amazon Pinpoint 使用 HTTPS 和传输层安全性协议 (TLS) 1.2 或更高版本来与您的客户端和应用程序进行通信。为了与其他 AWS 服务进行通信，Amazon Pinpoint 使用 HTTPS 和 TLS 1.2。此外，当

您使用控制台、AWS 软件开发工具包或创建和管理 Amazon Pinpoint 资源时 AWS Command Line Interface，所有通信都使用 HTTPS 和 TLS 1.2 进行保护。

## 密钥管理

为了加密您的亚马逊 Pinpoint 数据，Amazon Pinpoint 使用该服务代表您拥有和维护的 AWS KMS 内部密钥。我们会定期轮换这些密钥。您不能配置和使用自己的密钥 AWS KMS 或其他密钥来加密存储在 Amazon Pinpoint 中的数据。

## 互连网络流量隐私保护

互联网流量隐私是指保护 Amazon Pinpoint 与您的本地客户端和应用程序之间，以及 Amazon Pinpoint 与同一地区 AWS 其他资源之间的连接和流量。AWS 以下功能和实践可以帮助您确保 Amazon Pinpoint 的互联网流量隐私保护。

### Amazon Pinpoint 与本地客户端和应用程序之间的流量

要在 Amazon Pinpoint 与您的本地网络上的客户端和应用程序之间建立私有连接，您可以使用 AWS Direct Connect。这使您能够使用标准的光纤以太网电缆将您的网络链接到一个 AWS Direct Connect 位置。电缆的一端连接您的路由器，另一端连接到 AWS Direct Connect 路由器。有关更多信息，请参阅 AWS Direct Connect 《用户指南》中的 [什么是 AWS Direct Connect ?](#)。

为了有助于通过已发布的 API 安全地访问 Amazon Pinpoint，我们建议您遵守关于 API 调用的 Amazon Pinpoint 要求。Amazon Pinpoint 要求客户端使用传输层安全性协议 (TLS) 1.2 或更高版本。客户端还必须支持具有完全向前保密 (PFS) 的密码套件，例如临时 Diffie-Hellman (DHE) 或临时椭圆曲线 Diffie-Hellman (ECDHE)。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

此外，必须使用访问密钥 ID 和与 AWS 账户的 AWS Identity and Access Management (IAM) 委托人关联的私有访问密钥对请求进行签名。或者，您可以使用 [AWS Security Token Service](#) (AWS STS) 生成临时安全凭证来签名请求。

### 亚马逊 Pinpoint 与其他资源之间的流量 AWS

为了保护亚马逊 Pinpoint 与同一 AWS 地区其他 AWS 资源之间的通信，Amazon Pinpoint 默认使用 HTTPS 和 TLS 1.2。

### 为 Amazon Pinpoint 创建接口 VPC 端点

您可以通过创建接口 VPC 端点在虚拟私有云 (VPC) 与 Amazon Pinpoint 中的端点之间建立专用连接。

接口终端节点由一项技术提供支持 [AWS PrivateLink](#)，该技术允许您在没有互联网网关、NAT 设备、VPN 连接或的情况下私密访问 Amazon Pinpoint API。AWS Direct Connect VPC 中的实例不需要公有 IP 地址即可与集成了 AWS PrivateLink 的 Amazon Pinpoint API 进行通信。

有关更多信息，请参阅 [AWS PrivateLink 指南](#)。

## 创建接口 VPC 端点

您可以使用 Amazon VPC 控制台或 AWS Command Line Interface (AWS CLI) 创建接口终端节点。有关更多信息，请参阅 AWS PrivateLink 指南中的 [创建接口终端节点](#)。

Amazon Pinpoint 支持以下服务名称：

- `com.amazonaws.region.pinpoint`
- `com.amazonaws.region.pinpoint-sms-voice-v2`

例如，如果您为接口终端节点开启私有 DNS，则可以使用默认 DNS 名称向 Amazon Pinpoint 发出 AWS 区域 API 请求。`com.amazonaws.us-east-1.pinpoint` 有关更多信息，请参阅《AWS PrivateLink 指南》中的 [DNS 主机名](#)。

有关当前可用 Amazon Pinpoint 的所有区域和端点的列表，请参阅《Amazon Web Services 一般参考》中的 [AWS 服务端点](#)。

## 创建 VPC 端点策略

您可以为 VPC 端点附加控制访问权限的端点策略。该策略指定以下信息：

- 可执行操作的主体。
- 可执行的操作。
- 可对其执行操作的资源。

有关更多信息，请参阅《AWS PrivateLink 指南》中的 [使用端点策略控制对服务的访问](#)。

## 示例：VPC 端点策略

以下 VPC 端点策略授权所有资源上的所有主体可访问列出的 Amazon Pinpoint 操作。

```
{
```

```
"Statement": [  
  {  
    "Principal": "*",  
    "Action": [  
      "mobiletargeting:CreateCampaign",  
      "mobiletargeting:CreateApp",  
      "mobiletargeting>DeleteApp",  
    ],  
    "Effect": "Allow",  
    "Resource": "*"  }  
]
```

## Amazon Pinpoint 的身份和访问管理

AWS Identity and Access Management (IAM) AWS 服务 可帮助管理员安全地控制对 AWS 资源的访问权限。IAM 管理员控制谁可以通过身份验证（登录）和获得授权（具有权限）来使用 Amazon Pinpoint 资源。您可以使用 IAM AWS 服务，无需支付额外费用。

### 主题

- [受众](#)
- [使用身份进行身份验证](#)
- [使用策略管理访问](#)
- [Amazon Pinpoint 如何与 IAM 配合使用](#)
- [用于 IAM 策略的 Amazon Pinpoint 操作](#)
- [Amazon Pinpoint 基于身份的策略示例](#)
- [用于常见 Amazon Pinpoint 任务的 IAM 角色](#)
- [Amazon Pinpoint 身份和访问管理问题排查](#)

## 受众

您的使用方式 AWS Identity and Access Management (IAM) 会有所不同，具体取决于您在亚马逊 Pinpoint 中所做的工作。

服务用户 – 如果您使用 Amazon Pinpoint 服务来完成任务，则您的管理员会为您提供所需的凭证和权限。随着您使用更多 Amazon Pinpoint 功能来完成工作，您可能需要额外权限。了解如何管理访问权

限可帮助您向管理员请求适合的权限。如果您无法访问 Amazon Pinpoint 中的功能，请参阅 [Amazon Pinpoint 身份和访问管理问题排查](#)。

**服务管理员** – 如果您在公司负责管理 Amazon Pinpoint 资源，您可能对 Amazon Pinpoint 具有完全访问权限。您有责任确定您的服务用户应访问哪些 Amazon Pinpoint 功能和资源。然后，您必须向 IAM 管理员提交请求以更改服务用户的权限。请查看该页面上的信息以了解 IAM 的基本概念。要了解有关您的公司如何将 IAM 与 Amazon Pinpoint 搭配使用的更多信息，请参阅 [Amazon Pinpoint 如何与 IAM 配合使用](#)。

**IAM 管理员** – 如果您是 IAM 管理员，您可能需要了解如何编写策略以管理对 Amazon Pinpoint 的访问的详细信息。要查看您可在 IAM 中使用的 Amazon Pinpoint 基于身份的策略示例，请参阅 [Amazon Pinpoint 基于身份的策略示例](#)。

## 使用身份进行身份验证

身份验证是您 AWS 使用身份凭证登录的方式。您必须以 IAM 用户身份或通过担 AWS 账户根用户任 IAM 角色进行身份验证（登录 AWS）。

您可以使用通过身份源提供的凭据以 AWS 联合身份登录。AWS IAM Identity Center（IAM Identity Center）用户、贵公司的单点登录身份验证以及您的 Google 或 Facebook 凭据就是联合身份的示例。当您以联合身份登录时，您的管理员以前使用 IAM 角色设置了身份联合验证。当你使用联合访问 AWS 时，你就是在间接扮演一个角色。

根据您的用户类型，您可以登录 AWS Management Console 或 AWS 访问门户。有关登录的更多信息 AWS，请参阅《AWS 登录 用户指南》中的 [如何登录到您 AWS 账户的](#)。

如果您 AWS 以编程方式访问，则会 AWS 提供软件开发套件 (SDK) 和命令行接口 (CLI)，以便使用您的凭据对请求进行加密签名。如果您不使用 AWS 工具，则必须自己签署请求。有关使用推荐的方法自行签署请求的更多信息，请参阅 IAM 用户指南中的 [签署 AWS API 请求](#)。

无论使用何种身份验证方法，您可能需要提供其他安全信息。例如，AWS 建议您使用多重身份验证 (MFA) 来提高账户的安全性。要了解更多信息，请参阅《AWS IAM Identity Center 用户指南》中的 [多重身份验证](#) 和《IAM 用户指南》中的 [在 AWS 中使用多重身份验证 \(MFA\)](#)。

## AWS 账户 root 用户

创建时 AWS 账户，首先要有一个登录身份，该身份可以完全访问账户中的所有资源 AWS 服务和资源。此身份被称为 AWS 账户 root 用户，使用您创建帐户时使用的电子邮件地址和密码登录即可访问该身份。强烈建议您不要使用根用户执行日常任务。保护好根用户凭证，并使用这些凭证来执行仅根用户可以执行的任务。有关需要您以根用户身份登录的任务的完整列表，请参阅 IAM 用户指南 中的 [需要根用户凭证的任务](#)。



## IAM 用户和群组

[IAM 用户](#)是您 AWS 账户 内部对个人或应用程序具有特定权限的身份。在可能的情况下，我们建议使用临时凭证，而不是创建具有长期凭证（如密码和访问密钥）的 IAM 用户。但是，如果您有一些特定的使用场景需要长期凭证以及 IAM 用户，我们建议您轮换访问密钥。有关更多信息，请参阅《IAM 用户指南》中的[对于需要长期凭证的使用场景定期轮换访问密钥](#)。

[IAM 组](#)是一个指定一组 IAM 用户的身份。您不能使用组的身份登录。您可以使用组来一次性为多个用户指定权限。如果有大量用户，使用组可以更轻松地管理用户权限。例如，您可能具有一个名为 IAMAdmins 的组，并为该组授予权限以管理 IAM 资源。

用户与角色不同。用户唯一地与某个人或应用程序关联，而角色旨在让需要它的任何人代入。用户具有永久的长期凭证，而角色提供临时凭证。要了解更多信息，请参阅 IAM 用户指南中的[何时创建 IAM 用户（而不是角色）](#)。

## IAM 角色

[IAM 角色](#)是您内部具有特定权限 AWS 账户 的身份。它类似于 IAM 用户，但与特定人员不关联。您可以 AWS Management Console 通过[切换角色在中临时担任 IAM 角色](#)。您可以通过调用 AWS CLI 或 AWS API 操作或使用自定义 URL 来代入角色。有关使用角色的方法的更多信息，请参阅《IAM 用户指南》中的[使用 IAM 角色](#)。

具有临时凭证的 IAM 角色在以下情况下很有用：

- 联合用户访问 – 要向联合身份分配权限，请创建角色并为角色定义权限。当联合身份进行身份验证时，该身份将与角色相关联并被授予由此角色定义的权限。有关联合身份验证的角色的信息，请参阅《IAM 用户指南》中的[为第三方身份提供商创建角色](#)。如果您使用 IAM Identity Center，则需要配置权限集。为控制您的身份在进行身份验证后可以访问的内容，IAM Identity Center 将权限集与 IAM 中的角色相关联。有关权限集的信息，请参阅《AWS IAM Identity Center 用户指南》中的[权限集](#)。
- 临时 IAM 用户权限 – IAM 用户可代入 IAM 用户或角色，以暂时获得针对特定任务的不同权限。
- 跨账户存取 – 您可以使用 IAM 角色以允许不同账户中的某个人（可信主体）访问您的账户中的资源。角色是授予跨账户访问权限的主要方式。但是，对于某些资源 AWS 服务，您可以将策略直接附加到资源（而不是使用角色作为代理）。要了解用于跨账户访问的角色和基于资源的策略之间的差别，请参阅《IAM 用户指南》中的[IAM 角色与基于资源的策略有何不同](#)。
- 跨服务访问 — 有些 AWS 服务 使用其他 AWS 服务 服务中的功能。例如，当您在某个服务中进行调用时，该服务通常会在 Amazon EC2 中运行应用程序或在 Amazon S3 中存储对象。服务可能会使用发出调用的主体的权限、使用服务角色或使用服务相关角色来执行此操作。

- 转发访问会话 (FAS) — 当您使用 IAM 用户或角色在中执行操作时 AWS，您被视为委托人。使用某些服务时，您可能会执行一个操作，此操作然后在不同服务中启动另一个操作。FAS 使用调用委托人的权限以及 AWS 服务 向下游服务发出请求的请求。AWS 服务只有当服务收到需要与其他 AWS 服务 或资源交互才能完成的请求时，才会发出 FAS 请求。在这种情况下，您必须具有执行这两个操作的权限。有关发出 FAS 请求时的策略详情，请参阅[转发访问会话](#)。
- 服务角色 - 服务角色是服务代表您在您的账户中执行操作而分派的 [IAM 角色](#)。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的[创建向 AWS 服务委派权限的角色](#)。
- 服务相关角色-服务相关角色是一种链接到的服务角色。AWS 服务服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 AWS 账户，并且归服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。
- 在 Amazon EC2 上运行的应用程序 — 您可以使用 IAM 角色管理在 EC2 实例上运行并发出 AWS CLI 或 AWS API 请求的应用程序的临时证书。这优先于在 EC2 实例中存储访问密钥。要向 EC2 实例分配 AWS 角色并使其可供其所有应用程序使用，您需要创建附加到该实例的实例配置文件。实例配置文件包含角色，并使 EC2 实例上运行的程序能够获得临时凭证。有关更多信息，请参阅《IAM 用户指南》中的[使用 IAM 角色为 Amazon EC2 实例上运行的应用程序授予权限](#)。

要了解是使用 IAM 角色还是 IAM 用户，请参阅 IAM 用户指南中的[何时创建 IAM 角色 \(而不是用户\)](#)。

## 使用策略管理访问

您可以 AWS 通过创建策略并将其附加到 AWS 身份或资源来控制中的访问权限。策略是其中的一个对象 AWS，当与身份或资源关联时，它会定义其权限。AWS 在委托人 (用户、root 用户或角色会话) 发出请求时评估这些策略。策略中的权限确定是允许还是拒绝请求。大多数策略都以 JSON 文档的 AWS 形式存储在中。有关 JSON 策略文档的结构和内容的更多信息，请参阅《IAM 用户指南》中的[JSON 策略概览](#)。

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体 可以对什么资源执行操作，以及在什么条件下执行。

默认情况下，用户和角色没有权限。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM policy。管理员随后可以向角色添加 IAM policy，用户可以代入角色。

IAM policy 定义操作的权限，无关乎您使用哪种方法执行操作。例如，假设您有一个允许 `iam:GetRole` 操作的策略。拥有该策略的用户可以从 AWS Management Console AWS CLI、或 AWS API 获取角色信息。

## 基于身份的策略

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅 IAM 用户指南中的[创建 IAM policy](#)。

基于身份的策略可以进一步归类为内联策略或托管策略。内联策略直接嵌入单个用户、组或角色中。托管策略是独立的策略，您可以将其附加到中的多个用户、群组和角色 AWS 账户。托管策略包括 AWS 托管策略和客户托管策略。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的[在托管策略与内联策略之间进行选择](#)。

Amazon Pinpoint 支持使用基于身份的策略来控制对 Amazon Pinpoint 资源的访问。

## 基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Simple Storage Service ( Amazon S3 ) 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中[指定主体](#)。委托人可以包括账户、用户、角色、联合用户或 AWS 服务。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 AWS 托管策略。

Amazon Pinpoint 支持使用基于资源的策略来控制对 Amazon Pinpoint 资源的访问。

## 访问控制列表 (ACL)

访问控制列表 ( ACL ) 控制哪些主体 ( 账户成员、用户或角色 ) 有权访问资源。ACL 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

Amazon S3 和 Amazon VPC 就是支持 ACL 的服务示例。AWS WAF 要了解有关 ACL 的更多信息，请参阅《Amazon Simple Storage Service 开发人员指南》中的[访问控制列表 \(ACL\) 概览](#)。

Amazon Pinpoint 不支持使用 ACL 来控制对 Amazon Pinpoint 资源的访问。

## 其它策略类型

AWS 支持其他不太常见的策略类型。这些策略类型可以设置更常用的策略类型向您授予的最大权限。

- 权限边界 – 权限边界是一个高级特征，用于设置基于身份的策略可以为 IAM 实体 ( IAM 用户或角色 ) 授予的最大权限。您可为实体设置权限边界。这些结果权限是实体基于身份的策略及其权限边界

的交集。在 Principal 中指定用户或角色的基于资源的策略不受权限边界限制。任一项策略中的显式拒绝将覆盖允许。有关权限边界的更多信息，请参阅 IAM 用户指南中的 [IAM 实体的权限边界](#)。

- 服务控制策略 (SCP)-SCP 是 JSON 策略，用于指定组织或组织单位 (OU) 的最大权限。AWS Organizations AWS Organizations 是一项用于对您的企业拥有的多 AWS 账户 项进行分组和集中管理的服务。如果在组织内启用了所有特征，则可对任意或全部账户应用服务控制策略 (SCP)。SCP 限制成员账户中的实体 (包括每个 AWS 账户根用户实体) 的权限。有关 Organizations 和 SCP 的更多信息，请参阅 AWS Organizations 用户指南中的 [SCP 的工作原理](#)。
- 会话策略 – 会话策略是当您以编程方式为角色或联合身份用户创建临时会话时作为参数传递的高级策略。结果会话的权限是用户或角色的基于身份的策略和会话策略的交集。权限也可以来自基于资源的策略。任一项策略中的显式拒绝将覆盖允许。有关更多信息，请参阅《IAM 用户指南》中的 [会话策略](#)。

Amazon Pinpoint 支持使用这些类型的策略来控制对 Amazon Pinpoint 资源的访问。

## 多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 AWS 确定是否允许请求，请参阅 IAM 用户指南中的 [策略评估逻辑](#)。

## Amazon Pinpoint 如何与 IAM 配合使用

要使用 Amazon Pinpoint，您 AWS 账户中的用户需要权限才能查看分析数据、创建项目、定义用户群体、部署活动等。如果将移动或 Web 应用程序与 Amazon Pinpoint 集成，则应用程序的用户还需要具有 Amazon Pinpoint 的访问权限。此访问权限允许您的应用向 Amazon Pinpoint 注册端点并报告使用情况数据。要授予对亚马逊 Pinpoint 功能的访问权限，请创建 AWS Identity and Access Management (IAM) 策略，允许亚马逊 Pinpoint 对 IAM 身份或 Amazon Pinpoint 资源执行 Pinpoint 操作。

