aws

開發人員指南

Amazon WorkDocs

Amazon WorkDocs: 開發人員指南

Table of Contents

注意：Amazon WorkDocs 不再提供新客戶註冊和帳戶升級。在此處了解遷移步驟：如何從 Amazon 遷移資料 WorkDocs。

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。

什麼是亞馬遜 WorkDocs？

亞馬遜 WorkDocs 是一個文檔存儲，協作和共享系統。Amazon WorkDocs 完全受管、安全且具有企業規模。它提供強大的管理控制，以及有助於提高用戶生產力的反饋功能。檔案會存放在雲端，安全無虞。使用者的檔案只有他們能看見，而他們可以指定參與者和檢視者。除非他們特別授與存取，否則組織中其他成員無法存取其他使用者的任何檔案。

使用者可以與組織的其他成員共用檔案，以執行協同合作或檢閱。Amazon 用 WorkDocs 戶端應用程式可用來檢視許多不同類型的檔案，具體取決於檔案的網際網路媒體類型。亞馬遜 WorkDocs 支持所有常見的文檔和圖像格式，並不斷添加對其他媒體類型的支持。

如有資訊，請參閱 Amazon WorkDocs

訪問亞馬遜 WorkDocs

最終使用者使用用戶端應用程式來存取其檔案。非管理使用者永遠不需要使用 Amazon WorkDocs 主控台或管理儀表板。Amazon WorkDocs 提供數種不同的用戶端應用程式和公用程

- 用來文件管理和檢閱的 web 應用程式。
- 用來文件檢閱的行動裝置原生應用程式。
- 亞馬遜 WorkDocs 驅動器用於將 Mac 或 Windows 桌面上的文件夾與亞馬遜 WorkDocs 文件同步。

定價

使用亞馬遜 WorkDocs，沒有預付費用或承諾。您只需為使用中的使用者帳戶和使用的儲存空間付費。如需詳資訊，請前往定價

資源

以下相關資源可協助您使用此服務。

- 課程和研討會 – 連結至以角色為基礎的專門課程以及自主進度實驗室，協助加強您的 AWS 技能，並取得實際體驗。
- AWS 開發人員中心 – 研究教學課程、下載工具，以及瞭解 AWS 開發人員活動。
- AWS 開發人員工具 – 連結至開發人員工具、軟體開發套件、IDE 工具組和命令列工具，用來開發及管理 AWS 應用程式。

- 入門資源中心 – 瞭解如何設定 AWS 帳戶、加入 AWS 社群，並啟動您的第一個應用程式。
- 實用的教學課程-按照 step-by-step 教學課程啟動第一個應用程式AWS。
- AWS 白皮書 – 連結至完整的技術 AWS 白皮書清單，其中涵蓋了架構、安全和成本等主題，並由 AWS 解決方案架構師或其他技術專家撰寫。
- AWS Support 中心 – 建立和管理您的 AWS Support 案例的中心。這也包含與其他實用資源的連結，例如論壇、技術常見問答集、服務運作狀態以及 AWS Trusted Advisor。
- AWS Support— 有關的資訊的主要網頁AWS Support，它是一個快速回應支援頻道 one-on-one，可協助您在雲端中建置並執行應用程式。
- 聯絡我們 – 查詢有關 AWS 帳單、帳戶、事件、濫用與其他問題的聯絡中心。
- AWS 網站條款 – 我們的著作權與商標；您的帳戶、授權與網站存取；以及其他主題的詳細資訊。

開始使用

下列程式碼片段可協助您開始使用 Amazon 開 WorkDocs 發套件。

> **Note**
>
> 為了獲得更高的安全性，請盡可能建立聯合身分使用者而非 IAM 使用者。

範例

- 使用 IAM 使用者登入資料 WorkDocs 與使用者查詢 Connect 到 Amazon
- WorkDocs 通過假定一個角色連接到 Amazon
- 上傳文件
- 下載文件
- 設定通知
- 建立使用者
- 授與使用者資源的權限

