aws

开发人员指南

Amazon WorkDocs

Amazon WorkDocs: 开发人员指南

Table of Contents

注意：亚马逊 WorkDocs不再提供新买家注册和账户升级服务。在此处了解迁移步骤：如何从 Amazon 迁移数据 WorkDocs。

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。

什么是 Amazon WorkDocs？

Amazon WorkDocs 是一个文档存储、协作和共享系统。Amazon WorkDocs 是一项适用于企业环境的完全托管式安全服务。它提供了强大的管理控制，以及有助于提高用户工作效率的反馈功能。文件安全地存储在云中。您的用户的文件仅对用户和用户指定的参与者和查看者可见。贵公司的其他成员无法访问其他用户的文件，除非他们被专门授予访问权限。

用户可以与其公司的其他成员共享您的文件用来协作或审查。Amazon WorkDocs 客户端应用程序可用于查看多种不同类型的文件，具体取决于文件的 Internet 媒体类型。Amazon WorkDocs 支持所有常用文档和图像格式，并支持不断添加的其他媒体类型。

有关更多信息，请参阅 Amazon WorkDocs。

访问 Amazon WorkDocs

最终用户使用客户端应用程序访问其文件。非管理员用户从不需要使用 Amazon WorkDocs 控制台或管理控制面板。Amazon WorkDocs 提供了多个不同的客户端应用程序和实用工具：

- 一个用于文档管理和审核的 Web 应用程序。
- 用于查看文档的移动设备本机应用程序。
- Amazon WorkDocs Drive 可用于将您的 Mac 或 Windows 桌面上的文件夹与 Amazon WorkDocs 文件同步。

定价

Amazon WorkDocs 没有预付费用或预先承诺。您只需为活动用户账户以及您使用的存储量付费。有关更多信息，请转到定价。

资源

下列相关资源在您使用此服务的过程中会有所帮助。

- 课程和研讨会 – 指向基于角色的专业课程和自主进度动手实验室的链接，这些课程和实验室旨在帮助您增强 AWS 技能并获得实践经验。
- AWS 开发人员中心 – 浏览教程、下载工具并了解 AWS 开发人员活动。

- AWS 开发人员工具 – 指向开发人员工具、开发工具包、IDE 工具包和命令行工具的链接，这些资源用于开发和管理 AWS 应用程序。
- 入门资源中心 – 了解如何设置 AWS 账户、加入 AWS 社区和启动您的第一个应用程序。
- 动手实践教程 – 按照分步教程在 AWS 上启动您的第一个应用程序。
- AWS 白皮书 – 指向 AWS 技术白皮书的完整列表的链接，这些资料涵盖了架构、安全性、经济性等主题，由 AWS 解决方案架构师或其他技术专家编写。
- AWS Support 中心 – 用于创建和管理 AWS Support 案例的中心。还提供指向其他有用资源的链接，如论坛、技术常见问题、服务运行状况以及AWS Trusted Advisor。
- AWS Support – 提供有关 AWS Support 的信息的主要网页，这是一个一对一的快速响应支持渠道，可以帮助您在云中构建和运行应用程序。
- 联系我们 – 用于查询有关AWS账单、账户、事件、滥用和其他问题的中央联系点。
- AWS 网站条款 – 有关我们的版权和商标、您的账户、许可、网站访问和其他主题的详细信息。

开始使用

以下代码段可帮助您开始使用 Amazon WorkDocs 开发工具包。

> **Note**
>
> 为了提高安全性，请尽可能创建联合用户而不是 IAM 用户。

示例

- 使用 IAM 用户凭证连接到 Amazon WorkDocs 并查询用户
- 通过代入角色连接到 Amazon WorkDocs
- 上传文档
- 下载文档
- 为 IAM 用户或角色设置通知
- 创建用户
- 向用户授予对资源的权限

使用 IAM 用户凭证连接到 Amazon WorkDocs 并查询用户

以下代码显示了如何使用 IAM 用户的 API 凭证进行 API 调用。在这种情况下，API 用户和 Amazon WorkDocs 站点属于同一 AWS 账户。

> **Note**
>
> 为了提高安全性，请尽可能创建联合用户而不是 IAM 用户。

确保已通过适当的 IAM 策略向 IAM 用户授予了 Amazon WorkDocs API 访问权限。

代码示例使用 DescribeUsers API 搜索用户并为用户获取元数据。用户元数据提供了诸如名字、姓氏、用户 ID 和根文件夹 ID 等详细信息。如果您要代表用户执行任何内容上传或下载操作，根文件夹 ID 会特别有用。

代码要求您获取 Amazon WorkDocs 组织 ID。

按照以下步骤从 AWS 控制台获取 Amazon WorkDocs 组织 ID：

获取组织 ID

1. 在 AWS Directory Service 控制台导航窗格中，选择目录。
2. 记下与您的 Amazon WorkDocs 站点对应的目录 ID 值。这是该站点的组织 ID。