IAM 是一项服务，可帮助管理员安全地控制对 AWS 资源的访问。IAM 策略包含允许或拒绝特定用户或针对特定资源执行的特定操作的语句。Amazon Pinpoint 提供用于 IAM 策略的 [一组操作](#)，可以使用这些操作为 Amazon Pinpoint 用户和资源指定细粒度权限。这意味着，您可以授予对 Amazon Pinpoint 的适当级别的访问权限，而不会创建可能会泄漏重要数据或危害资源的过于宽松的策略。例如，您可以向 Amazon Pinpoint 管理员授予不受限制的访问权限，向只需访问特定项目的个人授予只读访问权限。

在使用 IAM 管理对 Amazon Pinpoint 的访问权限之前，您应该了解哪些 IAM 功能可用于 Amazon Pinpoint。要全面了解 Amazon Pinpoint 和其他 AWS 服务如何与 IAM 配合使用，请参阅 IAM 用户指南中与 IAM 配合使用的 [AWS 服务](#)。

## 主题

- [Amazon Pinpoint 基于身份的策略](#)
- [Amazon Pinpoint 基于资源的权限策略](#)
- [基于 Amazon Pinpoint 标签的授权](#)
- [Amazon Pinpoint IAM 角色](#)

## Amazon Pinpoint 基于身份的策略

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。Amazon Pinpoint 支持特定的操作、资源和条件键。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 [IAM JSON 策略元素参考](#)。

### 操作

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 Action 元素描述可用于在策略中允许或拒绝访问的操作。策略操作通常与关联的 AWS API 操作同名。有一些例外情况，例如没有匹配 API 操作的仅限权限操作。还有一些操作需要在策略中执行多个操作。这些附加操作称为相关操作。

在策略中包含操作以授予执行相关操作的权限。

这意味着，策略操作控制用户可以在 Amazon Pinpoint 控制台上执行的操作。它们还可以通过直接使用软件 AWS 开发工具包、AWS Command Line Interface (AWS CLI) 或 Amazon Pinpoint API 来控制用户可以以编程方式执行的操作。

Amazon Pinpoint 中的策略操作使用以下前缀：

- **mobiletargeting** – 适用于从 Amazon Pinpoint API 派生的操作，该 API 是 Amazon Pinpoint 的主要 API。
- **sms-voice** – 适用于从 Amazon Pinpoint SMS 和 Voice API 派生的操作，这是一个补充 API，它提供在 Amazon Pinpoint 中使用和管理短信和语音渠道的高级选项。

例如，要授予某人查看有关某项目所有分段的信息的权限（该操作与 Amazon Pinpoint API 中的 GetSegments 操作相对应），请将 mobiletargeting:GetSegments 操作包含在其策略中。策略语句必须包含 Action 或 NotAction 元素。Amazon Pinpoint 定义了一组自己的操作，以描述您可以使用该服务执行的任务。

要在单个语句中指定多项操作，请使用逗号将它们隔开：

```
"Action": [  
    "mobiletargeting:action1",  
    "mobiletargeting:action2"
```

您也可以使用通配符 (\*) 指定多项操作。例如，要指定以单词 Get 开头的的所有操作，包括以下操作：

```
"Action": "mobiletargeting:Get*"
```

但作为最佳实践，您应创建遵循最低权限 原则的策略。换句话说，您应创建仅包含执行特定操作所需的权限的策略。

有关您可以在 IAM 策略中使用的 Amazon Pinpoint 操作的列表，请参阅[用于 IAM 策略的 Amazon Pinpoint 操作](#)。

## 资源

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体 可以对什么资源 执行操作，以及在什么条件 下执行。

Resource JSON 策略元素指定要向其应用操作的一个或多个对象。语句必须包含 Resource 或 NotResource 元素。作为最佳实践，请使用其 [Amazon 资源名称 \(ARN\)](#) 指定资源。对于支持特定资源类型（称为资源级权限）的操作，您可以执行此操作。

对于不支持资源级权限的操作（如列出操作），请使用通配符 (\*) 指示语句应用于所有资源。

```
"Resource": "*"
```

例如，mobiletargeting:GetSegments 操作检索与特定 Amazon Pinpoint 项目关联的所有分段的信息。您可以使用以下格式标识具有 ARN 的项目：

```
arn:aws:mobiletargeting:${Region}:${Account}:apps/${projectId}
```

有关 ARN 格式的更多信息，请参阅《AWS 一般参考》中的 [Amazon 资源名称 \(ARN\)](#)。

在 IAM 策略中，您可以为以下类型的 Amazon Pinpoint 资源指定 ARN：

- 活动
- 历程
- 消息模板 ( 在某些情况下称为模板 )
- 项目 ( 在某些情况下称为应用程序 )
- 推荐器模型 ( 在某些情况下称为推荐器 )
- 分段

例如，要为具有项目 ID 810c7aab86d42fb2b56c8c966example 的项目创建策略语句，请使用以下 ARN：

```
"Resource": "arn:aws:mobiletargeting:us-east-1:123456789012:apps/810c7aab86d42fb2b56c8c966example"
```

要指定属于特定账户的所有项目，请使用通配符 (\*)：

```
"Resource": "arn:aws:mobiletargeting:us-east-1:123456789012:apps/*"
```

某些 Amazon Pinpoint 操作 ( 如用于创建资源的某些操作 ) 不能在特定资源上执行。在这些情况下，您必须使用通配符 (\*)：

```
"Resource": "*"
```

在 IAM 策略中，您可以为以下类型的 Amazon Pinpoint SMS 和 Voice 资源指定 ARN：

- 配置集
- 退订列表
- 电话号码
- 池
- 发件人 ID

例如，要为具有电话号码 ID phone-12345678901234567890123456789012 的电话号码创建策略声明，请使用以下 ARN：

```
"Resource": "arn:aws:sms-voice:us-east-1:123456789012:phone-number/phone-12345678901234567890123456789012"
```

要指定属于特定账户的所有电话号码，请使用通配符 (\*) 代替电话号码 ID：

```
"Resource": "arn:aws:sms-voice:us-east-1:123456789012:phone-number/*"
```

某些 Amazon Pinpoint SMS 和 Voice 操作不能在特定资源上执行，例如用于管理账户级别设置（如支出限制）的资源。在这些情况下，您必须使用通配符 (\*)：

```
"Resource": "*"
```

一些 Amazon Pinpoint API 操作涉及多种资源。例如，TagResource 操作可以向多个项目添加标签。要在单个语句中指定多个资源，请使用逗号分隔 ARN：

```
"Resource": [  
    "resource1",  
    "resource2"
```

要查看 Amazon Pinpoint 资源类型及其 ARN 的列表，请参阅《IAM 用户指南》中的 [Amazon Pinpoint 定义的资源](#)。要了解您可以使用哪些操作指定每个资源的 ARN，请参阅《IAM 用户指南》中的 [Amazon Pinpoint 定义的操作](#)。

## 条件键

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

在 Condition 元素（或 Condition 块）中，可以指定语句生效的条件。Condition 元素是可选的。您可以创建使用 [条件运算符](#)（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。

如果您在一个语句中指定多个 Condition 元素，或在单个 Condition 元素中指定多个键，则 AWS 使用逻辑 AND 运算评估它们。如果您为单个条件键指定多个值，则使用逻辑 OR 运算来 AWS 评估条件。在授予语句的权限之前必须满足所有的条件。

在指定条件时，您也可以使用占位符变量。例如，只有在使用 IAM 用户名标记 IAM 用户时，您才能为其授予访问资源的权限。有关更多信息，请参阅 IAM 用户指南中的 [IAM policy 元素：变量和标签](#)。

AWS 支持全局条件密钥和特定于服务的条件键。要查看所有 AWS 全局条件键，请参阅 IAM 用户指南中的 [AWS 全局条件上下文密钥](#)。



Amazon Pinpoint 定义了自己的一组条件键，同时还支持一些全局条件键。要查看所有 AWS 全局条件键的列表，请参阅 IAM 用户指南中的[AWS 全局条件上下文密钥](#)。要查看 Amazon Pinpoint 条件键的列表，请参阅《IAM 用户指南》中的[Amazon Pinpoint 的条件键](#)。要了解您可以对哪些操作和资源使用条件键，请参阅《IAM 用户指南》中的[Amazon Pinpoint 定义的操作](#)。

## 示例

要查看 Amazon Pinpoint 基于身份的策略的示例，请参阅[Amazon Pinpoint 基于身份的策略示例](#)。

## Amazon Pinpoint 基于资源的权限策略

基于资源的权限策略是 JSON 策略文档，它们指定了某个指定主体可以在 Amazon Pinpoint 资源上执行哪些操作以及在什么条件下可执行。Amazon Pinpoint 支持针对活动、旅程、消息模板（模板）、推荐器模型（推荐器）、项目（应用程序）和分段的基于资源的权限策略。

## 示例

要查看 Amazon Pinpoint 基于资源的策略的示例，请参阅[the section called “基于身份的策略示例”](#)，

## 基于 Amazon Pinpoint 标签的授权

您可以将标签与特定类型的 Amazon Pinpoint 资源关联，或将请求中的标签传递给 Amazon Pinpoint。要基于标签控制访问，您需要使用 `aws:ResourceTag/${TagKey}`、`aws:RequestTag/${TagKey}` 或 `aws:TagKeys` 条件键在策略的[条件元素](#)中提供标签信息。

有关标记 Amazon Pinpoint 资源（包括示例 IAM 策略）的信息，请参阅[为 Amazon Pinpoint 资源添加标签](#)。

## Amazon Pinpoint IAM 角色

[IAM 角色](#)是 AWS 账户中具有特定权限的实体。

### 将临时凭证用于 Amazon Pinpoint

您可以使用临时凭证进行联合身份登录、代入 IAM 角色或代入跨账户角色。您可以通过调用 AWS Security Token Service (AWS STS) API 操作（例如[AssumeRole](#)或）来获取临时安全证书[GetFederationToken](#)。

Amazon Pinpoint 支持使用临时凭证。

## 服务相关角色

[服务相关角色](#) 允许 AWS 服务访问其他服务中的资源以代表您完成操作。服务相关角色显示在 IAM 账户中，并归该服务所有。IAM 管理员可以查看但不能编辑服务相关角色的权限。

Amazon Pinpoint 不使用服务相关角色。

## 服务角色

此功能允许服务代表您担任 [服务角色](#)。此角色允许服务访问其他服务中的资源以代表您完成操作。服务角色显示在 IAM 账户中，并归该账户所有。这意味着，IAM 管理员可以更改该角色的权限。但是，这样做可能会中断服务的功能。

Amazon Pinpoint 支持使用服务角色。

## 用于 IAM 策略的 Amazon Pinpoint 操作

要管理对您 AWS 账户中亚马逊 Pinpoint 资源的访问权限，您可以在 (IAM) 策略中添加亚马逊 Pinpoint 操作 AWS Identity and Access Management。通过在策略中使用操作，您可以控制用户可以在 Amazon Pinpoint 控制台上执行的操作。您还可以直接使用软件 AWS 开发工具包、AWS Command Line Interface (AWS CLI) 或 Amazon Pinpoint API，以编程方式控制用户可以执行的操作。

在策略中，可以使用后跟冒号和操作名称（如 `GetSegments`）的适当 Amazon Pinpoint 命名空间指定每个操作。大多数操作都与使用特定 URI 和 HTTP 方法对 Amazon Pinpoint API 进行的请求相对应。例如，如果您在用户的策略中允许 `mobiletargeting:GetSegments` 操作，则允许用户通过向 [/apps/projectId/segments](#) URI 提交 HTTP GET 请求来检索有关项目所有分段的信息。此策略还允许用户在控制台上查看该信息，并使用 AWS 软件开发工具包或检索该信息 AWS CLI。

每个操作在特定 Amazon Pinpoint 资源（在策略语句中通过 Amazon 资源名称 (ARN) 来标识）上执行。例如，`mobiletargeting:GetSegments` 操作在使用 ARN `arn:aws:mobiletargeting:region:accountId:apps/projectId` 标识的特定项目上执行。

本主题介绍您可以添加到 AWS 账户的 IAM 策略的 Amazon Pinpoint 操作。要查看有关如何使用策略中的操作来管理对 Amazon Pinpoint 资源的访问的示例，请参阅 [Amazon Pinpoint 基于身份的策略示例](#)。

### 主题

- [Amazon Pinpoint API 操作](#)
- [Amazon Pinpoint SMS 和 Voice API 第 1 版操作](#)

- [Amazon Pinpoint SMS 和 Voice API 第 2 版操作](#)

## Amazon Pinpoint API 操作

本节介绍 Amazon Pinpoint API 中提供的功能的操作，该 API 是 Amazon Pinpoint 的主要 API。要了解有关此 API 的更多信息，请参阅 [Amazon Pinpoint API 参考](#)。

类别：

- [分析和指标](#)
- [市场活动](#)
- [渠道](#)
- [端点](#)
- [事件流](#)
- [事件](#)
- [导出任务](#)
- [导入任务](#)
- [旅程](#)
- [消息模板](#)
- [消息](#)
- [一次性密码](#)
- [电话号码验证](#)
- [项目](#)
- [推荐器模型](#)
- [分段](#)
- [标签](#)
- [用户](#)

### 分析和指标

以下权限与在 Amazon Pinpoint 控制台上查看分析数据相关。它们还与检索（查询）适用于项目、活动和旅程的标准指标的聚合数据相关，这些指标也称为关键绩效指标 (KPI)。

## mobiletargeting:GetReports

在 Amazon Pinpoint 控制台上查看分析数据。要使用 Amazon Pinpoint 控制台创建包含自定义属性的分段，也需要此权限。要在 Amazon Pinpoint 控制台中获取分段大小的估计值，同样需要此权限。

- URI – 不适用
- 方法 – 不适用
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:*`

## mobiletargeting:GetApplicationDateRangeKpi

检索 ( 查询 ) 标准应用程序指标的聚合数据。这是适用于与某个项目关联的所有活动或事务性消息的指标。

- URI – [/apps/projectId/kpis/daterange/kpi-name](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/kpis/daterange/kpi-name`

## mobiletargeting:GetCampaignDateRangeKpi

检索 ( 查询 ) 标准活动指标的聚合数据。这是适用于单个活动的指标。

- URI – [/apps/projectId/campaigns/campaignId/kpis/daterange/kpi-name](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/campaigns/campaignId/kpis/daterange/kpi-name`

## mobiletargeting:GetJourneyDateRangeKpi

检索 ( 查询 ) 标准旅程参与指标的聚合数据。这是适用于单独旅程的参与度指标，例如，对于某个旅程的所有活动，参与者打开的消息的数量。

- URI – [/apps/projectId/journeys/journeyId/kpis/daterange/kpi-name](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/journeyId/kpis/daterange/kpi-name`

## mobiletargeting:GetJourneyExecutionMetrics

检索 ( 查询 ) 适用于单独旅程的标准执行指标的汇总数据，例如，对于某个旅程的所有活动，积极推进各项活动的参与者的数量。

- URI – [/apps/\*projectId\*/journeys/\*journeyId\*/execution-metrics](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/execution-metrics

### **mobiletargeting:GetJourneyExecutionActivityMetrics**

检索 ( 查询 ) 适用于旅程中单个活动的标准执行指标的汇总数据，例如，开始或完成某项活动的参与者的数量。

- URI – [/apps/\*projectId\*/journeys/\*journeyId\*/activities/\*journey-activity-id\*/execution-metrics](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/activities/*journey-activity-id*/execution-metrics

## 市场活动

以下权限与管理您的 Amazon Pinpoint 账户中的活动有关。

### **mobiletargeting:CreateCampaign**

为项目创建活动。

- URI – [/apps/\*projectId\*/campaigns](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns

### **mobiletargeting>DeleteCampaign**

删除特定活动。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

### **mobiletargeting:GetCampaign**

检索有关特定活动的信息。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

### **mobiletargeting:GetCampaignActivities**

检索有关活动执行的活动的信息。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*/activities](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

### **mobiletargeting:GetCampaigns**

检索有关项目的所有活动的信息。

- URI – [/apps/\*projectId\*/campaigns](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

### **mobiletargeting:GetCampaignVersion**

检索有关特定活动版本的信息。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*/versions/\*versionId\*](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

### **mobiletargeting:GetCampaignVersions**

检索有关活动的当前和以前版本的信息。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*/versions](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/campaigns/*campaignId*

### **mobiletargeting:UpdateCampaign**

更新特定活动。

- URI – [/apps/\*projectId\*/campaigns/\*campaignId\*](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/campaigns/campaignId`

## 渠道

以下权限与管理您的 Amazon Pinpoint 账户中的渠道有关。在 Amazon Pinpoint 中，渠道是指您用于联系客户的方法，例如发送电子邮件、短信或推送通知。

### **mobiletargeting:DeleteAdmChannel**

禁用项目的 Amazon Device Messaging (ADM) 渠道。

- URI – [/apps/\*projectId\*/channels/adm](#)
- 方法 – DELETE
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/adm`

### **mobiletargeting:GetAdmChannel**

检索有关项目的 ADM 渠道的信息。

- URI – [/apps/\*projectId\*/channels/adm](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/adm`

### **mobiletargeting:UpdateAdmChannel**

启用或更新项目的 ADM 渠道。

- URI – [/apps/\*projectId\*/channels/adm](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/adm`

### **mobiletargeting>DeleteApnsChannel**

禁用项目的 Apple Push Notification service (APNs) 渠道。

- URI – [/apps/\*projectId\*/channels/apns](#)

- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

### **mobiletargeting:GetApnsChannel**

检索有关项目的 APNs 渠道的信息。

- URI – [/apps/\*projectId\*/channels/apns](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

### **mobiletargeting:UpdateApnsChannel**

启用或更新项目的 APNs 渠道。

- URI – [/apps/\*projectId\*/channels/apns](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns

### **mobiletargeting>DeleteApnsSandboxChannel**

禁用项目的 APNs 沙盒渠道。

- URI – [/apps/\*projectId\*/channels/apns\\_sandbox](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_sandbox

### **mobiletargeting:GetApnsSandboxChannel**

检索有关项目的 APNs 沙盒渠道的信息。

- URI – [/apps/\*projectId\*/channels/apns\\_sandbox](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_sandbox

### **mobiletargeting:UpdateApnsSandboxChannel**

启用或更新项目的 APNs 沙盒渠道。



- URI – [/apps/\*projectId\*/channels/apns\\_sandbox](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_sandbox

### **mobiletargeting:DeleteApnsVoipChannel**

禁用项目的 APNs VoIP 渠道。

- URI – [/apps/\*projectId\*/channels/apns\\_voip](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip

### **mobiletargeting:GetApnsVoipChannel**

检索有关项目的 APNs VoIP 渠道的信息。

- URI – [/apps/\*projectId\*/channels/apns\\_voip](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip

### **mobiletargeting:UpdateApnsVoipChannel**

启用或更新项目的 APNs VoIP 渠道。

- URI – [/apps/\*projectId\*/channels/apns\\_voip](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip

### **mobiletargeting:DeleteApnsVoipSandboxChannel**

禁用项目的 APNs VoIP 沙盒渠道。

- URI – [/apps/\*projectId\*/channels/apns\\_voip\\_sandbox](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip\_sandbox

### **mobiletargeting:GetApnsVoipSandboxChannel**

检索有关项目的 APNs VoIP 沙盒渠道的信息。

- URI – [/apps/\*projectId\*/channels/apns\\_voip\\_sandbox](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip\_sandbox

### **mobiletargeting:UpdateApnsVoipSandboxChannel**

启用或更新项目的 APNs VoIP 沙盒渠道。

- URI – [/apps/\*projectId\*/channels/apns\\_voip\\_sandbox](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/apns\_voip\_sandbox

### **mobiletargeting>DeleteBaiduChannel**

禁用项目的百度云推送渠道。

- URI – [/apps/\*projectId\*/channels/baidu](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

### **mobiletargeting:GetBaiduChannel**

检索有关项目的百度云推送渠道的信息。

- URI – [/apps/\*projectId\*/channels/baidu](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

### **mobiletargeting:UpdateBaiduChannel**

启用或更新项目的百度云推送渠道。

- URI – [/apps/\*projectId\*/channels/baidu](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/channels/baidu

### **mobiletargeting>DeleteEmailChannel**

禁用项目的电子邮件渠道。

- URI – [/apps/projectId/channels/email](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/email

### **mobiletargeting:GetEmailChannel**

检索有关项目的电子邮件渠道的信息。

- URI – [/apps/projectId/channels/email](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/email

### **mobiletargeting:UpdateEmailChannel**

启用或更新项目的电子邮件渠道。

- URI – [/apps/projectId/channels/email](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/email

### **mobiletargeting>DeleteGcmChannel**

禁用项目的 Firebase Cloud Messaging (FCM) 渠道。该渠道允许 Amazon Pinpoint 通过 FCM 服务将推送通知发送到 Android 应用程序，FCM 服务取代了 Google Cloud Messaging (GCM) 服务。

- URI – [/apps/projectId/channels/gcm](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/gcm

### **mobiletargeting:GetGcmChannel**

检索有关项目的 FCM 渠道的信息。该渠道允许 Amazon Pinpoint 通过 FCM 服务将推送通知发送到 Android 应用程序，FCM 服务取代了 Google Cloud Messaging (GCM) 服务。

- URI – [/apps/projectId/channels/gcm](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/gcm

## mobiletargeting:UpdateGcmChannel

启用或更新项目的 FCM 渠道。该渠道允许 Amazon Pinpoint 通过 FCM 服务将推送通知发送到 Android 应用程序，FCM 服务取代了 Google Cloud Messaging (GCM) 服务。

- URI – [/apps/projectId/channels/gcm](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/gcm

## mobiletargeting>DeleteSmsChannel

禁用项目的短信渠道。

- URI – [/apps/projectId/channels/sms](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/sms

## mobiletargeting:GetSmsChannel

检索有关项目的短信渠道的信息。

- URI – [/apps/projectId/channels/sms](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/sms

## mobiletargeting:UpdateSmsChannel

启用或更新项目的短信渠道。

- URI – [/apps/projectId/channels/sms](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/channels/sms

## mobiletargeting:GetChannels

检索有关应用程序的每个渠道的历史和状态的信息。

- URI – [/apps/application-id/channels](#)
- 方法 – GET

- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/channels`

### **mobiletargeting:DeleteVoiceChannel**

禁用应用程序的语音通道并删除该通道的任何现有设置。

- URI – [`/apps/application-id/channels/voice`](#)
- 方法 – DELETE
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectid/channels/voice`

### **mobiletargeting:GetVoiceChannel**

检索有关应用程序的语音通道的状态和设置的信息。

- URI – [`/apps/application-id/channels/voice`](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectid/channels/voice`

### **mobiletargeting:UpdateVoiceChannel**

启用应用程序的语音通道或更新应用程序语音通道的状态和设置。

- URI – [`/apps/application-id/channels/voice`](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectid/channels/voice`

## 端点

以下权限与管理您的 Amazon Pinpoint 账户中的端点有关。在 Amazon Pinpoint 中，端点是您的消息的一个目的地。例如，端点可能是客户的电子邮件地址、电话号码或移动设备令牌。

### **mobiletargeting:DeleteEndpoint**

删除端点。

- URI – [`/apps/projectId/endpoints/endpointId`](#)
- 方法 – DELETE

- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/endpoints/endpointId`

### **mobiletargeting:GetEndpoint**

检索有关特定端点的信息。

- URI – [/apps/projectId/endpoints/endpointId](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/endpoints/endpointId`

### **mobiletargeting:RemoveAttributes**

从与某个应用程序关联的所有端点中移除一个或多个具有相同属性类型的属性。

- URI – [apps/application-id/attributes/attribute-type](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/attributes/attribute-type`

### **mobiletargeting:UpdateEndpoint**

创建端点或更新端点的信息。

- URI – [/apps/projectId/endpoints/endpointId](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/endpoints/endpointId`

### **mobiletargeting:UpdateEndpointsBatch**

以批处理操作形式创建或更新端点。

- URI – [/apps/projectId/endpoints](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

## 事件流

以下权限与管理您的 Amazon Pinpoint 账户的事件流有关。

## mobiletargeting:DeleteEventStream

删除项目的事件流。

- URI – [/apps/\*projectId\*/eventstream/](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

## mobiletargeting:GetEventStream

检索有关项目的事件流的信息。

- URI – [/apps/\*projectId\*/eventstream/](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

## mobiletargeting:PutEventStream

创建或更新项目的事件流。

- URI – [/apps/\*projectId\*/eventstream/](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/eventstream

## 事件

以下权限与管理您的 Amazon Pinpoint 账户中的事件任务有关。在 Amazon Pinpoint 中，您可以创建导入任务，以根据存储在 Amazon S3 存储桶中的端点定义创建分段。

## mobiletargeting:PutEvents

为端点创建要记录的新事件，或者创建或更新与现有事件关联的端点数据。

- URI – [/apps/\*application-id\*/events](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/events

## 导出任务

以下权限与管理您的 Amazon Pinpoint 账户中的导出任务有关。在 Amazon Pinpoint 中，您可以创建导出任务，以将有关端点的信息发送到 Amazon S3 存储桶进行存储或分析。

### **mobiletargeting:CreateExportJob**

创建导出任务以将端点定义导出到 Amazon S3。

- URI – [/apps/projectId/jobs/export](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/jobs/export

### **mobiletargeting:GetExportJob**

检索有关项目的特定导出任务的信息。

- URI – [/apps/projectId/jobs/export/jobId](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/jobs/export/jobId

### **mobiletargeting:GetExportJobs**

检索项目的所有导出任务的列表。

- URI – [/apps/projectId/jobs/export](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/jobs/export

## 导入任务

以下权限与管理您的 Amazon Pinpoint 账户中的导入任务有关。在 Amazon Pinpoint 中，您可以创建导入任务，以根据存储在 Amazon S3 存储桶中的端点定义创建分段。

### **mobiletargeting:CreateImportJob**

从 Amazon S3 导入端点定义以创建分段。

- URI – [/apps/projectId/jobs/import](#)
- 方法 – POST



- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

### **mobiletargeting:GetImportJob**

检索有关项目的特定导入任务的信息。

- URI – [/apps/projectId/jobs/import/jobId](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/jobs/import/jobId`

### **mobiletargeting:GetImportJobs**

检索有关项目的所有导入任务的信息。

- URI – [/apps/projectId/jobs/import](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

## 旅程

以下权限与管理您的 Amazon Pinpoint 账户中的旅程相关。

### **mobiletargeting:CreateJourney**

为项目创建旅程。

- URI – [/apps/projectId/journeys](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys`

### **mobiletargeting:GetJourney**

检索有关特定旅程的信息。

- URI – [/apps/projectId/journeys/journeyId](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/journeys/journeyId`

### **mobiletargeting:ListJourneys**

检索有关项目的所有旅程的信息。

- URI – [/apps/\*projectId\*/journeys](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys

### **mobiletargeting:UpdateJourney**

更新特定旅程的配置和其他设置。

- URI – [/apps/\*projectId\*/journeys/\*journeyId\*](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*

### **mobiletargeting:UpdateJourneyState**

取消活动旅程。

- URI – [/apps/\*projectId\*/journeys/\*journeyId\*/state](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*/state

### **mobiletargeting>DeleteJourney**

删除特定旅程。

- URI – [/apps/\*projectId\*/journeys/\*journeyId\*](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/journeys/*journeyId*

## 消息模板

以下权限与为您的 Amazon Pinpoint 账户创建和管理消息模板有关。消息模板 是一组内容和设置，您可以在为任何 Amazon Pinpoint 项目发送的消息中定义、保存和重用它们。

### **mobiletargeting:ListTemplates**

检索有关与您的 Amazon Pinpoint 账户关联的所有消息模板的信息。

- URI – [/templates](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates`

### **mobiletargeting:ListTemplateVersions**

检索有关特定消息模板的所有版本的信息。

- URI – [/templates/template-name/template-type/versions](#)
- 方法 – GET
- 资源 ARN – 不适用

### **mobiletargeting:UpdateTemplateActiveVersion**

将消息模板的特定版本指定为模板的活动版本。

- URI – [/templates/template-name/template-type/active-version](#)
- 方法 – GET
- 资源 ARN – 不适用

### **mobiletargeting:GetEmailTemplate**

检索有关通过电子邮件渠道发送的消息的消息模板的信息。

- URI – [/templates/template-name/email](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/EMAIL`

### **mobiletargeting>CreateEmailTemplate**

为通过电子邮件渠道发送的消息创建消息模板。

- URI – [/templates/template-name/email](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/EMAIL`

## mobiletargeting:UpdateEmailTemplate

更新通过电子邮件渠道发送的消息的现有消息模板。

- URI – [/templates/\*template-name\*/email](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

## mobiletargeting>DeleteEmailTemplate

删除通过电子邮件渠道发送的消息的消息模板。

- URI – [/templates/\*template-name\*/email](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/EMAIL

## mobiletargeting:GetPushTemplate

检索有关通过推送通知渠道发送的消息的消息模板的信息。

- URI – [/templates/\*template-name\*/push](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

## mobiletargeting>CreatePushTemplate

为通过推送通知渠道发送的消息创建消息模板。

- URI – [/templates/\*template-name\*/push](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

## mobiletargeting:UpdatePushTemplate

更新通过推送通知渠道发送的消息的现有消息模板。

- URI – [/templates/\*template-name\*/push](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

### mobiletargeting:DeletePushTemplate

删除通过推送通知渠道发送的消息的消息模板。

- URI – [/templates/\*template-name\*/push](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/PUSH

### mobiletargeting:GetSmsTemplate

检索有关通过短信渠道发送的消息的消息模板的信息。

- URI – [/templates/\*template-name\*/sms](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

### mobiletargeting:CreateSmsTemplate

为通过短信渠道发送的消息创建消息模板。

- URI – [/templates/\*template-name\*/sms](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:templates/*template-name*/SMS

### mobiletargeting:UpdateSmsTemplate

更新通过短信渠道发送的消息的现有消息模板。

- URI – [/templates/\*template-name\*/sms](#)
- 方法 – PUT

- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/SMS`

### **mobiletargeting:DeleteSmsTemplate**

删除通过短信渠道发送的消息的消息模板。

- URI – [/templates/template-name/sms](#)
- 方法 – DELETE
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/SMS`

### **mobiletargeting:GetVoiceTemplate**

检索有关通过语音渠道发送的消息的消息模板的信息。

- URI – [/templates/template-name/voice](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/VOICE`

### **mobiletargeting:CreateVoiceTemplate**

为通过语音渠道发送的消息创建消息模板。

- URI – [/templates/template-name/voice](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/VOICE`

### **mobiletargeting:UpdateVoiceTemplate**

更新通过语音渠道发送的消息的现有消息模板。

- URI – [/templates/template-name/voice](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/VOICE`

### **mobiletargeting>DeleteVoiceTemplate**

删除通过语音渠道发送的消息的消息模板。

- URI – [/templates/\*template-name\*/voice](#)
- 方法 – DELETE
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:templates/template-name/VOICE`

## 消息

以下权限与从您的 Amazon Pinpoint 账户发送消息和推送通知有关。您可以使用 `SendMessage` 和 `SendUsersMessages` 操作将消息发送到特定端点，而不先创建分段和活动。

### **mobiletargeting:SendMessage**

将消息或推送通知发送到特定端点。

- URI – [/apps/\*projectId\*/messages](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/messages`

### **mobiletargeting:SendUsersMessages**

向与特定用户 ID 关联的所有端点发送消息或推送通知。

- URI – [/apps/\*projectId\*/users-messages](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/messages`

## 一次性密码

以下权限与在 Amazon Pinpoint 中发送和验证一次性密码 (OTP) 有关。

### **mobiletargeting:SendOTPMessage**

发送包含一次性密码的文本消息。

- URI – [/apps/\*projectId\*/otp](#)
- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/otp`

## mobiletargeting:VerifyOTPMessage

检查使用 SendOTPMessage 操作生成的一次性密码 (OTP) 的有效性。

- URI – [/apps/projectId/verify-otp](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/verify-otp

## 电话号码验证

以下权限与使用 Amazon Pinpoint 中的电话号码验证服务有关。

## mobiletargeting:PhoneNumberValidate

检索有关电话号码的信息。

- URI – [/phone/number/validate](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:phone/number/validate

## 项目

以下权限与管理您的 Amazon Pinpoint 账户中的项目有关。最初，项目被称为应用程序。出于这些操作的目的，Amazon Pinpoint 应用程序与 Amazon Pinpoint 项目是一回事。

## mobiletargeting:CreateApp

创建 Amazon Pinpoint 项目。

- URI – [/apps](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps

## mobiletargeting>DeleteApp

删除 Amazon Pinpoint 项目。

- URI – [/apps/projectId](#)
- 方法 – DELETE



- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

### **mobiletargeting:GetApp**

检索有关 Amazon Pinpoint 项目的信息。

- URI – [/apps/projectId](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

### **mobiletargeting:GetApps**

检索有关与您的 Amazon Pinpoint 账户关联的所有项目的信息。

- URI – [/apps](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps`

### **mobiletargeting:GetApplicationSettings**

检索 Amazon Pinpoint 项目的默认设置。

- URI – [/apps/projectId/settings](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

### **mobiletargeting:UpdateApplicationSettings**

更新 Amazon Pinpoint 项目的默认设置。

- URI – [/apps/projectId/settings](#)
- 方法 – PUT
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId`

## 推荐器模型

以下权限与管理 Amazon Pinpoint 配置以从推荐器模型中检索和处理推荐数据有关。推荐器模型 是一种机器学习模型，可以查找数据中的模式以预测和生成个性化建议。

### **mobiletargeting:CreateRecommenderConfiguration**

为推荐器模型创建 Amazon Pinpoint 配置。

- URI – [/recommenders](#)

- 方法 – POST
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:recommenders`

### **mobiletargeting:GetRecommenderConfigurations**

检索有关与 Amazon Pinpoint 账户关联的所有推荐器模型配置的信息。

- URI – [/recommenders](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:recommenders`

### **mobiletargeting:GetRecommenderConfiguration**

检索有关某个推荐器模型的单独 Amazon Pinpoint 配置的信息。

- URI – [/recommenders/recommenderId](#)
- 方法 – GET
- 资源 ARN –  
`arn:aws:mobiletargeting:region:accountId:recommenders/recommenderId`

### **mobiletargeting:UpdateRecommenderConfiguration**

更新推荐器模型的 Amazon Pinpoint 配置。

- URI – [/recommenders/recommenderId](#)
- 方法 – PUT
- 资源 ARN –  
`arn:aws:mobiletargeting:region:accountId:recommenders/recommenderId`

### **mobiletargeting>DeleteRecommenderConfiguration**

删除推荐器模型的 Amazon Pinpoint 配置。

- URI – [/recommenders/recommenderId](#)
- 方法 – DELETE
- 资源 ARN –  
`arn:aws:mobiletargeting:region:accountId:recommenders/recommenderId`

## 分段

以下权限与管理您的 Amazon Pinpoint 账户中的分段有关。在 Amazon Pinpoint 中，分段 是您的活动中具有您定义的某些共同属性的接收人组。

## mobiletargeting:CreateSegment

创建分段。要允许用户通过从 Amazon Pinpoint 外部导入端点数据来创建分段，请允许 mobiletargeting:CreateImportJob 操作。

- URI – [/apps/projectId/segments](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

## mobiletargeting>DeleteSegment

删除分段。

- URI – [/apps/projectId/segments/segmentId](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

## mobiletargeting:GetSegment

检索有关特定分段的信息。

- URI – [/apps/projectId/segments/segmentId](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*

## mobiletargeting:GetSegmentExportJobs

检索有关为分段导出端点定义的任务的信息。

- URI – [/apps/projectId/segments/segmentId/jobs/export](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/segments/*segmentId*/jobs/export

## mobiletargeting:GetSegments

检索有关项目的所有分段的信息。

- URI – [/apps/projectId/segments](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*

## mobiletargeting:GetSegmentImportJobs

检索有关通过从 Amazon S3 导入端点定义来创建分段的任务的信息。

- URI – [/apps/projectId/segments/segmentId/jobs/import](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/segments/segmentId

## mobiletargeting:GetSegmentVersion

检索有关特定分段版本的信息。

- URI – [/apps/projectId/segments/segmentId/versions/versionId](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/segments/segmentId

## mobiletargeting:GetSegmentVersions

检索有关分段的当前和以前版本的信息。

- URI – [/apps/projectId/segments/segmentId/versions](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/segments/segmentId

## mobiletargeting:UpdateSegment

更新特定分段。

- URI – [/apps/projectId/segments/segmentId](#)
- 方法 – PUT
- 资源 ARN – arn:aws:mobiletargeting:region:accountId:apps/projectId/segments/segmentId

## 标签

以下权限与查看和管理 Amazon Pinpoint 资源的标签有关。

## mobiletargeting:ListTagsForResource

检索有关与项目、活动、消息模板或分段关联的标签的信息。

- URI – [/tags/resource-arn](#)
- 方法 – GET
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:\*

### **mobiletargeting:TagResource**

向项目、活动、消息模板或分段添加一个或多个标签。

- URI – [/tags/resource-arn](#)
- 方法 – POST
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:\*

### **mobiletargeting:UntagResource**

从项目、活动、消息模板或分段中删除一个或多个标签。

- URI – [/tags/resource-arn](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:\*