使用 IAM 使用者登入資料 WorkDocs 與使用者查詢 Connect 到 Amazon

下列程式碼示範如何使用 IAM 使用者的 API 登入資料進行 API 呼叫。在這種情況下，API 用戶和 Amazon WorkDocs 網站屬於同一個AWS帳戶。

> **Note**
>
> 為了獲得更高的安全性，請盡可能建立聯合身分使用者而非 IAM 使用者。

確保 IAM 使用者已透過適當的 IAM 政策授予 Amazon WorkDocs API 存取權限。

程式碼範例使用 DescribeUsersAPI 來搜尋使用者，並取得使用者的中繼資料。使用者中繼資料提供詳細資訊，例如名字、姓氏、使用者 ID 和根資料夾 ID。如果您想要代表使用者執行任何內容上傳或下載作業，則根資料夾 ID 特別有用。

該代碼要求您獲得 Amazon WorkDocs 組織 ID。

請依照下列步驟從AWS主控台取得 Amazon WorkDocs 組織 ID：

取得組織 ID

1. 在 AWS Directory Service 主控台導覽窗格中，選擇 Directories (目錄)。
2. 請注意與您的 Amazon WorkDocs 網站對應的目錄 ID 值。這是場地的組織 ID。

下列範例顯示如何使用 IAM 登入資料進行 API 呼叫。

```
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());
    }
  }
}
```

WorkDocs 通過假定一個角色連接到 Amazon

此範例使用 AWS Java SDK 擔任角色，並使用該角色的臨時安全登入資料存取 Amazon WorkDocs。程式碼範例會使用 DescribeFolderContentsAPI 列出使用者資料夾中的項目。

```
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());
  }
}
```

上傳文件

> **Note**
>
> 您必須是軟體開發人員才能完成本節中的步驟。如需使用 Amazon 上傳檔案 WorkDocs 的相關資訊，請參閱 Amazon 使用 WorkDocs 者指南中的上傳檔案。

使用以下步驟將文件上傳到 Amazon WorkDocs。

若要上傳文件

1. 建立 AmazonWorkDocsClient 執行個體，如下所示：

   如果您使用 IAM 使用者登入資料，請參閱使用 IAM 使用者登入資料 WorkDocs 與使用者查詢 Connect 到 Amazon。如果您擔任 IAM 角色，請參閱以取 WorkDocs 通過假定一個角色連接到 Amazon得更多資訊。

   > **Note**
   >
   > 為了獲得更高的安全性，請盡可能建立聯合身分使用者而非 IAM 使用者。

   ```
   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. 為上傳取得簽章的 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. 使用簽章的 URL 上傳文件，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. 變更文件狀態為 ACTIVE 以完成上傳程序，如下所示：

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

下載文件

> **Note**
>
> 您必須是軟體開發人員才能完成本節中的步驟。如需使用 Amazon 下載檔案 WorkDocs 的相關資訊，請參閱 Amazon 使用 WorkDocs 者指南中的下載檔案。

若要從 Amazon 下載文件 WorkDocs，請取得下載的 URL，如下所示，然後使用開發平台提供的 API 動作，使用 URL 下載檔案。

```
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());
```

設定通知

您可以依照下列程序來設定通知：

1. 設定 IAM 使用者或角色許可，以允許來電者存取通知訂閱管理 API。
2. 呼叫通知訂閱 API，以啟用或停用將 SNS 訊息發佈到您的端點。

> **Note**
>
> 為了獲得更高的安全性，請盡可能建立聯合身分使用者而非 IAM 使用者。

若要設定 IAM 使用者許可

- 使用 IAM 主控台為使用者設定下列許可：

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

啟用通知

啟用通知可讓您在訂閱通知CreateNotificationSubscription後撥打電話。

1. 在以下位置打開 Amazon WorkDocs 控制台 https://console.aws.amazon.com/zocalo/。
2. 在「管理您的 WorkDocs 網站」頁面上，選取所需的目錄並選擇「動作」，然後選擇「管理通知」。
3. 在 Manage Notifications (管理通知) 頁面，選擇 Enable Notifications (啟用通知)。

4. 針對您要允許從 Amazon WorkDocs 網站接收通知的使用者或角色輸入 ARN。

如需啟用 Amazon 使用通知的相關資訊，請參閱 WorkDocs 將 Amazon WorkDocs API 與適用於 Python 和 AWS AWS Lambda 開發套件搭配使用。啟用通知後，您和您的使用者即可訂閱通知。

若要訂閱 WorkDocs 通知

1. 準備您的端點以處理 Amazon SNS 訊息。如需詳細資訊，請參閱 Amazon 簡單通知服務開發人員指南中的散播到 HTTP/S 端點。

   > **Important**
   >
   > SNS 會將確認訊息傳送至您設定的端點。您必須確認此訊息才能接收通知。此外，如果您在透過命令列界面或 API 存取 AWS 時需要經過 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

2. 請執行下列操作：

   - 取得組織 ID
     1. 在 AWS Directory Service 主控台導覽窗格中，選取目錄。
     2. 與您的 Amazon WorkDocs 網站對應的目錄 ID 也可做為該網站的組織 ID。
   - 建立訂閱請求，如下所示：

