



參考指南

# AWS SDKs和工具



# AWS SDKs和工具: 參考指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能附屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

# Table of Contents

AWS SDKs和工具參考指南 .....	1
開發人員資源 .....	2
工具組遙測通知 .....	3
Configuration .....	4
共用 config和 credentials 檔案 .....	4
個人檔案 .....	5
組態檔案的格式 .....	6
登入資料檔案格式 .....	9
共用檔案的位置 .....	9
主目錄解析 .....	10
變更這些檔案的預設位置 .....	10
環境變數 .....	11
如何設定環境變數 .....	12
無伺服器環境變數設定 .....	13
JVM 系統屬性 .....	13
如何設定 JVM 系統屬性 .....	13
身分驗證與存取 .....	16
選擇驗證應用程式碼的方法 .....	16
身分驗證方法 .....	19
AWS 建構家 ID .....	21
使用主控台登入資料登入 .....	21
運作方式 .....	21
IAM Identity Center 驗證 .....	22
先決條件 .....	22
使用 IAM Identity Center 設定程式設計存取 .....	22
重新整理入口網站存取工作階段 .....	25
了解 IAM Identity Center 身分驗證 .....	25
IAM Roles Anywhere .....	28
步驟 1：隨處設定 IAM 角色 .....	28
步驟 2：隨處使用 IAM 角色 .....	29
擔任角色 .....	30
擔任 IAM 角色 .....	30
擔任角色 (Web) .....	32
與 Web 身分或 OpenID Connect 聯合 .....	32

AWS 存取金鑰 .....	34
使用短期登入資料 .....	34
使用長期登入資料 .....	34
短期憑證 .....	35
長期登入資料 .....	36
EC2 執行個體的 IAM 角色 .....	39
建立 IAM 角色 .....	39
啟動 Amazon EC2 執行個體並指定您的 IAM 角色 .....	40
連線至 EC2 執行個體 .....	40
在 EC2 執行個體上執行您的應用程式 .....	40
信任的身分傳播 .....	41
使用 TIP 外掛程式的先決條件 .....	41
在程式碼中使用 TIP 外掛程式 .....	42
使用 TIP 的程式碼範例 .....	44
設定參考 .....	51
建立服務用戶端 .....	51
設定的優先順序 .....	51
了解本指南的設定頁面 .....	52
Config 檔案設定清單 .....	53
Credentials 檔案設定清單 .....	57
環境變數清單 .....	58
JVM 系統屬性清單 .....	63
標準化登入資料提供者 .....	66
了解登入資料提供者鏈結 .....	67
SDK 特定和工具特定登入資料提供者鏈結 .....	68
AWS 存取金鑰 .....	69
登入提供者 .....	72
擔任角色提供者 .....	74
容器提供者 .....	80
IAM Identity Center 供應商 .....	83
IMDS 提供者 .....	89
程序提供者 .....	94
標準化功能 .....	97
帳戶型端點 .....	98
應用程式 ID .....	101
Amazon EC2 執行個體中繼資料 .....	103

Amazon S3 存取點 .....	106
Amazon S3 多區域存取點 .....	108
S3 Express One Zone 工作階段身分驗證 .....	110
身分驗證機制 .....	113
AWS 區域 .....	115
AWS STS 區域端點 .....	118
資料完整性保護 .....	123
雙堆疊和 FIPS 端點 .....	127
端點探索 .....	130
一般組態 .....	132
主機字首注入 .....	135
IMDS 用戶端 .....	139
重試行為 .....	142
請求壓縮 .....	148
服務特定的端點 .....	150
智慧組態預設值 .....	202
常見執行時間 .....	207
CRT 相依性 .....	207
維護政策 .....	209
概觀 .....	209
版本控制 .....	209
SDK 主要版本生命週期 .....	209
相依性生命週期 .....	210
通訊方法 .....	210
版本生命週期 .....	212
文件歷史紀錄 .....	215
.....	ccxviii

# AWS SDKs和工具參考指南涵蓋的內容

許多 SDKs和工具透過共用的設計規格或透過共用程式庫來共用一些常見功能。

本指南包含下列相關資訊：

- [全域設定 AWS SDKs和工具](#) – 如何使用共用 config和credentials檔案或環境變數來設定您的 AWS SDKs和工具。
- [使用 AWS SDKs和工具進行身分驗證和存取](#) – 建立您的程式碼或工具在開發 AWS 時使用 進行身分驗證的方式 AWS 服務。
- [AWS SDKs和工具設定參考](#) – 可用於身分驗證和組態的所有標準化設定的參考。
- [AWS 通用執行期 \(CRT\) 程式庫](#) – 共用 AWS 通用執行期 (CRT) 程式庫概觀，可供幾乎所有 SDKs 使用。
- [AWS SDKs和工具維護政策](#) 涵蓋 AWS 軟體開發套件 (SDKs) 和工具的維護政策和版本控制，包括行動和物聯網 (IoT) SDKs及其基礎相依性。

此 AWS SDKs和工具參考指南旨在成為適用於多個 SDKs 和工具的資訊基礎。除了此處提供的任何資訊之外，還應該使用您正在使用的 SDK 或工具的特定指南。以下是本指南中具有相關材料區段的 SDK 和工具：

如果您使用的是：	本指南的相關章節如下：
<ul style="list-style-type: none"> <li>• 任何 SDK 或工具</li> </ul>	<a href="#">AWS SDKs和工具維護政策</a>
<ul style="list-style-type: none"> <li>• <a href="#">AWS Cloud9</a></li> <li>• <a href="#">AWS CDK</a></li> <li>• <a href="#">AWS Toolkit for Azure DevOps</a></li> <li>• <a href="#">AWS Toolkit for JetBrains</a></li> <li>• <a href="#">AWS Toolkit for Visual Studio</a></li> <li>• <a href="#">AWS Toolkit for Visual Studio Code</a></li> <li>• <a href="#">AWS Serverless Application Model</a></li> <li>• <a href="#">AWS CodeArtifact</a></li> <li>• <a href="#">AWS CodeBuild</a></li> </ul>	<a href="#">全域設定 AWS SDKs和工具</a> <a href="#">使用 AWS SDKs和工具進行身分驗證和存取</a> <a href="#">AWS SDKs和工具維護政策</a>

如果您使用的是：	本指南的相關章節如下：
<ul style="list-style-type: none"> <li>• <a href="#">Amazon CodeCatalyst</a></li> <li>• <a href="#">AWS CodeCommit</a></li> <li>• <a href="#">AWS CodeDeploy</a></li> <li>• <a href="#">AWS CodePipeline</a></li> </ul>	
<ul style="list-style-type: none"> <li>• <a href="#">AWS CLI</a></li> <li>• <a href="#">適用於 C++ 的 AWS SDK</a></li> <li>• <a href="#">適用於 Go 的 AWS SDK</a></li> <li>• <a href="#">適用於 Java 的 AWS SDK</a></li> <li>• <a href="#">適用於 JavaScript 的 AWS SDK</a></li> <li>• <a href="#">適用於 Kotlin 的 AWS SDK</a></li> <li>• <a href="#">適用於 .NET 的 AWS SDK</a></li> <li>• <a href="#">適用於 PHP 的 AWS SDK</a></li> <li>• <a href="#">適用於 Python (Boto3) 的 AWS SDK</a></li> <li>• <a href="#">適用於 Ruby 的 AWS SDK</a></li> <li>• <a href="#">適用於 Rust 的 AWS SDK</a></li> <li>• <a href="#">適用於 Swift 的 AWS SDK</a></li> <li>• <a href="#">AWS Tools for Windows PowerShell</a></li> </ul>	<ul style="list-style-type: none"> <li>• <a href="#">全域設定 AWS SDKs和工具</a></li> <li>• <a href="#">使用 AWS SDKs和工具進行身分驗證和存取</a></li> <li>• <a href="#">AWS SDKs和工具設定參考</a></li> <li>• <a href="#">AWS 通用執行期 (CRT) 程式庫</a></li> <li>• <a href="#">AWS SDKs和工具維護政策</a></li> <li>• <a href="#">AWS SDKs和工具版本生命週期</a></li> </ul>

- 如需可協助您開發應用程式之工具的概觀 AWS，請參閱[建置工具 AWS](#)。
- 如需支援的相關資訊，請參閱 [AWS 知識中心](#)。
- 如需 AWS 術語，請參閱 AWS 詞彙表 參考中的[AWS 詞彙表](#)。

## 開發人員資源

Amazon Q Developer 是採用生成式 AI 技術的對話式助理，可協助您了解、建置、擴展和操作 AWS 應用程式。為了加速您的建置 AWS，支援 Amazon Q 的模型會擴增高品質 AWS 內容，以產生更完整、可行且參考的答案。如需詳細資訊，請參閱《Amazon Q Developer 使用者指南》中的 [What is Amazon Q Developer?](#) 部分。

## 工具組遙測通知

AWS 整合式開發環境 (IDE) Toolkit 是外掛程式和擴充功能，可讓您存取 IDE 中的 AWS 服務。Amazon Q IDE 外掛程式和延伸可在 IDE 中啟用生成式 AI 協助。如需每個 IDE Toolkit 的詳細資訊，請參閱上表中的 Toolkit 使用者指南。若要進一步了解如何在 IDE 中使用 Amazon Q，請參閱《[Amazon Q 開發人員指南](#)》中的 [IDE 主題中的使用 Amazon Q](#)。

AWS IDE Toolkits 和 Amazon Q 可能會收集和存放用戶端遙測資料，以為未來 AWS Toolkit 和 Amazon Q 版本的決策提供資訊。收集的資料會量化您對 AWS Toolkit 和 Amazon Q 的使用情況。

若要進一步了解所有 IDE Toolkits 和 Amazon Q AWS 所收集的遙測資料，請參閱 `aws-toolkit-common` Github 儲存庫中的 [commonDefinitions.json](#) 文件。

如需每個 AWS IDE Toolkits 和 Amazon Q 延伸模組所收集的遙測資料的詳細資訊，請參閱下列 AWS Toolkit GitHub 儲存庫中的資源文件：

- [AWS Visual Studio Toolkit 搭配 Amazon Q](#)
- [AWS Toolkit for Visual Studio Code VS 程式碼的 和 Amazon Q 延伸模組](#)
- [AWS Toolkit for JetBrains 適用於 JetBrains 的 和 Amazon Q 外掛程式](#)
- [Amazon Q for Eclipse](#)

工具 AWS 組中可存取的某些 AWS 服務可能會收集額外的用戶端遙測資料。如需有關每個個別 AWS 服務所收集的資料類型的詳細資訊，請參閱您感興趣的特定服務[AWS 的文件](#)主題。



# 全域設定 AWS SDKs和工具

透過 AWS SDKs和其他 AWS 開發人員工具，例如 AWS Command Line Interface (AWS CLI)，您可以與 AWS 服務 APIs 互動。不過，在嘗試這麼做之前，您必須使用執行請求的操作所需的資訊來設定 SDK 或工具。

此資訊包含下列項目：

- 識別誰呼叫 API 的登入資料資訊。登入資料用於加密對 AWS 伺服器的請求。使用此資訊，會 AWS 確認您的身分，並可以擷取與其相關聯的許可政策。然後，它可以判斷您可以執行哪些動作。
- 您用來告知 AWS CLI 或 SDK 如何處理請求、傳送請求的位置（哪個 AWS 服務端點），以及如何解譯或顯示回應的其他組態詳細資訊。

每個 SDK 或工具都支援多個來源，您可以用來提供所需的登入資料和組態資訊。有些來源對於 SDK 或工具是唯一的，您必須參考該工具或 SDK 的文件，以取得如何使用該方法的詳細資訊。

不過，AWS SDKs 和工具支援來自程式碼本身以外主要來源的常見設定。本節涵蓋下列主題：

主題

- [使用共用 config 和 credentials 檔案來全域設定 AWS SDKs 和工具](#)
- [尋找和變更共用和 AWS SDKs config 和工具 credentials 檔案的位置](#)
- [使用環境變數全域設定 AWS SDKs 和工具](#)
- [使用 JVM 系統屬性全域設定 適用於 Java 的 AWS SDK 和 適用於 Kotlin 的 AWS SDK](#)

## 使用共用 **config** 和 **credentials** 檔案來全域設定 AWS SDKs 和工具

共用 AWS config 和 credentials 檔案是您可以為 AWS SDK 或工具指定身分驗證和組態的最常見方式。

共用 config 和 credentials 檔案包含一組設定檔。設定檔是一組組態設定，用於 AWS SDKs、AWS Command Line Interface (AWS CLI) 和其他工具的鍵值對。組態值會連接到設定檔，以便在使用該設定檔時設定 SDK/工具的某些層面。這些檔案是「共用」，因為這些值會影響使用者本機環境上的任何應用程式、程序或 SDKs。

共用config和credentials檔案都是純文字檔案，只包含 ASCII 字元 (UTF-8 編碼)。它們採用通常稱為 [INI 檔案](#) 的形式。

## 個人檔案

共用 config和 credentials 檔案內的設定與特定設定檔相關聯。您可以在 檔案中定義多個設定檔，以建立不同的設定組態，以套用至不同的開發環境。

[default] 若未指定特定具名設定檔，設定檔會包含 SDK 或工具操作所使用的值。您也可以建立個別の設定檔，以便依名稱明確參考。每個設定檔都可以根據您的應用程式和案例使用不同的設定和值。

### Note

[default] 只是未命名的設定檔。此描述檔已命名，default因為如果使用者未指定描述檔，則它是 SDK 使用的預設描述檔。它不會將繼承的預設值提供給其他設定檔。如果您在[default]設定檔中設定了一些項目，但未在具名設定檔中設定它，則當您使用具名設定檔時，不會設定該值。

## 設定具名設定檔

[default] 設定檔和多個具名設定檔可以存在於同一個檔案中。執程式碼時，請使用下列設定來選取軟體開發套件或工具所使用的設定檔設定。使用時，也可以在程式碼內或每個命令中選取設定檔 AWS CLI。

透過設定下列其中一項來設定此功能：

### **AWS\_PROFILE** - 環境變數

當此環境變數設定為具名設定檔或「預設」時，所有 SDK 程式碼和 AWS CLI 命令都會使用該設定檔中的設定。

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_PROFILE="my_default_profile_name";
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_PROFILE "my_default_profile_name"
```

## aws.profile - JVM 系統屬性

對於 JVM 上的適用於 Kotlin 的 SDK 和適用於 Java 的 SDK 2.x，您可以[設定aws.profile系統屬性](#)。開發套件建立服務用戶端時，除非在程式碼中覆寫設定，否則會使用具名設定檔中的設定。適用於 Java 的 SDK 1.x 不支援此系統屬性。

### Note

如果您的應用程式位於執行多個應用程式的伺服器上，我們建議您一律使用具名設定檔，而非預設設定檔。預設設定檔會自動由環境中的任何 AWS 應用程式擷取，並在這些應用程式之間共用。因此，如果其他人更新其應用程式的預設設定檔，可能會無意中影響其他人。為了防止這種情況，請在共用config檔案中定義具名描述檔，然後在程式碼中設定具名描述檔，以在應用程式中使用該具名描述檔。如果您知道具名描述檔的範圍僅影響您的應用程式，您可以使用環境變數或 JVM 系統屬性來設定具名描述檔。

## 組態檔案的格式

config 檔案會組織成數個區段。區段是具名的設定集合，且會持續到發生其他區段定義列為止。

config 檔案是使用下列格式的純文字檔案：

- 區段中的所有項目均採用 `setting-name=value` 的一般形式。
- 您可以在列的開頭使用井字號 (`#`)，為列加上註解。

### 區段類型

區段定義是將名稱套用至設定集合的行。區段定義列以方括號 (`[`) 開頭和結尾 `]`。在括號內，有一個區段類型識別符和區段的自訂名稱。您可以使用字母、數字、連字號 (`-`) 和底線 `_` (`()`)，但不能使用空格。

區段類型：**default**

範例區段定義列：`[default]`

`[default]` 是唯一不需要profile區段識別符的設定檔。

下列範例顯示具有 `[default]` 設定檔的基本config檔案。它會設定 [region](#) 設定。在遇到另一個區段定義之前，遵循此行的所有設定都是此設定檔的一部分。

```
[default]
#Full line comment, this text is ignored.
region = us-east-2
```

### 區段類型：**profile**

範例區段定義列：`[profile dev]`

`profile` 區段定義列是具名組態群組，您可以套用至不同的開發案例。若要進一步了解具名設定檔，請參閱設定檔上的上一節。

下列範例顯示具有`profile`區段定義列和名為 `config` 之設定檔的檔案`foo`。此行之後直到遇到另一個區段定義為止的所有設定，都是此具名設定檔的一部分。

```
[profile foo]
...settings...
```

有些設定有自己的巢狀子集群組，例如下列範例中`s3`的設定和子集。以一個或多個空格縮排子集，將子集與群組建立關聯。

```
[profile test]
region = us-west-2
s3 =
    max_concurrent_requests=10
    max_queue_size=1000
```

### 區段類型：**sso-session**

範例區段定義列：`[sso-session my-sso]`

`sso-session` 區段定義列會為您用來設定設定檔以使用解析 AWS 登入資料的一組設定命名 AWS IAM Identity Center。如需設定單一登入身分驗證的詳細資訊，請參閱 [使用 IAM Identity Center 驗證 AWS SDK 和工具](#)。設定檔由索引鍵/值對連結到`sso-session`區段，其中 `sso-session` 是索引鍵，而`sso-session`區段的名稱是值，例如 `sso-session = <name-of-sso-session-section>`。

下列範例會設定設定檔，該設定檔會使用來自 "my-sso" 的字符，取得 "111122223333" 帳戶中 "SampleRole" IAM 角色的短期 AWS 登入資料。在 `sso-session` 區段中，「my-sso」`profile`區段使用 `sso-session`金鑰依名稱參考。

```
[profile dev]
```

```
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
```

區段類型：**services**

範例區段定義列：`[services dev]`

### Note

`services` 本節支援服務特定的端點自訂，且僅適用於包含此功能SDKs 和工具。若要查看您的 SDK 是否可使用此功能，請參閱 [支援 AWS SDKs和工具](#) 以取得服務特定的端點。

`services` 區段定義列會為設定 AWS 服務 請求自訂端點的設定群組命名。設定檔由索引鍵/值對連結到`services`區段，其中 `services`是索引鍵，而`services`區段的名稱是值，例如 `services = <name-of-services-section>`。

`services` 區段會依`<SERVICE>` = 行進一步分隔為子區段，其中 `<SERVICE>`是 AWS 服務 識別符索引鍵。AWS 服務 識別符是以 API 模型的 為基礎`serviceId`，方法是以底線取代所有空格，並縮小所有字母大小寫。如需要在 `services` 區段中使用的所有服務識別碼金鑰的清單，請參閱 [服務特定端點的識別符](#)。服務識別碼金鑰後面接續巢狀化設定，每個設定獨佔一列並縮排兩個空格。

下列範例使用 `services`定義來設定端點，以用於僅對 服務提出的 Amazon DynamoDB 請求。在 "local-dynamodb" `services` 區段中，使用 `services`金鑰依名稱參考 `profile`區段。AWS 服務 識別符索引鍵為 `dynamodb`。Amazon DynamoDB 服務子區段從第 行開始`dynamodb =`。任何緊接著縮排的列都包含在該子區段中，並適用於該服務。

```
[profile dev]
services = local-dynamodb

[services local-dynamodb]
dynamodb =
  endpoint_url = http://localhost:8000
```

如需自訂端點組態的詳細資訊，請參閱 [服務特定的端點](#)。



作業系統	檔案的預設位置和名稱
Linux 和 macOS	~/.aws/config ~/.aws/credentials
Windows	%USERPROFILE%\aws\config %USERPROFILE%\aws\credentials

## 主目錄解析

~ 僅在以下情況下用於主目錄解析：

- 啟動路徑
- 緊接著 / 或平台特定分隔符號。在 Windows 上，~/ 和 ~\ 都會解析為主目錄。

判斷主目錄時，會檢查下列變數：

- ( 所有平台 ) HOME 環境變數
- (Windows 平台 ) USERPROFILE 環境變數
- (Windows 平台 ) HOMEDRIVE 和 HOMEPATH 環境變數的串連 (\$HOMEDRIVE\$HOMEPATH)
- ( 每個 SDK 或工具選用 ) SDK 或工具特定的主路徑解析函數或變數

如果使用者的主目錄是在路徑的開頭指定 ( 例如，~username/ )，則會解析為請求的使用者名稱的主目錄 ( 例如，/home/username/.aws/config )。

## 變更這些檔案的預設位置

您可以使用下列任一項來覆寫 SDK 或工具從中載入這些檔案的位置。

### 使用環境變數

您可以設定下列環境變數，將這些檔案的位置或名稱從預設值變更為自訂值：

- config 檔案環境變數：**AWS\_CONFIG\_FILE**
- credentials 檔案環境變數：**AWS\_SHARED\_CREDENTIALS\_FILE**

## Linux/macOS

您可以在 Linux 或 macOS 上執行下列 [匯出](#) 命令來指定替代位置。