## 用户

以下权限与管理用户有关。在 Amazon Pinpoint 中，用户对应于收到您的消息的个人。单个用户可能与多个端点关联。

### **mobiletargeting>DeleteUserEndpoints**

删除与用户 ID 关联的所有端点。

- URI – [/apps/projectId/users/userId](#)
- 方法 – DELETE
- 资源 ARN – arn:aws:mobiletargeting:*region*:*accountId*:apps/*projectId*/users/*userId*

### **mobiletargeting:GetUserEndpoints**

检索与用户 ID 关联的所有端点的相关信息。

- URI – [/apps/projectId/users/userId](#)
- 方法 – GET
- 资源 ARN – `arn:aws:mobiletargeting:region:accountId:apps/projectId/users/userId`

## Amazon Pinpoint SMS 和 Voice API 第 1 版操作

本节介绍 Amazon Pinpoint SMS 和 Voice API 中提供的功能的操作。这是一个补充 API，它提供在 Amazon Pinpoint 中使用和管理短信和语音渠道的高级选项。要了解有关此 API 的更多信息，请参阅 [Amazon Pinpoint SMS 和 Voice API 参考](#)。

### **sms-voice:CreateConfigurationSet**

创建配置集以发送语音消息。

- URI – `/sms-voice/configuration-sets`
- 方法 – POST
- 资源 ARN – 不可用。使用 \*。

### **sms-voice>DeleteConfigurationSet**

删除配置集以发送语音消息。

- URI — `/sms-voice/配置集/ConfigurationSetName`
- 方法 – DELETE
- 资源 ARN – 不可用。使用 \*。

### **sms-voice:GetConfigurationSetEventDestinations**

检索有关配置集及其包含的事件目标的信息。

- URI — `/sms-voice/配置集//ConfigurationSetNameevent-destinations`
- 方法 – GET
- 资源 ARN – 不可用。使用 \*。

### **sms-voice:CreateConfigurationSetEventDestination**

为语音事件创建事件目标。

- URI — /sms-voice/配置集//*ConfigurationSetName*event-destinations
- 方法 – POST
- 资源 ARN – 不可用。使用 \*。

### **sms-voice:UpdateConfigurationSetEventDestination**

为语音事件更新事件目标。

- URI — /sms-voice/配置集/ /event-destinations/  
*ConfigurationSetNameEventDestinationName*
- 方法 – PUT
- 资源 ARN – 不可用。使用 \*。

### **sms-voice>DeleteConfigurationSetEventDestination**

删除语音事件的事件目标。

- URI — /sms-voice/配置集/ /event-destinations/  
*ConfigurationSetNameEventDestinationName*
- 方法 – DELETE
- 资源 ARN – 不可用。使用 \*。

### **sms-voice:SendVoiceMessage**

创建和发送语音消息。

- URI – /sms-voice/voice/message
- 方法 – POST
- 资源 ARN – 不可用。使用 \*。

## Amazon Pinpoint SMS 和 Voice API 第 2 版操作

本节介绍 Amazon Pinpoint SMS 和 Voice API 中提供的功能的操作。这是一个补充 API，提供使用和管理短信和语音渠道的高级选项。有关第 2 版中可用操作的完整列表，请参阅 [Amazon Pinpoint SMS 和 Voice API 第 2 版参考](#)。

### **sms-voice:AssociateOriginationIdentity**

将指定的源身份与池相关联。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountyCode`

### **sms-voice:CreateConfigurationSet**

创建新的配置集。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:CreateEventDestination**

在配置集中创建新的事件目标。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:CreateOptOutList**

创建新的退出列表。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`

### **sms-voice:CreatePool**

创建新池并将指定的源身份关联到该池。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountyCode`

### **sms-voice>DeleteConfigurationSet**

删除现有的配置集

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice>DeleteDefaultMessageType**

删除配置集上的现有默认消息类型。



- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:DeleteDefaultSenderId**

删除配置集上的现有默认发件人 ID。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/configuration-set/configurationSetName`

### **sms-voice:DeleteEventDestination**

删除现有事件目标。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:DeleteKeyword**

从源电话号码或池中删除现有关键字。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`

### **sms-voice:DeleteOptedOutNumber**

从指定的退出列表中删除现有的退出目标电话号码。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`

### **sms-voice:DeleteOptOutList**

删除现有的退出列表。退出列表中所有已选择退出的电话号码都将被删除。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`

### **sms-voice:DeletePool**

删除现有池。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`

### **sms-voice:DeleteTextMessageSpendLimitOverride**

删除用于发送文本消息的账户级别的每月支出限额覆盖。

- 资源 ARN – 不可用。使用 \*。

## **sms-voice:DeleteVoiceMessageSpendLimitOverride**

删除用于发送语音消息的账户级别的每月支出限额覆盖。

- 资源 ARN – 不可用。使用 \*。

## **sms-voice:DescribeAccountAttributes**

描述您 AWS 账户的属性。

- 资源 ARN – 不可用。使用 \*。

## **sms-voice:DescribeAccountLimits**

描述您的账户当前的 Amazon Pinpoint SMS 和 Voice V2 资源限额。

- 资源 ARN – 不可用。使用 \*。

## **sms-voice:DescribeConfigurationSets**

描述您的账户中的指定或全部配置集。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

## **sms-voice:DescribeKeywords**

描述您的源电话号码或池上的指定或全部关键字。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`

## **sms-voice:DescribeOptedOutNumbers**

描述退出列表中指定或全部退出目标号码。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`

## **sms-voice:DescribeOptOutLists**

描述您的账户中指定或全部退出列表。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`

## **sms-voice:DescribePhoneNumbers**

描述您的账户中指定的或全部源电话号码。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`

## **sms-voice:DescribePools**

检索与您的 AWS 账户关联的指定或全部池。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`

## **sms-voice:DescribeSenderIds**

描述与您的 AWS 账户 SenderIds 关联的指定 SenderIds 或所有关联项。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountryCode`

## **sms-voice:DescribeSpendLimits**

描述发送语音和文本消息的当前 Amazon Pinpoint 每月支出限额。

- 资源 ARN – 不可用。使用 \*。

## **sms-voice:DisassociateOriginationIdentity**

从现有池中移除指定的源身份。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountryCode`

## **sms-voice:ListPoolOriginationIdentities**

显示池中的源电话号码。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`

## **sms-voice:ListTagsForResource**

列出与资源相关的标签。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountryCode`

## sms-voice:PutKeyword

在源电话号码或池上添加或更新关键字。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:phone-number/*phoneNumberId*
- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:pool/*poolId*

## sms-voice:PutOptedOutNumber

将目标电话号码添加到退出列表中。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:opt-out-list/*optOutListName*

## sms-voice:ReleasePhoneNumber

从您的 Amazon Pinpoint 账户中删除源电话号码。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:phone-number/*phoneNumberId*

## sms-voice:RequestPhoneNumber

请求在您的账户中添加源电话号码。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:phone-number/*phoneNumberId*
- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:pool/*poolId*

## sms-voice:SendTextMessage

发送短信。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:phone-number/*phoneNumberId*
- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:pool/*poolId*
- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:sender-id/*senderId*/*isoCountryCode*

## sms-voice:SendVoiceMessage

发送语音消息。

- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:phone-number/*phoneNumberId*
- 资源 ARN – arn:aws:sms-voice:*region*:*accountId*:pool/*poolId*

## sms-voice:SetDefaultMessageType

设置短信的默认消息类型。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:SetDefaultSenderId**

设置语音消息的默认发件人 ID 值。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:SetTextMessageSpendLimitOverride**

设置短信的每月支出限额。

- 资源 ARN – 不可用。使用 \*。

### **sms-voice:SetVoiceMessageSpendLimitOverride**

为语音消息设置每月支出限额。

- 资源 ARN – 不可用。使用 \*。

### **sms-voice:TagResource**

将标签添加到资源中。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountryCode`

### **sms-voice:UntagResource**

从资源中删除标签。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:opt-out-list/optOutListName`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`
- 资源 ARN – `arn:aws:sms-voice:region:accountId:sender-id/senderId/isoCountryCode`

### **sms-voice:UpdateEventDestination**

更新现有事件目标。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:configuration-set/configurationSetName`

### **sms-voice:UpdatePhoneNumber**

更新源电话号码的配置。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:phone-number/phoneNumberId`

### **sms-voice:UpdatePool**

更新现有的电话号码池。

- 资源 ARN – `arn:aws:sms-voice:region:accountId:pool/poolId`

## Amazon Pinpoint 基于身份的策略示例

默认情况下，用户和角色没有创建或修改 Amazon Pinpoint 资源的权限，他们也无法使用 AWS Management Console AWS CLI、或 AWS API 执行任务。IAM 管理员必须创建 IAM 策略，以授权用户和角色对其所需的资源执行特定的 API 操作。然后，管理员必须将这些策略附加到需要这些权限的用户或组。

要了解如何使用这些示例 JSON 策略文档创建 IAM 基于身份的策略，请参阅《IAM 用户指南》中的[在 JSON 选项卡上创建策略](#)。

### 主题

- [策略最佳实践](#)
- [使用 Amazon Pinpoint 控制台](#)
- [示例：访问单个 Amazon Pinpoint 项目](#)
- [示例：基于标签查看 Amazon Pinpoint 资源](#)
- [示例：允许用户查看他们自己的权限](#)
- [示例：提供对 Amazon Pinpoint API 操作的访问权限](#)

- [示例：提供对 Amazon Pinpoint SMS 和 Voice API 操作的访问权限](#)
- [示例：将 Amazon Pinpoint 项目的访问权限限制为特定 IP 地址](#)
- [示例：基于标签限制 Amazon Pinpoint 的访问权限](#)
- [示例：允许 Amazon Pinpoint 使用在 Amazon SES 中验证的身份发送电子邮件](#)

## 策略最佳实践

基于身份的策略确定某个人能否在您的账户中创建、访问或删除 Amazon Pinpoint 资源。这些操作可能会使 AWS 账户产生成本。创建或编辑基于身份的策略时，请遵循以下准则和建议：

- 开始使用 AWS 托管策略并转向最低权限策略 — 要开始向用户和工作负载授予权限，请使用为许多常见用例授予权限的 AWS 托管策略。它们在你的版本中可用 AWS 账户。我们建议您通过定义针对您的用例的 AWS 客户托管策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的 [AWS 托管策略](#) 或 [工作职能的 AWS 托管策略](#)。
- 应用最低权限 – 在使用 IAM policy 设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 [IAM 中的策略和权限](#)。
- 使用 IAM policy 中的条件进一步限制访问权限 – 您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果服务操作是通过特定 AWS 服务的（例如）使用的，则也可以使用条件来授予对服务操作的访问权限 AWS CloudFormation。有关更多信息，请参阅《IAM 用户指南》中的 [IAM JSON 策略元素：条件](#)。
- 使用 IAM Access Analyzer 验证您的 IAM policy，以确保权限的安全性和功能性 – IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM policy 语言 (JSON) 和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的 [IAM Access Analyzer 策略验证](#)。
- 需要多重身份验证 (MFA)-如果 AWS 账户您的场景需要 IAM 用户或根用户，请启用 MFA 以提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的 [配置受 MFA 保护的 API 访问](#)。

有关 IAM 中的最佳实践的更多信息，请参阅《IAM 用户指南》中的 [IAM 中的安全最佳实践](#)。

## 使用 Amazon Pinpoint 控制台

要访问 Amazon Pinpoint 控制台，您必须具有一组最低权限许可。这些权限必须允许您列出和查看有关您 AWS 账户中的 Amazon Pinpoint 资源的详细信息。如果您创建的基于身份的策略所应用的许可比

最低要求许可严格，则对于附加了该策略的实体（用户或角色），控制台将不能提供预期功能。为确保这些实体可以使用 Amazon Pinpoint 控制台，可为其附加另外的策略。有关更多信息，请参阅《IAM 用户指南》中的[为用户添加权限](#)。

以下示例策略提供对特定 AWS 地区的 Amazon Pinpoint 控制台的只读访问权限。其中包括对 Amazon Pinpoint 控制台所依赖的其他服务的只读访问权限，例如 Amazon Simple Email Service (Amazon SES)、IAM 和 Amazon Kinesis。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UseConsole",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:Get*",
        "mobiletargeting:List*"
      ],
      "Resource": "arn:aws:mobiletargeting:region:accountId:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "firehose:ListDeliveryStreams",
        "iam:ListRoles",
        "kinesis:ListStreams",
        "s3:List*",
        "ses:Describe*",
        "ses:Get*",
        "ses:List*",
        "sns:ListTopics"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        }
      }
    }
  ]
}
```

在前面的策略示例中，将##替换为 AWS 区域名称，并将 `accountId` 替换为您的账户 ID。AWS



对于仅调用 AWS CLI 或 AWS API 的用户，您无需为其设置最低控制台权限。相反，只允许访问与其尝试执行的 API 操作相匹配的操作。

## 示例：访问单个 Amazon Pinpoint 项目

您还可以创建仅为特定项目提供访问权限的只读策略。以下示例策略允许用户登录到控制台并查看项目列表。用户还可查看有关 Amazon Pinpoint 控制台所依赖的其他 AWS 服务的相关资源的信息，例如 Amazon SES、IAM 和 Amazon Kinesis。但是，该策略只允许用户查看策略中指定的项目的信息。您可以修改此政策以允许访问其他项目或 AWS 区域。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewProject",
      "Effect": "Allow",
      "Action": "mobiletargeting:GetApps",
      "Resource": "arn:aws:mobiletargeting:region:accountId:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:Get*",
        "mobiletargeting:List*"
      ],
      "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/projectId",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/*",
        "arn:aws:mobiletargeting:region:accountId:reports"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "ses:Get*",
        "kinesis:ListStreams",
        "firehose:ListDeliveryStreams",
        "iam:ListRoles",
        "ses:List*",
        "sns:ListTopics",
        "ses:Describe*",
        "s3:List*"
      ],
    },
  ],
}
```

```

    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "accountId"
      }
    }
  }
]
}

```

在前面的示例中，将 *##* 替换为 AWS 区域名称，将 *ac coun tID* 替换为 AWS 您的账户 ID，并将 *P rojectID* 替换为您想要提供访问权限的亚马逊 Pinpoint 项目的 ID。

同样，您可以创建策略，向 AWS 账户中的用户授予对您的一个 Amazon Pinpoint 项目（例如具有项目 ID 的项目）的有限写入权限。810c7aab86d42fb2b56c8c966example 在此案例中，您希望允许用户查看、添加和更新项目组件，例如分段和活动，但不允许删除任何组件。

除了授予关于 `mobiletargeting:Get` 和 `mobiletargeting:List` 操作的权限，您还可以创建一个策略，来授予关于以下操作的权限：`mobiletargeting:Create`、`mobiletargeting:Update` 和 `mobiletargeting:Put`。这些是创建和管理大多数项目组件所需的额外权限。例如：

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "LimitedWriteProject",
      "Effect": "Allow",
      "Action": "mobiletargeting:GetApps",
      "Resource": "arn:aws:mobiletargeting:region:accountId:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:Get*",
        "mobiletargeting:List*",
        "mobiletargeting:Create*",
        "mobiletargeting:Update*",
        "mobiletargeting:Put*"
      ],
      "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/810c7aab86d42fb2b56c8c966example",

```

```

"arn:aws:mobiletargeting:region:accountId:apps/810c7aab86d42fb2b56c8c966example/*",
    "arn:aws:mobiletargeting:region:accountId:reports"
  ]
},
{
  "Effect": "Allow",
  "Action": [
    "ses:Get*",
    "kinesis:ListStreams",
    "firehose:ListDeliveryStreams",
    "iam:ListRoles",
    "ses:List*",
    "sns:ListTopics",
    "ses:Describe*",
    "s3:List*"
  ],
  "Resource": "*",
  "Condition": {
    "StringEquals": {
      "aws:SourceAccount": "accountId"
    }
  }
}
]
}

```

## 示例：基于标签查看 Amazon Pinpoint 资源

您可以在基于身份的策略中使用条件，以基于标签控制对于 Amazon Pinpoint 资源的访问。此示例策略展示了如何创建此类策略，以允许查看 Amazon Pinpoint 资源。不过，仅当 Owner 资源标签具有该用户的用户名的值时，才会授予权限。此策略还授予在控制台上完成此操作的必要权限。

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ListResources",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:Get*",
        "mobiletargeting:List*"
      ],

```

```

    "Resource": "*"
  },
  {
    "Sid": "ViewResourceIfOwner",
    "Effect": "Allow",
    "Action": [
      "mobiletargeting:Get*",
      "mobiletargeting:List*"
    ],
    "Resource": "arn:aws:mobiletargeting:*:*:*",
    "Condition": {
      "StringEquals": {
        "aws:ResourceTag/Owner": "userName"
      },
      "StringEquals": {
        "aws:SourceAccount": "accountId"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws:mobiletargeting:region:accountId:*"
      }
    }
  }
]
}

```

您可以将此策略附加到您的账户中的用户。如果名为 richard-roe 的用户想要查看某个 Amazon Pinpoint 资源，该资源必须标记为 Owner=richard-roe 或 owner=richard-roe。否则，将拒绝其访问。条件标签键 Owner 匹配 Owner 和 owner，因为条件键名称不区分大小写。有关更多信息，请参阅《IAM 用户指南》中的 [IAM JSON 策略元素：条件](#)。

### 示例：允许用户查看他们自己的权限

该示例展示了如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上或使用 AWS CLI 或 AWS API 以编程方式完成此操作的权限。

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",

```

```

        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
},
{
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
    ],
    "Resource": "*"
}
]
}

```

## 示例：提供对 Amazon Pinpoint API 操作的访问权限

本节介绍允许访问 Amazon Pinpoint API ( Amazon Pinpoint 的主要 API ) 所提供的功能的策略示例。要了解有关此 API 的更多信息，请参阅 [Amazon Pinpoint API 参考](#)。

### 只读访问权限

以下示例策略允许以只读方式访问您在特定 AWS 地区的 Amazon Pinpoint 账户中的所有资源。

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewAllResources",
            "Effect": "Allow",
            "Action": [
                "mobiletargeting:Get*",
                "mobiletargeting:List*"
            ]
        }
    ]
}

```

```

    ],
    "Resource": "arn:aws:mobiletargeting:region:accountId:*"
  }
]
}

```

在前面的示例中，将 **##** 替换为 AWS 区域名称，并将 **ac*countID*** 替换为您的账户 ID。AWS 管理员访问权限

以下示例策略允许对您的 Amazon Pinpoint 账户中的所有 Amazon Pinpoint 操作和资源进行完全访问：

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FullAccess",
      "Effect": "Allow",
      "Action": [
        "mobiletargeting:*"
      ],
      "Resource": "arn:aws:mobiletargeting:region:accountId:*"
    }
  ]
}

```

在前面的示例中，将 **accountId** 替换为您的 AWS 账户 ID。

### 示例：提供对 Amazon Pinpoint SMS 和 Voice API 操作的访问权限

本节介绍允许访问 Amazon Pinpoint SMS 和 Voice API 提供的功能的策略示例。这是一个补充 API，它提供在 Amazon Pinpoint 中使用和管理短信和语音渠道的高级选项。要了解有关此 API 的更多信息，请参阅 [Amazon Pinpoint SMS 和 Voice API 参考](#)。

#### 只读访问权限

以下示例策略允许对您账户中的所有 Amazon Pinpoint 短信和语音 API 操作和资源进行只读访问：  
AWS