```
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

訊息包括下列資訊：

- `organizationId`— 組織的識別碼。
- `parentEntityType`— 父對象的類型（`Document`| `DocumentVersion` |`Folder`）。
- `parentEntityId`— 父項的識別碼。
- `entityType`— 實體的類型（`Document`| `DocumentVersion` |`Folder`）。
- `entityId`— 實體的識別碼。
- 動作 — 動作，可以是下列其中一個值：
  - `delete_document`
  - `move_document`
  - `recycle_document`
  - `rename_document`
  - `revoke_share_document`
  - `share_document`
  - `upload_document_version`

若要關閉通知

1. 在以下位置打開 Amazon WorkDocs 控制台 https://console.aws.amazon.com/zocalo/。
2. 在「管理您的 WorkDocs 網站」頁面上，選取所需的目錄並選擇「動作」，然後選擇「管理通知」。
3. 在 Manage Notifications (管理通知) 頁面上，選擇您想要停用通知的 ARN，並選擇 Disable Notifications (停用通知)。

建立使用者

下面的例子演示了在 Amazon 創建一個用戶 WorkDocs。

> **Note**
>
> 這不是 Connected AD configuration (連接 AD 組態) 的有效操作。若要在 [連線的 AD] 組態中建立使用者，使用者必須已經存在於企業目錄中。然後，您必須對 ActivateUserAPI 進行調用以激活 Amazon 中的用戶 WorkDocs。

下列範例會示範如何建立儲存配額為 1 GB 的使用者。

```
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(1048576l));
    request.setStorageRule(storageRule);
    CreateUserResult result = workDocsClient.createUser(request);
```

請依照下列步驟從AWS主控台取得 Amazon WorkDocs 組織 ID：

取得組織 ID

1. 在 AWS Directory Service 主控台導覽窗格中，選擇 Directories (目錄)。
2. 請注意與您的 Amazon WorkDocs 網站對應的目錄 ID 值。這是場地的組織 ID。

授與使用者資源的權限

下列範例顯示如何使用 AddResourcePermissionsAPI 將CONTRIBUTOR權限授與USER資源。您也可以使用 API 授予資料夾或文件上的使用者或群組的權限。

```
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);
```

管理應用程式的身分驗證與存取控制

亞馬遜 WorkDocs 管理 API 通過 IAM 政策進行身份驗證和授權。IAM 管理員可以建立允許開發人員存取 API 的 IAM 政策，並將其連接至一個 IAM 角色或使用者。

以下提供說明範例：

任務

- 授與開發人員權限給亞馬遜 WorkDocs API
- 授與第三方開發人員使用 Amazon WorkDocs API 的權限
- 授與使用者假設 IAM 角色的權限
- 限制對特定 Amazon WorkDocs 執行個體的存取

授與開發人員權限給亞馬遜 WorkDocs API

> **Note**
>
> 為了獲得更高的安全性，請盡可能建立聯合身分使用者而非 IAM 使用者。

如果您是 IAM 管理員，則可以從同一AWS帳戶將 Amazon WorkDocs API 存取權授予 IAM 使用者。若要這麼做，請建立 Amazon WorkDocs API 許可政策並將其附加到 IAM 使用者。下列 API 政策會授與各種`Describe` API 的唯讀權限。