```
$ export AWS_CONFIG_FILE=/some/file/path/on/the/system/config-file-name
$ export AWS_SHARED_CREDENTIALS_FILE=/some/other/file/path/on/the/system/
credentials-file-name
```

## Windows

您可以在 Windows 上執行下列 [setx](#) 命令來指定替代位置。

```
C:\> setx AWS_CONFIG_FILE c:\some\file\path\on\the\system\config-file-name
C:\> setx AWS_SHARED_CREDENTIALS_FILE c:\some\other\file\path\on\the\system
\credentials-file-name
```

如需使用環境變數設定系統的詳細資訊，請參閱 [使用環境變數全域設定 AWS SDKs和工具](#)。

## 使用 JVM 系統屬性

對於在 JVM 上執行的適用於 Kotlin 的 SDK 和適用於 Java 的 SDK 2.x，您可以設定下列 JVM 系統屬性，將這些檔案的位置或名稱從預設值變更為自訂值：

- config 檔案 JVM 系統屬性：**aws.configFile**
- credentials 檔案環境變數：**aws.sharedCredentialsFile**

如需如何設定 JVM 系統屬性的說明，請參閱 [the section called “如何設定 JVM 系統屬性”](#)。適用於 Java 的 SDK 1.x 不支援這些系統屬性。

## 使用環境變數全域設定 AWS SDKs和工具

環境變數提供另一種方式，可在使用 AWS SDKs和工具時指定組態選項和登入資料。環境變數對於編寫指令碼或暫時將具名設定檔設定為預設值非常有用。如需大多數 SDKs 支援的環境變數清單，請參閱 [環境變數清單](#)。

### 選項的優先順序

- 如果您使用環境變數指定設定，它會覆寫從共用 AWS config和credentials檔案中的設定檔載入的任何值。



- 如果您在 AWS CLI 命令列上使用 參數指定設定，它會覆寫來自對應環境變數或組態檔案中設定檔的任何值。

## 如何設定環境變數

下列範例說明如何為預設使用者設定環境變數。

Linux, macOS, or Unix

```
$ export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
$ export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
$ export
  AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40Lgk
$ export AWS_REGION=us-west-2
```

設定環境變數會變更使用的數值，直到 Shell 工作階段結束或直到您將該變數設為其他數值。您可以在 Shell 的啟動指令碼中設定變數，讓它們跨未來的工作階段持續生效。

Windows Command Prompt

```
C:\> setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
C:\> setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
C:\> setx
  AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40Lgk
C:\> setx AWS_REGION us-west-2
```

使用 [set](#) 設定環境變數會變更使用的值，直到目前的命令提示工作階段結束，或直到您將變數設定為不同的值為止。使用 [setx](#) 設定環境變數會變更目前命令提示工作階段和您在執行命令後建立的所有命令提示工作階段中使用的值。不會影響您執行命令當時已執行的其他命令 Shell。

PowerShell

```
PS C:\> $Env:AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
PS C:\> $Env:AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY"
PS C:\>
  \> $Env:AWS_SESSION_TOKEN="AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40Lgk"
PS C:\> $Env:AWS_REGION="us-west-2"
```

如果您在 PowerShell 提示字元中設定環境變數 (如前一個範例所示)，它只會在目前的工作階段儲存該數值。若要讓環境變數設定在所有 PowerShell 和命令提示字元工作階段中持續存在，請使用

Control Panel (控制面板) 中的 System (系統) 應用程式。或者，您也可以將變數新增到 PowerShell 設定檔，為所有未來 PowerShell 工作階段設定變數。如需有關存放環境變數或跨工作階段持續存放的詳細資訊，請參閱 [PowerShell 文件](#)。

## 無伺服器環境變數設定

如果您使用無伺服器架構進行開發，您還有其他設定環境變數的選項。根據您的容器，您可以針對在這些容器中執行的程式碼使用不同的策略來查看和存取環境變數，類似於非雲端環境。

例如，使用 AWS Lambda，您可以直接設定環境變數。如需詳細資訊，請參閱《AWS Lambda 開發人員指南》中的 [使用 AWS Lambda 環境變數](#)。

在無伺服器架構中，您通常可以在環境設定下的提供者金鑰下，在 `serverless.yml` 檔案中設定 SDK 環境變數。如需 `serverless.yml` 檔案的資訊，請參閱無伺服器架構文件中 [的一般函數設定](#)。

無論您使用哪種機制來設定容器環境變數，容器都會保留一些機制，例如在 [定義的執行期環境變數](#) 中記錄 Lambda 的機制。請務必參閱您所使用的容器官方文件，以判斷如何處理環境變數，以及是否有任何限制。

## 使用 JVM 系統屬性全域設定 適用於 Java 的 AWS SDK 和 適用於 Kotlin 的 AWS SDK

[JVM 系統屬性](#) 提供另一種方式，為在 JVM 上執行 SDKs 指定組態選項和登入資料，例如 適用於 Java 的 AWS SDK 和 適用於 Kotlin 的 AWS SDK。如需 SDKs 支援的 JVM 系統屬性清單，請參閱 [設定參考](#)。

### 選項的優先順序

- 如果您使用其 JVM 系統屬性指定設定，它會覆寫環境變數中找到的任何值，或從共用 AWS `config` 和 `credentials` 檔案中的設定檔載入的任何值。
- 如果您使用環境變數指定設定，它會覆寫從共用 AWS `config` 和 `credentials` 檔案中的設定檔載入的任何值。

## 如何設定 JVM 系統屬性

您可以透過多種方式設定 JVM 系統屬性。

## 在命令列上

使用 `-D` 切換叫用 `java` 命令時，在命令列上設定 JVM 系統屬性。除非您明確覆寫程式碼中的值，否則下列命令會為所有服務用戶端 AWS 區域 全域設定。

```
java -Daws.region=us-east-1 -jar <your_application.jar> <other_arguments>
```

如果您需要設定多個 JVM 系統屬性，請指定多次 `-D` 切換。

## 使用環境變數

如果您無法存取命令列來叫用 JVM 來執行應用程式，您可以使用 `JAVA_TOOL_OPTIONS` 環境變數來設定命令列選項。此方法適用於在 Java 執行時間執行 AWS Lambda 函數或在內嵌 JVM 中執行程式碼等情況。

下列範例會為所有服務用戶端 AWS 區域 全域設定，除非您明確覆寫程式碼中的值。

Linux, macOS, or Unix

```
$ export JAVA_TOOL_OPTIONS="-Daws.region=us-east-1"
```

設定環境變數會變更使用的數值，直到 Shell 工作階段結束或直到您將該變數設為其他數值。您可以在 Shell 的啟動指令碼中設定變數，讓它們跨未來的工作階段持續生效。

Windows Command Prompt

```
C:\> setx JAVA_TOOL_OPTIONS -Daws.region=us-east-1
```

使用 `set` 設定環境變數會變更使用的值，直到目前的命令提示工作階段結束，或直到您將變數設定為不同的值為止。使用 `setx` 設定環境變數會變更目前命令提示工作階段和您在執行命令後建立的所有命令提示工作階段中使用的值。不會影響您執行命令當時已執行的其他命令 Shell。

## 在執行時間

您也可以使用 `System.setProperty` 方法在程式碼中的執行時間設定 JVM 系統屬性，如下列範例所示。