```

{
  "Version": "2012-10-17",
  "Statement": [

```

```

    {
      "Sid": "SMSVoiceReadOnly",
      "Effect": "Allow",
      "Action": [
        "sms-voice:Get*",
        "sms-voice:List*"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:sms-voice:region:accountId:*"
        }
      }
    }
  ]
}

```

## 管理员访问权限

以下示例策略允许完全访问您账户中的所有 Amazon Pinpoint 短信和语音 API 操作和资源：AWS

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "SMSVoiceFullAccess",
      "Effect": "Allow",
      "Action": [
        "sms-voice:*",
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:sms-voice:region:accountId:*"
        }
      }
    }
  ]
}

```

```
}
```

## 示例：将 Amazon Pinpoint 项目的访问权限限制为特定 IP 地址

以下示例策略向任何用户授予对指定项目 (*projectId*) 执行任何 Amazon Pinpoint 操作的权限。但是，请求必须来自在条件中指定的 IP 地址范围。

此语句中的条件确定允许的 Internet 协议版本 4 (IPv4) 地址范围为 54.240.143.\*，只有一个例外：54.240.143.188。该 Condition 块使用 IpAddress 和 NotIpAddress 条件和 aws:SourceIp 条件键，后者是一个 AWS 宽范围的条件键。有关这些条件键的更多信息，请参阅《IAM 用户指南》中的[在策略中指定条件](#)。aws:SourceIp IPv4 值使用标准 CIDR 表示法。有关更多信息，请参阅《IAM 用户指南》中的[IP 地址条件运算符](#)。

```
{
  "Version": "2012-10-17",
  "Id": "AMZPinpointPolicyId1",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "mobiletargeting:*",
      "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/projectId",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
      ],
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": "54.240.143.0/24"
        },
        "NotIpAddress": {
          "aws:SourceIp": "54.240.143.188/32"
        }
      }
    }
  ]
}
```

## 示例：基于标签限制 Amazon Pinpoint 的访问权限

以下示例策略授予对指定项目 (*projectId*) 执行任何 Amazon Pinpoint 操作的权限。但是，只有当请求来自其名称是项目的 Owner 资源标签中的值的用户（如条件中所指定）时，才会授予权限。



```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ModifyResourceIfOwner",
      "Effect": "Allow",
      "Action": "mobiletargeting:*",
      "Resource": [
        "arn:aws:mobiletargeting:region:accountId:apps/projectId",
        "arn:aws:mobiletargeting:region:accountId:apps/projectId/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Owner": "userName"
        }
      }
    }
  ]
}

```

### 示例：允许 Amazon Pinpoint 使用在 Amazon SES 中验证的身份发送电子邮件

当您通过 Amazon Pinpoint 控制台验证电子邮件身份（例如电子邮件地址或域）时，系统会自动配置该身份，以便 Amazon Pinpoint 和 Amazon SES 都可以使用该身份。但是，如果您通过 Amazon SES 验证电子邮件身份，并且想要在 Amazon Pinpoint 中使用该身份，则必须对该身份应用策略。

以下示例策略授予 Amazon Pinpoint 使用已通过 Amazon SES 验证的电子邮件身份发送电子邮件的权限。

```

{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "PinpointEmail",
      "Effect": "Allow",
      "Principal": {
        "Service": "pinpoint.amazonaws.com"
      },
      "Action": "ses:*",
      "Resource": "arn:aws:ses:region:accountId:identity/emailId",
      "Condition": {
        "StringEquals": {

```

```

        "aws:SourceAccount": "accountId"
    },
    "StringLike": {
        "aws:SourceArn": "arn:aws:mobiletargeting:region:accountId:apps/*"
    }
}
]
}

```

如果您在 AWS GovCloud ( 美国西部 ) 地区使用 Amazon Pinpoint , 请改用以下政策示例 :

```

{
  "Version": "2008-10-17",
  "Statement": [
    {
      "Sid": "PinpointEmail",
      "Effect": "Allow",
      "Principal": {
        "Service": "pinpoint.amazonaws.com"
      },
      "Action": "ses:*",
      "Resource": "arn:aws-us-gov:ses:us-gov-west-1:accountId:identity/emailId",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "StringLike": {
          "aws:SourceArn": "arn:aws-us-gov:mobiletargeting:us-gov-
west-1:accountId:apps/*"
        }
      }
    }
  ]
}

```

## 用于常见 Amazon Pinpoint 任务的 IAM 角色

[IAM 角色](#) 是一种 AWS Identity and Access Management (IAM) 身份，您可以在自己的 AWS 账户中创建并授予特定权限。IAM 角色是一种具有权限策略的 AWS 身份，该策略决定了该身份可以做什么和不能做什么 AWS。但是，角色并不是唯一地与某个人关联，而是任何需要它的人都可以代入。

此外，角色没有长期关联的标准凭证。相反，它为会话提供临时安全凭证。您可以使用 IAM 角色将访问权限委托给通常无法访问您的 AWS 资源的用户、应用程序、应用程序或服务。

出于这些原因，您可以使用 IAM 角色将 Amazon Pinpoint 与您的账户的某些 AWS 服务和资源相集成。例如，您可能想要允许 Amazon Pinpoint 访问您存储在 Amazon Simple Storage Service (Amazon S3) 存储桶中并想要用于分段的端点定义。或者，您可能想要允许 Amazon Pinpoint 将事件数据流式传输到您的账户的 Amazon Kinesis 流。同样，您可能希望使用 IAM 角色来允许网络或移动应用程序注册终端节点或报告 Amazon Pinpoint 项目的使用数据，而不必在应用程序中嵌入 AWS 密钥（这些密钥可能难以轮换，用户有可能提取密钥）。

对于这些情况，您可以使用 IAM 角色委派对 Amazon Pinpoint 的访问权限。本节解释并提供了将 IAM 角色与其他 AWS 服务配合使用的常见 Amazon Pinpoint 任务示例。有关具体如何将 IAM 角色与 Web 和移动应用程序配合使用的信息，请参阅《IAM 用户指南》中的[向经过外部身份验证的用户（身份联合验证）提供访问权限](#)。

## 主题

- [用于导入端点或分段的 IAM 角色](#)
- [用于导出端点或分段的 IAM 角色](#)
- [用于从 Amazon Personalize 检索建议的 IAM 角色](#)
- [用于将事件流式传输到 Kinesis 的 IAM 角色](#)

## 用于导入端点或分段的 IAM 角色

借助 Amazon Pinpoint，您可以通过从账户中的亚马逊简单存储服务 (Amazon S3) 存储桶导入终端节点定义来定义用户细分。AWS 导入之前，必须向 Amazon Pinpoint 委派所需权限。为此，您需要创建一个 AWS Identity and Access Management (IAM) 角色并将以下策略附加到该角色：

- AmazonS3ReadOnlyAccess AWS 托管策略。此策略由创建和管理 AWS，它授予对您的 Amazon S3 存储桶的只读访问权限。
- 允许 Amazon Pinpoint 代入角色的信任策略。

在创建该角色之后，您可以使用 Amazon Pinpoint 从 Amazon S3 存储桶导入分段。有关使用控制台创建存储桶、创建端点文件以及导入分段的信息，请参阅《Amazon Pinpoint 用户指南》中的[导入分段](#)。有关如何使用以编程方式导入区段的示例 AWS SDK for Java，请参阅本指南[导入客户细分](#)中的。

## 创建 IAM 角色 (AWS CLI)

完成以下步骤，使用 AWS Command Line Interface (AWS CLI) 创建 IAM 角色。如果您尚未安装 AWS CLI，请参阅 [AWS Command Line Interface 用户指南 AWS CLI 中的安装](#)。

### 使用创建 IAM 角色 AWS CLI

1. 创建包含角色的信任策略的 JSON 文件，并在本地保存该文件。可以复制本主题中提供的信任策略。
2. 在命令行上，使用 `create-role` 命令创建角色并附加信任策略：

```
aws iam create-role --role-name PinpointSegmentImport --assume-role-policy-document file://PinpointImportTrustPolicy.json
```

在 `file://` 前缀后面，指定包含信任策略的 JSON 文件的路径。

运行此命令后，您会在终端上看到类似如下的输出：

```
{
  "Role": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",
          "Effect": "Allow",
          "Principal": {
            "Service": "pinpoint.amazonaws.com"
          },
          "Condition": {
            "StringEquals": {
              "aws:SourceAccount": "accountId"
            },
            "ArnLike": {
              "aws:SourceArn":
                "arn:aws:mobiletargeting:region:accountId:apps/application-id"
            }
          }
        }
      ]
    },
    "RoleId": "AIDACKCEVSQ6C2EXAMPLE",
```

```
"CreateDate": "2016-12-20T00:44:37.406Z",
"RoleName": "PinpointSegmentImport",
"Path": "/",
"Arn": "arn:aws:iam::accountId:role/PinpointSegmentImport"
}
}
```

3. 使用 [attach-role-policy](#) 命令将 AmazonS3ReadOnlyAccess AWS 托管策略附加到角色：

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess --role-name PinpointSegmentImport
```

## 用于导出端点或分段的 IAM 角色

您可以通过创建导出任务来获取端点列表。创建导出任务时，必须指定项目 ID，并可以选择指定分段 ID。然后，Amazon Pinpoint 将与项目或分段关联的端点列表导出到 Amazon Simple Storage Service (Amazon S3) 存储桶。生成的文件包含采用 JSON 格式的端点列表及其属性（如渠道、地址、选择加入/选择退出状态、创建日期和端点 ID）。

要创建导出任务，必须配置一个 IAM 角色，以允许 Amazon Pinpoint 向 Amazon S3 存储桶中写入。配置角色的过程包含以下两个步骤：

1. 创建 IAM 策略，以允许实体（此处为 Amazon Pinpoint）写入到特定 Amazon S3 存储桶。
2. 创建 IAM 角色并向其附加此策略。

本主题包含完成这两个步骤的过程。这些过程假定您已创建了 Amazon S3 存储桶，并在该存储桶创建了文件夹以用于存储导出的分段。有关创建存储桶的信息，请参阅《Amazon Simple Storage Service 用户指南》中的 [创建存储桶](#)。

这些过程同样假定您已安装和配置 AWS Command Line Interface (AWS CLI)。有关设置的信息 AWS CLI，请参阅 [《AWS Command Line Interface 用户指南》AWS CLI 中的安装](#)。

### 步骤 1：创建 IAM 策略

IAM 策略用于定义实体（如身份或资源）的权限。要创建用于导出 Amazon Pinpoint 端点的角色，必须创建一个策略，以授予写入到特定 Amazon S3 存储桶中的特定文件夹的权限。以下策略示例遵循授予最低权限这一安全实践，也就是说，仅授予执行某项任务所需的权限。

## 创建 IAM policy

1. 在文本编辑器中，创建一个新文件。将以下代码粘贴到该文件中：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowUserToSeeBucketListInTheConsole",
      "Action": [
        "s3:ListAllMyBuckets",
        "s3:GetBucketLocation"
      ],
      "Effect": "Allow",
      "Resource": [ "arn:aws:s3:::*" ]
    },
    {
      "Sid": "AllowRootAndHomeListingOfBucket",
      "Action": [
        "s3:ListBucket"
      ],
      "Effect": "Allow",
      "Resource": [ "arn:aws:s3:::example-bucket" ],
      "Condition": {
        "StringEquals": {
          "s3:delimiter": [ "/" ],
          "s3:prefix": [
            "",
            "Exports/"
          ]
        }
      }
    },
    {
      "Sid": "AllowListingOfUserFolder",
      "Action": [
        "s3:ListBucket"
      ],
      "Effect": "Allow",
      "Resource": [ "arn:aws:s3:::example-bucket" ],
      "Condition": {
        "StringLike": {
          "s3:prefix": [
```

```

        "Exports/*"
    ]
}
},
{
    "Sid": "AllowAllS3ActionsInUserFolder",
    "Action": [ "s3:*" ],
    "Effect": "Allow",
    "Resource": [ "arn:aws:s3:::example-bucket/Exports/*" ]
}
]
}

```

在之前的代码中，将 *example-bucket* 的所有实例替换为包含您要向其中导出分段信息的文件夹的 Amazon S3 存储桶的名称。此外，将 *Exports* 的所有实例替换为文件夹本身的名称。

完成后，将文件另存为 `s3policy.json`。

2. 通过使用 AWS CLI，导航到 `s3policy.json` 文件所在的目录。然后，键入以下命令以创建策略：

```
aws iam create-policy --policy-name s3ExportPolicy --policy-document
file://s3policy.json
```

如果策略创建成功，则您会看到类似于以下内容的输出：

```
{
  "Policy": {
    "CreateDate": "2018-04-11T18:44:34.805Z",
    "IsAttachable": true,
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PolicyId": "ANPAJ2YJQRJCG3EXAMPLE",
    "UpdateDate": "2018-04-11T18:44:34.805Z",
    "Arn": "arn:aws:iam::123456789012:policy/s3ExportPolicy",
    "PolicyName": "s3ExportPolicy",
    "Path": "/"
  }
}
```

复制策略的 Amazon 资源名称 (ARN) ( 之前示例中的 `arn:aws:iam::123456789012:policy/s3ExportPolicy` )。在下一部分，当您创建角色时，必须提供此 ARN。

### Note

如果弹出一条消息说，您的账户无权执行 `CreatePolicy` 操作，则需要将一个允许您创建新的 IAM 策略和角色的策略附加到用户。有关更多信息，请参阅《IAM 用户指南》中的 [添加和删除 IAM 身份权限](#)。

## 步骤 2：创建 IAM 角色

创建 IAM 策略后，您可以创建一个角色并向其附加该策略。每个 IAM 角色都包含一个信任策略，即，用来指定哪些实体被允许代入角色的一组规则。在本部分中，您将创建一个信任策略，以允许 Amazon Pinpoint 代入角色。接下来，您可以创建角色本身，然后附加您在前一部分创建的策略。

### 创建 IAM 角色

1. 在文本编辑器中，创建一个新文件。将以下代码粘贴到该文件中：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pinpoint.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "accountId"
        },
        "ArnLike": {
          "aws:SourceArn":
            "arn:aws:mobiletargeting:region:accountId:apps/applicationId"
        }
      }
    }
  ]
}
```



```
}
```

将该文件保存为 `trustpolicy.json`。

2. 通过使用 AWS CLI，导航到 `trustpolicy.json` 文件所在的目录。然后，键入以下命令以创建新角色：

```
aws iam create-role --role-name s3ExportRole --assume-role-policy-document  
file://trustpolicy.json
```

3. 在命令行中，键入以下命令，将您在前一部分创建的策略附加到您刚创建的角色：

```
aws iam attach-role-policy --policy-arn arn:aws:iam::123456789012:policy/  
s3ExportPolicy --role-name s3ExportRole
```

在前面的命令中，将 `arn:aws:iam::123456789012:policy/s3ExportPolicy` 的 ARN。

## 用于从 Amazon Personalize 检索建议的 IAM 角色

您可以配置 Amazon Pinpoint，以从部署为 Amazon Personalize 活动的 Amazon Personalize 解决方案中检索建议数据。您可以使用该数据根据每个消息接收人的属性和行为向接收人发送个性化建议。要了解更多信息，请参阅《Amazon Pinpoint 用户指南》中的[机器学习模型](#)。

要从一个 Amazon Personalize 活动中检索建议数据，必须先创建一个 AWS Identity and Access Management (IAM) 角色，以允许 Amazon Pinpoint 从该活动中检索数据。当您使用控制台在 Amazon Pinpoint 中设置推荐器模型时，Amazon Pinpoint 可以自动为您创建此角色。您也可以手动创建此角色。

要手动创建此角色，请使用 IAM API 完成以下步骤：

1. 创建一个 IAM 策略，以允许实体（此处为 Amazon Pinpoint）从 Amazon Personalize 活动中检索建议数据。
2. 创建 IAM 角色并向其附加此 IAM 策略。

本主题介绍如何使用 AWS Command Line Interface (AWS CLI) 完成这些步骤。它假定您已经创建了 Amazon Personalize 解决方案并将其部署为 Amazon Personalize 活动。有关创建和部署活动的信息，请参阅《Amazon Personalize 开发人员指南》中的[创建活动](#)。

本主题还假定您已安装并配置了 AWS CLI。有关设置的信息 AWS CLI，请参阅 [《AWS Command Line Interface 用户指南》AWS CLI 中的安装](#)。

## 步骤 1：创建 IAM 策略

IAM 策略定义实体（例如身份或资源）的权限。要创建一个角色以允许 Amazon Pinpoint 从 Amazon Personalize 活动中检索建议数据，必须先为该角色创建一个 IAM 策略。该策略需要允许 Amazon Pinpoint：

- 检索活动部署的解决方案的配置信息 (DescribeSolution)。
- 检查活动的状态 (DescribeCampaign)。
- 从活动中检索建议数据 (GetRecommendations)。

在以下过程中，示例策略允许特定 Amazon Personalize 活动部署的特定 Amazon Personalize 解决方案进行该访问。

### 创建 IAM policy

1. 在文本编辑器中，创建一个新文件。将以下代码粘贴到该文件中：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RetrieveRecommendationsOneCampaign",
      "Effect": "Allow",
      "Action": [
        "personalize:DescribeSolution",
        "personalize:DescribeCampaign",
        "personalize:GetRecommendations"
      ],
      "Resource": [
        "arn:aws:personalize:region:accountId:solution/solutionId",
        "arn:aws:personalize:region:accountId:campaign/campaignId"
      ]
    }
  ]
}
```

在前面的示例中，将## 文本替换为您的信息：

- *region* – 托管 Amazon Personalize 解决方案和活动的 AWS 区域的名称。
  - *accountId* – 您的 AWS 账户 ID。
  - *solutionId* – 活动部署的 Amazon Personalize 解决方案的唯一资源 ID。
  - *campaignId* – 从中检索建议数据的 Amazon Personalize 活动的唯一资源 ID。
2. 完成后，将文件另存为 `RetrieveRecommendationsPolicy.json`。
  3. 通过使用命令行界面，导航到保存 `RetrieveRecommendationsPolicy.json` 文件的目录。
  4. 输入以下命令以创建一个策略，并将其命名为 `RetrieveRecommendationsPolicy`。要使用其他名称，*RetrieveRecommendationsPolicy* 请更改为所需的名称。