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

```
        }
    ]
}
```

授與第三方開發人員使用 Amazon WorkDocs API 的權限

您可以向第三方開發人員、或是使用不同的 AWS 帳戶之使用者授予存取權。若要這麼做，請建立 IAM 角色，並附加 Amazon WorkDocs API 允許政策。

下列情況需要使用這種存取權限：

- 開發人員屬於同一個組織，但開發人員的AWS帳戶與 Amazon WorkDocs AWS 帳戶不同。
- 企業想要將 Amazon WorkDocs API 存取權授予第三方應用程式開發人員時。

在這兩種情況下，都涉及兩個AWS帳戶，一個開發人員的AWS帳戶和託管 Amazon WorkDocs 網站的不同帳戶。

開發人員需要提供下列資訊，帳戶管理員才能建立 IAM 角色：

- 您的 AWS 帳戶 ID
- 讓客戶能識別您的一個唯一的 `External ID`。如需詳細資訊，請參閱將 AWS 資源的存取權授予第三方時如何使用外部 ID。
- 您的應用程式需要存取的 Amazon WorkDocs API 清單。以 IAM 為基礎的政策控制提供精細控制，能夠在個別 API 層級定義允許或拒絕政策。有關亞馬遜 WorkDocs API 的列表，請參閱亞馬遜 WorkDocs API 參考。

下列程序說明涉及了跨帳戶存取的 IAM 設定步驟。

若要設定跨帳戶存取的 IAM

1. 創建一個亞馬遜 WorkDocs API 許可政策，稱之為`WorkDocsAPIReadOnly`政策。
2. 在託管 Amazon WorkDocs 網站的AWS帳戶的 IAM 主控台中建立新角色：
    a. 登入 AWS Management Console，並開啟位於 https://console.aws.amazon.com/iam/ 的 IAM 主控台。
    b. 在主控台的導覽窗格中，按一下 Roles (角色)，然後按一下 Create New Role (建立新角色)。

c. 關於 Role name (角色名稱)，請鍵入角色名稱，例如 `workdocs_app_role`，以協助您識別此角色的用途。角色名稱在您的 AWS 帳戶內必須是獨一無二的。輸入名稱之後，請按 Next Step (下一步)。

d. 在 Select Role Type (選取角色類型) 頁面，選取 Role for Cross-Account Access (跨帳戶存取的角色) 區段，然後選擇您想建立的角色類型：

- 若您同為使用者帳戶與資源帳戶的管理員，或兩個帳戶同屬一間公司，請選擇 Provide access between AWS accounts you own (在您擁有帳戶間提供存取權限)。當即將存取的使用者、角色與資源都屬於相同帳戶時，這也是您應選的選項。
- 如果您是擁有 Amazon WorkDocs 網站的AWS帳戶管理員，並且想要從應用程式開發人員帳戶授予使用者許可，請選取 [在帳戶和第AWS三方帳戶之間提供存取權限]。使用這個選項時，您需要指定一個外部 ID (這是第三方必須提供給您的) 來額外提供對於環境的控制，以供第三方在其中使用角色來存取您的資源。如需詳細資訊，請參閱如何在將 AWS 資源的存取權授予第三方時使用外部 ID。

e. 在下一頁面，請指定 AWS 帳戶 ID，以授予它對您資源存取的權限，並輸入 External ID (外部 ID)，以備第三方存取。

f. 按一下 Next Step (下一步) 以連接政策。

3. 在「連接政策」頁面上，搜尋先前建立的 Amazon WorkDocs API 權限政策，然後選取政策旁邊的核取方塊，然後按下一步。
4. 檢視該詳細資訊，複製角色 ARN 以供未來參考，並按下 Create Role (建立角色) 以完成角色建立。
5. 與開發人員共享角色 ARN。以下是角色 ARN 的範例：