以下示例显示了如何使用 IAM 凭证进行 API 调用。

```
import java.util.ArrayList;
import java.util.List;

import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.workdocs.AmazonWorkDocs;
import com.amazonaws.services.workdocs.AmazonWorkDocsClient;
import com.amazonaws.services.workdocs.model.DescribeUsersRequest;
import com.amazonaws.services.workdocs.model.DescribeUsersResult;
import com.amazonaws.services.workdocs.model.User;

public class GetUserDemo {

  public static void main(String[] args) throws Exception {
    AWSCredentials longTermCredentials =
        new BasicAWSCredentials("accessKey", "secretKey");
    AWSStaticCredentialsProvider staticCredentialProvider =
        new AWSStaticCredentialsProvider(longTermCredentials);

    AmazonWorkDocs workDocs =
        AmazonWorkDocsClient.builder().withCredentials(staticCredentialProvider)
            .withRegion(Regions.US_WEST_2).build();

    List<User> wdUsers = new ArrayList<>();
    DescribeUsersRequest request = new DescribeUsersRequest();

    // The OrganizationId used here is an example and it should be replaced
    // with the OrganizationId of your WorkDocs site.
    request.setOrganizationId("d-123456789c");
    request.setQuery("joe");
```

```
    String marker = null;
    do {
      request.setMarker(marker);
      DescribeUsersResult result = workDocs.describeUsers(request);
      wdUsers.addAll(result.getUsers());
      marker = result.getMarker();
    } while (marker != null);

    System.out.println("List of users matching the query string: joe ");

 for (User wdUser : wdUsers) {
      System.out.printf("Firstname:%s | Lastname:%s | Email:%s | root-folder-id:%s\n",
          wdUser.getGivenName(), wdUser.getSurname(), wdUser.getEmailAddress(),
          wdUser.getRootFolderId());
    }
  }
}
```

通过代入角色连接到 Amazon WorkDocs

此示例使用 AWS Java 开发工具包代入角色并使用该角色的临时安全凭证访问 Amazon WorkDocs。代码示例使用 DescribeFolderContents API 列出用户文件夹中的项目。

```
import java.util.ArrayList;
import java.util.List;

import com.amazonaws.auth.AWSCredentials;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.auth.BasicSessionCredentials;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.securitytoken.AWSSecurityTokenService;
import com.amazonaws.services.securitytoken.AWSSecurityTokenServiceClientBuilder;
import com.amazonaws.services.securitytoken.model.AssumeRoleRequest;
import com.amazonaws.services.securitytoken.model.AssumeRoleResult;
import com.amazonaws.services.workdocs.AmazonWorkDocs;
import com.amazonaws.services.workdocs.AmazonWorkDocsClient;
import com.amazonaws.services.workdocs.model.DescribeFolderContentsRequest;
import com.amazonaws.services.workdocs.model.DescribeFolderContentsResult;
import com.amazonaws.services.workdocs.model.DocumentMetadata;
import com.amazonaws.services.workdocs.model.FolderMetadata;
```

```
public class AssumeRoleDemo {
  private static final String DEMO_ROLE_ARN = "arn:aws:iam::111122223333:role/workdocs-
readonly-role";
  private static AmazonWorkDocs workDocs;

  public static void main(String[] args) throws Exception {

    AWSCredentials longTermCredentials =
        new BasicAWSCredentials("accessKey", "secretKey");

    // Use developer's long-term credentials to call the AWS Security Token Service
 (STS)
 // AssumeRole API, specifying the ARN for the role workdocs-readonly-role in
 // 3rd party AWS account.

    AWSSecurityTokenService stsClient =
        AWSSecurityTokenServiceClientBuilder.standard()
            .withCredentials(new AWSStaticCredentialsProvider(longTermCredentials))
            .withRegion(Regions.DEFAULT_REGION.getName()).build();;

    // If you are accessing a 3rd party account, set ExternalId
    // on assumeRequest using the withExternalId() function.
    AssumeRoleRequest assumeRequest =
        new AssumeRoleRequest().withRoleArn(DEMO_ROLE_ARN).withDurationSeconds(3600)
            .withRoleSessionName("demo");

    AssumeRoleResult assumeResult = stsClient.assumeRole(assumeRequest);

    // AssumeRole returns temporary security credentials for the
 // workdocs-readonly-role

    BasicSessionCredentials temporaryCredentials =
        new BasicSessionCredentials(assumeResult.getCredentials().getAccessKeyId(),
 assumeResult
            .getCredentials().getSecretAccessKey(),
 assumeResult.getCredentials().getSessionToken());

    // Build WorkDocs client using the temporary credentials.
    workDocs =
        AmazonWorkDocsClient.builder()
            .withCredentials(new AWSStaticCredentialsProvider(temporaryCredentials))
            .withRegion(Regions.US_WEST_2).build();
```

```
    // Invoke WorkDocs service calls using the temporary security credentials
    // obtained for workdocs-readonly-role. In this case a call has been made
 // to get metadata of Folders and Documents present in a user's root folder.

    describeFolder("root-folder-id");
  }

  private static void describeFolder(String folderId) {
    DescribeFolderContentsRequest request = new DescribeFolderContentsRequest();
    request.setFolderId(folderId);
    request.setLimit(2);
    List<DocumentMetadata> documents = new ArrayList<>();
    List<FolderMetadata> folders = new ArrayList<>();

    String marker = null;

    do {
      request.setMarker(marker);
      DescribeFolderContentsResult result = workDocs.describeFolderContents(request);
      documents.addAll(result.getDocuments());
      folders.addAll(result.getFolders());
      marker = result.getMarker();
    } while (marker != null);

    for (FolderMetadata folder : folders)
      System.out.println("Folder:" + folder.getName());
    for (DocumentMetadata document : documents)
      System.out.println("Document:" + document.getLatestVersionMetadata().getName());
  }
}
```