```
System.setProperty("aws.region", "us-east-1");
```

**⚠ Important**

在初始化 SDK 服務用戶端之前設定任何 JVM 系統屬性，否則服務用戶端可能會使用其他值。

# 使用 AWS SDKs和工具進行身分驗證和存取

當您開發 AWS SDK 應用程式或使用 AWS 工具來使用時 AWS 服務，您必須建立程式碼或工具的驗證方式 AWS。您可以根據程式碼執行的環境和您可用的存取權，以不同的方式設定 AWS 資源的程式設計 AWS 存取。

以下選項是[登入資料提供者鏈結](#)的一部分。這表示透過相應地設定您的共用 AWS config和credentials檔案，您的 AWS SDK 或工具將自動探索和使用該身分驗證方法。

## 選擇驗證應用程式碼的方法

選擇一種方法來驗證應用程式 AWS 對發出的呼叫。

您是否在 內部執行程式碼 AWS 服務（例如 Amazon EC2、Lambda、Amazon ECS、Amazon EKS、CodeBuild）？

如果您的程式碼在 上執行 AWS，登入資料可以自動提供給您的應用程式。例如，如果您的應用程式託管在 Amazon Elastic Compute Cloud 上，而且有與該資源相關聯的 IAM 角色，則登入資料會自動提供給您的應用程式。同樣地，如果您使用 Amazon ECS 或 Amazon EKS 容器，IAM 角色的登入資料集可以透過 SDK 的[登入資料提供者鏈](#)，由容器內執行的程式碼自動取得。

您的程式碼是否在 Amazon Elastic Compute Cloud 執行個體中？

[使用 IAM 角色來驗證部署到 Amazon EC2 的應用程式](#) – 使用 IAM 角色在 Amazon EC2 執行個體上安全地執行應用程式。

您的程式碼是否在 AWS Lambda 函數中？

當您建立 Lambda [函數時](#)，[Lambda 會建立](#)具有最少許可的執行角色。軟體 AWS 開發套件或工具接著會透過 Lambda 執行環境，自動使用在執行時間連接至 Lambda 的 IAM 角色。

您的程式碼是否在 Amazon Elastic Container Service 中（在 Amazon EC2 或 AWS Fargate Amazon ECS 上）？

使用任務的 IAM 角色。您必須[建立任務角色](#)，並在 [Amazon ECS 任務定義](#)中指定該角色。軟體 AWS 開發套件或工具接著會透過 Amazon ECS 中繼資料，自動使用在執行時間指派給任務的 IAM 角色。

您的程式碼是否在 Amazon Elastic Kubernetes Service 中？

我們建議您使用 [Amazon EKS Pod 身分](#)。

注意：如果您認為[服務帳戶 \(IRSA\) 的 IAM 角色](#)可能更符合您的獨特需求，請參閱《[Amazon EKS 使用者指南](#)》中的[比較 EKS Pod 身分和 IRSA](#)。

您的程式碼是否在 中執行 AWS CodeBuild

請參閱[使用 CodeBuild 的身分型政策](#)。

您的程式碼是否在另一個程式碼中 AWS 服務？

請參閱的專用指南 AWS 服務。當您在 上執行程式碼時 AWS，開發套件[登入資料提供者鏈](#)可以自動為您取得和重新整理登入資料。

您要建立行動應用程式或用戶端型 Web 應用程式嗎？

如果您要建立需要存取的行動應用程式或用戶端型 Web 應用程式 AWS，請建置您的應用程式，以便使用 Web 聯合身分動態請求臨時 AWS 安全登入資料。

有了 Web 聯合身分，您就不需要建立自訂登入代碼，或管理您自己的使用者身分。反之，應用程式使用者可以使用知名的外部身分提供者 (IdP) 登入，例如 Login with Amazon、Facebook、Google 或任何其他與 OpenID Connect (OIDC) 相容的 IdP。他們可以接收身分驗證字符，然後交換該字符以取得中的臨時安全登入資料 AWS，該登入資料對應至具有許可的 IAM 角色，以使用您 中的資源 AWS 帳戶。

若要了解如何為您的 SDK 或工具設定此項目，請參閱[使用 Web 身分或 OpenID Connect 擔任角色以驗證 AWS SDKs和工具](#)。

對於行動應用程式，請考慮使用 Amazon Cognito。Amazon Cognito 充當身分經紀人，並為您執行大部分聯合工作。如需詳細資訊，請參閱《IAM 使用者指南》中的[將 Amazon Cognito 用於行動應用程式](#)。

您是否正在開發和執程式碼 LOCALLY？

我們建議使用[使用主控台登入資料來驗證 AWS SDKs和工具](#)。

在快速瀏覽器型身分驗證流程之後，AWS 會自動產生臨時登入資料，以跨 CLI AWS Tools for PowerShell 和 AWS SDKs AWS 等本機開發工具運作。

如果您使用 Identity Center 進行 AWS 帳戶存取

如果您已經擁有 AWS 帳戶的存取權和/或需要管理人力資源的存取權，請使用 IAM Identity Center 來驗證 AWS SDK 和工具。作為安全最佳實務，我們建議您使用 AWS Organizations 搭配 IAM Identity Center 來管理所有 AWS 帳戶的存取權。您可以在 IAM Identity Center 中建立使用者、使用 Microsoft

Active Directory、使用 SAML 2.0 身分提供者 (IdP)，或將 IdP 個別聯合到 AWS 帳戶。若要檢查您的區域是否支援 IAM Identity Center，請參閱《Amazon Web Services 一般參考》中的 [使用 IAM Identity Center 驗證 AWS SDK 和工具](#) IAM Identity Center 端點和配額。

如果您正在尋找其他方法來驗證

在您的目標角色中建立具有許可的最低權限 IAM `sts:AssumeRole` 使用者。然後，設定您的設定檔以使用該使用者的 `source_profile` 設定來擔任角色。

您也可以透過環境變數或共用登入資料檔案使用臨時 IAM AWS 登入資料。請參閱使用短期憑證來驗證 AWS SDKs和工具。

注意：僅在沙盒或學習環境中，您可以考慮使用長期登入資料來驗證 AWS SDKs和工具。

此程式碼是在內部部署或在混合式/隨需 VM 中執行（例如從 Amazon S3 讀取或寫入 Amazon S3 的伺服器，或是部署到雲端的 Jenkins）？

您是否使用 X.509 用戶端憑證？

是：請參閱 [使用 IAM Roles Anywhere 驗證 AWS SDKs和工具](#)。您可以使用 IAM Roles Anywhere 在 IAM 中取得臨時安全登入資料，例如在外部執行的伺服器、容器和應用程式 AWS。若要使用 IAM Roles Anywhere，您的工作負載必須使用 X.509 憑證。

環境是否可以安全地連線至聯合身分提供者（例如 Microsoft Entra 或 Okta）來請求臨時 AWS 憑證？

是：使用 [程序登入資料提供者](#)

使用 [程序登入資料提供者](#) 在執行時間自動擷取登入資料。這些系統可能會使用協助程式工具或外掛程式來取得登入資料，並且可能會使用在幕後擔任 IAM 角色 `sts:AssumeRole`。

否：使用透過注入的臨時登入資料 AWS Secrets Manager

使用透過注入的臨時登入資料 AWS Secrets Manager。如需取得短期存取金鑰的選項，請參閱《IAM 使用者指南》中的 [請求臨時安全登入資料](#)。如需存放這些臨時登入資料的選項，請參閱 [AWS 存取金鑰](#)。

您可以使用這些登入資料，從 [Secrets Manager](#) 安全地擷取更廣泛的應用程式許可，您可以在其中存放生產秘密或長期的角色型登入資料。

您是否使用不在 中的第三方工具 AWS？

使用第三方供應商撰寫的文件，以取得取得登入資料的最佳指引。

如果您的第三方未提供文件，您可以安全地插入臨時登入資料嗎？

是：使用環境變數和暫時 AWS STS 登入資料。

否：使用存放在加密秘密管理器（上次排序）中的靜態存取金鑰。

## 身分驗證方法

在 AWS 環境中執行之程式碼的身分驗證方法

如果您的程式碼在 上執行 AWS，登入資料可以自動提供給您的應用程式。例如，如果您的應用程式託管在 Amazon Elastic Compute Cloud 上，而且有與該資源相關聯的 IAM 角色，則登入資料會自動提供給您的應用程式。同樣地，如果您使用 Amazon ECS 或 Amazon EKS 容器，IAM 角色的登入資料集可以透過 SDK 的登入資料提供者鏈，由容器內執行的程式碼自動取得。

- [使用 IAM 角色來驗證部署到 Amazon EC2 的應用程式](#) – 使用 IAM 角色在 Amazon EC2 執行個體上安全地執行應用程式。
- 您可以使用 IAM Identity Center 以程式設計方式與 AWS 互動，方法如下：
  - 使用 從主控台[AWS CloudShell](#)執行 AWS CLI 命令。
  - 若要為軟體開發團隊嘗試雲端協作空間，請考慮使用 [Amazon CodeCatalyst](#)。

透過 Web 型身分提供者進行身分驗證 - 行動或用戶端型 Web 應用程式

如果您要建立需要存取的行動應用程式或用戶端型 Web 應用程式 AWS，請建置您的應用程式，以便使用 Web 聯合身分動態請求臨時 AWS 安全登入資料。

有了 Web 聯合身分，您就不需要建立自訂登入代碼，或管理您自己的使用者身分。反之，應用程式使用者可以使用知名的外部身分提供者 (IdP) 登入，例如 Login with Amazon、Facebook、Google 或任何其他與 OpenID Connect (OIDC) 相容的 IdP。他們可以接收身分驗證字符，然後交換該字符以取得中的臨時安全登入資料 AWS，該登入資料對應至具有許可的 IAM 角色，以使用您 中的資源 AWS 帳戶。

若要了解如何為您的 SDK 或工具設定此項目，請參閱 [使用 Web 身分或 OpenID Connect 擔任角色以驗證 AWS SDKs和工具](#)。

對於行動應用程式，請考慮使用 Amazon Cognito。Amazon Cognito 充當身分經紀人，並為您執行大部分聯合工作。如需詳細資訊，請參閱《IAM 使用者指南》中的[將 Amazon Cognito 用於行動應用程式](#)。



在本機執行之程式碼的身分驗證方法 (不在中 AWS)

- [使用主控台登入資料來驗證 AWS SDKs和工具](#) – 此功能可與 AWS Command Line Interface 和 Tools for PowerShell 搭配使用，並提供可重新整理的登入資料，可用於 CLI、Tools for PowerShell AWS 和 等本機開發工具 AWS。
- [使用 IAM Identity Center 驗證 AWS SDK 和工具](#) – 作為安全最佳實務，我們建議您使用 AWS Organizations 搭配 IAM Identity Center 來管理所有的存取 AWS 帳戶。您可以在 中建立使用者 AWS IAM Identity Center、使用 Microsoft Active Directory、使用 SAML 2.0 身分提供者 (IdP)，或個別聯合您的 IdP AWS 帳戶。若要檢查您的區域是否支援 IAM Identity Center，請參閱 中的 [AWS IAM Identity Center 端點和配額](#) Amazon Web Services 一般參考。
- [使用 IAM Roles Anywhere 驗證 AWS SDKs和工具](#) – 您可以使用 IAM Roles Anywhere 在 IAM 中取得臨時安全登入資料，例如在 外部執行的伺服器、容器和應用程式 AWS。若要使用 IAM Roles Anywhere，您的工作負載必須使用 X.509 憑證。
- [使用 AWS 登入資料來擔任角色以驗證 AWS SDKs和工具](#) – 您可以擔任 IAM 角色來暫時存取您可能無法存取 AWS 的資源。
- [使用 AWS 存取金鑰來驗證 AWS SDKs和工具](#) – 可能較不方便或可能會增加 AWS 資源安全風險的其他選項。

有關存取管理的詳細資訊

IAM 使用者指南提供下列有關安全控制 AWS 資源存取的資訊：

- [IAM 身分 \(使用者、使用者群組和角色\)](#) – 了解其中身分的基本概念 AWS。
- [IAM 中的安全最佳實務](#) – 根據 [共同責任模型](#) 開發 AWS 應用程式時應遵循的安全建議。

Amazon Web Services 一般參考 具有下列基本概念：

- [了解並取得您的 AWS 登入資料](#) – 主控台和程式設計存取的存取金鑰選項和管理實務。

要存取的 IAM Identity Center 受信任身分傳播 (TIP) 外掛程式 AWS 服務

- [使用 TIP 外掛程式存取 AWS 服務](#) – 如果您要為 Amazon Q Business 或其他支援受信任身分傳播的服務建立應用程式，並使用 適用於 Java 的 AWS SDK 或 適用於 JavaScript 的 AWS SDK，您可以使用 TIP 外掛程式來簡化授權體驗。

# AWS 建構家 ID

您的 AWS 建構家 ID 補充 AWS 帳戶 您可能已經擁有或想要建立的任何。雖然 AWS 帳戶 充當您建立 AWS 之資源的容器，並為這些資源提供安全界限，但 AWS 建構家 ID 代表您做為個人。您可以使用 登入 AWS 建構家 ID，以存取開發人員工具和服務，例如 Amazon Q 和 Amazon CodeCatalyst。

- AWS 登入 《使用者指南》中的[使用 登入 AWS 建構家 ID](#) – 了解如何建立和使用 AWS 建構家 ID，並了解建置器 ID 提供的內容。
- [CodeCatalyst 概念 - AWS 建構家 ID](#) 在 Amazon CodeCatalyst 使用者指南中 - 了解 CodeCatalyst 如何使用 AWS 建構家 ID。

## 使用主控台登入資料來驗證 AWS SDKs和工具

在本機環境或其他非AWS 運算服務環境中開發 AWS 應用程式時，建議使用主控台登入資料來提供 AWS 登入資料。如果您是在 AWS 資源上進行開發，例如 Amazon Elastic Compute Cloud (Amazon EC2) 或 AWS CloudShell，我們建議您改為從該服務取得登入資料。

您也可以透過 IAM Identity Center 進行身分驗證[使用 IAM Identity Center 驗證 AWS SDK 和工具](#)。此選項是組織管理人力資源存取權的常見方式，需要啟用 Identity Center。

### 其運作方式？

[使用主控台登入資料進行 AWS 本機開發](#)可讓您使用現有的 AWS 管理主控台登入資料，以程式設計方式存取 AWS 服務。在瀏覽器型身分驗證流程之後，AWS 會產生臨時登入資料，可用於 CLI、Tools for PowerShell AWS 和 AWS SDKs等本機開發工具。此功能可簡化設定和管理 AWS CLI 登入資料的程序，尤其是如果您偏好互動式身分驗證，而不是管理長期存取金鑰。

透過此程序，您可以使用在初始帳戶設定期間建立的根登入資料、IAM 使用者或身分提供者的聯合身分進行身分驗證。

如果您使用SDKs進行開發，軟體開發套件用戶端將透過 [使用臨時登入資料AWS SDKs和工具標準化登入資料提供者](#)。您也可以設定 [登入憑證提供者](#)。

CLI AWS 和 Tools for PowerShell 都支援透過登入命令進行驗證：

- [使用主控台登入資料登入 AWS 進行本機開發](#)
- AWS Tools for PowerShell 使用者指南中的[使用主控台登入資料登入](#)

# 使用 IAM Identity Center 驗證 AWS SDK 和工具

AWS IAM Identity Center 可用於在非AWS 運算服務環境中開發 AWS 應用程式時提供 AWS 登入資料。如果您是在 AWS 資源上進行開發，例如 Amazon Elastic Compute Cloud (Amazon EC2) 或 AWS Cloud9，我們建議您改為從該服務取得登入資料。

如果您已經使用 Identity Center 進行 AWS 帳戶存取，或需要管理組織的存取，請使用 IAM Identity Center 身分驗證。

在本教學課程中，您將建立 IAM Identity Center 存取權，並使用 AWS 存取入口網站和 `awscli`，為您的 SDK 或工具進行設定 AWS CLI。

- AWS 存取入口網站是您手動登入 IAM Identity Center 的 Web 位置。URL 的格式為 `d-xxxxxxxxxx.awsapps.com/start` 或 `your_subdomain.awsapps.com/start`。登入 AWS 存取入口網站時，您可以檢視已為該使用者設定的 AWS 帳戶 和 角色。此程序使用 AWS 存取入口網站來取得 SDK/工具身分驗證程序所需的組態值。
- AWS CLI 用於設定 SDK 或工具，以使用 IAM Identity Center 身分驗證進程式碼發出的 API 呼叫。此一次性程序會更新您的共用 AWS config 檔案，然後在您執程式碼時由軟體開發套件或工具使用。

## 先決條件

開始此程序之前，您應該已完成下列操作：

- 如果您沒有 AWS 帳戶，[請註冊 AWS 帳戶](#)。
- 如果您尚未啟用 IAM Identity Center，請依照 AWS IAM Identity Center 使用者指南中的指示[啟用 IAM Identity Center](#)。

## 使用 IAM Identity Center 設定程式設計存取

### 步驟 1：建立存取權並選取適當的許可集

選擇下列其中一種方法來存取您的 AWS 登入資料。

我沒有透過 IAM Identity Center 建立存取權

1. 遵循 AWS IAM Identity Center 《使用者指南》中的[使用預設 IAM Identity Center 目錄程序設定使用者存取權](#)，以新增使用者和新增管理許可。

2. AdministratorAccess 許可集不應用於定期開發。我們建議您改為使用預先定義的PowerUserAccess許可集，除非您的僱主已為此目的建立自訂許可集。

再次遵循相同的[設定使用者存取與預設的 IAM Identity Center 目錄](#)程序，但這次：

- 與其建立Admin team群組，請建立Dev team群組，並在之後於指示中取代此群組。
- 您可以使用現有的使用者，但必須將使用者新增至新Dev team群組。
- 與其建立AdministratorAccess許可集，請建立PowerUserAccess許可集，並在之後於指示中取代。

完成後，您應該具備下列項目：

- Dev team 群組。
  - 連接至 Dev team群組的PowerUserAccess許可集。
  - 您的使用者已新增至 Dev team群組。
3. 退出口網站並再次登入，以查看 Administrator或的 AWS 帳戶 和 選項PowerUserAccess。使用工具/SDK PowerUserAccess時選取。

我已經 AWS 可以透過我的僱主管理的聯合身分提供者（例如 Microsoft Entra 或 Okta）存取

AWS 透過身分提供者的入口網站登入。如果您的雲端管理員已授予您 PowerUserAccess（開發人員）許可，您會看到您有權存取 AWS 帳戶的 和許可集。您會在許可集名稱旁，看到使用該許可集手動或以程式設計方式存取帳戶的選項。

若您自訂實作，可能會產生不同體驗，例如不同的許可集名稱。若您不確定要使用哪個許可集，請聯絡您的 IT 團隊尋求協助。

我已經 AWS 可以透過我的僱主管理的 AWS 存取入口網站存取

透過 AWS AWS 存取入口網站登入。若雲端管理員已授予您 PowerUserAccess（開發人員）權限，您會看到您有權存取的 AWS 帳戶 和許可集。您會在許可集名稱旁，看到使用該許可集手動或以程式設計方式存取帳戶的選項。

我已經 AWS 可以透過我的僱主管理的聯合身分自訂身分提供者存取

請聯絡您的 IT 團隊尋求協助。

## 步驟 2：設定 SDKs和工具以使用 IAM Identity Center

1. 在開發機器上安裝最新的 AWS CLI。
  - a. 請參閱AWS Command Line Interface 《使用者指南》中的[安裝或更新最新版本的 AWS CLI](#)。
  - b. (選用) 若要驗證 AWS CLI 是否正常運作，請開啟命令提示字元並執行 `aws --version` 命令。
2. 登入 AWS 存取入口網站。您的僱主可能會提供此 URL，您也可以依照步驟 1：建立存取權，在電子郵件中取得此 URL。如果沒有，請在 <https://console.aws.amazon.com/singlesignon/>：// 儀表板上尋找您的AWS 存取入口網站 URL。
  - a. 在 AWS 存取入口網站的帳戶索引標籤中，選取要管理的個別帳戶。使用者的角色隨即顯示。選擇存取金鑰以取得命令列的登入資料，或為適當的許可集取得程式設計存取。使用預先定義的PowerUserAccess許可集，或您或您的僱主建立的任何許可集，以套用開發的最低權限許可。
  - b. 在取得憑證對話方塊中，選擇MacOS 和 Linux 或 Windows，具體取決於您的作業系統。
  - c. 選擇 IAM Identity Center 登入資料方法，以取得下一個步驟所需的 Issuer URL 和 SSO Region 值。注意：SSO Start URL 可與 互換使用 Issuer URL。
3. 在 AWS CLI 命令提示字元中，執行 `aws configure sso` 命令。出現提示時，輸入您在上一個步驟中收集的組態值。如需此 AWS CLI 命令的詳細資訊，請參閱[使用aws configure sso精靈設定設定檔](#)。
  - a. 針對提示 SSO Start URL，輸入您為 取得的值 Issuer URL。
  - b. 對於 CLI 設定檔名稱，我們建議您在開始使用時輸入 `###`。如需如何設定非預設（已命名）設定檔及其相關聯環境變數的資訊，請參閱 [個人檔案](#)。
4. (選用) 在 AWS CLI 命令提示字元中，執行 `aws sts get-caller-identity` 命令以確認作用中的工作階段身分。回應應會顯示您設定的 IAM Identity Center 許可集。
5. 如果您使用的是 AWS 開發套件，請在開發環境中為您的開發套件建立應用程式。
  - a. 對於某些 SDKs，SSO IDC 您必須將 SSO 和 等其他套件新增至您的應用程式，才能使用 IAM Identity Center 身分驗證。如需詳細資訊，請參閱您的特定 SDK。
  - b. 如果您先前已設定的存取權 AWS，請檢閱共用 AWS credentials 檔案是否有任何 [AWS 存取金鑰](#)。由於[了解登入資料提供者鏈結](#)優先順序，您必須移除任何靜態登入資料，開發套件或工具才會使用 IAM Identity Center 登入資料。

如需深入了解SDKs和工具如何使用此組態來使用和重新整理登入資料，請參閱 [如何解決 AWS SDKs IAM Identity Center 身分驗證](#)。

若要直接在共用config檔案中設定 IAM Identity Center 提供者設定，請參閱本指南 [IAM Identity Center 憑證提供者](#) 中的。

## 重新整理入口網站存取工作階段

您的存取最終將過期，開發套件或工具將遇到身分驗證錯誤。當此過期發生時，取決於您設定的工作階段長度。若要在需要時再次重新整理存取入口網站工作階段，請使用 AWS CLI 執行 `aws sso login` 命令。

您可以同時延長 IAM Identity Center 存取入口網站工作階段持續時間和許可集工作階段持續時間。這可延長您可以執行程式碼的時間，之後才需要再次使用 手動登入 AWS CLI。如需詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的以下主題：

- IAM Identity Center 工作階段持續時間 – [設定使用者 AWS 存取入口網站工作階段的持續時間](#)
- 許可集工作階段持續時間 – [設定工作階段持續時間](#)

## 如何解決 AWS SDKs IAM Identity Center 身分驗證

### 相關的 IAM Identity Center 術語

下列術語可協助您了解背後的程序和組態 AWS IAM Identity Center。適用於 AWS SDK APIs 的文件針對其中一些身分驗證概念使用與 IAM Identity Center 不同的名稱。知道這兩個名稱會很有幫助。

下表顯示替代名稱如何互相關聯。

IAM Identity Center 名稱	SDK API 名稱	Description
Identity Center	sso	雖然已重新命名 AWS 單一登入，但 sso API 命名空間會保留其原始名稱，以用於回溯相容性。如需詳細資訊，請參閱《AWS IAM Identity Center 使用者指南》中的 <a href="#">IAM Identity Center 重新命名</a> 。

IAM Identity Center 名稱	SDK API 名稱	Description
IAM Identity Center 主控台 管理主控台		您用來設定單一登入的主控台。
AWS 存取入口網站 URL		IAM Identity Center 帳戶的唯一 URL，例如 <code>https://xxx.awsapps.com/start</code> 。您可以使用 IAM Identity Center 登入憑證登入此入口網站。
IAM Identity Center Access Portal 工作階段 許可集工作階段	身分驗證工作階段	為發起人提供承載存取字符。  軟體開發套件在內部用來進行 AWS 服務 呼叫的 IAM 工作階段。在非正式討論中，您可能會看到此錯誤稱為「角色工作階段」。
許可集登入資料	AWS 登入資料 sigv4 登入資料	軟體開發套件實際用於大多數 AWS 服務 呼叫的登入資料 (特別是所有 sigv4 AWS 服務 calls)。在非正式討論中，您可能會看到這不正確地稱為「角色登入資料」。
IAM Identity Center 憑證提供者	SSO 登入資料提供者	如何取得登入資料，例如提供功能的類別或模組。

## 了解的 SDK 登入資料解析 AWS 服務

IAM Identity Center API 會交換 sigv4 登入資料的承載字符登入資料。大多數 AWS 服務 是 sigv4 APIs，但有一些例外狀況，例如 Amazon CodeWhisperer 和 Amazon CodeCatalyst。以下說明登入資料解析程序，支援透過 對應用程式碼進行大多數 AWS 服務 呼叫 AWS IAM Identity Center。

## 啟動 AWS 存取入口網站工作階段

- 使用您的登入資料登入工作階段，以啟動程序。
  - 在 AWS Command Line Interface () 中使用 `aws sso login` 命令 AWS CLI。如果您還沒有作用中的工作階段，這會啟動新的 IAM Identity Center 工作階段。
- 當您啟動新的工作階段時，您會收到來自 IAM Identity Center 的重新整理權杖和存取權杖。AWS CLI 也會使用新的存取權杖更新 SSO 快取 JSON 檔案，並重新整理權杖，讓軟體 SDKs 使用。
- 如果您已經有作用中的工作階段，AWS CLI 命令會重複使用現有的工作階段，並會在現有的工作階段過期時過期。若要了解如何設定 IAM Identity Center 工作階段的長度，請參閱 AWS IAM Identity Center 《使用者指南》中的 [設定使用者 AWS 存取入口網站工作階段的持續時間](#)。
  - 工作階段長度上限已延長至 90 天，以減少頻繁登入的需求。

## 開發套件如何取得 AWS 服務 呼叫的登入資料

當您執行個體化每個服務的用戶端物件 AWS 服務 時，SDKs 會提供 的存取權。為 IAM Identity Center 登入資料解析設定共用 AWS config 檔案的所選設定檔時，IAM Identity Center 會用來解析應用程式的登入資料。

- 建立用戶端時，[登入資料解析程序](#) 會在執行時間完成。

若要使用 IAM Identity Center 單一登入來擷取 sigv4 APIs 的登入資料，開發套件會使用 IAM Identity Center 存取字符來取得 IAM 工作階段。此 IAM 工作階段稱為許可集工作階段，它透過擔任 IAM 角色來提供 SDK 的 AWS 存取權。

- 許可集工作階段持續時間與 IAM Identity Center 工作階段持續時間獨立設定。
  - 若要了解如何設定許可集工作階段持續時間，請參閱 AWS IAM Identity Center 《使用者指南》中的 [設定工作階段持續時間](#)。
- 請注意，在大多數 AWS SDK API 文件中，許可集登入資料也稱為 AWS 登入資料和 sigv4 登入資料。

許可集登入資料會從 IAM Identity Center API 的 [getRoleCredentials](#) 呼叫傳回至 SDK。SDK 的用戶端物件使用 擔任的 IAM 角色來呼叫 AWS 服務，例如要求 Amazon S3 列出您帳戶中的儲存貯體。用戶端物件可以繼續使用這些許可集憑證來操作，直到許可集工作階段過期為止。



## 工作階段過期和重新整理

使用時 [SSO 權杖提供者組態](#)，從 IAM Identity Center 取得的每小時存取權杖會使用重新整理權杖自動重新整理。

- 如果存取字符在 SDK 嘗試使用時已過期，則 SDK 會使用重新整理字符來嘗試取得新的存取字符。IAM Identity Center 會將重新整理字符與您的 IAM Identity Center 存取入口網站工作階段持續時間進行比較。如果重新整理字符未過期，IAM Identity Center 會以另一個存取字符回應。
- 此存取權杖可用來重新整理現有用戶端的許可集工作階段，或解析新用戶端的登入資料。

不過，如果 IAM Identity Center 存取入口網站工作階段已過期，則不會授予新的存取字符。因此，無法續約許可集持續時間。每當快取的許可設定現有用戶端的工作階段長度逾時時，就會過期（且將失去存取權）。

一旦 IAM Identity Center 工作階段過期，任何建立新用戶端的程式碼都會失敗身分驗證。這是因為未快取許可集登入資料。在您擁有有效的存取權杖之前，您的程式碼將無法建立新的用戶端並完成登入資料解析程序。

總而言之，當開發套件需要新的許可集登入資料時，開發套件會先檢查任何有效的現有登入資料，並使用這些登入資料。無論登入資料是用於新用戶端，還是具有過期登入資料的現有用戶端，這都適用。如果找不到登入資料或登入資料無效，則 SDK 會呼叫 IAM Identity Center API 以取得新的登入資料。若要呼叫 API，它需要存取權杖。如果存取權杖已過期，開發套件會使用重新整理權杖嘗試從 IAM Identity Center 服務取得新的存取權杖。如果您的 IAM Identity Center 存取入口網站工作階段未過期，則會授予此權杖。

## 使用 IAM Roles Anywhere 驗證 AWS SDKs和工具

您可以使用 IAM Roles Anywhere 在 IAM 中取得臨時安全登入資料，例如在外部執行的伺服器、容器和應用程式 AWS。若要使用 IAM Roles Anywhere，您的工作負載必須使用 X.509 憑證。您的雲端管理員應提供所需的憑證和私有金鑰，以將 IAM Roles Anywhere 設定為您的憑證提供者。

### 步驟 1：隨處設定 IAM 角色

IAM Roles Anywhere 提供取得在外部執行之工作負載或程序的臨時登入資料的方法 AWS。信任錨點會與憑證授權單位建立，以取得相關聯 IAM 角色的臨時憑證。當您的程式碼向 IAM Roles Anywhere 驗證時，該角色會設定工作負載將擁有的許可。

如需設定信任錨點、IAM 角色和 IAM Roles Anywhere 描述檔的步驟，請參閱《IAM [Roles Anywhere 使用者指南](#)》中的在 [AWS Identity and Access Management Roles Anywhere](#) 中建立信任錨點和描述檔。

### Note

IAM Roles Anywhere 使用者指南中的設定檔是指 IAM Roles Anywhere 服務中的唯一概念。它與共用 AWS config 檔案中的設定檔無關。

## 步驟 2：隨處使用 IAM 角色

若要從 IAM Roles Anywhere 取得臨時安全登入資料，請使用 IAM Roles Anywhere 提供的登入資料協助程式工具。登入資料工具會實作 IAM Roles Anywhere 的簽署程序。

如需下載登入資料協助程式工具的指示，請參閱《IAM [AWS Identity and Access Management Roles Anywhere 使用者指南](#)》中的從 [Roles Anywhere](#) 取得臨時安全登入資料。

若要使用 IAM Roles Anywhere AWS SDKs 和 的臨時安全登入資料 AWS CLI，您可以在共用 AWS config 檔案中設定 `credential_process` 設定。SDKs 和 AWS CLI 支援使用 `credential_process` 驗證的程序登入資料提供者。以下顯示設定的一般結構 `credential_process`。

```
credential_process = [path to helper tool] [command] [--parameter1 value] [--parameter2 value] [...]
```

協助程式工具的 `credential-process` 命令會以與 `credential_process` 設定相容的標準 JSON 格式傳回臨時登入資料。請注意，命令名稱包含連字號，但設定名稱包含底線。命令需要下列參數：

- `private-key` – 簽署請求之私有金鑰的路徑。
- `certificate` – 憑證的路徑。
- `role-arn` – 要取得暫時登入資料的角色 ARN。
- `profile-arn` – 提供指定角色映射之設定檔的 ARN。
- `trust-anchor-arn` – 用於驗證的信任錨點 ARN。

您的雲端管理員應該提供憑證和私有金鑰。您可以從複製這三個 ARN 值 AWS 管理主控台。下列範例顯示共用 config 檔案，用於設定從協助程式工具擷取臨時登入資料。

```
[profile dev]
credential_process = ./aws_signing_helper credential-process --certificate /
/path/to/certificate --private-key /path/to/private-key --trust-anchor-
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-
arn arn:aws:iam::account:role/ROLE_ID
```

如需選用參數和其他協助程式工具詳細資訊，請參閱 GitHub 上的 [IAM Roles Anywhere Credential Helper](#)。

如需 SDK 組態設定本身和程序登入資料提供者的詳細資訊，請參閱本指南[程序登入資料提供者](#)中的。

## 使用 AWS 登入資料來擔任角色以驗證 AWS SDKs和工具

假設角色涉及使用一組臨時安全登入資料來存取 AWS 您可能無法存取的資源。這些臨時登入資料由存取金鑰 ID、私密存取金鑰和安全字符組成。若要進一步了解 AWS Security Token Service (AWS STS) API 請求，請參閱 AWS Security Token Service API 參考中的[動作](#)。

若要設定軟體開發套件或工具以擔任角色，您必須先建立或識別要擔任的特定角色。IAM 角色由角色 Amazon Resource Name ([ARN](#)) 唯一識別。角色會與其他實體建立信任關係。使用角色的信任實體可能是 AWS 服務 或另一個 AWS 帳戶。若要進一步了解 IAM 角色，請參閱《[IAM 使用者指南](#)》中的[使用 IAM 角色](#)。

識別 IAM 角色之後，如果您受該角色信任，您可以將 SDK 或工具設定為使用角色授予的許可。

### Note

AWS 最佳實務是盡可能使用區域端點並設定您的 [AWS 區域](#)。

## 擔任 IAM 角色

擔任角色時，會 AWS STS 傳回一組臨時安全登入資料。這些登入資料來自另一個設定檔，或是您的程式碼執行所在的執行個體或容器。當您有一個帳戶的 AWS 登入資料，但您的應用程式需要存取另一個帳戶中的資源時，通常會使用這種類型的擔任角色。