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

授與使用者假設 IAM 角色的權限

具有管理AWS帳戶的開發人員可以允許使用者擔任 IAM 角色。若要這麼做，您可以建立新原則並將其附加到該使用者。

此政策必須包含對`sts:AssumeRole`動作有`Allow`影響的陳述式，以及`Resource`元素中角色的 Amazon 資源名稱 (ARN)，如下列範例所示。透過群組成員或直接連接取得政策的使用者，可以切換至指定的角色。

```
{
```

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

限制對特定 Amazon WorkDocs 執行個體的存取

如果您在一個AWS帳戶中有多個 Amazon 網 WorkDocs 站，並且想要授予特定網站的 API 存取權限，則可以定義一個Condition元素。Condition (條件) 元素可讓您於政策生效時指定條件。

下列範例會顯示一個條件元素：

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

在政策中有上述條件後，使用者只能使用 ID 存取 Amazon WorkDocs 執行個體d-123456789c5。Amazon WorkDocs 執行個體 ID 有時稱為組織 ID 或目錄 ID。如需詳細資訊，請參閱限制對特定 Amazon WorkDocs 執行個體的存取。

請依照下列步驟從AWS主控台取得 Amazon WorkDocs 組織 ID：

取得組織 ID

1. 在 AWS Directory Service 主控台導覽窗格中，選擇 Directories (目錄)。
2. 請注意與您的 Amazon WorkDocs 網站對應的目錄 ID 值。這是場地的組織 ID。

使用者應用程式的身份驗證與存取控制

亞馬遜 WorkDocs 使用者層級應用程式是透過 Amazon 註冊並受管 WorkDocs 主控台。開發人員應註冊他們的應用程序My Applications亞馬遜上的頁面 WorkDocs 主控台將提供每個應用程式唯一的 ID。在註冊期間，開發人員應指定重新導向 URI，以讓他們接收存取字符以及應用程式範圍。

目前，應用程序只能訪問亞馬遜 WorkDocs 相同內的網站AWS他們註冊的帳戶。

內容

授予撥打亞馬遜的許可 WorkDocs API

命令列介面使用者必須擁有 Amazon 的完整許可 WorkDocs 和AWS Directory Service。如果沒有權限，任何 API 調用都會返回UnauthorizedResourceAccessException訊息。以下政策允許完整許可。

```
{
  "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": "*"
      }
    ]
}
```

如果您想要授予唯讀許可，請使用此政策。

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

在政策中，第一個動作授予訪問所有亞馬遜 WorkDocs `Describe`操作。所以此`DescribeDirectories` 行動獲得有關您的信息AWS Directory Service目錄。Amazon EC2 操作啟用亞馬遜 WorkDocs 以取得您的 VPC 與子網路清單。

在 API 呼叫中使用資料夾 ID

每當 API 呼叫存取資料夾時，您都必須使用資料夾 ID，而不是資料夾名稱。例如，如果您通過`client.get_folder(FolderId='MyDocs')`時，API 呼叫會傳回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.
```

為了避免這種情況，請在文件夾的 URL 中使用 ID。

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

傳遞該 ID 會傳回正確的結果。