上传文档

按照以下步骤将文档上传到 Amazon WorkDocs。

上传文档

1. 如下所示创建 AmazonWorkDocsClient 的实例：

   如果您使用的是 IAM 用户凭证，请参阅使用 IAM 用户凭证连接到 Amazon WorkDocs 并查询用户。如果您代入 IAM 角色，请参阅通过代入角色连接到 Amazon WorkDocs 来了解更多信息。

> **Note**
>
> 为了提高安全性，请尽可能创建联合用户而不是 IAM 用户。

```
AWSCredentials longTermCredentials =
  new BasicAWSCredentials("accessKey", "secretKey");
AWSStaticCredentialsProvider staticCredentialProvider =
  new AWSStaticCredentialsProvider(longTermCredentials);

// Use the region specific to your WorkDocs site.
AmazonWorkDocs amazonWorkDocsClient =
  AmazonWorkDocsClient.builder().withCredentials(staticCredentialProvider)
    .withRegion(Regions.US_WEST_2).build();
```

2. 获取上传的签名 URL，如下所示：

```
InitiateDocumentVersionUploadRequest request = new
 InitiateDocumentVersionUploadRequest();
request.setParentFolderId("parent-folder-id");
request.setName("my-document-name");
request.setContentType("application/octet-stream");
InitiateDocumentVersionUploadResult result =
 amazonWorkDocsClient.initiateDocumentVersionUpload(request);
UploadMetadata uploadMetadata = result.getUploadMetadata();
String documentId = result.getMetadata().getId();
String documentVersionId = result.getMetadata().getLatestVersionMetadata().getId();
String uploadUrl = uploadMetadata.getUploadUrl();
```

3. 使用签名的 URL 上传文档，如下所示：

```
URL url = new URL(uploadUrl);
HttpURLConnection connection = (HttpURLConnection) url.openConnection();
connection.setDoOutput(true);
connection.setRequestMethod("PUT");
// Content-Type supplied here should match with the Content-Type set
// in the InitiateDocumentVersionUpload request.
connection.setRequestProperty("Content-Type","application/octet-stream");
connection.setRequestProperty("x-amz-server-side-encryption", "AES256");
File file = new File("/path/to/file.txt");
FileInputStream fileInputStream = new FileInputStream(file);
```

```
OutputStream outputStream = connection.getOutputStream();
com.amazonaws.util.IOUtils.copy(fileInputStream, outputStream);
connection.getResponseCode();
```

4. 通过将文档状态更改为 ACTIVE 完成上传过程，如下所示：

```
UpdateDocumentVersionRequest request = new UpdateDocumentVersionRequest();
request.setDocumentId("document-id");
request.setVersionId("document-version-id");
request.setVersionStatus(DocumentVersionStatus.ACTIVE);
amazonWorkDocsClient.updateDocumentVersion(request);
```

下载文档

要从 Amazon WorkDocs 下载文档，请按如下方式获取下载的 URL，然后使用开发平台提供的 API 操作，使用该 URL 下载文件。

```
GetDocumentVersionRequest request = new GetDocumentVersionRequest();
request.setDocumentId("document-id");
request.setVersionId("document-version-id");
request.setFields("SOURCE");
GetDocumentVersionResult result = amazonWorkDocsClient.getDocumentVersion(request);
String downloadUrl =
 result.getMetadata().getSource().get(DocumentSourceType.ORIGINAL.name());
```

为 IAM 用户或角色设置通知

要在 Amazon WorkDocs 中创建和管理通知，管理员可以使用 IAM 和 Amazon WorkDocs 控制台。您可以使用 IAM 控制台设置用户权限，并使用 Amazon WorkDocs 控制台来启用通知。启用通知后，即可订阅通知。执行以下步骤。