```
region = us-east-1
output = json
role_arn = arn:aws:iam::123456789012:role/my-role-name
source_profile = profile-with-user-that-can-assume-role
role_session_name = my_session
```

如需所有擔任角色登入資料提供者設定的詳細資訊，請參閱本指南[擔任角色登入資料提供者](#)中的。

## 使用 Web 身分或 OpenID Connect 擔任角色以驗證 AWS SDKs和工具

假設角色涉及使用一組臨時安全登入資料來存取 AWS 您可能無法存取的資源。這些臨時登入資料由存取金鑰 ID、私密存取金鑰和安全字符組成。若要進一步了解 AWS Security Token Service (AWS STS) API 請求，請參閱 [AWS Security Token Service API 參考](#)中的[動作](#)。

若要設定軟體開發套件或工具以擔任角色，您必須先建立或識別要擔任的特定角色。IAM 角色由角色 Amazon Resource Name ([ARN](#)) 唯一識別。角色會與其他實體建立信任關係。使用角色的信任實體可能是 Web 身分提供者或 OpenID Connect(OIDC) 或 SAML 聯合。若要進一步了解 IAM 角色，請參閱《IAM 使用者指南》中的[擔任角色的方法](#)。

在軟體開發套件中設定 IAM 角色之後，如果該角色設定為信任您的身分提供者，您可以進一步設定軟體開發套件擔任該角色，以取得臨時 AWS 登入資料。

### Note

AWS 最佳實務是盡可能使用區域端點並設定您的 [AWS 區域](#)。

## 與 Web 身分或 OpenID Connect 聯合

您可以使用公有身分提供者的 JSON Web Token (JWTs)，例如 Login With Amazon、Facebook、Google，以使用取得臨時 AWS 登入資料 `AssumeRoleWithWebIdentity`。視其使用方式而定，這些 JWTs 可能會稱為 ID 字符或存取字符。您也可以使用從與 OIDC 探索通訊協定相容的身分提供者 (IdPs) 發行的 JWTs，例如 EntraId 或 PingFederate。

如果您使用的是 Amazon Elastic Kubernetes Service，此功能可讓您為 Amazon EKS 叢集中的每個服務帳戶指定不同的 IAM 角色。此 Kubernetes 功能會將 JWTs 分發到您的 Pod，然後由此登入資料提供者用來取得臨時 AWS 登入資料。如需此 Amazon EKS 組態的詳細資訊，請參閱《Amazon EKS 使

用者指南》中的[服務帳戶的 IAM 角色](#)。不過，對於更簡單的選項，如果您的 [SDK 支援](#)，建議您改用 [Amazon EKS Pod 身分](#)。

## 步驟 1：設定身分提供者和 IAM 角色

若要使用外部 IdP 設定聯合，請使用 IAM 身分提供者來通知 AWS 外部 IdP 及其組態。這會在您的 AWS 帳戶和外部 IdP 之間建立信任。設定 SDK 以使用 JSON Web Token (JWT) 進行身分驗證之前，您必須先設定身分提供者 (IdP) 和用來存取它的 IAM 角色。若要設定這些項目，請參閱《IAM 使用者指南》中的[為 Web 身分或 OpenID Connect Federation \(主控台\) 建立角色](#)。

## 步驟 2：設定 SDK 或工具

設定 SDK 或工具以使用來自的 JSON Web Token (JWT) AWS STS 進行身分驗證。

當您在設定檔中指定此項目時，軟體開發套件或工具會自動為您進行對應的 AWS STS [AssumeRoleWithWebIdentity](#) API 呼叫。若要使用 Web 聯合身分擷取和使用臨時憑證，請在共用 AWS config 檔案中指定下列組態值。如需這些設定的詳細資訊，請參閱 [擔任角色登入資料提供者設定](#) 一節。

- `role_arn` - 來自您在步驟 1 中建立的 IAM 角色
- `web_identity_token_file` - 從外部 IdP
- (選用) `duration_seconds`
- (選用) `role_session_name`

以下是使用 Web 身分擔任角色的共用 config 檔案組態範例：

```
[profile web-identity]  
role_arn=arn:aws:iam::123456789012:role/my-role-name  
web_identity_token_file=/path/to/a/token
```

### Note

對於行動應用程式，請考慮使用 Amazon Cognito。Amazon Cognito 充當身分中介裝置，並為您執行大部分聯合工作。不過，Amazon Cognito 身分提供者不包含在 SDKs 和工具核心程式庫中，就像其他身分提供者一樣。若要存取 Amazon Cognito API，請在 SDK 或工具的建置或程式庫中包含 Amazon Cognito 服務用戶端。如需使用 AWS SDKs 請參閱《Amazon Cognito 開發人員指南》中的[程式碼範例](#)。

如需所有擔任角色登入資料提供者設定的詳細資訊，請參閱本指南[擔任角色登入資料提供者](#)中的。

## 使用 AWS 存取金鑰來驗證 AWS SDKs和工具

使用 AWS SDKs和工具時，使用 AWS 存取金鑰是身分驗證的選項。

### 使用短期登入資料

我們建議您設定軟體開發套件或工具，[使用 IAM Identity Center 驗證 AWS SDK 和工具](#)以使用延長的工作階段持續時間選項。

不過，若要直接設定 SDK 或工具的臨時登入資料，請參閱[使用短期登入資料來驗證 AWS SDKs和工具](#)。

### 使用長期登入資料

#### Warning

為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如[AWS IAM Identity Center](#)。

## 管理跨 的存取 AWS 帳戶

作為安全最佳實務，我們建議您使用 AWS Organizations 搭配 IAM Identity Center 來管理所有的存取 AWS 帳戶。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的安全性最佳實務](#)。

您可以在 IAM Identity Center 中建立使用者、使用 Microsoft Active Directory、使用 SAML 2.0 身分提供者 (IdP)，或個別聯合您的 IdP AWS 帳戶。使用這些方法之一，您可以為使用者提供單一登入體驗。您也可以強制執行多重要素驗證 (MFA)，並使用臨時登入資料進行 AWS 帳戶 存取。這與 IAM 使用者不同，IAM 使用者是可共用的長期登入資料，可能會增加 AWS 資源的安全風險。

### 僅為沙盒環境建立 IAM 使用者

如果您是初次使用 AWS，您可以建立測試 IAM 使用者，然後使用它來執行教學課程，並探索 AWS 提供的內容。您可以在學習時使用此類型的登入資料，但我們建議您避免在沙盒環境之外使用。

對於下列使用案例，在 中開始使用 IAM 使用者可能很合理 AWS：

- 開始使用 AWS SDK 或工具，並在沙盒 AWS 服務 環境中探索。

- 在學習過程中，執行不支援人工登入程序的排程指令碼、任務和其他自動化程序。

如果您在這些使用案例之外使用 IAM 使用者，請 AWS 帳戶 盡快轉換至 IAM Identity Center 或聯合身分提供者至。如需詳細資訊，請參閱 [中的聯合身分 AWS](#)。

## 安全 IAM 使用者存取金鑰

您應該定期輪換 IAM 使用者存取金鑰。請遵循《IAM 使用者指南》中的[輪換存取金鑰](#)的指引。如果您認為自己不小心共用了 IAM 使用者存取金鑰，請輪換存取金鑰。

IAM 使用者存取金鑰應存放在本機電腦上的共用 AWS credentials 檔案中。請勿將 IAM 使用者存取金鑰存放在程式碼中。請勿在任何原始碼管理軟體中包含包含 IAM 使用者存取金鑰的組態檔案。開放原始碼專案 [git-secrets](#) 等外部工具可協助您避免不小心將敏感資訊遞交至 Git 儲存庫。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 身分 \(使用者、使用者群組和角色\)](#)。

若要設定 IAM 使用者以開始使用，請參閱 [使用長期登入資料來驗證 AWS SDKs和工具](#)。

## 使用短期登入資料來驗證 AWS SDKs和工具

我們建議您設定軟體 AWS 開發套件或工具，以[使用 IAM Identity Center 驗證 AWS SDK 和工具](#)搭配延長的工作階段持續時間選項使用。不過，您可以複製並使用 AWS 存取入口網站中提供的臨時登入資料。若這些憑證過期，便需要複製新憑證。您可以在設定檔中使用臨時憑證，或用作系統屬性和環境變數的值。

最佳實務：我們建議您的應用程式使用從下列位置交付的臨時登入資料，而不是手動管理登入資料檔案中的存取金鑰和權杖：

- AWS 運算服務，例如在 Amazon Elastic Compute Cloud 或 中執行應用程式 AWS Lambda。
- 登入資料提供者鏈結中的另一個選項，例如 [使用 IAM Identity Center 驗證 AWS SDK 和工具](#)。
- 或使用 [程序登入資料提供者](#)擷取臨時登入資料。

使用從 AWS 存取入口網站擷取的短期登入資料來設定登入資料檔案

1. [建立共用的登入資料檔案](#)。
2. 在登入資料檔案中，貼上下列預留位置文字，直到您貼上運作中的臨時登入資料為止。

```
[default]
aws_access_key_id=<value from AWS access portal>
```





## 憑證的重要警告和指引

### 憑證警告

- 請勿使用您帳戶的根憑證存取 AWS 資源。這些登入資料可讓未管制的帳戶存取和很難撤銷這些帳戶。
- 請勿在應用程式檔案中放置常值存取金鑰或憑證資訊。如果您不小心這麼做了，則會有暴露您登入資料的風險，例如，當您上傳專案到公有儲存庫時。
- 請勿在您的專案區域中放入包含憑證的檔案。
- 請注意，存放在共用 AWS credentials 檔案中的任何登入資料都會以純文字儲存。

### 安全管理憑證的其他指引

如需如何安全管理 AWS 登入資料的一般討論，請參閱《》中的[管理 AWS 存取金鑰的最佳實務AWS 一般參考](#)。除了討論之外，請考慮下列事項：

- 使用適用於 Amazon Elastic Container Service (Amazon ECS) 任務的[任務 IAM 角色](#)。
- 使用在 Amazon EC2 執行個體上執行的應用程式的 [IAM 角色](#)。

## 先決條件：建立 AWS 帳戶

若要使用 IAM 使用者存取 AWS 服務，您需要 AWS 帳戶和 AWS 登入資料。

### 1. 建立帳戶。

若要建立 AWS 帳戶，請參閱 AWS 帳戶管理 參考指南中的[入門：您是第一次 AWS 使用嗎？](#)。

### 2. 建立管理使用者。

避免使用根使用者帳戶 (您建立的初始帳戶) 來存取管理主控台與服務。反之，請依據 IAM 使用者指南的[建立管理使用者](#)中的說明，建立管理使用者帳戶。

建立管理使用者帳戶並記錄登入詳細資料之後，請務必登出您的根使用者帳戶，然後使用管理帳戶重新登入。

這些帳戶都不適用於在 上執行開發 AWS 或在 上執行應用程式 AWS。最佳實務是，您需要建立適合這些任務的使用者、許可集或服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[套用最低權限許可](#)。

## 步驟 1：建立 IAM 使用者

- 按照 IAM 使用者指南中的[建立 IAM 使用者 \(主控台\)](#) 程序來建立 IAM 使用者。建立 IAM 使用者時：
  - 我們建議您選取提供使用者對的存取權 AWS 管理主控台。這可讓您檢視與在視覺化環境中執行的程式碼 AWS 服務相關的，例如檢查 AWS CloudTrail 診斷日誌或將檔案上傳至 Amazon Simple Storage Service，這在偵錯程式碼時很有幫助。
  - 針對設定許可 - 許可選項，選取直接連接政策，以了解如何將許可指派給此使用者。
    - 多數「入門」SDK 教學都使用 Amazon S3 服務做為範例。若要讓應用程式能夠完整存取 Amazon S3，請選取要連接至此使用者的 AmazonS3FullAccess 政策。
  - 您可以忽略該程序有關設定許可界限或標籤的選用步驟。

## 步驟 2：取得您的存取金鑰

1. 在 IAM 主控台的導覽窗格中，選取使用者，然後選取您先前建立使用者的 **User name**。
2. 在使用者頁面上，選取安全憑證頁面。接著，在存取金鑰下，選取建立存取金鑰。
3. 針對建立存取金鑰步驟 1，選擇命令列界面 (CLI) 或本機程式碼。這兩個選項都會產生與 AWS CLI 和 SDKs 搭配使用的相同類型金鑰。
4. 在建立存取金鑰步驟 2 中，輸入選用標籤並選取下一步。
5. 在建立存取金鑰步驟 3 中，選取下載 .csv 檔案，以儲存包含 IAM 使用者存取金鑰和私密存取金鑰的 .csv 檔案。您之後將會用到此資訊。

### Warning

使用適當的安全措施來保護這些登入資料的安全。

6. 選取 Done (完成)。

## 步驟 3：更新共用 **credentials** 檔案

1. 建立或開啟共用的 AWS credentials 檔案。這個檔案是位於 Linux 和 macOS 系統上的 `~/.aws/credentials` 和 Windows 上的 `%USERPROFILE%\aws\credentials`。如需詳細資訊，請參閱[登入資料檔案的位置](#)。

- 將以下文字新增至共用的 `credentials` 檔案。將範例 ID 值和範例索引鍵值取代為您先前下載的 `.csv` 檔案中的值。

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

- 儲存檔案。

共用 `credentials` 檔案是存放登入資料的最常見方式。這些也可以設定為環境變數，請參閱 [AWS 存取金鑰](#) 以取得環境變數名稱。這是讓您開始使用的方法，但我們建議您盡快轉換到 IAM Identity Center 或其他臨時登入資料。在您不使用長期登入資料後，請記得從共用 `credentials` 檔案刪除這些登入資料。

## 使用 IAM 角色來驗證部署到 Amazon EC2 的應用程式

此範例涵蓋設定具有 Amazon S3 存取權 AWS Identity and Access Management 的角色，以便在部署到 Amazon Elastic Compute Cloud 執行個體的應用程式中使用。

若要在 Amazon Elastic Compute Cloud 執行個體上執行 AWS SDK 應用程式，請建立 IAM 角色，然後授予 Amazon EC2 執行個體對該角色的存取權。如需更多詳細資訊，請參閱 Amazon EC2 使用者指南中的 [Amazon EC2 的 IAM 角色](#)。

### 建立 IAM 角色

您開發的 AWS SDK 應用程式可能存取至少一個 AWS 服務來執行動作。建立 IAM 角色，授予應用程式執行所需的必要許可。

此程序會建立角色，以授予 Amazon S3 的唯讀存取權做為範例。許多 AWS SDK 指南都有從 Amazon S3 讀取的「入門」教學課程。

- 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/iam/> 的 IAM 主控台。
- 在導覽窗格中，選取角色，然後選取建立角色。
- 對於選取信任的實體，在信任的實體類型下，選擇 AWS 服務。
- 在使用案例中，選擇 Amazon EC2，然後選擇下一步。
- 對於新增許可，請從政策清單中選取 Amazon S3 唯讀存取的核取方塊，然後選取下一步。

6. 輸入角色的名稱，然後選取建立角色。請記住此名稱，因為您在建立 Amazon EC2 執行個體時需要此名稱。

## 啟動 Amazon EC2 執行個體並指定您的 IAM 角色

您可以使用 IAM 角色建立和啟動 Amazon EC2 執行個體，方法如下：

- 遵循《Amazon EC2 使用者指南》中的[快速啟動執行個體](#)。不過，在最終提交步驟之前，也請執行下列動作：
  - 在進階詳細資訊下，針對 IAM 執行個體描述檔，選擇您在上一個步驟中建立的角色。

透過此 IAM 和 Amazon EC2 設定，您可以將應用程式部署到 Amazon EC2 執行個體，而您的應用程式將具有 Amazon S3 服務的讀取存取權。

## 連線至 EC2 執行個體

連線至 Amazon EC2 執行個體，以便您可以將應用程式轉移到該執行個體，然後執行應用程式。建立執行個體時，您需要包含金鑰對（登入）下所用金鑰對私有部分的檔案，也就是 PEM 檔案。

您可以遵循執行個體類型的指引來執行此操作：[連接至 Linux 執行個體](#)或[連接至 Windows 執行個體](#)。當您連線時，請以可讓您將檔案從開發機器傳輸到執行個體的方式執行此操作。

### Note

在 Linux 或 macOS 終端機上，您可以使用安全複製命令來複製應用程式。若要 scp 搭配金鑰對使用，您可以使用下列命令：`scp -i path/to/key file/to/copy ec2-user@ec2-xx-xx-xxx-xxx.compute.amazonaws.com:~`。

如需 Windows 的詳細資訊，請參閱[將檔案傳輸至 Windows 執行個體](#)。

如果您使用的是 AWS Toolkit，通常也可以使用 Toolkit 連線到執行個體。如需詳細資訊，請參閱您使用之 Toolkit 的特定使用者指南。

## 在 EC2 執行個體上執行您的應用程式

1. 將您的應用程式檔案從本機磁碟機複製到 Amazon EC2 執行個體。
2. 啟動應用程式，並確認其執行的結果與開發機器上的結果相同。

3. (選用) 驗證應用程式是否使用 IAM 角色提供的登入資料。
  - a. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/ec2/>:// 開啟 Amazon EC2 主控台。
  - b. 選取執行個體。
  - c. 選擇動作、安全性，然後選擇修改 IAM 角色。
  - d. 對於 IAM 角色，選擇無 IAM 角色來分離 IAM 角色。
  - e. 選擇更新 IAM 角色。
  - f. 再次執行應用程式並確認傳回授權錯誤。

## 使用 TIP 外掛程式存取 AWS 服務

信任的身分傳播 (TIP) 是的一項功能 AWS IAM Identity Center，可讓的管理員根據 群組關聯等使用者屬性 AWS 服務 授予許可。透過受信任的身分傳播，身分內容會新增至 IAM 角色，以識別請求存取 AWS 資源的使用者。此內容會傳播到其他 AWS 服務。

身分內容包含的資訊，AWS 服務 用於在收到存取請求時做出授權決策。此資訊包含識別請求者的中繼資料（例如，IAM Identity Center 使用者）、請求存取的 AWS 服務（例如，Amazon Redshift），以及存取範圍（例如，唯讀存取）。接收 AWS 服務 使用此內容，以及指派給使用者的任何許可，來授權存取其資源。如需詳細資訊，請參閱 AWS IAM Identity Center 《使用者指南》中的[信任身分傳播概觀中的](#)。

TIP 外掛程式可與支援受信任身分傳播 AWS 服務的 搭配使用。做為參考使用案例，請參閱《[Amazon Q Business 使用者指南](#)》中的[使用 設定 AWS IAM Identity Center Amazon Q Business 應用程式](#)。

### Note

如果您使用的是 Amazon Q Business，請參閱[使用 設定 Amazon Q Business 應用程式 AWS IAM Identity Center](#)以取得服務特定指示。

## 使用 TIP 外掛程式的先決條件

需要下列資源，外掛程式才能運作：

1. 您必須使用 適用於 Java 的 AWS SDK 或 適用於 JavaScript 的 AWS SDK。
2. 確認您正在使用的服務支援信任的身分傳播。

請參閱《AWS IAM Identity Center 使用者指南》中[AWS 透過與 IAM Identity Center 整合之受管應用程式](#)的 IAM Identity Center 啟用受信任身分傳播欄。

### 3. 啟用 IAM Identity Center 和信任的身分傳播。

請參閱AWS IAM Identity Center 《使用者指南》中的 [TIP 先決條件和考量事項](#)。

### 4. 您必須擁有 Identity-Center-integrated的應用程式。

請參閱AWS IAM Identity Center 《使用者指南》中的[AWS 受管應用程式](#)或[客戶受管應用程式](#)。

### 5. 您必須設定信任的權杖發行者 (TTI)，並將您的服務連線至 IAM Identity Center。

請參閱《AWS IAM Identity Center 使用者指南》中[設定信任權杖發行者的先決條件](#)和任務。

## 在程式碼中使用 TIP 外掛程式

1. 建立信任身分傳播外掛程式的執行個體。
2. 建立服務用戶端執行個體以與您的 互動，AWS 服務 並透過新增信任的身分傳播外掛程式來自訂服務用戶端。

TIP 外掛程式採用下列輸入參數：

- **webTokenProvider**：客戶實作以從外部身分提供者取得 OpenID 字符的函數。
- **accessRoleArn**：由具有使用者身分內容的外掛程式擔任的 IAM 角色 ARN，以取得身分增強憑證。
- **applicationArn**：用戶端或應用程式的唯一識別符字串。此值是已設定 OAuth 授予的應用程式 ARN。
- **ssoOidcClient**：(選用) SSO OIDC 用戶端，例如[SsoOidcClient](#)適用於 Java 的 或[client-sso-oidc](#)適用於 JavaScript 的，具有客戶定義的組態。若未提供，則會使用 applicationRoleArn 建立並啟用預設的 OIDC 用戶端。
- **stsClient**：(選用) AWS STS 具有客戶定義組態的用戶端，用於accessRoleArn以使用者的身分內容擔任。如果未提供，applicationRoleArn則會執行個體化並使用的 AWS STS 用戶端。
- **applicationRoleArn**：(選用) 要擔任的 IAM 角色 ARN，AssumeRoleWithWebIdentity以便可以引導 OIDC 和 AWS STS 用戶端。
  - 如果未提供，則必須同時提供 ssoOidcClient和 stsClient 參數。

- 如果提供，`applicationRoleArn`不能與 `accessRoleArn` 參數的值相同。  
`applicationRoleArn` 用於建置 `stsClient`，其用於擔任 `accessRole`。如果 `applicationRole`和 `accessRole` 都使用相同的角色，則表示使用角色來擔任自己（自我角色假設），這會不建議這麼做 AWS。如需詳細資訊，請參閱[公告](#)。

## `ssoOidcClient`、`stsClient`和 `applicationRoleArn` 參數的考量事項

設定 TIP 外掛程式時，請根據您提供的參數，考慮下列許可要求：

- 如果您要提供 `ssoOidcClient`和 `stsClient`：
  - 上的登入資料`ssoOidcClient`應具有呼叫身分中心的`oauth:CreateTokenWithIAM`許可，以取得身分中心特定的使用者內容。
  - 上的登入資料`stsClient`應具有 `sts:AssumeRole`和 `sts:SetContext`許可`accessRole`。`accessRole`也需要設定與 `stsClient`上登入資料的信任關係。
- 如果您要提供 `applicationRoleArn`：
  - `applicationRole` 應該具有必要資源 (`IdC` 執行個體 `accessRole`) 的 `oauth:CreateTokenWithIAM``sts:AssumeRole`和 `sts:SetContext`許可，因為它將用於建置 `OIDC` 和 `STS` 用戶端。
  - `applicationRole` 應與用於產生的身分提供者具有信任關係`webToken`，因為 `webToken`將用於透過外掛程式的 [AssumeRoleWithWebIdentity](#) 呼叫擔任 `applicationRole`。

ApplicationRole 組態範例：

具有 Web 權杖提供者的信任政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::ACCOUNT_ID:oidc-provider/IDENTITY_PROVIDER_URL"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "IDENTITY_PROVIDER_URL:aud": "CLIENT_ID_TO_BE_TRUSTED"
        }
      }
    }
  ]
}
```



```
    }
  }
}
]
```