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

#### Note

如果弹出一条消息说，您的账户无权执行 `CreatePolicy` 操作，则您需要将一个允许您为账户创建新的 IAM 策略和角色的策略附加到用户。有关更多信息，请参阅《IAM 用户指南》中的[添加和删除 IAM 身份权限](#)。

5. 复制策略的 Amazon 资源名称 (ARN) (之前示例中的 `arn:aws:iam::123456789012:policy/RetrieveRecommendationsPolicy`)。在下一部分中，您需要使用该 ARN 以创建 IAM 角色。

## 步骤 2：创建 IAM 角色

在创建 IAM 策略后，您可以创建一个 IAM 角色并将策略附加到该角色。

每个 IAM 角色都包含一个信任策略，即，用来指定哪些实体被允许代入角色的一组规则。在本部分中，您将创建一个信任策略，以允许 Amazon Pinpoint 代入角色。接下来，您创建角色本身。然后，您将策略附加到该角色。

### 创建 IAM 角色

1. 在文本编辑器中，创建一个新文件。将以下代码粘贴到该文件中：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```

    "Effect": "Allow",
    "Principal": {
      "Service": "pinpoint.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "AWS:SourceAccount": "accountId"
      },
      "ArnLike": {
        "AWS:SourceArn":
          "arn:aws:mobiletargeting:region:accountId:apps/*"
      }
    }
  }
]
}

```

2. 将该文件保存为 `RecommendationsTrustPolicy.json`。
3. 通过使用命令行界面，导航到保存 `RecommendationsTrustPolicy.json` 文件的目录。
4. 输入以下命令以创建一个新角色，并将其命名为 `PinpointRoleforPersonalize`。要使用其他名称，`PinpointRoleforPersonalize` 请更改为所需的名称。

```
aws iam create-role --role-name PinpointRoleforPersonalize --assume-role-policy-document file://RecommendationsTrustPolicy.json
```

5. 输入以下命令，以将在上一节中创建的策略附加到刚创建的角色：

```
aws iam attach-role-policy --policy-arn arn:aws:iam::123456789012:policy/RetrieveRecommendationsPolicy --role-name PinpointRoleforPersonalize
```

在前面的命令中，将 `arn:aws:iam::123456789012:policy/########RetrieveRecommendationsPolicy###` 的 ARN。此外，如果您 `PinpointRoleforPersonalize` 为角色指定了不同的名称，请替换为在步骤 4 中指定的角色名称。

## 用于将事件流式传输到 Kinesis 的 IAM 角色

Amazon Pinpoint 可以自动将应用程序使用数据或事件数据从您的应用程序发送到您账户中的亚马逊 Kinesis 数据流或亚马逊 Kinesis Data Firehose 传输流。AWS 您必须先向 Amazon Pinpoint 委派所需权限，然后 Amazon Pinpoint 才能开始流式传输事件数据。

如果您使用控制台设置事件流式传输，则 Amazon Pinpoint 会自动创建具备所需权限的 AWS Identity and Access Management (IAM) 角色。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的[将 Amazon Pinpoint 事件流式传输到 Amazon Kinesis](#)。

如果您想要手动创建角色，请将以下策略附加到角色：

- 允许 Amazon Pinpoint 将事件数据发送到流的权限策略。
- 允许 Amazon Pinpoint 代入角色的信任策略。

创建角色之后，您可以将 Amazon Pinpoint 配置为自动将事件发送到流。有关更多信息，请参阅本指南中的[将 Amazon Pinpoint 事件流式传输到 Kinesis](#)。

### 创建 IAM 角色 (AWS CLI)

完成以下步骤，以使用 AWS Command Line Interface (AWS CLI) 手动创建 IAM 角色。要了解如何使用 Amazon Pinpoint 控制台创建角色，请参阅《Amazon Pinpoint 用户指南》中的[将 Amazon Pinpoint 事件流式传输到 Kinesis](#)。

如果您尚未安装 AWS CLI，请参阅AWS Command Line Interface 用户指南 [AWS CLI中的安装](#)。

### 使用创建 IAM 角色 AWS CLI

1. 创建新的文件。将以下策略粘贴到文档中：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pinpoint.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
```

```

        "aws:SourceAccount": "accountId"
    },
    "ArnLike": {
        "aws:SourceArn":
"arn:aws:mobiletargeting:region:accountId:apps/applicationId"
    }
}
]
}

```

完成后，将文件另存为 `PinpointEventStreamTrustPolicy.json`。

2. 使用 `create-role` 命令创建角色并附加信任策略：

```
aws iam create-role --role-name PinpointEventStreamRole --assume-role-policy-document file://PinpointEventStreamTrustPolicy.json
```

3. 创建新的包含角色的权限策略的文件。

如果您要将 Amazon Pinpoint 配置为向 Kinesis 流发送数据，请将以下策略粘贴到文件中：

```

{
  "Version": "2012-10-17",
  "Statement": {
    "Action": [
      "kinesis:PutRecords",
      "kinesis:DescribeStream"
    ],
    "Effect": "Allow",
    "Resource": [
      "arn:aws:kinesis:region:accountId:stream/streamName"
    ]
  }
}

```

或者，如果您要将 Amazon Pinpoint 配置为向 Kinesis Data Firehose 流发送数据，请将以下策略粘贴到文件中：

```

{
  "Version": "2012-10-17",
  "Statement": {

```

```
    "Effect": "Allow",
    "Action": [
      "firehose:PutRecordBatch",
      "firehose:DescribeDeliveryStream"
    ],
    "Resource": [
      "arn:aws:firehose:region:accountId:deliverystream/delivery-stream-name"
    ]
  }
}
```

完成后，将文件另存为 `PinpointEventStreamPermissionsPolicy.json`。

4. 使用 `put-role-policy` 命令将权限策略附加到角色：

```
aws iam put-role-policy --role-name PinpointEventStreamRole --policy-name PinpointEventStreamPermissionsPolicy --policy-document file://PinpointEventStreamPermissionsPolicy.json
```

## Amazon Pinpoint 身份和访问管理问题排查

可以使用以下信息，以帮助您诊断和修复在使用 Amazon Pinpoint 和 IAM 时可能遇到的常见问题。

### 主题

- [我无权在 Amazon Pinpoint 中执行操作](#)
- [我无权执行 iam : PassRole](#)
- [我想允许 AWS 账户之外的用户访问我的 Amazon Pinpoint 资源](#)