> **Note**
>
> 为了提高安全性，请尽可能创建联合用户而不是 IAM 用户。

设置 IAM 用户权限

- 使用 IAM 控制台为用户设置以下权限：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": [
            "workdocs:CreateNotificationSubscription",
            "workdocs:DeleteNotificationSubscription",
            "workdocs:DescribeNotificationSubscriptions"
            ],
        "Resource": "*"
        }
    ]
}
```

启用通知

启用通知后，您就可在订阅通知后调用 CreateNotificationSubscription。

1. 打开 Amazon WorkDocs 控制台，网址为：https://console.aws.amazon.com/zocalo/。
2. 在管理您的 WorkDocs 站点页面上，选择所需目录并选择操作，然后选择管理通知。
3. 在管理通知页面上，选择启用通知。
4. 输入您要允许从 Amazon WorkDocs 站点接收通知的用户或角色的 ARN。

有关让 Amazon WorkDocs 使用通知的信息，请参阅将 Amazon WorkDocs API 与适用于 Python 的 AWS SDK 和 AWS Lambda 结合使用。启用通知后，您和您的用户即可订阅通知。

订阅 WorkDocs 通知

1. 准备端点以处理 Amazon SNS 消息。有关更多信息，请参阅《Amazon Simple Notification Service 开发人员指南》中的扇出到 HTTP/S 端点。

> **Important**
>
> SNS 会向您配置的端点发送确认消息。您必须确认此消息才能收到通知。另外，如果在通过命令行界面或 API 访问 AWS 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》。

2. 执行以下操作：

   - 获取组织 ID

     1. 在 AWS Directory Service 控制台导航窗格中，选择目录。
     2. 与您的 Amazon WorkDocs 站点对应的目录 ID 也可用作该站点的组织 ID。

   - 如下所示创建订阅请求：

```
CreateNotificationSubscriptionRequest request = new
 CreateNotificationSubscriptionRequest();
request.setOrganizationId("d-1234567890");
request.setProtocol(SubscriptionProtocolType.Https);
request.setEndpoint("https://my-webhook-service.com/webhook");
request.setSubscriptionType(SubscriptionType.ALL);
CreateNotificationSubscriptionResult result =
 amazonWorkDocsClient.createNotificationSubscription(request);
System.out.println("WorkDocs notifications subscription-id: "
 result.getSubscription().getSubscriptionId());
```

SNS 通知

消息包含以下信息：

- `organizationId` – 组织的 ID。
- `parentEntityType` – 父级的类型 (`Document` | `DocumentVersion` | `Folder`)。
- `parentEntityId` – 父级的 ID。
- `entityType` – 实体的类型 (`Document` | `DocumentVersion` | `Folder`)。
- `entityId` – 实体的 ID。
- 操作 – 操作，可以为以下值之一：
  - `delete_document`
  - `move_document`
  - `recycle_document`
  - `rename_document`

- revoke_share_document
- share_document
- upload_document_version

禁用通知

1. 打开 Amazon WorkDocs 控制台，网址为：https://console.aws.amazon.com/zocalo/。
2. 在管理您的 WorkDocs 站点页面上，选择所需目录并选择操作，然后选择管理通知。
3. 在管理通知页面上，选择您希望对其禁用通知的 ARN，然后选择禁用通知。

创建用户

以下示例展示了如何在 Amazon Workdocs 中创建用户。

> **Note**
>
> 对于已连接的 AD 配置，这不是有效的操作。要在已连接的 AD 配置中创建用户，该用户必须已存在于企业目录中。然后，您必须调用 ActivateUser API 来激活 Amazon WorkDocs 中的用户。

以下示例演示了如何创建存储配额为 1 GB 的用户。

```
CreateUserRequest request = new CreateUserRequest();
    request.setGivenName("GivenName");
    request.setOrganizationId("d-12345678c4");
    // Passwords should:
    //   Be between 8 and 64 characters
    //   Contain three of the four below:
    //   A Lowercase Character
    //   An Uppercase Character
    //   A Number
    //   A Special Character
    request.setPassword("Badpa$$w0rd");
    request.setSurname("surname");
    request.setUsername("UserName");
    StorageRuleType storageRule = new StorageRuleType();
    storageRule.setStorageType(StorageType.QUOTA);
```

```
    storageRule.setStorageAllocatedInBytes(new Long(10485761));
    request.setStorageRule(storageRule);
    CreateUserResult result = workDocsClient.createUser(request);
```

按照以下步骤从 AWS 控制台获取 Amazon WorkDocs 组织 ID：

获取组织 ID

1. 在 AWS Directory Service 控制台导航窗格中，选择目录。
2. 记下与您的 Amazon WorkDocs 站点对应的目录 ID 值。这是该站点的组织 ID。

向用户授予对资源的权限

以下示例显示了如何使用 AddResourcePermissions API 向 USER 授予对资源的 CONTRIBUTOR 权限。您还可以使用该 API 向文件夹或文档的用户或组授予权限。