```
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'}}
```

建立應用程式

作為 Amazon WorkDocs 管理者，可以運用下列步驟來建立您的應用程式。

建立應用程式

1. 開啟 Amazon WorkDocs 位於的主控台https://console.aws.amazon.com/zocalo/。
2. 選擇 My Applications (我的應用程式)、Create an Application (建立應用程式)。
3. 輸入下列值：

   Application Name (應用程式名稱)

   應用程式名稱。

   電子郵件

   與應用程式建立關聯的電子郵件地址。

   Application Description (應用程式描述)

   應用程式的描述。

   Redirect URIs (重新導向 URI)

   您想要 Amazon 的位置 WorkDocs 以重新導向流量。

   Application Scopes (應用程式範圍)

   您希望您的應用程式擁有的讀取或寫入範圍。如需詳細資訊，請參閱 應用程式範圍。
4. 選擇 Create (建立)。

應用程式範圍

亞馬遜 WorkDocs 支援下列應用程式範圍：

- 內容已閱讀 (`workdocs.content.read`) 可讓您的應用程式存取下列 Amazon WorkDocs 應用程式介面
  - Get* (取得)
  - Describe* (描述)
- 內容寫入 (英文)`workdocs.content.write`) 可讓您的應用程式存取下列 Amazon WorkDocs 應用程式介面
  - 建立*
  - 更新*
  - 刪除*

- Initiate* (啟動)
- Abort* (中止)
- Add* (新增)
- Remove* (移除)

授權

應用程式註冊完成後，應用程式便可代表任何 Amazon 請求授權 WorkDocs 使用者。若要執行此操作，該應用程式應造訪 Amazon WorkDocs OAuth 端點，`https://auth.amazonworkdocs.com/oauth`，並提供下列查詢參數：

- [必要]`app_id`當應用程式註冊後，即產生應用程式 ID。
- [必要]`auth_type`請求的 OAuth 類型。支援的值為 `ImplicitGrant`。
- [必要]`redirect_uri`為應用程式註冊的重新導向 URI 以接收存取字符。
- [選用]`scopes`逗號分隔範圍清單。若未指定，將會使用該註冊期間選擇的範圍清單。
- [選用]`state`與存取字符一同傳回的字串。

> **Note**
>
> 如果您在透過命令列介面或 API 存取 AWS 時，需要 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的詳細資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

範例 GET 請求以啟動 OAuth 流程以取得存取字符：

```
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
```

下列項目會在 OAuth 授權流程期間發生：

1. 應用程式使用者會收到輸入 Amazon 的提示訊息 WorkDocs 網站名稱。
2. 使用者被重新導向至 Amazon WorkDocs 輸入其憑證的驗證頁面。

3. 在成功驗證身分後，同意畫面將會出現，允許該使用者授與或拒絕您應用程式存取 Amazon 的授權 WorkDocs。
4. 在使用者選擇同意畫面上的 Accept 後，他們的瀏覽器將重新導向至您應用程式的回呼 URL，以及做為查詢參數的存取字符與區域資訊。

的範例 GET 請求 WorkDocs：

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

除了訪問令牌，亞馬遜 WorkDocs OAuth 服務也會傳回`region`作為所選亞馬遜的查詢參數 WorkDocs 網站。外部應用程式應該使用`region`參數來確定亞馬遜 WorkDocs 服務端點。

如果您在透過命令列介面或 API 存取 AWS 時，需要 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的詳細資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

呼叫 Amazon WorkDocs API

在取得存取字符之後，您的應用程式可以進行 API 呼叫至 Amazon WorkDocs服務。

> **Important**

這個例子演示了如何使用 curl GET 請求來獲取文檔的元數據。

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

範例 JavaScript 描述使用者根資料夾的函數：

```
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 Java 為基礎的 API 呼叫範例描述如下：

```
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());
}
```

亞馬遜 WorkDocs 內容管理員

亞馬遜 WorkDocs 內容管理器是一種高級實用工具，可以上傳內容或從亞馬遜下載內容 WorkDocs 網站。

主題

建構亞馬遜 WorkDocs 內容管理員

您可以使用亞馬遜 WorkDocs 用於管理和使用者應用程式的內容管理員。

對於使用者應用程式，開發人員必須建構 Amazon WorkDocs 內容管理員與匿名AWS身份驗證字符。

對於管理應用程序，亞馬遜 WorkDocs 用戶端必須初始化AWS Identity and Access Management(IAM)登入資料。此外，在後續的 API 呼叫中一定會省略驗證權杖。

下面的代碼演示瞭如何初始化亞馬遜 WorkDocs 使用 Java 或 C# 的用戶應用程序的內容管理器。

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);
```

下載文件

開發者可以使用亞馬遜 WorkDocs 內容管理器從亞馬遜下載特定版本或最新版本的文檔 WorkDocs。下面範例示範如何使用 Java 與 C# 下載文件的特定版本。

> **Note**
>
> 若要下載文件的最新版本，請勿在建構 `GetDocumentStream` 請求時指定 `VersionId`。

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();
```

上傳文件

亞馬遜 WorkDocs 內容管理器提供用於將內容上傳到亞馬遜的 API WorkDocs 網站。以下範例示範如何使用 Java 與 C# 來上傳文件。

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();
```