### 我无权在 Amazon Pinpoint 中执行操作

如果 AWS Management Console 告诉您您无权执行某项操作，则必须联系管理员寻求帮助。管理员是向您提供登录凭证的人。

当 `mateojackson` 用户尝试使用控制台查看有关项目的详细信息但不具有 `mobiletargeting:GetApp` 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: mobiletargeting:GetApp on resource: my-example-project
```

在这种情况下，Mateo 请求他的管理员更新其策略，以允许他使用 `mobiletargeting:GetApp` 操作访问 `my-example-project` 资源。

## 我无权执行 `iam:PassRole`

如果您收到一个错误，称您无权执行 `iam:PassRole` 操作，则必须更新策略以允许您将角色传递给 Amazon Pinpoint。

有些 AWS 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为 `marymajor` 的 IAM 用户尝试使用控制台在 Amazon Pinpoint 中执行操作时，会发生以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 `iam:PassRole` 操作。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

## 我想允许 AWS 账户之外的用户访问我的 Amazon Pinpoint 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以担任角色。对于支持基于资源的策略或访问控制列表 (ACL) 的服务，您可以使用这些策略向人员授予对您的资源的访问权。

要了解更多信息，请参阅以下内容：

- 要了解 Amazon Pinpoint 是否支持这些功能，请参阅 [Amazon Pinpoint 如何与 IAM 配合使用](#)。
- 要了解如何提供对您拥有的资源的访问权限 AWS 账户，请参阅 [IAM 用户指南中的向您拥有 AWS 账户的另一个 IAM 用户提供访问权限](#)。
- 要了解如何向第三方提供对您的资源的访问权限 AWS 账户，请参阅 [IAM 用户指南中的向第三方提供访问权限](#)。AWS 账户
- 要了解如何通过联合身份验证提供访问权限，请参阅 IAM 用户指南中的 [为经过外部身份验证的用户 \(联合身份验证\) 提供访问权限](#)。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 [IAM 角色与基于资源的策略有何不同](#)。



## Amazon Pinpoint 中的日志记录和监控

日志记录和监控是保持 Amazon Pinpoint 项目和其他类型的 Amazon Pinpoint 资源的可靠性、可用性和性能的重要部分。您应该记录并收集来自 Amazon Pinpoint 项目和资源的各个部分的监控数据，以便在出现多点故障时更轻松地进行调试。AWS 提供了多种工具，可以帮助您记录和收集这些数据，并对潜在事件做出响应：

### AWS CloudTrail

Amazon Pinpoint 与集成，后者是一项服务，可记录用户 AWS CloudTrail、角色或其他服务在 Amazon Pinpoint 中执行的操作。AWS 这包括来自 Amazon Pinpoint 控制台的操作以及对 Amazon Pinpoint API 操作的编程调用。通过使用收集的信息 CloudTrail，您可以确定哪些请求是向 Amazon Pinpoint 提出的。对于每个请求，您可以识别它是何时发出的、来源 IP 地址、谁发出的以及其他详细信息。有关更多信息，请参阅本指南中的[使用 AWS CloudTrail 记录 Amazon Pinpoint API 调用](#)。

### 亚马逊 CloudWatch

您可以使用亚马逊 CloudWatch 收集、查看和分析与您的亚马逊 Pinpoint 账户和项目相关的几个重要指标。您还可以使用 CloudWatch 创建警报，当某个指标的值满足某些条件并且在或超过您定义的阈值之内时会通知您。如果您创建警报，则 CloudWatch 会向您指定的亚马逊简单通知服务 (Amazon SNS) 主题发送通知。有关更多信息，请参阅亚马逊 Pinpoint 用户指南 [CloudWatch 中的通过亚马逊监控亚马逊 Pinpoint](#)。

### AWS Health 仪表盘

通过使用 AWS Health 控制面板，您可以检查和监控 Amazon Pinpoint 环境的状态。要查看亚马逊 Pinpoint 服务的整体状态，请使用服务运行状况控制面板 AWS 板。要检查、监控和查看可能对您的 AWS 环境产生更具体影响的任何事件或问题的历史数据，请使用 Personal Health Dashboard。要了解有关这些控制面板的更多信息，请参阅 [AWS Health 用户指南](#)。

### AWS Trusted Advisor

AWS Trusted Advisor 检查您的 AWS 环境并就解决安全漏洞、提高系统可用性和性能以及节省资金的机会提供建议。所有 AWS 客户都可以访问一组核心支持 Trusted Advisor 票。拥有商业或企业支持计划的客户可以获得其他 Trusted Advisor 支票。

其中许多检查可以帮助您评估作为 AWS 账户一部分的 Amazon Pinpoint 资源的安全状况。例如，核心的 Trusted Advisor 检查包括以下内容：

- 您 AWS 账户的日志配置，适用于每个受支持 AWS 区域。

- Amazon Simple Storage Service (Amazon S3) 存储桶的访问权限，其中可能包含您导入 Amazon Pinpoint 以创建分段的文件。
- 使用 AWS Identity and Access Management 用户、群组 and 角色来控制对 Amazon Pinpoint 资源的访问权限。
- 可能会危及您的 AWS 环境和 Amazon Pinpoint 资源安全的 IAM 配置和策略设置。

有关更多信息，请参阅《AWS Support 用户指南》中的 [AWS Trusted Advisor](#)。

## Amazon Pinpoint 的合规性验证

作为多个 AWS 合规性计划的一部分，第三方审计员将评估 Amazon Pinpoint 的安全性和合规性。其中包括用于安全管理控制的 AWS 系统和组织控制 (SOC)、FedRAMP、HIPAA、用于安全管理控制的 ISO/IEC 27001:2013、用于云特定控制的 ISO/IEC 27017:2015、用于个人数据保护的 ISO/IEC 27018:2014、用于质量管理体系的 ISO/IEC 9001:2015 等。

有关特定合规计划范围内 AWS 服务的列表，请参阅合规计划范围内按合规 [计划提供的范围内的 AWS 服务 \(按分\)](#)。有关一般信息，请参阅 [AWS 合规计划 AWS](#)。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅 [Artifact 中的下载报表在 AWS r](#)。

您在使用 Amazon Pinpoint 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。AWS 提供了以下资源来帮助实现合规性：

- [安全与合规性快速入门指南](#) — 这些部署指南讨论了架构注意事项，并提供了在上面部署以安全性和合规性为重点的基准环境的步骤。AWS
- [HIPAA 安全与合规架构白皮书 — 本白皮书](#) 描述了各公司如何使用它来 AWS 创建符合 HIPAA 标准的应用程序。
- [AWS 合规资源](#) — 此工作簿和指南集合可能适用于您的行业和所在地区。
- [使用 AWS Config 开发人员指南中的规则评估资源](#) — 该 AWS Config 服务评估您的资源配置在多大程度上符合内部实践、行业准则和法规。
- [AWS Security Hub](#) — 此 AWS 服务可全面了解您的安全状态 AWS，帮助您检查是否符合安全行业标准和最佳实践。

当买家使用适当的沟通渠道时，Amazon AWS Pinpoint 是一项符合 HIPAA 资格的服务。如果您想使用 Amazon Pinpoint 运行包含 HIPAA 和相关法律法规所定义的受保护健康信息 (PHI) 的工作负载，则

应使用电子邮件渠道、推送通知渠道或短信渠道发送包含 PHI 的消息。如果您使用 SMS 渠道发送包含 PHI 的消息，则应使用您为 AWS 账户请求的[专用短代码](#)发送这些消息，其明确目的是发送将包含或可能包含 PHI 的消息。语音频道不符合 AWS HIPAA 资格；请勿使用语音频道发送包含 PHI 的消息。

## Amazon Pinpoint 中的故障恢复能力

AWS 全球基础设施是围绕 AWS 区域和可用区构建的。AWS 区域提供多个物理隔离和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络相连。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错性和可扩展性。

有关参考架构的更多信息，请参阅[Amazon Pinpoint 弹性架构指南](#)。

有关 AWS 区域和可用区的更多信息，请参阅[AWS 全球基础设施](#)。

## Amazon Pinpoint 中的基础设施安全性

作为一项托管服务，Amazon Pinpoint 受到 AWS 全球网络安全的保护。有关 AWS 安全服务以及如何 AWS 保护基础设施的信息，请参阅[AWS 云安全](#)。要使用基础设施安全的最佳实践来设计您的 AWS 环境，请参阅 S AWS security Pillar Well-Architected Framework 中的[基础设施保护](#)。

您可以使用 AWS 已发布的 API 调用通过网络访问 Amazon Pinpoint。客户端必须支持以下内容：

- 传输层安全性协议 (TLS) 我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密 (PFS) 的密码套件，例如 DHE (临时 Diffie-Hellman) 或 ECDHE (临时椭圆曲线 Diffie-Hellman)。大多数现代系统 (如 Java 7 及更高版本) 都支持这些模式。

此外，必须使用访问密钥 ID 和与 IAM 委托人关联的秘密访问密钥来对请求进行签名。或者，您可以使用[AWS Security Token Service](#) (AWS STS) 生成临时安全凭证来对请求进行签名。

尽管您可以从任何网络位置进行这些 API 调用，但 Amazon Pinpoint 支持基于资源的访问策略。这些策略可以包括基于源 IP 地址的限制。要了解此类策略的更多信息，请参阅[使用策略管理访问](#)。

此外，您还可以配置和使用各种 AWS 安全功能，控制您通过与 Amazon Pinpoint 集成的任何移动或网络应用程序访问亚马逊 Pinpoint 资源。这包括对添加端点、更新端点数据、提交事件数据和报告使用数据等任务的 API 调用进行限制。

要使用这些功能，我们建议您使用 AWS 移动软件开发工具包或 AWS Amplify JavaScript 库将移动和网络应用程序与 Amazon Pinpoint 集成。对于 Android 或 iOS 应用程序，我们建议您分别使用 AWS

Mobile SDK for Android 或。AWS Mobile SDK for iOS对于 JavaScript基于移动或网络的应用程序，我们建议您使用适用于 Web 的 AWS Amplify JavaScript库或适用于 React Native 的 AWS Amplify JavaScript 库。要了解有关这些资源的更多信息，请参阅[AWS 移动 SDK 入门](#)、[网络版 Amp AWS lify 库入门](#)和[原生反应的 AWS Amplify 库入门](#)。

## Amazon Pinpoint 中的配置和漏洞分析

作为一项托管服务，Amazon Pinpoint 受[亚马逊网络服务：安全流程概述白皮书中描述的 AWS 全球网络安全程序](#)的保护。这意味着它 AWS 管理和执行基本的安全任务和程序，以强化、修补、更新和以其他方式维护您的 Amazon Pinpoint 账户和资源的底层基础设施。这些流程已经过适当的第三方审核和认证。

有关更多信息，请参阅以下资源：

- [Amazon Pinpoint 的合规性验证](#)
- [责任共担模式](#)
- [亚马逊云科技：安全流程概览](#) ( 白皮书 )

## Amazon Pinpoint 的安全最佳实践

使用 AWS 身份和访问管理 (IAM) 账户控制 API 操作的 Amazon Pinpoint 访问权限，尤其是创建、修改或 Amazon Pinpoint 删除资源的操作。对于 Amazon Pinpoint API，此类资源包括项目、活动和旅程。对于 Amazon Pinpoint SMS 和 Voice API，此类资源包括电话号码、资源池和配置集。

- 为每个管理 Amazon Pinpoint 资源的人创建一个单独的用户，包括你自己。请勿使用 AWS 根凭证来管理 Amazon Pinpoint 资源。
- 授予每位用户执行其职责所需的最低权限集。
- 使用 IAM 组有效地管理适用于多个用户的权限。
- 定期轮换您的 IAM 凭证。

有关 Amazon Pinpoint 安全的更多信息，请参阅 [Amazon Pinpoint 中的安全](#)。有关 IAM 的更多信息，请参阅 [AWS Identity and Access Management](#)。有关 IAM 最佳实践的信息，请参阅 [IAM 最佳实践](#)。

# Amazon Pinpoint 限额

以下各节列出并介绍了适用于 Amazon Pinpoint 资源和操作的限额（以前称为限制）。一些限额可以提升，而另一些限额不能提升。要确定您是否可以请求提升限额，请参阅每个部分中的符合提升条件列或声明。

## 主题

- [项目限额](#)
- [API 请求限额](#)
- [SMS 和 Voice API 第 2 版请求限额](#)
- [活动限额](#)
- [电子邮件限额](#)
- [端点限额](#)
- [端点导入限额](#)
- [事件提取限额](#)
- [旅程限额](#)
- [Lambda 限额](#)
- [机器学习限额](#)
- [消息模板限额](#)
- [推送通知限额](#)
- [应用程序内消息限额](#)
- [分段限额](#)
- [短信限额](#)
- [10DLC 限额](#)
- [语音限额](#)
- [请求提高限额](#)

## 项目限额

下表列出了与 Amazon Pinpoint 中项目相关联的限额。

资源	默认限额	是否符合提高配额的条件？
项目	在每个项目中 AWS 区域，您最多可以有 100 个项目。	不支持

## API 请求限额

Amazon Pinpoint 实施配额，限制您可以从账户向亚马逊 Pinpoint API 发出的请求的大小和数量。  
AWS

除非针对特定类型的资源另行指定，否则调用（请求和响应）负载的最大大小为 7 MB。要确定某个资源是否具有不同的限额，请参阅本主题下对应于该类型资源的部分。

最大请求次数因限额类型和 API 操作而异。Amazon Pinpoint 对 API 请求实施了两种类型的限额：

- **速率限额** – 也称为速率限制，这种限额定义了您每秒可以针对特定操作发出的最大请求数。它控制每个账户发送或接收请求的速率。
- **限额暴增** – 也称为限制暴增或容量暴增，这种限额定义了账户同时进行的最大请求数。

下表列出了 Amazon Pinpoint API 的速率限额和限额暴增。

操作	默认限额暴增/速率限额（每秒请求数）
CreateCampaign	25
CreateEmailTemplate	10
CreateInAppTemplate	10
CreateImportJob	300
CreatePushTemplate	10
CreateSegment	25
CreateSmsTemplate	10
CreateVoiceTemplate	10

操作	默认限额暴增/速率限额 (每秒请求数)
DeleteCampaign	25
DeleteEndpoint	5
DeleteSegment	25
GetEndpoint	10
PhoneNumberValidate	20
PutEvents	15
SendMessages	4,000
SendUsersMessages	6000
UpdateCampaign	25
UpdateEmailTemplate	10
UpdateEndpoint	10
UpdateEndpointsBatch	2
UpdateInAppTemplate	10
UpdatePushTemplate	10
UpdateSegment	25
UpdateSmsTemplate	10
UpdateVoiceTemplate	10
所有其他操作	300

下表列出了 CreateImportJob 的文件导入限额。

操作	默认限额	是否符合提高配额的条件？
最大导入文件数	每个导入任务 10,000 个文件	不支持

如果超过其中一个限额，Amazon Pinpoint 会限制请求，即，拒绝本应有效的请求，并返回 `TooManyRequests` 错误。限制将根据您的账户针对特定 AWS 区域中的特定操作发出的请求总数来实施。此外，系统会为每个操作单独计算限制决定。例如，如果 Amazon Pinpoint 限制针对 `SendMessage` 的请求，则针对 `UpdateEndpoint` 操作的并发请求可以成功完成。

## SMS 和 Voice API 第 2 版请求限额

下表列出了 Amazon Pinpoint API 第 2 版中每个资源的每秒请求数 (RPS) 限额。对于所有资源，您只按照 [请求增加限额](#) 中的说明进行操作，就可以为其增加限额。

资源	默认速率限额 (每秒请求数)
<code>DescribeAccountAttributes</code>	1
<code>DescribeAccountLimits</code>	1
<code>DescribeSpendLimits</code>	1
<code>SetTextMessageSpendLimitOverride</code>	1
<code>SetVoiceMessageSpendLimitOverride</code>	1
<code>DeleteTextMessageSpendLimitOverride</code>	1
<code>DeleteVoiceMessageSpendLimitOverride</code>	1
<code>DescribePhoneNumbers</code>	1
<code>DescribeSenderId</code>	1
<code>RequestPhoneNumber</code>	1
<code>UpdatePhoneNumber</code>	1
<code>ReleasePhoneNumber</code>	1



资源	默认速率限额 (每秒请求数)
DescribePools	1
CreatePool	1
UpdatePool	1
DeletePool	1
ListPoolOriginationIdentities	1
AssociateOriginationIdentity	1
DisassociateOriginationIdentity	1
DescribeConfigurationSets	1
CreateConfigurationSet	1
DeleteConfigurationSet	1
CreateEventDestination	1
UpdateEventDestination	1
DeleteEventDestination	1
SetDefaultMessageType	1
SetDefaultSenderId	1
DeleteDefaultMessageType	1
DeleteDefaultSenderId	1
DescribeOptOutLists	1
CreateOptOutList	1
DeleteOptOutList	1

资源	默认速率限额 (每秒请求数)
DescribeOptedOutNumbers	1
PutOptedOutNumber	10
DeleteOptedOutNumber	10
DescribeKeywords	1
PutKeyword	1
DeleteKeyword	1
ListTagsForResource	1
TagResource	1
UntagResource	1

## 活动限额

以下限额适用于 Amazon Pinpoint API 的[活动](#)资源。

以下配额适用 AWS 区域，有些可以增加。有关更多信息，请参阅《服务限额用户指南》中的[请求提高限额](#)。

资源	默认限额	是否符合提高配额的条件？
有效活动数	每个账户 200 个	不支持

**Note**

有效活动 是尚未完成或已失败的活动。有效活动的状态包括 SCHEDULED、EXECUTING

资源	默认限额	是否符合提高配额的条件？
	或 PENDING_N EXT_RUN 。	
最大分段大小	对于导入的分段：每个活动 100,000,000 个  对于动态分段：无限制	不支持
基于事件的活动	<p>每个项目最多可以包含在事件发生时发送的 25 个活动。</p> <p>使用基于事件的触发器的活动必须使用动态分段。不能使用导入的分段。</p> <p>如果您使用 AWS 移动软件开发工具包将您的应用程序与 Amazon Pinpoint 集成，则来自基于事件的广告系列的消息仅发送给其应用程序运行 AWS Mobile SDK for Android 版本 2.7.2 或更高版本、版本 2.6.30 或更高 AWS Mobile SDK for iOS 版本的客户。</p> <p>如果 Amazon Pinpoint 无法在 5 分钟内从基于事件的活动发送消息，它将丢弃该消息而不会尝试重新传送。</p>	不支持

## 电子邮件限额

以下部分中的限额适用于电子邮件渠道。

## 电子邮件限额

资源	默认限额	是否符合提高配额的条件？
最大邮件大小（包括附件）	每封邮件 10 MB	不支持
验证的身份数	10,000 个身份	不支持

**Note**

身份 是指电子邮件地址、域或者二者的任意组合。您使用 Amazon Pinpoint 发送的每封电子邮件必须发送自经过验证的身份。

## 电子邮件发送人和接收人限额


资源	默认限额	是否符合提高配额的条件？
发送人地址	必须验证所有发送地址或域。	不支持
接收人地址	如果您的账户处于沙盒中，则必须验证所有接收人电子邮件地址或域。  如果您的账户在沙盒以外，则可以发送到任何有效的地址。	<u>是</u>
每个消息的接收人数	每个消息 50 个接收人	不支持
可以验证的身份数	每个 AWS 区域 10,000 个身份	不支持

资源	默认限额	是否符合提高配额的条件？
	<p><b>Note</b></p> <p>身份 是指电子邮件地址、域或者二者的任意组合。您使用 Amazon Pinpoint 发送的每封电子邮件必须发送自经过验证的身份。</p>	

## 电子邮件发送限额

发送限额、发送速率和沙盒限制由同一区域内的两个服务共享。如果您在 us-east-1 中使用 Amazon SES，同时已从沙盒中移除，且您的发送限额/速率已提高，那么这些更改均适用于您在 us-east-1 中的 Pinpoint 账户。

资源	默认限额	是否符合提高配额的条件？
每 24 小时周期可发送的电子邮件数（发送限额）	<p>如果您的账户在沙盒中，则每 24 小时周期 200 封电子邮件。</p> <p>如果您的账户在沙盒之外，限额将根据您的具体使用情形而异。</p> <p><b>Note</b></p> <p>此限额基于接收人的数量，而不是发送的唯一消息数量。接收人是“To:”行上的任何电子邮件地址。</p>	<u>是</u>
每秒可发送的电子邮件数（发送速率）	如果您的账户在沙盒中，则每秒 1 封电子邮件。	<u>是</u>

资源	默认限额	是否符合提高配额的条件？
	<p>如果您的账户在沙盒之外，速率将根据您的具体使用情形而异。</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> <b>Note</b></p> <p>此速率基于接收人的数量，而不是发送的唯一消息数量。接收人是“To:”行上的任何电子邮件地址。</p> </div>	

## 端点限额

以下限额适用于 Amazon Pinpoint API 的[端点](#)资源。

每个端点支持的最大属性数量为 250，最大端点大小为 15 KB。但是，此属性数量（包括所有属性）可能会受到端点总大小的限制。如果您在向模板添加属性时遇到错误，请考虑减少每个属性中的数据量或减少属性的数量。

资源	默认限额	是否符合提高配额的条件？
端点大小	最大大小 15 KB	不支持
分配给 <code>Attributes</code> 、 <code>Metrics</code> 和 <code>UserAttributes</code> 参数的属性总计	每个应用程序的所有属性参数数量最多为 250 个	不支持
分配给 <code>Attributes</code> 参数的属性	每个应用程序的所有属性参数数量最多为 250 个	不支持
分配给 <code>Metrics</code> 参数的属性	每个应用程序的所有属性参数数量最多为 250 个	不支持

资源	默认限额	是否符合提高配额的条件？
分配给 UserAttributes 参数的属性	每个应用程序的所有属性参数数量最多为 250 个	不支持
属性名称长度	50 个字符	不支持
属性值长度	100 个字符	不支持
EndpointBatchRequest 负载中的 EndpointBatchItem 对象	每个负载 100 个。负载大小不能超过 7 MB。	不支持
具有相同用户 ID 的端点	每个用户 ID 最多 15 个唯一点	不支持
分配给 Attributes 参数属性的值	每个属性 50 个	不支持
分配给 UserAttributes 参数属性的值	每个属性 50 个	不支持

## 端点导入限额

以下限额适用于将端点导入 Amazon Pinpoint。

资源	默认限额	是否符合提高配额的条件？
活动导入任务	每个账户 10 个  导入任务只有在运行时才计入此限额。导入任务完成后，将不再计入此限额。	不支持
导入大小	每个导入任务 1 GB	不支持

资源	默认限额	是否符合提高配额的条件？
	例如，如果每个端点不超过 4 KB，您可以导入 250,000 个端点。	

## 事件提取限额

以下配额适用于使用 AWS 移动软件开发工具包和 Amazon Pinpoint API [的事件资源提取事件](#)。

资源	默认限额	是否符合提高配额的条件？
自定义事件类型的最大数量	每个应用程序 1,500 个	不支持
自定义属性键的最大数量	每个应用程序 500 个	不支持
每个属性键的最大自定义属性值数量	100,000。任何超过 100,000 的数字仍会被登记，但不会在 Amazon Pinpoint 分析控制台中显示。	不支持
每个属性键的最大字符数	50	不可以
每个属性值的最大字符数	200。如果字符数超过 200，则删除事件。	不支持
自定义指标键的最大数量	每个应用程序 500 个	不支持
请求中的最大事件数	每请求 100 个事件	不支持
请求的最大大小	4 MB	不支持
单个事件的最大大小	1,000 KB	不支持
每个事件的最大属性键和指标键数量	每请求 40 个事件	不支持



## 旅程限额

以下限额适用于旅程。

以下配额适用 AWS 区域，有些可以增加。有关更多信息，请参阅《服务限额用户指南》中的[请求提高限额](#)。

资源	默认限额	是否符合提高配额的条件？
最大活动旅程数	每个账户 50 个	不支持
最大活跃人数 EventTriggeredJourneys	每个账户 40 个	不支持
最大旅程活动数	每个旅程 40 个	不支持
最大分段大小	对于导入的分段：每个旅程 100,000,000 个。 对于动态分段：无限制	不支持
联系中心活动次数上限	每个旅程 3 个	不支持
最大关闭日期规则数	每个渠道 20 个	不支持
关闭日期规则名称的最大长度	150 个字符	不支持