```
AddResourcePermissionsRequest request = new AddResourcePermissionsRequest();
    request.setResourceId("resource-id");
    Collection<SharePrincipal> principals = new ArrayList<>();;
    SharePrincipal principal = new SharePrincipal();
    principal.setId("user-id");
    principal.setType(PrincipalType.USER);
    principal.setRole(RoleType.CONTRIBUTOR);
    principals.add(principal);
    request.setPrincipals(principals);
    AddResourcePermissionsResult result =
 workDocsClient.addResourcePermissions(request);
```

管理应用程序的身份验证和访问控制

Amazon WorkDocs 管理 API 通过 IAM 策略进行身份验证和授权。IAM 管理员可以创建 IAM 策略，并将其附加到开发人员用于访问该 API 的 IAM 角色或用户。

下面提供了示例：

任务

- 向开发人员授予对 Amazon WorkDocs API 的使用权限
- 向第三方开发人员授予对 Amazon WorkDocs API 的使用权限
- 授予用户代入 IAM 角色的权限
- 限制对特定 Amazon WorkDocs 实例的访问权限

向开发人员授予对 Amazon WorkDocs API 的使用权限

> **Note**
>
> 为了提高安全性，请尽可能创建联合用户而不是 IAM 用户。

如果您是 IAM 管理员，则可以向同一 AWS 账户中的 IAM 用户授予 Amazon WorkDocs API 访问权限。为此，请创建一个 Amazon WorkDocs API 权限策略并将其附加到 IAM 用户。以下 API 策略可授予对各种 `Describe` API 的只读权限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "WorkDocsAPIReadOnly",
            "Effect": "Allow",
            "Action": [
                "workdocs:Get*",
      "workdocs:Describe*"
            ],
            "Resource": [
                "*"
            ]
        }
```

```
    ]
}
```

向第三方开发人员授予对 Amazon WorkDocs API 的使用权限

您可以将访问权限授予第三方开发人员，或使用不同 AWS 账户的用户。为此，请创建一个 IAM 角色，并附加 Amazon WorkDocs API 允许策略。

在下列情况下，需要使用这种形式的访问权限：

- 开发人员属于同一组织，但开发人员的 AWS 账户与 Amazon WorkDocs AWS 账户不同。
- 当企业希望向第三方应用程序开发人员授予 Amazon WorkDocs API 访问权限时。

在这两种情况下，都会涉及两个 AWS 账户，一个开发人员的 AWS 账户和一个不同的托管 Amazon WorkDocs 站点的账户。

开发人员将需要提供以下信息，以便账户管理员可以创建 IAM 角色：

- 您的 AWS 账户 ID
- 您的客户将用来标识您的唯一 `External ID`。有关更多信息，请参阅如何在向第三方授予对 AWS 资源的访问权时使用外部 ID。
- 您的应用程序需要访问的 Amazon WorkDocs API 列表。基于 IAM 的策略控制提供了粒度控制，即能够在单个 API 级别定义允许或拒绝策略。有关 Amazon WorkDocs API 的列表，请参阅 Amazon WorkDocs API 参考。

以下过程描述了为跨账户访问权限配置 IAM 所涉及的步骤。

配置 IAM 以进行跨账户存取

1. 创建 Amazon WorkDocs API 权限策略，将其称为 `WorkDocsAPIReadOnly` 策略。
2. 在托管 Amazon WorkDocs 站点的 AWS 账户的 IAM 控制台中创建新角色：
    a. 登录 AWS Management Console，然后通过以下网址打开 IAM 控制台：https://console.aws.amazon.com/iam/。
    b. 在控制台的导航窗格中，单击角色，然后单击创建新角色。
    c. 对于角色名称，键入有助于识别此角色作用的角色名称，例如 `workdocs_app_role`。角色名称在您的 AWS 账户内必须是唯一的。输入名称后，单击下一步。

d. 在选择角色类型页面上，选择跨账户存取部分，然后选择要创建的角色类型：

- 如果您是用户账户和资源账户的管理员，或这两个账户属于同一个公司，则选择提供在您的 AWS 账户之间进行相互访问的权限。这也是要访问的用户、角色和资源都在相同账户中时可供您选择的选项。
- 如果您是 Amazon WorkDocs 站点所有者账户的管理员，并且想要向应用程序开发人员账户的用户授予权限，请选择提供在您的 AWS 账户和第三方 AWS 账户之间进行相互访问的权限。此选项要求您指定外部 ID（必须由第三方向您提供），从而对第三方可以使用该角色访问您的资源的情况提供更多的控制。有关更多信息，请参阅如何在向第三方授予对 AWS 资源的访问权限时使用外部 ID。

e. 在下一页上，指定要向其授予资源访问权限的 AWS 账户 ID，并针对第三方访问输入外部 ID。

f. 单击下一步附加策略。

3. 在附加策略页面上，搜索之前创建的 Amazon WorkDocs API 权限策略并选择该策略旁边的框，然后单击下一步。
4. 查看详细信息，复制角色 ARN 以供将来参考，然后单击创建角色以完成角色的创建。
5. 与开发人员共享角色 ARN。以下是角色 ARN 的示例：

```
arn:aws:iam::AWS-ACCOUNT-ID:role/workdocs_app_role
```