許可政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext"
      ],
      "Resource": [
        "accessRoleArn"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "sso-oauth:CreateTokenWithIAM"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

## 使用 TIP 的程式碼範例

以下範例示範如何使用 適用於 Java 的 AWS SDK 或 在程式碼中實作 TIP 外掛程式 適用於 JavaScript 的 AWS SDK。

### Java

若要在 適用於 Java 的 AWS SDK 專案中使用 TIP 外掛程式，您需要在專案的 `pom.xml` 檔案中將其宣告為相依性。

```
<dependency>
<groupId>software.amazon.awssdk.trustedIdentityPropagation</groupId>
<artifactId>aws-sdk-java-trustedIdentityPropagation-java-plugin</artifactId>
  <version>2.0.0</version>
</dependency>
```

在您的原始程式碼中，包含所需的套件陳述式 `software.amazon.awssdk.trustedidentitypropagation`。

下列範例顯示建立受信任身分傳播外掛程式執行個體並將其新增至服務用戶端的兩種方式。這兩個範例都使用 Amazon S3 做為服務，並使用 `S3AccessGrantsPlugin` 來管理使用者特定許可，但可以套用到支援受信任身分傳播 (TIP) 的任何 AWS 服務。

### Note

對於這些範例，您需要從 S3 Access Grants 設定使用者特定許可。如需詳細資訊，請參閱 [S3 Access Grants 文件](#)。

## 選項 1：建置並傳遞 OIDC 和 STS 用戶端

```
SsoOidcClient oidcClient = SsoOidcClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

StsClient stsClient = StsClient.builder()
    .region(Region.US_EAST_1)
    .credentialsProvider(credentialsProvider).build();

TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
    TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .ssoOidcClient(oidcClient)
        .stsClient(stsClient)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
    .build();
```

```
S3Client s3Client =
    S3Client.builder().region(Region.US_EAST_1)
        .crossRegionAccessEnabled(true)
        .addPlugin(trustedIdentityPropagationPlugin)
        .addPlugin(accessGrantsPlugin)
        .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
    .key("path/to/object/fileName")
    .bucket("bucketName")
    .build());
```

## 選項 2：將 applicationRoleArn 和延遲用戶端建立傳遞至外掛程式

```
TrustedIdentityPropagationPlugin trustedIdentityPropagationPlugin =
    TrustedIdentityPropagationPlugin.builder()
        .webTokenProvider(() -> webToken)
        .applicationArn(idcApplicationArn)
        .accessRoleArn(accessRoleArn)
        .applicationRoleArn(applicationRoleArn)
        .build();

S3AccessGrantsPlugin accessGrantsPlugin = S3AccessGrantsPlugin.builder()
    .build();

S3Client s3Client =
    S3Client.builder().region(Region.US_EAST_1)
        .crossRegionAccessEnabled(true)
        .addPlugin(trustedIdentityPropagationPlugin)
        .addPlugin(accessGrantsPlugin)
        .build();

final var resp = s3Client.getObject(GetObjectRequest.builder()
    .key("path/to/object/fileName")
    .bucket("bucketName")
    .build());
```

如需其他詳細資訊和來源，請參閱 GitHub 上的 [trusted-identity-propagation-java](#)。

## JavaScript

執行下列命令，在您的適用於 JavaScript 的 AWS SDK 專案中安裝 TIP 身分驗證外掛程式套件：

```
$ npm i @aws-sdk-extension/trusted-identity-propagation
```

最終package.json應包含類似如下的相依性：

```
"dependencies": {  
  "@aws-sdk-extension/trusted-identity-propagation": "^2.0.0"  
},
```

在您的原始程式碼中，匯入所需的TrustedIdentityPropagationExtension相依性。

下列範例顯示建立受信任身分傳播外掛程式執行個體並將其新增至服務用戶端的兩種方式。這兩個範例都使用 Amazon S3 做為服務，並使用 Amazon S3 Access Grants 來管理使用者特定許可，但可以套用到支援受信任身分傳播 (TIP) 的任何 AWS 服務。

#### Note

如需這些範例，您需要從 Amazon S3 Access Grants 設定使用者特定許可，請參閱 [Amazon S3 Access Grants 文件](#) 以取得更多詳細資訊。

### 選項 1：建置並傳遞 OIDC 和 STS 用戶端

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";  
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";  
import { TrustedIdentityPropagationExtension } from "@aws-sdk-extension/trusted-identity-propagation";  
  
const s3ControlClient = new S3ControlClient({  
  region: "us-east-1",  
  extensions: [  
    TrustedIdentityPropagationExtension.create({  
      webTokenProvider: async () => {  
        return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';  
      },  
      ssoOidcClient: customOidcClient,  
      stsClient: customStsClient,  
      accessRoleArn: accessRoleArn,  
      applicationArn: applicationArn,  
    }  
  ]  
});
```

```
});

const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  // Create a new S3 client with the temporary credentials
  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });

  // Use the temporary S3 client to perform the operation
  const s3Params = {
    Bucket: "BUCKET_NAME",
    Key: "S3_OBJECT_KEY",
  };
  const getObjectCommand = new GetObjectCommand(s3Params);
  const s3Object = await temporaryS3Client.send(getObjectCommand);

  const fileContent = await s3Object.Body.transformToString();

  // Process the S3 object data
  console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

## 選項 2：將 applicationRoleArn 和延遲用戶端建立傳遞至外掛程式

```
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";
import { S3ControlClient, GetDataAccessCommand } from "@aws-sdk/client-s3-control";
import { TrustedIdentityPropagationTokenExtension } from "@aws-sdk-extension/trusted-identity-propagation";

const s3ControlClient = new S3ControlClient({
  region: "us-east-1",
  extensions: [
    TrustedIdentityPropagationTokenExtension.create({
      webTokenProvider: async () => {
        return 'ID_TOKEN_FROM_YOUR_IDENTITY_PROVIDER';
      },
      accessRoleArn: accessRoleArn,
      applicationRoleArn: applicationRoleArn,
      applicationArn: applicationArn,
    }),
  ],
});

// Same S3 AccessGrants workflow as Option 1
const getDataAccessParams = {
  Target: "S3_URI_PATH",
  Permission: "READ",
  AccountId: ACCOUNT_ID,
  InstanceArn: S3_ACCESS_GRANTS_ARN,
  TargetType: "Object",
};

try {
  const command = new GetDataAccessCommand(getDataAccessParams);
  const response = await s3ControlClient.send(command);

  const credentials = response.Credentials;

  const temporaryS3Client = new S3Client({
    region: "us-east-1",
    credentials: {
      accessKeyId: credentials.AccessKeyId,
      secretAccessKey: credentials.SecretAccessKey,
      sessionToken: credentials.SessionToken,
    },
  });
```

```
const s3Params = {
  Bucket: "BUCKET_NAME",
  Key: "S3_OBJECT_KEY",
};
const getObjectCommand = new GetObjectCommand(s3Params);
const s3object = await temporaryS3Client.send(getObjectCommand);

const fileContent = await s3object.Body.transformToString();

console.log("Successfully retrieved S3 object:", fileContent);
} catch (error) {
  console.error("Error accessing S3 data:", error);
}
```

如需其他詳細資訊和來源，請參閱 GitHub 上的 [trusted-identity-propagation-js](#)。

# AWS SDKs和工具設定參考

SDKs提供特定語言APIs AWS 服務。他們負責成功進行 API 呼叫所需的一些繁重工作，包括身分驗證、重試行為等。為此，SDKs具有靈活的策略，以取得用於請求的登入資料、維護用於每個服務的設定，以及取得用於全域設定的值。

您可以在以下章節中找到組態設定的詳細資訊：

- [AWS SDKs和工具標準化登入資料提供者](#) – 跨多個 SDKs常見登入資料提供者。
- [AWS SDKs和工具標準化功能](#) – 跨多個 SDKs常見功能。

## 建立服務用戶端

若要以程式設計方式存取 AWS 服務，SDKs會針對每個 SDK 使用用戶端類別/物件 AWS 服務。例如，如果您的應用程式需要存取 Amazon EC2，您的應用程式會建立 Amazon EC2 用戶端物件以與該服務連接。然後，您可以使用 服務用戶端向該用戶端提出請求 AWS 服務。在大多數 SDKs 中，服務用戶端物件是不可變的，因此您必須為您提出請求的每個服務建立新的用戶端，並使用不同的組態對相同的服務提出請求。

## 設定的優先順序

全域設定可設定大多數 SDKs 支援且具有廣泛影響的功能、登入資料提供者和其他功能 AWS 服務。所有 SDKs 都有一系列位置（或來源）進行檢查，以尋找全域設定的值。以下是設定查詢優先順序：

1. 在程式碼中或服務用戶端本身上設定的任何明確設定，都優先於任何其他設定。
  - 某些設定可以根據每個操作進行設定，也可以視需要為您叫用的每個操作進行變更。對於 AWS CLI 或 AWS Tools for PowerShell，這些採用您在命令列中輸入的每個操作參數形式。對於 SDK，明確指派可以採用您在執行個體化 AWS 服務 用戶端或組態物件時設定的參數形式，或者有時當您呼叫個別 API 時。
2. 僅限 Java/Kotlin：會檢查設定的 JVM 系統屬性。如果已設定，則會使用該值來設定用戶端。
3. 檢查 環境變數。如果已設定，則會使用該值來設定用戶端。
4. SDK 會檢查 設定的共用credentials檔案。如果已設定，用戶端會使用它。
5. 設定的共用config檔案。如果設定存在，則 SDK 會使用它。
  - AWS\_PROFILE 環境變數或 aws.profile JVM 系統屬性可用來指定 SDK 載入的設定檔。
6. 開發套件原始碼本身提供的任何預設值都會最後使用。



**Note**

有些 SDKs和工具可能會以不同的順序進行檢查。此外，某些 SDKs和工具支援其他儲存和擷取參數的方法。例如，適用於 .NET 的 AWS SDK 支援稱為 [SDK Store](#) 的其他來源。如需開發套件或工具唯一提供者的詳細資訊，請參閱您正在使用之開發套件或工具的特定指南。

順序會決定哪些方法優先並覆寫其他方法。例如，如果您在共用 config 檔案中設定設定檔，只有在軟體開發套件或工具先檢查其他位置之後，才會找到並使用它。這表示如果您在 credentials 檔案中放置設定，則會使用該設定，而不是在 config 檔案中找到的設定。如果您使用設定和值設定環境變數，它會覆寫 credentials 和 config 檔案中的設定。最後，個別操作 (AWS CLI 命令列參數或 API 參數) 或程式碼中的設定會覆寫該命令的所有其他值。

## 了解本指南的設定頁面

本指南的設定參考區段中的頁面詳細說明了可透過各種機制設定的可用設定。下表列出組態和登入資料檔案設定、環境變數，以及 (適用於 Java 和 Kotlin SDKs) 可在程式碼外部用來設定功能的 JVM 設定。每個清單中的每個連結主題都會帶您前往對應的設定頁面。

- [Config 檔案設定清單](#)
- [Credentials 檔案設定清單](#)
- [環境變數清單](#)
- [JVM 系統屬性清單](#)

每個登入資料提供者或功能都有一個頁面，其中會列出用於設定該功能的設定。對於每個設定，您通常可以透過將設定新增至組態檔案，或透過設定環境變數，或 (僅適用於 Java 和 Kotlin) 透過設定 JVM 系統屬性來設定值。每個設定都會列出所有支援的方法來設定描述詳細資訊上方區塊中的值。雖然 [優先順序](#) 不同，但無論如何設定，產生的功能都相同。

描述將包含預設值，如果有的話，如果不執行任何動作就會生效。它也會定義該設定的有效值。

例如，讓我們從 [請求壓縮](#) 功能頁面查看設定。

disable\_request\_compression 範例設定的資訊會記錄下列項目：

- 控制程式碼庫外部的請求壓縮有三種同等的方式。您可擇一方法：
  - 使用在組態檔案中設定 disable\_request\_compression

- 使用 將其設定為環境變數 `AWS_DISABLE_REQUEST_COMPRESSION`
- 或者，如果您使用的是 Java 或 Kotlin SDK，請使用 將其設定為 JVM 系統屬性 `aws.disableRequestCompression`

### Note

也可能有方法可以直接在您的程式碼中設定相同的功能，但此參考不會涵蓋此項目，因為它對於每個 SDK 都是唯一的。如果您想要在程式碼本身中設定組態，請參閱您的特定 SDK 指南或 API 參考。

- 如果您不執行任何動作，值會預設為 `false`。
- 此布林值設定的唯一有效值為 `true`和 `false`。

在每個功能頁面底部有一個 Support AWS SDKs和工具資料表。

此資料表顯示您的 SDK 是否支援頁面上列出的設定。Supported 資料欄會以下列值表示支援層級：

- Yes – 開發套件完全支援寫入的設定。
- Partial – 支援某些設定，或行為偏離描述。對於 Partial，額外的備註會指出偏差。
- No – 不支援任何設定。這不會宣告程式碼中是否可以實現相同的功能；這只會指出不支援列出的外部組態設定。

## Config 檔案設定清單

下表列出的設定可以在共用 AWS config檔案中指派。它們是全域的，會影響所有 AWS 服務。SDKs 和工具也可能支援唯一的設定和環境變數。若要查看僅個別 SDK 或工具支援的設定和環境變數，請參閱該特定 SDK 或工具指南。

設定名稱	詳細資訊
<code>account_id_endpoint_mode</code>	<a href="#">帳戶型端點</a>
<code>api_versions</code>	<a href="#">一般組態設定</a>

設定名稱	詳細資訊
auth_scheme_preference	<a href="#">身分驗證機制</a>
aws_access_key_id	<a href="#">AWS 存取金鑰</a>
aws_account_id	<a href="#">帳戶型端點</a>
aws_secret_access_key	<a href="#">AWS 存取金鑰</a>
aws_session_token	<a href="#">AWS 存取金鑰</a>
ca_bundle	<a href="#">一般組態設定</a>
credential_process	<a href="#">程序登入資料提供者</a>
credential_source	<a href="#">擔任角色登入資料提供者</a>
defaults_mode	<a href="#">智慧組態預設值</a>
disable_host_prefix_injection	<a href="#">主機字首注入</a>
disable_request_compression	<a href="#">請求壓縮</a>
duration_seconds	<a href="#">擔任角色登入資料提供者</a>
ec2_metadata_service_endpoint	<a href="#">IMDS 登入資料提供者</a>

設定名稱	詳細資訊
ec2_metadata_service_endpoint_mode	<a href="#">IMDS 登入資料提供者</a>
ec2_metadata_v1_disabled	<a href="#">IMDS 登入資料提供者</a>
endpoint_discovery_enabled	<a href="#">端點探索</a>
endpoint_url	<a href="#">服務特定的端點</a>
external_id	<a href="#">擔任角色登入資料提供者</a>
ignore_configured_endpoint_urls	<a href="#">服務特定的端點</a>
max_attempts	<a href="#">重試行為</a>
metadata_service_num_attempts	<a href="#">Amazon EC2 執行個體中繼資料</a>
metadata_service_timeout	<a href="#">Amazon EC2 執行個體中繼資料</a>
mfa_serial	<a href="#">擔任角色登入資料提供者</a>
output	<a href="#">一般組態設定</a>
parameter_validation	<a href="#">一般組態設定</a>

設定名稱	詳細資訊
region	<a href="#">AWS 區域</a>
request_checksum_calculation	<a href="#">Amazon S3 的資料完整性保護</a>
request_minimum_compression_size_bytes	<a href="#">請求壓縮</a>
response_checksum_validation	<a href="#">Amazon S3 的資料完整性保護</a>
retry_mode	<a href="#">重試行為</a>
role_arn	<a href="#">擔任角色登入資料提供者</a>
role_session_name	<a href="#">擔任角色登入資料提供者</a>
s3_disable_express_session_auth	<a href="#">S3 Express One Zone 工作階段身分驗證</a>
s3_disable_multiregion_access_points	<a href="#">Amazon S3 多區域存取點</a>
s3_use_arn_region	<a href="#">Amazon S3 存取點</a>
sdk_ua_app_id	<a href="#">應用程式 ID</a>

設定名稱	詳細資訊
sigv4a_signing_region_set	<a href="#">身分驗證機制</a>
source_profile	<a href="#">擔任角色登入資料提供者</a>
sso_account_id	<a href="#">IAM Identity Center 憑證提供者</a>
sso_region	<a href="#">IAM Identity Center 憑證提供者</a>
sso_registration_scopes	<a href="#">IAM Identity Center 憑證提供者</a>
sso_role_name	<a href="#">IAM Identity Center 憑證提供者</a>
sso_start_url	<a href="#">IAM Identity Center 憑證提供者</a>
sts_regional_endpoints	<a href="#">AWS STS 區域端點</a>
use_dualstack_endpoint	<a href="#">雙堆疊和 FIPS 端點</a>
use_fips_endpoint	<a href="#">雙堆疊和 FIPS 端點</a>
web_identity_token_file	<a href="#">擔任角色登入資料提供者</a>

## Credentials 檔案設定清單

下表列出的設定可以在共用 AWS credentials 檔案中指派。它們是全域的，會影響所有 AWS 服務。SDKs 和工具也可能支援唯一的設定和環境變數。若要查看僅個別 SDK 或工具支援的設定和環境變數，請參閱該特定 SDK 或工具指南。

設定名稱	詳細資訊
aws_access_key_id	<a href="#">AWS 存取金鑰</a>
aws_secret_access_key	<a href="#">AWS 存取金鑰</a>
aws_session_token	<a href="#">AWS 存取金鑰</a>

## 環境變數清單

下表列出大多數 SDKs支援的環境變數。它們是全域的，會影響所有人 AWS 服務。SDKs和工具也可能支援唯一的設定和環境變數。若要查看僅個別 SDK 或工具支援的設定和環境變數，請參閱該特定 SDK 或工具指南。

設定名稱	詳細資訊
AWS_ACCESS_KEY_ID	<a href="#">AWS 存取金鑰</a>
AWS_ACCOUNT_ID	<a href="#">帳戶型端點</a>
AWS_ACCOUNT_ID_ENDPOINT_MODE	<a href="#">帳戶型端點</a>
AWS_AUTH_SCHEME_PREFERENCE	<a href="#">身分驗證機制</a>
AWS_CA_BUNDLE	<a href="#">一般組態設定</a>
AWS_CONFIG_FILE	<a href="#">尋找和變更共用 config 和 AWS SDKs 和工具credentials 檔案的位置</a>

設定名稱	詳細資訊
AWS_CONTAINER_AUTHORIZATION_TOKEN	<a href="#">容器憑證提供者</a>
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE	<a href="#">容器憑證提供者</a>
AWS_CONTAINER_CREDENTIALS_FULL_URI	<a href="#">容器憑證提供者</a>
AWS_CONTAINER_CREDENTIALS_RELATIVE_URI	<a href="#">容器憑證提供者</a>
AWS_DEFAULTS_MODE	<a href="#">智慧組態預設值</a>
AWS_DISABLE_HOST_PREFIX_INJECTION	<a href="#">主機字首注入</a>
AWS_DISABLE_REQUEST_COMPRESSION	<a href="#">請求壓縮</a>
AWS_EC2_METADATA_DISABLED	<a href="#">IMDS 登入資料提供者</a>



設定名稱	詳細資訊
AWS_EC2_METADATA_SERVICE_ENDPOINT	<a href="#">IMDS 登入資料提供者</a>
AWS_EC2_METADATA_ENDPOINT_MODE	<a href="#">IMDS 登入資料提供者</a>
AWS_EC2_METADATA_V1_DISABLED	<a href="#">IMDS 登入資料提供者</a>
AWS_ENABLE_ENDPOINT_DISCOVERY	<a href="#">端點探索</a>
AWS_ENDPOINT_URL	<a href="#">服務特定的端點</a>
AWS_ENDPOINT_URL_<SERVICE>	<a href="#">服務特定的端點</a>
AWS_IGNORE_ENDPOINT_URLS	<a href="#">服務特定的端點</a>
AWS_MAX_ATTEMPTS	<a href="#">重試行為</a>