关闭日期规则的开始时间和结束时间之间的最大天数	7 days	不支持
最大开放小时规则数	每天 4 个	不支持

## Lambda 限额

以下限额适用于从机器学习 (ML) 模型中检索和处理数据的 Amazon Pinpoint 配置。

资源	默认限额	是否符合提高配额的条件？
Lambda 函数调用有效负载 ( 请求和响应 ) 的最大大小	6 MB	不支持
等待 Lambda 函数处理数据的最长时间	15 秒	不支持
每个端点的最大事件属性数	5	不支持
每个事件属性名称的最大字符数	128 个字符	不支持
每个事件属性名称的最大字符数	128 个字符	不支持
旅程可以运行的最大天数	540 天	不支持

## 机器学习限额

以下限额适用于从机器学习 (ML) 模型中检索和处理数据的 Amazon Pinpoint 配置。

资源	默认限额	是否符合提高配额的条件？
最大模型配置数	每个消息模板 1 个 每个账户 100 个	不支持
最大建议数	每个端点或用户 5 个	不支持
每个端点或用户的最大建议属性数	1 个 - 如果 AWS Lambda 函数不处理属性值 10 个 - 如果 AWS Lambda 函数处理属性值	不支持
建议的属性名称的最大长度	属性名称为 50 个字符	不支持

资源	默认限额	是否符合提高配额的条件？
	属性显示名称（在控制台中的属性查找器中显示的名称）为 25 个字符	
从 Amazon Personalize 中检索的建议属性值的最大长度	100 个字符	不支持
Lambda 函数调用有效负载（请求和响应）的最大大小	6 MB	不支持
等待 Lambda 函数处理数据的最长时间	15 秒	不支持
调用 Lambda 函数的最大尝试次数	3 次尝试	不支持

根据您配置 Amazon Pinpoint 以使用 ML 模型的方式，可能需要应用额外的限额。要了解有关 Amazon Personalize 限额的信息，请参阅《Amazon Personalize 开发人员指南》中的[限额](#)。要了解 AWS Lambda 限额，请参阅《AWS Lambda 开发人员指南》中的[限额](#)。

## 消息模板限额

以下限额适用于您的 Amazon Pinpoint 账户的消息模板。

资源	默认限额	是否符合提高配额的条件？
最大消息模板数	每个账户 20,000 个	不支持
最大版本数量	每个模板 5,000 个	不支持
电子邮件模板中的最大字符数	600,000 个字符	不支持
应用程序内模板中的最大字符数	200,000 个字符	不支持
推送通知模板的默认模板部分中的最大字符数	4000 个字符	不支持

资源	默认限额	是否符合提高配额的条件？
推送通知模板的 ADM 特定的模板部分中的最大字符数	6,000 个字符	不支持
推送通知模板的 APNs 特定模板部分中的最大字符数	4000 个字符	不支持
推送通知模板的百度特定的模板部分中的最大字符数	4000 个字符	不支持
推送通知模板的 FCM 特定模板部分中的最大字符数	4000 个字符	不支持
SMS 模板中的最大字符数	1,600 个字符	不支持
语音模板中的最大字符数	10,000 个字符	不支持

## 推送通知限额

以下限额适用于 Amazon Pinpoint 通过推送通知渠道发送的消息。

资源	默认限额	是否符合提高配额的条件？
一个活动中每秒可发送的推送通知的最大数量	每秒 25,000 个通知	<a href="#">是</a>
Amazon Device Messaging (ADM) 消息有效负载大小	每个消息 6 KB	不支持
Apple Push Notification service (APNs) 消息有效负载大小	每封邮件 4 KB	不支持
APNs 沙盒消息有效负载大小	每封邮件 4 KB	不支持
百度云推送消息有效负载大小	每封邮件 4 KB	不支持

资源	默认限额	是否符合提高配额的条件？
Firebase Cloud Message (FCM) 消息有效负载大小	每封邮件 4 KB	不支持

## 应用程序内消息限额

以下限额适用于您使用 Amazon Pinpoint 管理的应用程序内消息。

以下配额适用 AWS 区域，有些可以增加。有关更多信息，请参阅《服务限额用户指南》中的[请求提高限额](#)。

资源	默认限额	是否符合提高配额的条件？
每秒可以调用 GetInAppMessages API 的最大次数。	每秒 5,000 个请求	<a href="#">是</a>
应用程序内消息活动	每个项目最多可以包含 25 个使用应用程序内消息收发渠道的活动。	是，请参阅《服务限额用户指南》中的 <a href="#">请求提高限额</a>

## 分段限额

以下限额适用于 Amazon Pinpoint API 的[分段](#)资源。

资源	默认限额	是否符合提高配额的条件？
可用于创建分段的维度的最大数量	每个分段 100 个	不支持
每个分段的最大分段组数	5	不支持
每个分段的最大源分段数	5	不支持
源分段的最大深度。	5	不支持

资源	默认限额	是否符合提高配额的条件？
例如，如果一个分段的源分段中还有一个源分段，则深度链不再超过这个限制。		

## 短信限额

以下限额适用于短信渠道。

有关 SMS 费用的更多信息，请参阅《Amazon Pinpoint 用户指南》中的[了解 Amazon Pinpoint 的短信计费和使用报告](#)。

资源	默认限额	是否符合提高配额的条件？
支出阈值	每个账户 1.00 美元	<a href="#">是</a> ，但支出限制因区域而异。您必须指定需要增加限额的区域。
每秒可发送的短信数（发送速率）	因目的地国家/地区和源电话号码而异。有关更多信息，请参阅《Amazon Pinpoint 用户指南》中的 <a href="#">每秒消息部分限制</a> 。	<a href="#">是</a> ，但您可能需要获取支持更高吞吐量的电话号码。如果您不确定要使用哪种号码类型，请联系 AWS Support 或您的 AWS 客户经理了解更多信息  如果您使用字母数字的发件人 ID 发送消息，则可以提高吞吐率。要了解您的发件人 ID 是否可以提高吞吐量，请在支持中心控制台中 <a href="#">创建发件人 ID 请求</a> 。在您的请求中，请包括您现有的发件人 ID、您在哪个国家/地区使用该发件人 ID 以及您要请求的吞吐率。
每秒可发送给单个接收人的短信数	每秒 1 条消息	不支持

资源	默认限额	是否符合提高配额的条件？
双向短信的 Amazon SNS 主题数	每个账户 100,000 个	<a href="#">是</a>
双向短信的关键词数	每个号码 30 个关键词	<a href="#">是</a>
短信和语音号码的数量	每个账户和区域 25 个	<a href="#">是</a>
专用电话号码的数量	每个账户 25 个	<a href="#">是</a>
选择退出列表的数量  注意：必需的默认选择退出列表计入此限额。	每个账户 25 个	<a href="#">是</a>
配置集数	每个账户 25 个	<a href="#">是</a>
事件目标的数量	每个配置集 5 个	不支持
短信沙盒中经过验证的目标电话号码数量	每个账户 10 个	<a href="#">是</a>
电话号码池的数量	每个账户 25 个	<a href="#">是</a>
可以与电话号码池关联的源身份数量	每个电话号码池 100	<a href="#">是</a>

## 10DLC 限额

以下限额适用于使用 10DLC 电话号码发送的短信。10DLC 号码只能用于向美国境内的收件人发送消息。

资源	默认限额	是否符合提高配额的条件？
每人最多 10 个 DLC 公司 AWS 账户	25	<a href="#">是</a>

资源	默认限额	是否符合提高配额的条件？
每个 10DLC 公司最多 10DLC 活动数	10	<a href="#">是</a>
每个 10DLC 公司最多 10DLC 号码数	49	不支持

## 语音限额

以下限额适用于语音渠道。

### Note

从沙盒中删除您的账户后，您将自动有资格获得下表所示的最大限额。

资源	默认限额	是否符合提高配额的条件？
您在 24 小时内可以发送的语音消息数	如果您的账户位于沙盒环境中：20 条消息	不支持
您在 24 小时内可以发送给单个接收人的语音消息数	5 条消息	不支持
您每分钟可以发送的语音消息数	如果您的账户位于沙盒环境中：每分钟 5 次呼叫 如果您的账户没有位于沙盒环境中：每分钟 20 次呼叫	不支持
每秒从单个原始电话号码可以发送的语音消息数	每秒 1 条消息	不支持
语音消息长度	如果您的账户位于沙盒环境中：30 秒	不支持



资源	默认限额	是否符合提高配额的条件？
	如果您的账户没有位于沙盒环境中：5 分钟	
能够将语音消息发送到国际电话号码	<p>如果您的账户在沙盒中，您只能将消息发送给以下国家/地区的接收人：</p> <ul style="list-style-type: none"><li>• 澳大利亚</li><li>• 加拿大</li><li>• 德国</li><li>• 中国香港</li><li>• 以色列</li><li>• 日本</li><li>• 墨西哥</li><li>• 新加坡</li><li>• 瑞典</li><li>• 美国</li><li>• 英国</li></ul> <p>如果您的账户不在沙盒环境中，则可以将消息发送到任何国家/地区的接收人。</p> <div data-bbox="594 1371 1029 1640" style="border: 1px solid #00a0e3; border-radius: 10px; padding: 10px;"><p> <b>Note</b></p><p>国际通话收取额外费用，具体因目标国家/地区而异。</p></div>	不支持

资源	默认限额	是否符合提高配额的条件？
语音消息中的字符数	3,000 个应计费字符（说出的单词中的字符）  总共 6,000 个字符（包括应计费字符和 SSML 标签）	不支持
配置集数	每个 AWS 区域 10,000 个语音配置集	不支持

## 请求提高限额

如果上述任何表中的符合提升限额的条件列中的值为是，则可以请求提升该限额。

### 要请求提高限额

1. 登录[网址为 AWS Management Console https://console.aws.amazon.com/](https://console.aws.amazon.com/)。
2. [通过 https://console.aws.amazon.com/support/home#/case/create](https://console.aws.amazon.com/support/home#/case/create) 创建新的 S AWS support 案例。
3. 在您的支持案例窗格上，选择创建案例。
4. 选择想要提高服务限制？链接。
5. 在案例分类下，对于限额类型，选择以下选项之一：
  - 要请求提高与电子邮件渠道相关的限额，请选择 Pinpoint 电子邮件。
  - 要请求提高短信支出限额或短信发送率限额，请选择 Pinpoint 短信。要提高所有其他短信限额，请选择 Pinpoint。
  - 要请求提高与语音渠道相关的限额，请选择 Pinpoint 语音。
  - 要请求提高与任何其他 Amazon Pinpoint 功能相关的限额，请选择 Pinpoint。
6. 根据您在上一步中选择的选项，在可选字段和必填字段中输入请求的信息来说明您的使用情形。
7. 在请求下，执行以下操作之一：
  - 如果您的请求与短信渠道相关，对于资源类型，选择常规限额。
  - 如果您的请求与其他渠道或任何其他 Amazon Pinpoint 功能有关，请在“AWS 区域”中选择您要申请增加配额的区域。要在其他区域中请求提高到相同限额，请选择添加其他请求，然后选择其他区域。

8. 选择要提升的限额，然后为该限额输入所需的新值。
9. 在案例描述下，对于使用案例描述，说明为什么请求提高限额。
10. 在“联系人选项”下，在“首选联系语言”中，选择您与 Su AWS pport 团队沟通时首选使用的语言。
11. 对于联系方式，请选择您首选的与 Su AWS pport 团队沟通的方法。
12. 选择提交。

Su AWS pport 团队会在 24 小时内对您的请求做出初步回应。

为了防止我们的系统被用于发送未经请求或恶意的内容，我们必须仔细审查每个请求。如果我们能做到这一点，我们将在 24 小时内准予您的请求。但是，如果我们需要从您那里获得其他信息，则可能需要更长的时间来解决您的请求。

如果您的使用情形与我们的策略不符，我们可能无法准予您的请求。

# Amazon Pinpoint 文档历史记录

下表介绍了 2018 年 12 月之后，《Amazon Pinpoint 开发人员指南》的每个版本中的重要更改。要获得本文档的更新通知，您可以订阅 RSS 源。

- 文档最新更新时间：2023 年 11 月 16 日

变更	说明	日期
<a href="#">Amazon Pinpoint 限额</a>	添加了“最大关闭日期规则数”、“关闭日期规则名称的最大长度”、“关闭日期规则的开始时间和结束时间之间的最大天数”和“最大开放小时规则数”的限额。有关更多信息，请参阅 <a href="#">Amazon Pinpoint 限额</a> 。	2023 年 12 月 19 日
<a href="#">Amazon Pinpoint 已更新其用户指南文档</a>	要获取有关如何创建、配置和管理您的 Amazon Pinpoint SMS 和语音资源的最新信息，请参阅新的 <a href="#">Amazon Pinpoint SMS 用户指南</a> 。	2023 年 11 月 16 日
<a href="#">Amazon Pinpoint 限额</a>	更新了 UpdateEndpointsBatch、UpdateEndpointPutEvents DeleteEndpoint、和的配额 GetEndpoint。有关更多信息，请参阅 <a href="#">Amazon Pinpoint 限额</a> 。	2023 年 9 月 22 日
<a href="#">Amazon Pinpoint 限额</a>	更新了 CreateEmailTemplate、CreateSmsTemplate、CreatePushTemplate、CreateInAppTemplate、CreateVoiceTemplate、UpdateEmailTemplate UpdateSms	2023 年 9 月 12 日

	<p>Template UpdatePushTemplate、UpdateVoiceTemplate 和的配额 CreateImportJob。有关更多信息，请参阅 <a href="#">Amazon Pinpoint 限额</a>。</p>	
<a href="#">旅程和活动执行指标</a>	<p>为旅程和活动添加了新的分析指标。有关更多信息，请参阅 <a href="#">旅程和活动执行指标</a>。</p>	2023 年 4 月 25 日
<a href="#">为 Amazon Pinpoint 创建接口 VPC 端点</a>	<p>Amazon Pinpoint 现在支持接口 VPC 端点。有关更多信息，请参阅 <a href="#">为 Amazon Pinpoint 创建接口 VPC 端点</a>。</p>	2023 年 4 月 11 日
<a href="#">传输中加密</a>	<p>从 2023 年 3 月 22 日开始，Amazon Pinpoint 将不再支持 TLS 1.0，但您仍然可以使用 TLS 1.2 或更高版本。有关更多信息，请参阅 <a href="#">传输中加密</a>。</p>	2023 年 3 月 20 日
<a href="#">Amazon Pinpoint 限额</a>	<p>更新了申请增加活动、旅程和应用程序内消息限额的流程。有关更多信息，请参阅 <a href="#">Amazon Pinpoint 限额</a>。</p>	2022 年 12 月 16 日
<a href="#">区域可用性</a>	<p>Amazon Pinpoint 现已在以下区域推出：美国东部（俄亥俄州）区域。</p>	2022 年 10 月 5 日
<a href="#">IAM 角色示例更新</a>	<p>更新了整个文档中的几个 IAM 角色示例，以更好地与安全最佳实践保持一致。</p>	2022 年 5 月 27 日

## [SMS 和 Voice API，第 2 版](#)

Amazon Pinpoint 现在包含用于发送短信和语音消息的专用 API。此 API 包括配置集、资源池和退出列表等新功能，这些功能对以事务性方式发送短信和语音消息的客户很有用。有关更多信息，请参阅[使用 Amazon Pinpoint SMS 和 Voice API](#)。

2022 年 4 月 1 日

## [一次性密码的创建和验证](#)

Amazon Pinpoint 现在包含一项功能，可生成一次性密码 (OTP) 并将其作为短信发送给您的用户。它还包括一个 API，用于在用户将 OTP 代码输入到您的应用程序或网站时对其进行验证。有关更多信息，请参阅[发送和验证一次性密码 \(OTP\)](#)。

2021 年 11 月 26 日

## [应用程序内消息](#)

添加了有关将 Amazon Pinpoint 的[应用程序内消息](#)功能与您的应用程序集成的信息。

2021 年 11 月 10 日

## [代码示例](#)

为常见 Amazon Pinpoint 操作添加了[代码示例库](#)。

2021 年 11 月 3 日

## [项目限额](#)

[Amazon Pinpoint 项目的最大数量](#)仍为 100 个，但现在可以通过向打开服务限制提高请求来增加配额。AWS Support

2021 年 10 月 11 日

<a href="#">Lambda 策略更新。</a>	某些 Lambda 权限策略现在必须包含一个 <code>AWS:SourceAccount</code> 条件中。更新了在 <a href="#">在 Amazon Pinpoint 中创建自定义渠道</a> 和 <a href="#">使用 AWS Lambda 自定义分段</a> 主题中的示例策略。	2021 年 10 月 7 日
<a href="#">UpdateEndpoint</a>	亚马逊 Pinpoint <a href="#">UpdateEndpoint API</a> 现已被登录。 CloudTrail	2020 年 11 月 16 日
<a href="#">自定义属性</a>	Amazon Pinpoint 现在在电子邮件消息模板中支持 250 个属性。请参阅 <a href="#">限额</a> 。	2020 年 9 月 18 日
<a href="#">区域可用性</a>	Amazon Pinpoint 现已在以下区域推出：亚太地区（东京）区域、欧洲地区（伦敦）区域以及加拿大（中部）区域。请注意，Amazon Pinpoint SMS 和 Voice API 未在这些区域推出。	2020 年 9 月 10 日
<a href="#">区域可用性</a>	Amazon Pinpoint 现已在亚太地区（东京）区域推出。请注意，Amazon Pinpoint SMS 和 Voice API 不支持在该区域中使用语音。	2020 年 9 月 2 日
<a href="#">活动事件</a>	在 <a href="#">活动事件</a> 中添加了有关新的活动事件 <code>delivery_type</code> 参数的信息。	2020 年 8 月 2 日
<a href="#">区域可用性</a>	Amazon Pinpoint 现已在亚太地区（首尔）区域推出。请注意，Amazon Pinpoint API 不支持在该区域中使用语音或短信。	2020 年 7 月 31 日

<a href="#">区域可用性</a>	亚马逊 Pinpoint 现已在该地区上 AWS GovCloud (US) 市。	2020 年 4 月 30 日
<a href="#">自定义渠道</a>	更新了有关 <a href="#">使用 Lambda 函数或 Webhook 创建自定义渠道</a> 的信息。	2020 年 4 月 23 日
<a href="#">机器学习</a>	添加了有关从推荐模型中检索个性化推荐以及可选地使用 <a href="#">AWS Lambda 函数增强这些推荐</a> 的信息。	2020 年 3 月 4 日
<a href="#">安全性</a>	添加了 <a href="#">安全章节</a> ，其中提供了有关 Amazon Pinpoint 的各种安全控制和功能的信息。	2020 年 2 月 4 日
<a href="#">旅程</a>	添加了有关使用 Amazon Pinpoint 旅程开发执行项目的消息传送活动的自动化工作流程的信息。还添加了有关 <a href="#">查询分析数据</a> 以获取一部分适用于旅程的指标的信息。	2019 年 10 月 31 日
<a href="#">分析</a>	添加了说明如何 <a href="#">查询活动和事务性邮件的分析数据</a> 的过程，并添加了有关 <a href="#">使用查询结果</a> 的信息。	2019 年 10 月 17 日
<a href="#">分析</a>	添加了有关针对一部分适用于事务性电子邮件和短信的指标 <a href="#">查询分析数据</a> 的信息。	2019 年 9 月 6 日
<a href="#">代码示例</a>	增加了 <a href="#">代码示例</a> ，您可以使用这些示例，通过 Amazon Pinpoint 支持的所有服务来发送事务性推送通知。	2019 年 7 月 30 日



<a href="#">分析</a>	添加了有关 <a href="#">查询分析数据</a> 以获取适用于项目（应用程序）和活动的指标子集数据的信息。	2019 年 7 月 24 日
<a href="#">分段</a>	添加了一个 <a href="#">教程</a> ，以介绍将客户数据从外部系统（如 Salesforce 或 Marketo）导入到 Amazon Pinpoint 的解决方案。	2019 年 5 月 14 日
<a href="#">区域可用性</a>	Amazon Pinpoint 现已在 AWS 亚太地区（孟买）和亚太地区（悉尼）地区上市。	2019 年 4 月 25 日
<a href="#">将 Postman 用于 Amazon Pinpoint</a>	添加了一个 <a href="#">教程</a> ，以介绍如何使用 Postman 与 Amazon Pinpoint API 进行交互。	2019 年 4 月 8 日
<a href="#">标记</a>	添加了有关 <a href="#">标记 Amazon Pinpoint 资源</a> 的信息。	2019 年 2 月 27 日
<a href="#">短信注册</a>	添加了一个 <a href="#">教程章节</a> 和一个教程，以介绍如何创建 <a href="#">处理短信用户注册的解决方案</a> 。	2019 年 2 月 27 日
<a href="#">代码示例</a>	添加了多种编程语言的 <a href="#">代码示例</a> ，演示如何以编程方式发送 <a href="#">电子邮件</a> 、 <a href="#">SMS</a> 和 <a href="#">语音消息</a> 。	2019 年 2 月 6 日

## 早期更新

下表介绍了 2018 年 12 月之后，《Amazon Pinpoint 开发人员指南》的每个版本中的重要更改。

更改	描述	日期
区域可用性	Amazon Pinpoint 现已在 AWS 在美国西部（俄勒冈）和欧洲（法兰克福）地区上市。	2018 年 12 月 21 日
语音渠道	您可以使用新的 Amazon Pinpoint 语音渠道创建语音消息，然后通过电话向客户发送这些消息。目前，您只能使用 Amazon Pinpoint SMS 和 Voice API 发送语音消息。	2018 年 11 月 15 日
欧洲地区（爱尔兰）可用性	Amazon Pinpoint 现已在 AWS 欧洲（爱尔兰）地区上市。	2018 年 10 月 25 日
事件 API	使用 Amazon Pinpoint API 可 <a href="#">记录事件</a> 并将其与端点关联。	2018 年 8 月 7 日
定义和查找端点的代码示例	添加了代码示例以展示如何定义、更新、删除和查找端点。提供了 AWS CLI AWS SDK for Java、和 Amazon Pinpoint API 的示例。有关更多信息，请参阅 <a href="#">向 Amazon Pinpoint 定义您的受众</a> 和 <a href="#">在 Amazon Pinpoint 中访问受众数据</a> 。	2018 年 8 月 7 日
端点导出权限	<a href="#">配置 IAM 策略</a> ，以便您将 Amazon Pinpoint 端点导出到 Amazon S3 存储桶。	2018 年 5 月 1 日
用于短信的电话号码验证	使用 Amazon Pinpoint API 来 <a href="#">验证电话号码</a> ，以确定其是否为短信的有效目标号码。	2018 年 4 月 23 日

更改	描述	日期
Amazon Pinpoint 集成主题更新	使用 AWS 软件开发工具包或库@@@ <a href="#">将 Amazon Pinpoint</a> 与您的安卓、iOS 或 JavaScript 应用程序集成。	2018 年 3 月 23 日
AWS CloudTrail 日志记录	添加了有关 <a href="#">使用记录 Amazon Pinpoint API 调用的</a> 信息。 CloudTrail	2018 年 2 月 6 日
更新了服务限额	使用有关电子邮件限额的其他信息更新了 <a href="#">限额</a> 。	2018 年 1 月 19 日
Amazon Pinpoint 扩展程序的公开测试版	使用 AWS Lambda 函数 <a href="#">自定义细分</a> 或 <a href="#">创建自定义消息渠道</a> 。	2017 年 11 月 28 日
推送通知负载限额	限额包括 <a href="#">移动推送消息的负载大小</a> 。	2017 年 10 月 25 日
更新了服务限额	向 <a href="#">限额</a> 添加了短信和电子邮件渠道信息。	2017 年 10 月 9 日
ADM 和百度移动推送	更新您的应用程序代码，以处理来自百度和 ADM 移动推送渠道的推送通知。	2017 年 9 月 27 日
使用 Amazon Cognito 用户池的用户 ID 和身份验证事件。	如果您使用 Amazon Cognito 用户池来管理您的移动应用程序的用户登录，Amazon Cognito 会将用户 ID 分配给端点，并将身份验证事件报告给 Amazon Pinpoint。	2017 年 9 月 26 日
用户 ID	将用户 ID 分配给端点来监控单独用户的应用程序使用情况。针对 <a href="#">AWS Mobile SDK</a> 和 <a href="#">SDK for Java</a> 提供了示例。	2017 年 8 月 31 日

更改	描述	日期
身份验证事件	报告身份验证事件，以了解用户在您的应用程序上进行身份验证的频率。 <a href="#">在应用程序中报告事件</a> 中提供了示例。	2017 年 8 月 31 日
更新了示例事件	<a href="#">示例事件</a> 包括 Amazon Pinpoint 为电子邮件和短信活动流式传输的事件。	2017 年 6 月 08 日
Android 会话管理	可使用 AWS Mobile Hub 示例应用程序提供的类在 Android 应用程序中管理会话。	2017 年 4 月 20 日
更新了货币化事件示例	针对报告货币化事件更新了示例代码。	2017 年 3 月 31 日
事件流	您可以将 Amazon Pinpoint 配置为 <a href="#">将应用程序和活动事件发送给 Kinesis 流</a> 。	2017 年 3 月 24 日
权限	<a href="#">Amazon Pinpoint 如何与 IAM 配合使用</a> 有关向账户中的用户和移动应用程序的 AWS 用户授予访问 Amazon Pinpoint 权限的信息，请参阅。	2017 年 1 月 12 日
Amazon Pinpoint 正式版	此版本引入了 Amazon Pinpoint。	2016 年 12 月 1 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。