授予用户代入 IAM 角色的权限

拥有 AWS 管理账户的开发人员可以允许用户代入 IAM 角色。为此，您需要创建新策略并将其附加到相应用户。

策略必须包含一个语句，该语句对 `sts:AssumeRole` 操作以及 `Resource` 元素中的角色的 Amazon 资源名称 (ARN) 具有 `Allow` 效果，如以下示例所示。获得了该策略（通过组成员资格或直接附加）的用户可以切换到指定角色。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::<aws_account_id>:role/workdocs_app_role"
  }
```

```
}
```

限制对特定 Amazon WorkDocs 实例的访问权限

如果您在 AWS 账户上有多个 Amazon WorkDocs 站点，并且要向特定站点授予 API 访问权限，则可以通过定义 `Condition` 元素来执行此操作。`Condition` 元素允许您在策略生效时指定策略的条件。

以下示例展示了条件元素：

```
    "Condition":
    {
                "StringEquals": {
                    "Resource.OrganizationId": "d-123456789c5"
                }
    }
```

在策略中放置上述条件后，用户只能访问 ID 为 `d-123456789c5` 的 Amazon WorkDocs 实例。Amazon WorkDocs 实例 ID 有时被称为组织 ID 或目录 ID。有关更多信息，请参阅限制对特定 Amazon WorkDocs 实例的访问权限。

按照以下步骤从 AWS 控制台获取 Amazon WorkDocs 组织 ID：

获取组织 ID

1. 在 AWS Directory Service 控制台导航窗格中，选择目录。
2. 记下与您的 Amazon WorkDocs 站点对应的目录 ID 值。这是该站点的组织 ID。

用户应用程序的身份验证和访问控制

Amazon WorkDocs 用户级应用程序是通过 Amazon WorkDocs 控制台注册和管理的。开发人员应在 Amazon WorkDocs 控制台上的 My Applications 页面上注册其应用程序，该页面将为每个应用程序提供唯一的 ID。在注册过程中，开发人员应指定要接收访问令牌的重定向 URI 以及应用程序范围。

当前，应用程序只能访问在其注册的同一 AWS 账户中的 Amazon WorkDocs 站点。

目录

授予调用 Amazon WorkDocs API 的权限

命令行界面用户必须拥有对 Amazon WorkDocs 和 AWS Directory Service 的完全权限。如果没有这些权限，任何 API 调用都会返回 UnauthorizedResourceAccessException 消息。以下策略可授予完全权限。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
          "workdocs:*",
          "ds:*",
          "ec2:CreateVpc",
          "ec2:CreateSubnet",
          "ec2:CreateNetworkInterface",
          "ec2:CreateTags",
          "ec2:CreateSecurityGroup",
          "ec2:DescribeVpcs",
          "ec2:DescribeSubnets",
          "ec2:DescribeNetworkInterfaces",
```

```
            "ec2:DescribeAvailabilityZones",
            "ec2:AuthorizeSecurityGroupEgress",
            "ec2:AuthorizeSecurityGroupIngress",
            "ec2:DeleteSecurityGroup",
            "ec2:DeleteNetworkInterface",
            "ec2:RevokeSecurityGroupEgress",
            "ec2:RevokeSecurityGroupIngress"
            ],
        "Effect": "Allow",
        "Resource": "*"
      }
    ]
}
```

如果您需要授予只读权限，请使用以下策略。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
          "workdocs:Describe*",
          "ds:DescribeDirectories",
          "ec2:DescribeVpcs",
          "ec2:DescribeSubnets"
          ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

在该策略中，第一个操作授予对所有 Amazon WorkDocs `Describe` 操作的访问权限。`DescribeDirectories` 操作获取有关 AWS Directory Service 目录的信息。Amazon EC2 操作使 Amazon WorkDocs 能够获取您的 VPC 和子网的列表。

在 API 调用中使用文件夹 ID

每当 API 调用访问文件夹时，您都必须使用文件夹 ID，而不是文件夹名称。例如，如果您传递 `client.get_folder(FolderId='MyDocs')`，API 调用会返回 UnauthorizedResourceAccessException 消息和以下 404 消息。

```
client.get_folder(FolderId='MyDocs')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "C:\Users\user-name\AppData\Local\Programs\Python\Python36-32\lib\site-packages
\botocore\client.py", line 253, in _api_call
    return self._make_api_call(operation_name, kwargs)
  File "C:\Users\user-name\AppData\Local\Programs\Python\Python36-32\lib\site-packages
\botocore\client.py", line 557, in _make_api_call
    raise error_class(parsed_response, operation_name)
botocore.errorfactory.UnauthorizedResourceAccessException: An error occurred
 (UnauthorizedResourceAccessException) when calling the GetFolder operation:
Principal [arn:aws:iam::395162986870:user/Aman] is not allowed to execute
 [workdocs:GetFolder] on the resource.
```

为避免这种情况，请使用文件夹 URL 中的 ID。

site.workdocs/index.html#/folder/
abc123def456ghi789jkl789mno4be7024df198736472dd50ca970eb22796082e3d489577.