設定名稱	詳細資訊
AWS_METAD ATA_SERVI CE_NUM_AT TEMPTS	<a href="#">Amazon EC2 執行個體中繼資料</a>
AWS_METAD ATA_SERVI CE_TIMEOUT	<a href="#">Amazon EC2 執行個體中繼資料</a>
AWS_PROFILE	<a href="#">使用共用 config和 credentials 檔案來全域設定 AWS SDKs和工具</a>
AWS_REGION	<a href="#">AWS 區域</a>
AWS_REQUE ST_CHECKS UM_CALCULATION	<a href="#">Amazon S3 的資料完整性保護</a>
AWS_REQUE ST_MIN_CO MPRESSION _SIZE_BYTES	<a href="#">請求壓縮</a>
AWS_RESPO NSE_CHECK SUM_VALIDATION	<a href="#">Amazon S3 的資料完整性保護</a>
AWS_RETRY_MODE	<a href="#">重試行為</a>
AWS_ROLE_ARN	<a href="#">擔任角色登入資料提供者</a>
AWS_ROLE_ SESSION_NAME	<a href="#">擔任角色登入資料提供者</a>

設定名稱	詳細資訊
AWS_S3_DISABLE_EXPRESS_SESSION_AUTH	<a href="#">S3 Express One Zone 工作階段身分驗證</a>
AWS_S3_DISABLE_MULTIREGION_ACCESS_POINTS	<a href="#">Amazon S3 多區域存取點</a>
AWS_S3_US_E_ARN_REGION	<a href="#">Amazon S3 存取點</a>
AWS_SDK_UA_APP_ID	<a href="#">應用程式 ID</a>
AWS_SECRET_ACCESS_KEY	<a href="#">AWS 存取金鑰</a>
AWS_SESSION_TOKEN	<a href="#">AWS 存取金鑰</a>
AWS_SHARED_CREDENTIALS_FILE	<a href="#">尋找和變更 AWS SDKs config 和工具的共用和credentials 檔案的位置</a>
AWS_SIGV4_A_SIGNING_REGION_SET	<a href="#">身分驗證機制</a>
AWS_STS_REGIONAL_ENDPOINTS	<a href="#">AWS STS 區域端點</a>
AWS_USE_DUALSTACK_ENDPOINT	<a href="#">雙堆疊和 FIPS 端點</a>

設定名稱	詳細資訊
AWS_USE_FIPS_ENDPOINT	<a href="#">雙堆疊和 FIPS 端點</a>
AWS_WEB_IDENTITY_TOKEN_FILE	<a href="#">擔任角色登入資料提供者</a>

## JVM 系統屬性清單

您可以針對適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK（以 JVM 為目標）使用下列 JVM 系統屬性。如需如何設定 JVM 系統屬性的說明，請參閱 [the section called “如何設定 JVM 系統屬性”](#)，請參閱。

設定名稱	詳細資訊
aws.accessKeyId	<a href="#">AWS 存取金鑰</a>
aws.accountId	<a href="#">帳戶型端點</a>
aws.accountIdEndpointMode	<a href="#">帳戶型端點</a>
aws.authSchemePreference	<a href="#">身分驗證機制</a>
aws.configFile	<a href="#">尋找和變更共用和 AWS SDKs config和工具credentials 檔案的位置</a>
aws.defaultsMode	<a href="#">智慧組態預設值</a>
aws.disableEc2MetadataV1	<a href="#">IMDS 登入資料提供者</a>

設定名稱	詳細資訊
<code>aws.disableHostPrefixInjection</code>	<a href="#">主機字首注入</a>
<code>aws.disableRequestCompression</code>	<a href="#">請求壓縮</a>
<code>aws.disableS3ExpressAuth</code>	<a href="#">S3 Express One Zone 工作階段身分驗證</a>
<code>aws.ec2MetadataServiceEndpoint</code>	<a href="#">IMDS 登入資料提供者</a>
<code>aws.ec2MetadataServiceEndpointMode</code>	<a href="#">IMDS 登入資料提供者</a>
<code>aws.endpointDiscoveryEnabled</code>	<a href="#">端點探索</a>
<code>aws.endpointUrl</code>	<a href="#">服務特定的端點</a>
<code>aws.endpointUrl&lt;ServiceName&gt;</code>	<a href="#">服務特定的端點</a>
<code>aws.ignoreConfiguredEndpointUrls</code>	<a href="#">服務特定的端點</a>
<code>aws.maxAttempts</code>	<a href="#">重試行為</a>

設定名稱	詳細資訊
<code>aws.profile</code>	<a href="#">使用共用 config和 credentials 檔案來全域設定 AWS SDKs和工具</a>
<code>aws.region</code>	<a href="#">AWS 區域</a>
<code>aws.requestChecksumCalculation</code>	<a href="#">Amazon S3 的資料完整性保護</a>
<code>aws.requestMinCompressionSizeBytes</code>	<a href="#">請求壓縮</a>
<code>aws.responseChecksumValidation</code>	<a href="#">Amazon S3 的資料完整性保護</a>
<code>aws.retryMode</code>	<a href="#">重試行為</a>
<code>aws.roleArn</code>	<a href="#">擔任角色登入資料提供者</a>
<code>aws.roleSessionName</code>	<a href="#">擔任角色登入資料提供者</a>
<code>aws.s3DisableMultiRegionAccessPoints</code>	<a href="#">Amazon S3 多區域存取點</a>
<code>aws.s3UseArnRegion</code>	<a href="#">Amazon S3 存取點</a>
<code>aws.secretAccessKey</code>	<a href="#">AWS 存取金鑰</a>

設定名稱	詳細資訊
<code>aws.sessionToken</code>	<a href="#">AWS 存取金鑰</a>
<code>aws.sharedCredentialsFile</code>	<a href="#">尋找和變更 AWS SDKs config 和工具的共用和credentials 檔案的位置</a>
<code>aws.useDualstackEndpoint</code>	<a href="#">雙堆疊和 FIPS 端點</a>
<code>aws.useFipsEndpoint</code>	<a href="#">雙堆疊和 FIPS 端點</a>
<code>aws.webIdentityTokenFile</code>	<a href="#">擔任角色登入資料提供者</a>
<code>sdk.ua.appId</code>	<a href="#">應用程式 ID</a>

## AWS SDKs和工具標準化登入資料提供者

許多登入資料提供者已標準化為一致的預設值，並在許多 SDKs 中以相同的方式運作。此一致性可提高跨多個 SDKs 編碼時的生產力和清晰度。所有設定都可以在程式碼中覆寫。如需詳細資訊，請參閱您的特定 SDK API。

### Important

並非所有 SDKs 都支援所有供應商，甚至是供應商內的所有層面。

### 主題

- [了解登入資料提供者鏈結](#)
- [SDK 特定和工具特定登入資料提供者鏈結](#)
- [AWS 存取金鑰](#)

- [登入憑證提供者](#)
- [擔任角色登入資料提供者](#)
- [容器憑證提供者](#)
- [IAM Identity Center 憑證提供者](#)
- [IMDS 登入資料提供者](#)
- [程序登入資料提供者](#)

## 了解登入資料提供者鏈結

所有 SDKs 都有一系列位置（或來源）可供他們檢查，以尋找可用於向提出請求的有效登入資料 AWS 服務。找到有效的憑證後，系統就會停止搜尋。此系統搜尋稱為登入資料提供者鏈結。

使用其中一個標準化憑證提供者時，AWS SDKs 一律會在憑證過期時嘗試自動續約憑證。內建的登入資料提供者鏈結可讓您的應用程式重新整理登入資料，無論您在鏈結中使用哪個提供者。不需要額外的程式碼，軟體開發套件即可執行此操作。

雖然每個 SDK 使用的不同鏈結各不相同，但它們通常包含下列來源：

登入資料提供者	Description
<a href="#">AWS 存取金鑰</a>	AWS IAM 使用者的存取金鑰（例如 <code>AWS_ACCESS_KEY_ID</code> 、和 <code>AWS_SECRET_ACCESS_KEY</code> ）。
<a href="#">與 Web 身分或 OpenID Connect 聯合 - 擔任角色登入資料提供者</a>	使用知名的外部身分提供者 (IdP) 登入，例如 Login with Amazon、Facebook、Google 或任何其他與 OpenID Connect (OIDC) 相容的 IdP。使用來自 AWS Security Token Service () 的 JSON Web Token (JWT) 來擔任 IAM 角色的許可AWS STS。
<a href="#">登入憑證提供者</a>	取得您登入的新或現有主控台工作階段的登入資料。
<a href="#">IAM Identity Center 憑證提供者</a>	從 取得登入資料 AWS IAM Identity Center。
<a href="#">擔任角色登入資料提供者</a>	擔任 IAM 角色的許可，以取得其他 資源的存取權。（擷取並使用角色的臨時登入資料）。
<a href="#">容器憑證提供者</a>	Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 登入



登入資料提供者	Description
	資料。容器憑證提供者會為客戶的容器化應用程式擷取憑證。
<a href="#">程序登入資料提供者</a>	自訂登入資料提供者。從外部來源或程序取得您的登入資料，包括 IAM Roles Anywhere。
<a href="#">IMDS 登入資料提供者</a>	Amazon Elastic Compute Cloud (Amazon EC2) 執行個體設定檔憑證。將 IAM 角色與每個 EC2 執行個體建立關聯。該角色的臨時登入資料可供執行個體中執行的程式碼使用。憑證是透過 Amazon EC2 中繼資料服務傳遞。

對於鏈結中的每個步驟，有多種方式可以指派設定值。設定在程式碼中指定的值一律優先。不過，也有[環境變數](#)和[使用共用 config和 credentials 檔案來全域設定 AWS SDKs和工具](#)。如需詳細資訊，請參閱[設定的優先順序](#)。

## SDK 特定和工具特定登入資料提供者鏈結

若要直接前往軟體開發套件或工具的特定登入資料提供者鏈結詳細資訊，請從下列內容中選擇軟體開發套件或工具：

- [AWS CLI](#)
- [適用於 C++ 的 SDK](#)
- [適用於 Go 的 SDK](#)
- [適用於 Java 的開發套件](#)
- [適用於 JavaScript 的 SDK](#)
- [適用於 Kotlin 的 SDK](#)
- [適用於 .NET 的 SDK](#)
- [適用於 PHP 的 SDK](#)
- [適用於 Python 的 SDK \(Boto3\)](#)
- [適用於 Ruby 的 SDK](#)
- [適用於 Rust 的 SDK](#)
- [適用於 Swift 的 SDK](#)
- [PowerShell 的工具](#)

## AWS 存取金鑰

### Warning

為避免安全風險，在開發專用軟體或使用真實資料時，請勿使用 IAM 使用者進行身分驗證。相反地，搭配使用聯合功能和身分提供者，例如 [AWS IAM Identity Center](#)。

AWS IAM 使用者的存取金鑰可以用作您的 AWS 登入資料。AWS 開發套件會自動使用這些 AWS 登入資料來簽署 API 請求 AWS，讓您的工作負載可以安全方便地存取您的 AWS 資源和資料。建議一律使用 `aws_session_token` 讓登入資料暫時且過期後不再有效。不建議使用長期登入資料。

### Note

如果 AWS 無法重新整理這些臨時登入資料，AWS 可能會延長登入資料的有效性，讓您的工作負載不受影響。

共用 AWS `credentials` 檔案是存放登入資料資訊的建議位置，因為它位於應用程式來源目錄外，並與共用 `config` 檔案的 SDK 特定設定分開。

若要進一步了解 AWS 登入資料和使用存取金鑰，請參閱《IAM 使用者指南》中的 [AWS 安全登入資料和管理 IAM 使用者的存取金鑰](#)。 [https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_credentials\\_access-keys.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_access-keys.html)

使用下列項目設定此功能：

**`aws_access_key_id`** - 共用 AWS `config` 檔案設定, **`aws_access_key_id`** - 共用 AWS `credentials` 檔案設定（建議的方法），**`AWS_ACCESS_KEY_ID`** - 環境變數, **`aws.accessKeyId`** - JVM 系統屬性：僅限 Java/Kotlin

指定用作登入資料一部分的 AWS 存取金鑰，以驗證使用者。

**`aws_secret_access_key`** - 共用 AWS `config` 檔案設定, **`aws_secret_access_key`** - 共用 AWS `credentials` 檔案設定（建議的方法），**`AWS_SECRET_ACCESS_KEY`** - 環境變數, **`aws.secretAccessKey`** - JVM 系統屬性：僅限 Java/Kotlin

指定用作登入資料一部分以驗證使用者的 AWS 私密金鑰。

**aws\_session\_token** - 共用 **AWS config** 檔案設定, **aws\_session\_token** - 共用 **AWS credentials** 檔案設定 ( 建議的方法 ), **AWS\_SESSION\_TOKEN** - 環境變數, **aws.sessionToken** - JVM 系統屬性 : 僅限 Java/Kotlin

指定做為登入資料一部分的 AWS 工作階段字符, 以驗證使用者。您會收到此值, 這是成功請求擔任角色所傳回暫時登入資料的一部分。只有當您手動指定臨時的安全憑證時, 才需要工作階段字符。不過, 我們建議您一律使用臨時安全登入資料, 而非長期登入資料。如需安全建議, 請參閱 [IAM 中的安全最佳實務](#)。

如需如何取得這些值的說明, 請參閱 [使用短期登入資料來驗證 AWS SDKs和工具](#)。

在 **config** 或 **credentials** 檔案中設定這些必要值的範例 :

```
[default]
aws_access_key_id = AKIAIOSFODNN7EXAMPLE
aws_secret_access_key = wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
aws_session_token = AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40lgk
```

透過命令列設定環境變數的 Linux/macOS 範例 :

```
export AWS_ACCESS_KEY_ID=AKIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
export
AWS_SESSION_TOKEN=AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40lgk
```

透過命令列設定環境變數的 Windows 範例 :

```
setx AWS_ACCESS_KEY_ID AKIAIOSFODNN7EXAMPLE
setx AWS_SECRET_ACCESS_KEY wJalrXUtnFEMI/K7MDENG/bPxrFiCYEXAMPLEKEY
setx
AWS_SESSION_TOKEN AQoEXAMPLEH4aoAH0gNCAPy...truncated...zrkuWJ0gQs8IZZaIv2BXIa2R40lgk
```

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	不支援共用config檔案。
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	不支援環境變數。

## 登入憑證提供者

您可以使用[現有的 AWS 管理主控台登入憑證](#)，取得可用於程式設計存取的短期憑證。完成瀏覽器身分驗證流程後，AWS 會產生臨時登入資料，可用於 CLI、AWS Tools for PowerShell AWS 和 AWS SDKs 等本機開發工具。

若要產生這些登入資料，請在 CLI AWS 中執行 `aws login` 命令，或在 AWS Tools for PowerShell 中執行 `Invoke-AWSLogin cmdlet`。產生的短期登入資料將在本機快取，AWS SDKs 可在其中重複使用。短期登入資料會在 15 分鐘內過期，但 CLI 和 SDKs 將視需要自動重新整理它們，最長可達 12 小時。當重新整理字符過期時，系統會提示您透過 CLI 或 PowerShell 再次登入。

登入命令將使用 `login_session` 設定更新您指定的設定檔，該設定會存放您在登入工作流程期間選取的管理主控台工作階段的身分。

```
[profile console]
login_session = arn:aws:iam::0123456789012:user/username
region = us-west-2
```

根據預設，短期憑證和重新整理字符存放在 Linux 和 macOS 或 `%USERPROFILE%\.aws\login\cache` Windows 的 `~/.aws/login/cache` 目錄中的 JSON 檔案中。檔案名稱是以登入工作階段名稱為基礎。您可以設定 `AWS_LOGIN_CACHE_DIRECTORY` 環境變數來覆寫目錄。

## 登入提供者設定

使用下列項目設定此功能：

### **AWS\_LOGIN\_CACHE\_DIRECTORY** - 環境變數

CLI 和 SDKs 將存放對應至登入工作階段設定檔之快取登入資料的替代目錄。

預設值：`~/.aws/login/cache` 在 Linux 和 macOS 上，或在 Windows `%USERPROFILE%\.aws\login\cache` 上。

## 支援 AWS SDKs 和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	否	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	需要 CRT
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	否	

## 擔任角色登入資料提供者

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

假設角色涉及使用一組臨時安全登入資料來存取 AWS 您可能無法存取的資源。這些臨時登入資料由存取金鑰 ID、私密存取金鑰和安全字符組成。

若要設定軟體開發套件或工具以擔任角色，您必須先建立或識別要擔任的特定角色。IAM 角色由角色 Amazon Resource Name ([ARN](#)) 唯一識別。角色會與其他實體建立信任關係。使用角色的信任實體可能是 AWS 服務、另一個 AWS 帳戶、Web 身分提供者或 OIDC 或 SAML 聯合。

識別 IAM 角色之後，如果您受該角色信任，您可以將 SDK 或工具設定為使用角色授予的許可。若要這樣做，請使用下列設定。

如需開始使用這些設定的指引，請參閱本指南[使用 AWS 登入資料來擔任角色以驗證 AWS SDKs和工具](#)中的。

### 擔任角色登入資料提供者設定

使用下列項目設定此功能：

#### **credential\_source** - 共用 AWS **config**檔案設定

在 Amazon EC2 執行個體或 Amazon Elastic Container Service 容器內使用，以指定 SDK 或工具可以尋找登入資料的位置，這些登入資料具有使用 `role_arn` 參數擔任您指定之角色的許可。

預設值：無

有效值：

- 環境 – 指定 SDK 或工具是從環境變數 [AWS\\_ACCESS\\_KEY\\_ID](#)和 [AWS\\_SECRET\\_ACCESS\\_KEY](#)擷取來源憑證。
- `Ec2InstanceMetadata` – 指定 SDK 或工具使用[連接到 EC2 執行個體描述檔的 IAM 角色](#)來取得來源憑證。
- `EcsContainer` – 指定 SDK 或工具使用[連接至 Amazon ECS 容器的 IAM 角色](#)或[連接至 Amazon EKS 容器的 IAM 角色](#)來取得來源憑證。

您無法在同一個描述檔中同時指定 `credential_source` 和 `source_profile`。

在 `config` 檔案中設定此項目以表示登入資料應來自 Amazon EC2 的範例：

```
credential_source = Ec2InstanceMetadata
role_arn = arn:aws:iam::123456789012:role/my-role-name
```

### **duration\_seconds** - 共用 AWS `config` 檔案設定

指定角色工作階段的最大持續時間 (以秒為單位)。

此設定只有在設定檔指定 擔任角色時才適用。

預設值：3600 秒 (一小時)

有效值：值的範圍可以從 900 秒 (15 分鐘) 到為角色設定的工作階段持續時間上限 (最多可達 43200 秒或 12 小時)。如需詳細資訊，請參閱 [《IAM 使用者指南》中的檢視角色的工作階段持續時間上限設定](#)。

在 `config` 檔案中設定的範例：

```
duration_seconds = 43200
```

### **external\_id** - 共用 AWS `config` 檔案設定

指定一個唯一識別符，第三方用來在其客戶帳戶擔任角色。

此設定僅適用於設定檔指定 擔任角色，且角色的信任政策需要的值時 `ExternalId`。當設定檔指定角色時，值會映射至傳遞至 `AssumeRole` 操作的 `ExternalId` 參數。

預設值：無。

有效值：請參閱《IAM 使用者指南》中的 [如何在將 AWS 資源的存取權授予第三方時使用外部 ID](#)。

在 `config` 檔案中設定的範例：

```
external_id = unique_value_assigned_by_3rd_party
```

### **mfa\_serial** - 共用 AWS `config` 檔案設定

指定使用者擔任角色時必須使用的多重驗證 (MFA) 裝置的識別或序號。



擔任角色時為必要，其中該角色的信任政策包含需要 MFA 身分驗證的條件。如需 MFA 的詳細資訊，請參閱 [《AWS IAM 使用者指南》](#) 中的 [IAM 中的多重要素驗證](#)。

預設值：無。

有效值：該值可以是硬體裝置的序號（例如 GAHT12345678），也可以是虛擬 MFA 裝置的 Amazon Resource Name (ARN)。ARN 的格式為：`arn:aws:iam::account-id:mfa/mfa-device-name`

在 config 檔案中設定的範例：

此範例假設已為帳戶 MyMFADevice 建立並為使用者啟用的虛擬 MFA 裝置，稱為。

```
mfa_serial = arn:aws:iam::123456789012:mfa/MyMFADevice
```

**role\_arn** - 共用 AWS config 檔案設定, **AWS\_ROLE\_ARN** - 環境變數, **aws.roleArn** - JVM 系統屬性：僅限 Java/Kotlin

指定您要用來執行使用此設定檔請求之操作的 IAM 角色的 Amazon Resource Name (ARN)。

預設值：無。

有效值：該值必須是 IAM 角色的 ARN，格式如下：`arn:aws:iam::account-id:role/role-name`

此外，您還必須指定下列其中一項設定：

- **source\_profile** – 識別另一個設定檔，用來尋找具有在此設定檔中擔任角色之許可的登入資料。
- **credential\_source** – 使用目前環境變數識別的登入資料，或連接至 Amazon EC2 執行個體描述檔的登入資料，或 Amazon ECS 容器執行個體。
- **web\_identity\_token\_file** – 針對已在行動或 Web 應用程式中驗證的使用者，使用公有身分提供者或任何 OpenID Connect (OIDC) 相容身分提供者。