传递该 ID 将返回正确的结果。

```
client.get_folder(FolderId='abc123def456ghi789jkl789mno4be7024df198736472dd50ca970eb22796082e3d
{'ResponseMetadata': {'RequestId': 'f8341d4e-4047-11e7-9e70-afa8d465756c',
 'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amzn-requestid': 'f234564e-1234-56e7-89e7-
a10fa45t789c', 'cache-control': 'private, no-cache, no-store, max-age=0',
 'content-type': 'application/json', 'content-length': '733', 'date':
 'Wed, 24 May 2017 06:12:30 GMT'}, 'RetryAttempts': 0}, 'Metadata': {'Id':
 'abc123def456ghi789jkl789mno4be7024df198736472dd50ca970eb22796082e3d489577', 'Name':
 'sentences', 'CreatorId':
 'S-1-5-21-2125721135-1643952666-3011040551-2105&d-906724f1ce', 'ParentFolderId':
 '0a811a922403ae8e1d3c180f4975f38f94372c3d6a2656c50851c7fb76677363',
 'CreatedTimestamp': datetime.datetime(2017, 5, 23, 12, 59, 13, 8000,
 tzinfo=tzlocal()), 'ModifiedTimestamp': datetime.datetime(2017, 5, 23, 13,
 13, 9, 565000, tzinfo=tzlocal()), 'ResourceState': 'ACTIVE', 'Signature':
 'b7f54963d60ae1d6b9ded476f5d20511'}}
```

创建应用程序

作为 Amazon WorkDocs 管理员，请使用以下步骤创建应用程序。

创建应用程序

1. 打开 Amazon WorkDocs 控制台，网址为：https://console.aws.amazon.com/zocalo/。
2. 依次选择我的应用程序和创建应用程序。
3. 输入以下值：

   应用程序名称

   应用程序的名称。

   电子邮件

   要与应用程序关联的电子邮件地址。

   应用程序描述

   应用程序的描述。

   重定向 URI

   希望 Amazon WorkDocs 将流量重定向到的位置。

   应用程序范围

   希望应用程序具有的读取或写入范围。有关更多详细信息，请参阅应用程序范围。
4. 选择创建。

应用程序范围

Amazon WorkDocs 支持以下应用程序范围：

- 内容读取 (`workdocs.content.read`)，它使您的应用程序可以访问以下 Amazon WorkDocs API：
  - Get*
  - Describe*
- 内容写入 (`workdocs.content.write`)，它使您的应用程序可以访问以下 Amazon WorkDocs API：
  - Create*
  - Update*
  - Delete*

- Initiate*
- Abort*
- Add*
- Remove*

授权

应用程序注册完成后，应用程序可以代表任何 Amazon WorkDocs 用户请求授权。为此，应用程序应访问 Amazon WorkDocs OAuth 端点 (`https://auth.amazonworkdocs.com/oauth`)，并提供以下查询参数：

- [必需] `app_id` – 注册应用程序时生成的应用程序 ID。
- [必需] `auth_type` – 请求的 OAuth 类型。支持的值为 `ImplicitGrant`。
- [必需] `redirect_uri` – 为应用程序注册以接收访问令牌的重定向 URI。
- [可选] `scopes` – 范围的逗号分隔列表。如果未指定，将会使用在注册期间选择的范围的列表。
- [可选] `state` – 与访问令牌一起返回的字符串。

> **Note**
>
> 如果在通过命令行界面或 API 访问 AWS 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》。

一个示例 GET 请求，用以启动 OAuth 流以获取访问令牌：

```
GET https://auth.amazonworkdocs.com/oauth?app_id=my-app-
id&auth_type=ImplicitGrant&redirect_uri=https://myapp.com/
callback&scopes=workdocs.content.read&state=xyz
```

以下情形发生在 OAuth 授权流程中：

1. 系统会提示应用程序用户输入 Amazon WorkDocs 站点名称。
2. 用户将被重定向到 Amazon WorkDocs 身份验证页面以输入其凭证。

3. 成功进行身份验证后，系统会向用户显示同意屏幕，允许用户向您的应用程序授予或拒绝访问 Amazon WorkDocs 的授权。
4. 在用户选择同意屏幕上的 `Accept` 后，他们的浏览器将连同作为查询参数的访问令牌和区域信息一起被重定向到应用程序的回调 URL。

来自 Amazon WorkDocs 的 GET 请求示例：

```
GET https://myapp.com/callback?acessToken=accesstoken&region=us-east-1&state=xyz
```

除访问令牌外，Amazon WorkDocs OAuth 服务还会将 `region` 作为选定 Amazon WorkDocs 站点的查询参数返回。外部应用程序应使用 `region` 参数确定 Amazon WorkDocs 服务端点。

如果在通过命令行界面或 API 访问 AWS 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》。

调用 Amazon WorkDocs API

在获取访问令牌后，应用程序可以对 Amazon WorkDocs 服务进行 API 调用。

> **Important**

以下示例展示了如何使用 curl GET 请求获取文档的元数据。

```
Curl "https://workdocs.us-east-1.amazonaws.com/api/v1/documents/{document-id}" -H
 "Accept: application/json" -H "Authentication: Bearer accesstoken"
```