**role\_session\_name** - 共用 AWS config 檔案設定, **AWS\_ROLE\_SESSION\_NAME** - 環境變數, **aws.roleSessionName** - JVM 系統屬性：僅限 Java/Kotlin

指定要連接到角色工作階段的名稱。此名稱會出現在與此工作階段相關聯的項目 AWS CloudTrail 日誌中，這在稽核時非常有用。如需詳細資訊，請參閱 AWS CloudTrail [《使用者指南》](#) 中的 [CloudTrail userIdentity 元素](#)。

預設值：選用參數。如果您未提供此值，則在設定檔擔任角色時，工作階段名稱會自動產生。

有效值：當 AWS CLI 或 AWS API 代表您呼叫 AssumeRole 操作（或操作等 AssumeRoleWithWebIdentity 操作）時，提供給 RoleSessionName 參數。此值會成為您可以查詢的擔任角色使用者 Amazon Resource Name (ARN) 的一部分，並顯示為此設定檔所調用之操作的 CloudTrail 日誌項目的一部分。

```
arn:aws:sts::123456789012:assumed-role/my-role-name/my-role_session_name.
```

在 config 檔案中設定的範例：

```
role_session_name = my-role-session-name
```

### source\_profile - 共用 AWS config 檔案設定

指定另一個設定檔，其登入資料用於擔任原始設定檔中 role\_arn 設定所指定的角色。若要了解如何在共用 AWS config 和 credentials 檔案中使用設定檔，請參閱 [共用 config 和 credentials 檔案](#)。

如果您指定的設定檔也是擔任角色設定檔，則每個角色都會依序擔任，以完全解析登入資料。當 SDK 遇到具有登入資料的設定檔時，此鏈結會停止。角色鏈結會將您的 AWS CLI 或 AWS API 角色工作階段限制為最多一小時，且無法增加。如需詳細資訊，請參閱《IAM 使用者指南》中的 [角色術語和概念](#)。

預設值：無。

有效值：文字字串，由 config 和 credentials 檔案中定義的設定檔名稱組成。您還必須在目前的設定檔 role\_arn 中指定的值。

您無法在同一個描述檔中同時指定 credential\_source 和 source\_profile。

在組態檔案中設定的範例：

```
[profile A]
source_profile = B
role_arn = arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_process = ./aws_signing_helper credential-process --certificate /
path/to/certificate --private-key /path/to/private-key --trust-anchor-
arn arn:aws:rolesanywhere:region:account:trust-anchor/TA_ID --profile-
arn arn:aws:rolesanywhere:region:account:profile/PROFILE_ID --role-arn
arn:aws:iam::account:role/ROLE_ID
```

在先前的範例中，A設定檔會告知 SDK 或工具自動查詢連結B設定檔的登入資料。在此情況下，B設定檔會使用提供的登入資料協助程式工具[使用 IAM Roles Anywhere 驗證 AWS SDKs和工具](#)來取得 SDK 的 AWS 登入資料。然後，您的程式碼會使用這些臨時登入資料來存取 AWS 資源。指定的角色必須已連接允許請求程式碼執行的 IAM 許可政策，例如命令 AWS 服務或 API 方法。設定檔採取的每個動作A都會包含 CloudTrail 日誌中的角色工作階段名稱。

對於角色鏈結的第二個範例，如果您在 Amazon Elastic Compute Cloud 執行個體上有應用程式，而且您想要讓該應用程式擔任另一個角色，則可以使用下列組態。

```
[profile A]
source_profile = B
role_arn = arn:aws:iam::123456789012:role/RoleA
role_session_name = ProfileARoleSession

[profile B]
credential_source=Ec2InstanceMetadata
```

設定檔A將使用來自 Amazon EC2 執行個體的登入資料擔任指定的角色，並自動續約登入資料。

**web\_identity\_token\_file** - 共用 AWS config檔案設定, **AWS\_WEB\_IDENTITY\_TOKEN\_FILE** - 環境變數, **aws.webIdentityTokenFile** - JVM 系統屬性：僅限 Java/Kotlin

指定檔案的路徑，其中包含來自[支援 OAuth 2.0 提供者](#)或[OpenID Connect ID 身分提供者](#)的存取權杖。

此設定可透過使用 [Google](#)、[Facebook](#) 和 [Amazon](#) 等 Web 聯合身分提供者來啟用身分驗證。SDK 或開發人員工具會載入此檔案的內容，並在代您呼叫AssumeRoleWithWebIdentity操作時將其做為WebIdentityToken引數傳遞。

預設值：無。

有效值：此值必須是路徑和檔案名稱。檔案必須包含身分提供者提供給您的 OAuth 2.0 存取字符或 OpenID Connect 字符。相對路徑視為相對於程序的工作目錄。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	部分	credential_source 不支援。duration_seconds 不支援。mfa_serial 不支援。
<a href="#">適用於 Go 的 SDK V2 (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	部分	mfa_serial 不支援。duration_seconds 不支援。
<a href="#">適用於 Java 的 SDK 1.x</a>	部分	credential_source 不支援。mfa_serial 不支援。不支援 JVM 系統屬性。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	部分	credential_source 不支援。
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	

SDK	支援 備註或更多資訊
<a href="#">PowerShell V5 的工具</a>	是
<a href="#">PowerShell V4 的工具</a>	是

## 容器憑證提供者

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

容器憑證提供者會為客戶的容器化應用程式擷取憑證。此登入資料提供者適用於 Amazon Elastic Container Service (Amazon ECS) 和 Amazon Elastic Kubernetes Service (Amazon EKS) 客戶。SDKs會嘗試透過 GET 請求從指定的 HTTP 端點載入登入資料。

如果您使用 Amazon ECS，我們建議您使用任務 IAM 角色來改善憑證隔離、授權和可稽核性。設定後，Amazon ECS 會設定軟體SDKs和工具用來取得登入資料AWS\_CONTAINER\_CREDENTIALS\_RELATIVE\_URI的環境變數。若要為此功能設定 Amazon ECS，請參閱《Amazon Elastic Container Service 開發人員指南》中的[任務 IAM 角色](#)。

如果您使用 Amazon EKS，我們建議您使用 Amazon EKS Pod Identity 來改善憑證隔離、最低權限、可稽核性、獨立操作、可重複使用性和可擴展性。您的 Pod 和 IAM 角色都與 Kubernetes 服務帳戶相關聯，以管理應用程式的登入資料。若要進一步了解 Amazon EKS Pod 身分，請參閱《[Amazon EKS 使用者指南](#)》中的[Amazon EKS Pod 身分](#)。設定後，Amazon EKS 會設定開發套件AWS\_CONTAINER\_CREDENTIALS\_FULL\_URI和工具用來取得憑證SDKs和AWS\_CONTAINER\_AUTHORIZATION\_TOKEN\_FILE環境變數。如需設定資訊，請參閱《[Amazon EKS 使用者指南](#)》中的設定 [Amazon EKS Pod Identity Agent](#) 或 [Amazon EKS Pod Identity 可簡化部落格網站上的 Amazon EKS 叢集應用程式 IAM 許可](#)。AWS

使用下列項目設定此功能：

### AWS\_CONTAINER\_CREDENTIALS\_FULL\_URI - 環境變數

指定 SDK 在請求登入資料時要使用的完整 HTTP URL 端點。其中包含通訊協定與主機。

預設值：無。

有效值：有效 URI。

注意：此設定是 的替代方案 `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`，只有在 `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` 未設定 時才會使用。

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credentials
```

或

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost:8080/get-credentials
```

### **AWS\_CONTAINER\_CREDENTIALS\_RELATIVE\_URI** - 環境變數

指定 SDK 在請求登入資料時要使用的相對 HTTP URL 端點。此值會附加到 的預設 Amazon ECS 主機名稱 `169.254.170.2`。

預設值：無。

有效值：有效的相對 URI。

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_CONTAINER_CREDENTIALS_RELATIVE_URI=/get-credentials?a=1
```

### **AWS\_CONTAINER\_AUTHORIZATION\_TOKEN** - 環境變數

以純文字指定授權字符。如果設定此變數，軟體開發套件會在 HTTP 請求上設定具有環境變數值的授權標頭。

預設值：無。

有效值：字串。

注意：此設定是 的替代方案 `AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE`，只有在 `AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE` 未設定 時才會使用。

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN=Basic abcd
```

## AWS\_CONTAINER\_AUTHORIZATION\_TOKEN\_FILE - 環境變數

指定檔案的絕對檔案路徑，其中包含純文字的授權字符。

預設值：無。

有效值：字串。

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_CONTAINER_CREDENTIALS_FULL_URI=http://localhost/get-credential
export AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE=/path/to/token
```

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	當 <a href="#">Lambda SnapStart</a> 啟用時，AWS_CONTAINER_CREDENTIALS_FULL_URI AWS_CONTAINER_AUTHORIZATION_TOKEN 會自動用於身分驗證。
<a href="#">適用於 Java 的 SDK 1.x</a>	是	當 <a href="#">Lambda SnapStart</a> 啟用時，AWS_CONTAINER_CREDENTIALS_FULL_URI AWS_CONTAINER_AUTHORIZATION_TOKEN 會自動用於身分驗證。

SDK	支援	備註或更多資訊
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	當 <a href="#">Lambda SnapStart</a> 啟用時，AWS_CONTAINER_CREDENTIALS_FULL_URI、AWS_CONTAINER_AUTHORIZATION_TOKEN 會自動用於身分驗證。
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	當 <a href="#">Lambda SnapStart</a> 啟用時，AWS_CONTAINER_CREDENTIALS_FULL_URI、AWS_CONTAINER_AUTHORIZATION_TOKEN 會自動用於身分驗證。
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	當 <a href="#">Lambda SnapStart</a> 啟用時，AWS_CONTAINER_CREDENTIALS_FULL_URI、AWS_CONTAINER_AUTHORIZATION_TOKEN 會自動用於身分驗證。
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## IAM Identity Center 憑證提供者

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。



此身分驗證機制使用 AWS IAM Identity Center 為您的程式碼取得 AWS 服務的單一登入 (SSO) 存取權。

#### Note

在 AWS SDK API 文件中，IAM Identity Center 憑證提供者稱為 SSO 憑證提供者。

啟用 IAM Identity Center 之後，您可以為共用 AWS config 檔案中的設定定義設定檔。此設定檔用於連線至 IAM Identity Center 存取入口網站。當使用者成功向 IAM Identity Center 進行身分驗證時，入口網站會傳回與該使用者相關聯之 IAM 角色的短期憑證。若要了解 SDK 如何從組態取得臨時登入資料並將其用於 AWS 服務 請求，請參閱 [如何解決 AWS SDKs IAM Identity Center 身分驗證](#)。

透過 config 檔案設定 IAM Identity Center 的方式有兩種：

- (建議) SSO 權杖提供者組態 – 延長工作階段持續時間。包括對自訂工作階段持續時間的支援。
- 傳統不可重新整理組態 – 使用固定的八小時工作階段。

在這兩種組態中，您需要在工作階段過期時再次登入。

下列兩個指南包含有關 IAM Identity Center 的其他資訊：

- [AWS IAM Identity Center 使用者指南](#)
- [AWS IAM Identity Center 入口網站 API 參考](#)

如需 SDKs 和工具如何使用和使用此組態重新整理登入資料的深入探討，請參閱 [如何解決 AWS SDKs IAM Identity Center 身分驗證](#)。

## 先決條件

您必須先啟用 IAM Identity Center。如需啟用 IAM Identity Center 身分驗證的詳細資訊，請參閱 AWS IAM Identity Center 《使用者指南》中的 [啟用 AWS IAM Identity Center](#)。

#### Note

或者，如需此頁面詳細說明的完整先決條件和必要的共用 config 檔案組態，請參閱設定的引導說明 [使用 IAM Identity Center 驗證 AWS SDK 和工具](#)。

## SSO 權杖提供者組態

當您使用 SSO 權杖提供者組態時，軟體 AWS 開發套件或工具會自動重新整理工作階段，直到延長工作階段期間為止。如需工作階段持續時間和最長持續時間的詳細資訊，請參閱AWS IAM Identity Center 《使用者指南》中的[設定 AWS 存取入口網站和 IAM Identity Center 整合應用程式的工作階段持續時間](#)。

config 檔案的 sso-session區段用於將用於取得 SSO 存取權杖的組態變數分組，然後可用於取得 AWS 登入資料。如需 config 檔案內本節的詳細資訊，請參閱 [組態檔案的格式](#)。

下列共用config檔案範例使用dev設定檔來設定 SDK 或工具，以請求 IAM Identity Center 登入資料。

```
[profile dev]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

上述範例顯示您定義 sso-session區段，並將其與設定檔建立關聯。一般而言，sso\_account\_id和 sso\_role\_name 必須在 profile區段中設定，以便 SDK 可以請求 AWS 登入資料。sso\_region、sso\_start\_url和 sso\_registration\_scopes 必須在 sso-session區段中設定。

sso\_account\_id 和 sso\_role\_name 並非 SSO 字符組態的所有案例都需要。如果您的應用程式只使用 AWS 服務 支援承載身分驗證，則不需要傳統 AWS 登入資料。承載身分驗證是一種 HTTP 身分驗證結構描述，使用稱為承載字符的安全字符。在這種情況下，sso\_account\_id 和 sso\_role\_name 並非必要資訊。請參閱個別 AWS 服務 指南，判斷服務是否支援承載字符授權。

註冊範圍會設定為 的一部分sso-session。範圍是 中的一種機制OAuth 2.0，用於限制應用程式對使用者帳戶的存取。先前的範例會設定 sso\_registration\_scopes以提供列出帳戶和角色的必要存取權。

下列範例示範如何在多個設定檔中重複使用相同的sso-session組態。

```
[profile dev]
```

```
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole

[profile prod]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole2

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_registration_scopes = sso:account:access
```

身分驗證字符會快取至~/.aws/sso/cache目錄下的磁碟，並以工作階段名稱為基礎的檔案名稱。

## 舊版不可重新整理的配置

使用舊版不可重新整理的組態，不支援自動字符重新整理。我們建議您[SSO 權杖提供者組態](#)改用。

若要使用舊版不可重新整理組態，您必須在設定檔中指定下列設定：

- sso\_start\_url
- sso\_region
- sso\_account\_id
- sso\_role\_name

您可以使用 sso\_start\_url 和 sso\_region 設定來指定設定檔的使用者入口網站。您可以使用 sso\_account\_id 和 sso\_role\_name 設定指定許可。

下列範例會設定 config 檔案中的四個必要值。

```
[profile my-sso-profile]
sso_start_url = https://my-sso-portal.awsapps.com/start
sso_region = us-west-2
sso_account_id = 111122223333
sso_role_name = SSOReadOnlyRole
```

身分驗證字符會快取至~/.aws/sso/cache目錄下的磁碟，並以 `sso_start_url` 為基礎的檔案名稱。

## IAM Identity Center 憑證提供者設定

使用下列項目設定此功能：

### **sso\_start\_url** - 共用 AWS **config**檔案設定

指向您組織的 IAM Identity Center 發行者 URL 或存取入口網站 URL 的 URL。如需詳細資訊，請參閱AWS IAM Identity Center 《使用者指南》中的[使用 AWS 存取入口網站](#)。

若要尋找此值，請開啟 [IAM Identity Center 主控台](#)、檢視儀表板、尋找AWS 存取入口網站 URL。

- 或者，從 2.22.0 版開始 AWS CLI，您可以改為使用AWS 發行者 URL 的值。

### **sso\_region** - 共用 AWS **config**檔案設定

包含 IAM Identity Center 入口網站主機 AWS 區域的，亦即您在啟用 IAM Identity Center 之前選取的區域。這與您的預設 AWS 區域無關，並且可以不同。

如需 AWS 區域及其代碼的完整清單，請參閱《》中的[區域端點](#)Amazon Web Services 一般參考。

若要尋找此值，請開啟 [IAM Identity Center 主控台](#)、檢視儀表板和尋找區域。

### **sso\_account\_id** - 共用 AWS **config**檔案設定

透過 AWS Organizations 服務 AWS 帳戶 新增用於身分驗證之 的數值 ID。

若要查看可用帳戶的清單，請前往 [IAM Identity Center 主控台](#)並開啟AWS 帳戶頁面。您也可以  
在AWS IAM Identity Center 入口網站 API 參考中使用 [ListAccounts](#) API 方法查看可用帳戶的清單。例如，您可以呼叫 AWS CLI 方法 [list-accounts](#)。

### **sso\_role\_name** - 共用 AWS **config**檔案設定

佈建為 IAM 角色的許可集名稱，定義使用者產生的許可。角色必須存在於 AWS 帳戶 指定的  
中sso\_account\_id。使用角色名稱，而非角色 Amazon Resource Name (ARN)。

許可集會連接 IAM 政策和自訂許可政策，並定義使用者對其指派的存取層級 AWS 帳戶。

若要查看每個的可用許可集清單 AWS 帳戶，請前往 [IAM Identity Center 主控台](#)並開啟AWS 帳  
戶頁面。選擇 AWS 帳戶 表格中列出的正確許可集名稱。您也可以  
在AWS IAM Identity Center 入口網站 API 參考中使用 [ListAccountRoles](#) API 方法查看可用許可集的清單。例如，您可以呼叫 AWS  
CLI 方法 [list-account-roles](#)。

### **sso\_registration\_scopes** - 共用 AWS **config**檔案設定

要授權給 的有效範圍字串逗號分隔清單sso-session。應用程式可以請求一個或多個範圍，則核  
發給應用程式的存取權杖僅限於授予的範圍。必須sso:account:access授予 的最小範圍，才能

從 IAM Identity Center 服務取得重新整理權杖。如需可用存取範圍選項的清單，請參閱AWS IAM Identity Center 《使用者指南》中的[存取範圍](#)。

這些範圍定義為已註冊 OIDC 用戶端和用戶端擷取的存取字符進行授權所需請求的許可。範圍授權對 IAM Identity Center 承載字符授權端點的存取。

此設定不適用於舊版不可重新整理組態。使用舊版組態發行的權杖僅限於 `sso:account:access` 隱含範圍。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	<code>credentials</code> 檔案中也支援組態值。
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	部分	僅限舊版不可重新整理組態。
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## IMDS 登入資料提供者

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

Instance Metadata Service (IMDS) 提供執行個體的資料，可用來設定或管理執行中的執行個體。如需可用資料的詳細資訊，請參閱《Amazon EC2 使用者指南》中的[使用執行個體中繼資料](#)。Amazon EC2 提供執行個體可用的本機端點，可將各種位元的資訊提供給執行個體。如果執行個體已連接角色，則可以提供一組對該角色有效的登入資料。SDKs可以使用該端點來解析登入資料，做為其[預設登入資料提供者鏈結](#)的一部分。執行個體中繼資料服務第 2 版 (IMDSv2) 是使用工作階段字符的更安全 IMDS 版本，預設為使用。如果因為不可重試的條件 (HTTP 錯誤代碼 403、404、405) 而失敗，則會使用 IMDSv1 做為備用。

使用下列項目設定此功能：

### **AWS\_EC2\_METADATA\_DISABLED** - 環境變數

是否嘗試使用 Amazon EC2 執行個體中繼資料服務 (IMDS) 來取得登入資料。

預設值：false。


有效值：

- **true** – 請勿使用 IMDS 取得登入資料。
- **false** – 使用 IMDS 取得登入資料。

**ec2\_metadata\_v1\_disabled** - 共用 AWS **config**檔案設定,

**AWS\_EC2\_METADATA\_V1\_DISABLED** - 環境變數, **aws.disableEc2MetadataV1** - JVM 系統屬性：  
僅限 Java/Kotlin

如果 IMDSv1) 做為備用。IMDSv2

 Note

新的 SDKs 不支援 IMDSv1，因此不支援此設定。如需詳細資訊，請參閱資料表 [支援 AWS SDKs和工具](#)。

預設值：false。

有效值：

- **true** – 請勿使用 IMDSv1 做為備用。
- **false** – 使用 IMDSv1 做為備用。

**ec2\_metadata\_service\_endpoint** - 共用 AWS **config**檔案設定,

**AWS\_EC2\_METADATA\_SERVICE\_ENDPOINT** - 環境變數, **aws.ec2MetadataServiceEndpoint** -  
JVM 系統屬性：僅限 Java/Kotlin

IMDS 的端點。此值會覆寫 AWS SDKs和工具將搜尋 Amazon EC2 執行個體中繼資料的預設位置。

預設值：如果 **ec2\_metadata\_service\_endpoint\_mode** 等於 IPv4，則預設端點為 `http://169.254.169.254`。如果 **ec2\_metadata\_service\_endpoint\_mode** 等於 IPv6，則預設端點為 `http://[fd00:ec2::254]`。