描述用户根文件夹的 JavaScript 函数示例：

```
function printRootFolders(accessToken, siteRegion) {
    var workdocs = new AWS.WorkDocs({region: siteRegion});
    workdocs.makeUnauthenticatedRequest("describeRootFolders", {AuthenticationToken:
 accessToken}, function (err, folders) {
        if (err) console.log(err);
        else console.log(folders);
    });
}
```

以下展示了基于 Java 的 API 调用示例：

```
AWSCredentialsProvider credentialsProvider = new AWSCredentialsProvider() {
  @Override
  public void refresh() {}

  @Override
  public AWSCredentials getCredentials() {
    new AnonymousAWSCredentials();
  }
};

// Set the correct region obtained during OAuth flow.
workDocs =
    AmazonWorkDocsClient.builder().withCredentials(credentialsProvider)
        .withRegion(Regions.US_EAST_1).build();

DescribeRootFoldersRequest request = new DescribeRootFoldersRequest();
request.setAuthenticationToken("access-token-obtained-through-workdocs-oauth");
DescribeRootFoldersResult result = workDocs.describeRootFolders(request);

for (FolderMetadata folder : result.getFolders()) {
  System.out.printf("Folder name=%s, Id=%s \n", folder.getName(), folder.getId());
}
```

Amazon WorkDocs 内容管理器

Amazon WorkDocs 内容管理器是一项高级实用工具，可从 Amazon WorkDocs 站点上传或下载内容。

主题

构造 Amazon WorkDocs 内容管理器

您可将 Amazon WorkDocs 内容管理器用于管理和用户应用程序。

对于用户应用程序，开发人员必须使用匿名 AWS 凭证和身份验证令牌构造 Amazon WorkDocs 内容管理器。

对于管理应用程序，必须使用 AWS Identity and Access Management (IAM) 凭证初始化 Amazon WorkDocs 客户端。此外，在后续 API 调用中必须省略身份验证令牌。

以下代码演示如何使用 Java 或 C# 初始化用户应用程序的 Amazon WorkDocs 内容管理器。

Java：

```
AWSStaticCredentialsProvider credentialsProvider = new AWSStaticCredentialsProvider(new
 AnonymousAWSCredentials());

AmazonWorkDocs client =
 AmazonWorkDocsClient.builder().withCredentials(credentialsProvider).withRegion("region").build

ContentManager contentManager =
 ContentManagerBuilder.standard().withWorkDocsClient(client).withAuthenticationToken("token").b
```

C#：

```
AmazonWorkDocsClient client = new AmazonWorkDocsClient(new AnonymousAWSCredentials(),
 "region");
ContentManagerParams params = new ContentManagerParams
{
```

```
WorkDocsClient = client,
AuthenticationToken = "token"
};
IContentManager workDocsContentManager = new ContentManager(param);
```

下载文档

开发人员可以使用 Amazon WorkDocs 内容管理器从 Amazon WorkDocs 下载特定版本或最新版本的文档。下面是如何使用 Java 和 C# 下载特定版本的文档的示例。

> **Note**
>
> 要下载文档的最新版本，请不要在构造 `VersionId` 请求时指定 `GetDocumentStream`。

Java

```
ContentManager contentManager =
 ContentManagerBuilder.standard().withWorkDocsClient(client).withAuthenticationToken("auth-
token").build();

// Download document.
GetDocumentStreamRequest request = new GetDocumentStreamRequest();
request.setDocumentId("document-id");
request.setVersionId("version-id");

// stream contains the content of the document version.
InputStream stream = contentManager.getDocumentStream(request).getStream();
```

C#

```
ContentManager contentManager =
 ContentManagerBuilder.standard().withWorkDocsClient(client).withAuthenticationToken("auth-
token").build();

// Download document.
GetDocumentStreamRequest request = new GetDocumentStreamRequest();
request.setDocumentId("document-id");
request.setVersionId("version-id");

// stream contains the content of the document version.
```

```
InputStream stream = contentManager.getDocumentStream(request).getStream();
```

上传文档

Amazon WorkDocs 内容管理器提供了一个 API，可用于将内容上传到 Amazon WorkDocs 站点。以下示例演示了如何使用 Java 和 C # 上传文档。

Java

```
File file = new File("file-path");
InputStream stream = new FileInputStream(file);
UploadDocumentStreamRequest request = new UploadDocumentStreamRequest();
request.setParentFolderId("destination-folder-id");
request.setContentType("content-type");
request.setStream(stream);
request.setDocumentName("document-name");
contentManager.uploadDocumentStream(request);
```

C#

```
var stream = new FileStream("file-path", FileMode.Open);

UploadDocumentStreamRequest uploadDocumentStreamRequest = new
 UploadDocumentStreamRequest()
{
ParentFolderId = "destination-id",
DocumentName = "document-name",
ContentType = "content-type",
Stream = stream
};

workDocsContentManager.UploadDocumentStreamAsync(uploadDocumentStreamRequest).Wait();
```