有效值：有效 URI。

**ec2\_metadata\_service\_endpoint\_mode** - 共用 AWS **config**檔

案設定, **AWS\_EC2\_METADATA\_SERVICE\_ENDPOINT\_MODE** - 環境變數,

**aws.ec2MetadataServiceEndpointMode** - JVM 系統屬性：僅限 Java/Kotlin

IMDS 的端點模式。

預設值：IPv4。

有效值：IPv4、IPv6。

### Note

IMDS 登入資料提供者是 的一部分[了解登入資料提供者鏈結](#)。不過，只有在此系列中的其他數個提供者之後，才會檢查 IMDS 登入資料提供者。因此，如果您希望程式使用此提供者的登入資料，您必須從組態中移除其他有效的登入資料提供者，或使用不同的設定檔。或者，不要依賴登入資料提供者鏈自動探索哪個提供者傳回有效的登入資料，而是在程式碼中指定 IMDS 登入資料提供者的使用。您可以在建立服務用戶端時直接指定登入資料來源。

## IMDS 登入資料的安全性

根據預設，當 AWS 軟體開發套件未設定有效的登入資料時，軟體開發套件會嘗試使用 Amazon EC2 執行個體中繼資料服務 (IMDS) 來擷取 AWS 角色的登入資料。將此 `AWS_EC2_METADATA_DISABLED` 環境變數設定為 `true`，即可停用此行為。這可防止不必要的網路活動，並增強可能模擬 Amazon EC2 執行個體中繼資料服務之不受信任網路的安全性。

### Note

AWS 使用有效登入資料設定的 SDK 用戶端絕不會使用 IMDS 擷取登入資料，無論這些設定為何。

## 停用 Amazon EC2 IMDS 登入資料

如何設定此環境變數取決於正在使用的作業系統，以及您是否希望變更持續存在。

### Linux 和 macOS

使用 Linux 或 macOS 的客戶可以使用下列命令來設定此環境變數：

```
$ export AWS_EC2_METADATA_DISABLED=true
```

如果您希望此設定在多個 shell 工作階段和系統重新啟動之間持續存在，您可以將上述命令新增至您的 shell 設定檔，例如 `.bash_profile`、`.zsh_profile` 或 `.profile`。



## Windows

使用 Windows 的客戶可以使用下列命令來設定此環境變數：

```
$ set AWS_EC2_METADATA_DISABLED=true
```

如果您希望此設定在多個 Shell 工作階段間持續存在，且系統重新啟動，則可改用下列命令：

```
$ setx AWS_EC2_METADATA_DISABLED=true
```

### Note

setx 命令不會將值套用至目前的 Shell 工作階段，因此您需要重新載入或重新開啟 Shell，變更才會生效。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	部分	JVM 系統屬性：使用 <code>com.amazonaws.sdk.disableEc2MetadataV1</code> 而非 <code>aws.disableEc2MetadataV1</code> ； <code>aws.ec2MetadataServiceEndpoint</code>

SDK	支援	備註或更多資訊
		<code>int aws.ec2MetadataServiceEndpointMode</code> 不支援。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 的 SDK 2.x</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	不使用 IMDSv1 備用。
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	不使用 IMDSv1 備用。
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	您可以使用 在程式碼中明確停用 IMDSv1 後援 <code>[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = \$true</code> 。
<a href="#">PowerShell V4 的工具</a>	是	您可以使用 在程式碼中明確停用 IMDSv1 後援 <code>[Amazon.Util.EC2InstanceMetadata]::EC2MetadataV1Disabled = \$true</code> 。

## 程序登入資料提供者

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

SDKs為自訂使用案例提供擴展憑證提供者鏈的方法。此提供者可用來提供自訂實作，例如從內部部署登入資料存放區擷取登入資料，或與您的內部部署識別提供者整合。

例如，IAM Roles Anywhere 會使用 代表您的應用程式 `credential_process` 取得臨時登入資料。若要 `credential_process` 為此用途設定，請參閱 [使用 IAM Roles Anywhere 驗證 AWS SDKs和工具](#)。

### Note

以下說明從外部程序取得登入資料的方法，如果您在 外部執行軟體，可能會使用 AWS。如果您要在 AWS 運算資源上建置，請使用其他登入資料提供者。如果使用此選項，您應該使用作業系統的安全最佳實務，確保組態檔案盡可能鎖定。確認您的自訂登入資料工具不會將任何秘密資訊寫入 `StdErr`，因為 SDKs和 AWS CLI 可以擷取並記錄這類資訊，因此可能會將其暴露給未經授權的使用者。

使用下列項目設定此功能：

### `credential_process` - 共用 AWS `config` 檔案設定

指定 SDK 或工具代表您執行的外部命令，以產生或擷取要使用的身分驗證憑證。設定會指定軟體開發套件將叫用的程式/命令名稱。當 SDK 調用程序時，會等待程序將 JSON 資料寫入 `stdout`。自訂提供者必須以特定格式傳回資訊。該資訊包含開發套件或工具可用來驗證您的登入資料。

### Note

程序登入資料提供者是 的一部分 [了解登入資料提供者鏈結](#)。不過，程序登入資料提供者只會在此系列中的幾個其他提供者之後進行檢查。因此，如果您希望程式使用此提供者的登入資料，您必須從組態中移除其他有效的登入資料提供者，或使用不同的設定檔。或者，不要依賴登入

資料提供者鍵自動探索哪個提供者傳回有效的登入資料，而是在程式碼中指定使用程序登入資料提供者。您可以在建立服務用戶端時直接指定登入資料來源。

## 指定登入資料程式的路徑

設定的值是一個字串，其中包含 SDK 或開發工具代表您執行之程式的路徑：

- 路徑和檔案名稱只能包含下列字元：A-Z、a-z、0-9、連字號 (-)、底線 (\_)、句點 (.)、正斜線 (/)、反斜線 (\) 和空格。
- 如果路徑或檔案名稱包含空格，完整的路徑和檔案名稱請以雙引號 (「」) 括住。
- 如果參數名稱或參數值包含空格，則該元素請以雙引號 (「」) 括住。僅括住名稱或值，而非對組。
- 請勿在字串中包含任何環境變數。例如，請勿包含 \$HOME 或 %USERPROFILE%。
- 請勿將主資料夾指定為 ~。\* 您必須指定完整路徑或基本檔案名稱。如果有基本檔案名稱，系統會嘗試在 PATH 環境變數指定的資料夾中尋找程式。路徑會根據作業系統而有所不同：

下列範例顯示在 Linux/macOS 的共用 config 檔案中設定 Credential\_process。

```
credential_process = "/path/to/credentials.sh" parameterWithoutSpaces "parameter with spaces"
```

下列範例顯示在 Windows 的共用 config 檔案中設定 Credential\_process。

```
credential_process = "C:\Path\To\credentials.cmd" parameterWithoutSpaces "parameter with spaces"
```

- 可以在專用設定檔中指定：

```
[profile cred_process]  
credential_process = /Users/username/process.sh  
region = us-east-1
```

## 登入資料程式的有效輸出

SDK 會執行設定檔中指定的 命令，然後從標準輸出串流讀取資料。您指定的命令，無論是指令碼或二進位程式，都必須在 上產生符合下列語法 STDOUT 的 JSON 輸出。

```
{
```

```

"Version": 1,
"AccessKeyId": "an AWS access key",
"SecretAccessKey": "your AWS secret access key",
"SessionToken": "the AWS session token for temporary credentials",
"Expiration": "RFC3339 timestamp for when the credentials expire"
}

```

### Note

截至本文編寫時，Version 索引鍵必須設定為 1。這可能隨著結構演進而逐漸遞增。

Expiration 金鑰是 RFC3339 格式的時間戳記。如果Expiration金鑰不存在於工具的輸出中，開發套件會假設登入資料是不會重新整理的長期登入資料。否則，登入資料會被視為臨時登入資料，並在登入資料過期之前重新執行 credential\_process命令來自動重新整理。

### Note

軟體開發套件不會像擔任角色登入資料一樣快取外部程序登入資料。如果需要快取，您必須在外部程序中實作它。

外部程序可能傳回非零傳回碼，以表示擷取憑證時發生錯誤。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支 備註或更多資訊 援
<a href="#">AWS CLI v2</a>	是
<a href="#">適用於 C++ 的 SDK</a>	是
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是 若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。

SDK	支援	備註或更多資訊
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## AWS SDKs和工具標準化功能

許多功能已標準化為一致的預設值，並在許多 SDKs 中以相同的方式運作。此一致性可提高跨多個 SDKs 編碼時的生產力和清晰度。所有設定都可以在程式碼中覆寫，如需詳細資訊，請參閱您的特定 SDK API。

**⚠ Important**

並非所有 SDKs都支援所有功能，甚至支援功能中的所有層面。

**主題**

- [帳戶型端點](#)
- [應用程式 ID](#)
- [Amazon EC2 執行個體中繼資料](#)
- [Amazon S3 存取點](#)
- [Amazon S3 多區域存取點](#)
- [S3 Express One Zone 工作階段身分驗證](#)
- [身分驗證機制](#)
- [AWS 區域](#)
- [AWS STS 區域端點](#)
- [Amazon S3 的資料完整性保護](#)
- [雙堆疊和 FIPS 端點](#)
- [端點探索](#)
- [一般組態設定](#)
- [主機字首注入](#)
- [IMDS 用戶端](#)
- [重試行為](#)
- [請求壓縮](#)
- [服務特定的端點](#)
- [智慧組態預設值](#)

**帳戶型端點****📘 Note**

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

帳戶型端點使用您的 AWS 帳戶 ID 路由支援此功能之服務的請求，有助於確保高效能和可擴展性。當您使用支援以帳戶為基礎的端點的 AWS SDK 和服務時，開發套件用戶端會建構並使用以帳戶為基礎的端點，而不是區域性端點。如果 SDK 用戶端看不到帳戶 ID，用戶端將使用區域端點。以帳戶為基礎的端點採用的形式 `https://<account-id>.ddb.<region>.amazonaws.com`，其中 `<account-id>` 和 `<region>` 是您的 AWS 帳戶 ID 和 AWS 區域。

使用下列項目設定此功能：

**aws\_account\_id** - 共用 AWS **config** 檔案設定, **AWS\_ACCOUNT\_ID** - 環境變數, **aws.accountId** - JVM 系統屬性：僅限 Java/Kotlin

AWS 帳戶 ID。用於以帳戶為基礎的端點路由。AWS 帳戶 ID 的格式類似 111122223333。

帳戶型端點路由可為某些服務提供更好的請求效能。

**account\_id\_endpoint\_mode** - 共用 AWS **config** 檔案設定, **AWS\_ACCOUNT\_ID\_ENDPOINT\_MODE** - 環境變數, **aws.accountIdEndpointMode** - JVM 系統屬性：僅限 Java/Kotlin

此設定用於在必要時關閉以帳戶為基礎的端點路由，並略過以帳戶為基礎的規則。

預設值：preferred

有效值：

- **preferred** – 如果可用，端點應包含帳戶 ID。
- **disabled** – 已解析的端點不包含帳戶 ID。
- **required** – 端點必須包含帳戶 ID。如果帳戶 ID 無法使用，SDK 會擲出錯誤。

## 支援 AWS SDKs 和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	在 SDK 版本中發行	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	2.25.0	
<a href="#">AWS CLI v1</a>	是	1.38.0	



SDK	支援	在 SDK 版本中發行	備註或更多資訊
<a href="#">適用於 C++ 的 SDK</a>	否		
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	1.35.0 版	
<a href="#">適用於 Go 1.x (V1) 的 SDK</a>	否		
<a href="#">適用於 Java 的 SDK 2.x</a>	是	2.28.4 版	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	1.12.771 版	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	v3.656.0	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否		
<a href="#">適用於 Kotlin 的 SDK</a>	是	v1.3.37	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	4.0.0	
<a href="#">適用於 .NET 3.x 的 SDK</a>	否		
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	v3.318.0	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	1.37.0	

SDK	支援	在 SDK 版本中發行	備註或更多資訊
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	1.123.0 版	
<a href="#">適用於 Rust 的 SDK</a>	是	Release-2 025-04-24	
<a href="#">適用於 Swift 的 SDK</a>	是	1.2.0	
<a href="#">PowerShell V5 的工具</a>	否		
<a href="#">PowerShell V4 的工具</a>	否		

## 應用程式 ID

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

單一 AWS 帳戶 可供多個客戶應用程式用來呼叫 AWS 服務。應用程式 ID 可讓客戶識別使用 AWS 帳戶。AWS SDKs和 服務進行一組呼叫的來源應用程式不使用或解譯此值，但 會在客戶通訊中呈現此值。例如，此值可以包含在操作電子郵件或 中 AWS Health 儀板表，以唯一識別哪些應用程式與通知相關聯。

使用下列項目設定此功能：

**sdk\_ua\_app\_id** - 共用 AWS **config**檔案設定, **AWS\_SDK\_UA\_APP\_ID** - 環境變數, **sdk.ua.appId** - JVM 系統屬性：僅限 Java/Kotlin

此設定是您指派給應用程式的唯一字串，用於識別特定 中的哪些應用程式會 AWS 帳戶 呼叫 AWS。

預設值：None

有效值：長度上限為 50 的字串。允許使用字母、數字和下列特殊字元：!、#、\$%、&、'、\*+、-、.、^\_、`、`、`、|~。

在 config 檔案中設定此值的範例：

```
[default]
sdk_ua_app_id=ABCDEF
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_SDK_UA_APP_ID=ABCDEF
export AWS_SDK_UA_APP_ID="ABC DEF"
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_SDK_UA_APP_ID ABCDEF
setx AWS_SDK_UA_APP_ID="ABC DEF"
```

如果您包含對正在使用的 shell 具有特殊意義的符號，請適當地逸出該值。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	不支援共用config檔案。
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	部分	不支援共用config檔案設定；不支援環境變數。

SDK	支援	備註或更多資訊
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	JVM 系統屬性為 <code>aws.userAgentAppId</code> 。
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## Amazon EC2 執行個體中繼資料

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

Amazon EC2 在名為執行個體中繼資料服務 (IMDS) 的執行個體上提供服務。若要進一步了解此服務，請參閱《Amazon EC2 使用者指南》中的[使用執行個體中繼資料](#)。嘗試在已設定 IAM 角色的 Amazon EC2 執行個體上擷取登入資料時，執行個體中繼資料服務的連線是可調整的。

使用下列項目設定此功能：

**metadata\_service\_num\_attempts** - 共用 AWS **config** 檔案設定，  
**AWS\_METADATA\_SERVICE\_NUM\_ATTEMPTS** - 環境變數

此設定指定嘗試從執行個體中繼資料服務擷取資料時，放棄之前要嘗試的總次數。

預設值：1

有效值：大於或等於 1 的數字。

**metadata\_service\_timeout** - 共用 AWS **config** 檔案設定，  
**AWS\_METADATA\_SERVICE\_TIMEOUT** - 環境變數

指定嘗試從執行個體中繼資料服務擷取資料時，逾時之前的秒數。

預設值：1

有效值：大於或等於 1 的數字。

在 config 檔案中設定這些值的範例：

```
[default]
metadata_service_num_attempts=10
metadata_service_timeout=10
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_METADATA_SERVICE_NUM_ATTEMPTS=10
export AWS_METADATA_SERVICE_TIMEOUT=10
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_METADATA_SERVICE_NUM_ATTEMPTS 10
setx AWS_METADATA_SERVICE_TIMEOUT 10
```

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	否	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	否	
<a href="#">適用於 Go 1.x (V1) 的 SDK</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	部分	僅支援 <code>AWS_METADATA_SERVICE_TIMEOUT</code> 。
<a href="#">適用於 Java 的 SDK 1.x</a>	部分	僅支援 <code>AWS_METADATA_SERVICE_TIMEOUT</code> 。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	否	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	否	
<a href="#">適用於 .NET 4.x 的 SDK</a>	否	
<a href="#">適用於 .NET 3.x 的 SDK</a>	否	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	否	
<a href="#">適用於 Rust 的 SDK</a>	否	

SDK	支援	備註或更多資訊
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	否	
<a href="#">PowerShell V4 的工具</a>	否	

## Amazon S3 存取點

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

Amazon S3 服務提供存取點做為與 Amazon S3 儲存貯體互動的替代方式。存取點具有可套用至它們的唯一政策和組態，而不是直接套用至儲存貯體。透過 AWS SDKs，您可以在 API 操作的儲存貯體欄位中使用存取點 Amazon Resource Name (ARNs)，而不是明確指定儲存貯體名稱。它們用於特定操作，例如使用存取點 ARN [GetObject](#) 搭配 從儲存貯體擷取物件，或使用存取點 ARN 搭配 [PutObject](#) 將物件新增至儲存貯體。

若要進一步了解 Amazon S3 存取點和 ARNs，請參閱《Amazon S3 使用者指南》中的 [使用存取點](#)。

使用下列項目設定此功能：

**s3\_use\_arn\_region** - 共用 AWS `config` 檔案設定, **AWS\_S3\_USE\_ARN\_REGION** - 環境變數, **aws.s3UseArnRegion** - JVM 系統屬性：僅限 Java/Kotlin, 若要直接在程式碼中設定值，請直接參閱您的特定 SDK。

此設定控制 SDK 是否使用存取點 ARN AWS 區域 來建構請求的區域端點。開發套件會驗證 ARN 是否由與用戶端設定的相同 AWS 分割區 AWS 區域 提供 AWS 區域，以防止很可能失敗的跨分割區呼叫。如果定義了乘法，則程式碼設定的設定優先，後面接著環境變數設定。

預設值：`false`

有效值：

- **true** – SDK 會在建構端點 AWS 區域 時使用 ARN 的 ，而不是用戶端的 設定 AWS 區域。例外狀況：如果用戶端的 設定為 AWS 區域 FIPS AWS 區域，則必須符合 ARN 的 AWS 區域。否則將會發生錯誤。
- **false** – SDK 會在建構端點時使用用戶端的 設定 AWS 區域。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支 備註或更多資訊 援
<a href="#">AWS CLI v2</a>	是
<a href="#">適用於 C++ 的 SDK</a>	是
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是 若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是
<a href="#">適用於 Java 的 SDK 1.x</a>	是 不支援 JVM 系統屬性。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是
<a href="#">適用於 Kotlin 的 SDK</a>	是
<a href="#">適用於 .NET 4.x 的 SDK</a>	是
<a href="#">適用於 .NET 3.x 的 SDK</a>	是 不遵循標準優先順序；共用config檔案值的優先順序高於環境變數。
<a href="#">適用於 PHP 的 SDK 3.x</a>	是



SDK	支援	備註或更多資訊
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	否	
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	是	不遵循標準優先順序；共用config檔案值的優先順序高於環境變數。
<a href="#">PowerShell V4 的工具</a>	是	不遵循標準優先順序；共用config檔案值的優先順序高於環境變數。

## Amazon S3 多區域存取點

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

Amazon S3 多區域存取點提供全域端點，應用程式可用來滿足來自位於多個 Amazon S3 儲存貯體的請求 AWS 區域。您可以使用多區域存取點，以單一區域中使用的相同架構建置多區域應用程式，然後在世界任何地方執行這些應用程式。

若要進一步了解多區域存取點，請參閱《[Amazon S3 使用者指南](#)》中的 [Amazon S3 中的多區域存取點](#)。Amazon S3

若要進一步了解多區域存取點 Amazon Resource Name (ARNs)，請參閱《[Amazon S3 使用者指南](#)》中的 [使用多區域存取點提出請求](#)。

若要進一步了解如何建立多區域存取點，請參閱《[Amazon S3 使用者指南](#)》中的 [管理多區域存取點](#)。

SigV4A 演算法是用來簽署全域區域請求的簽署實作。此演算法是由 SDK 透過 的相依性取得 [AWS 通用執行期 \(CRT\) 程式庫](#)。

使用下列項目設定此功能：

**s3\_disable\_multiregion\_access\_points** - 共用 AWS config 檔

案設定, **AWS\_S3\_DISABLE\_MULTIREGION\_ACCESS\_POINTS** - 環境變數,

**aws.s3DisableMultiRegionAccessPoints** - JVM 系統屬性：僅限 Java/Kotlin, 若要直接在程式碼中設定值，請直接參閱您的特定 SDK。

此設定會控制 SDK 是否可能嘗試跨區域請求。如果定義了乘法，則程式碼設定的設定優先，後面接著環境變數設定。

預設值：false

有效值：

- **true** – 停止使用跨區域請求。
- **false** – 使用多區域存取點啟用跨區域請求。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援 備註或更多資訊
<a href="#">AWS CLI v2</a>	是
<a href="#">適用於 C++ 的 SDK</a>	是
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否
<a href="#">適用於 Java 的 SDK 2.x</a>	是
<a href="#">適用於 Java 的 SDK 1.x</a>	否
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是

SDK	支援	備註或更多資訊
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## S3 Express One Zone 工作階段身分驗證

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

S3 Express One Zone 是 Amazon S3 的高效能儲存類別，可為經常存取的資料提供單一位數毫秒延遲。當您使用 S3 Express One Zone 儲存貯體時，AWS SDKs和工具會自動使用針對資料請求的低延遲授權最佳化的工作階段型身分驗證。您可以使用工作階段字符搭配區域（物件層級）操作，將與授權相關聯的延遲分配到工作階段中的多個請求，從而減少身分驗證額外負荷並改善整體請求效能。

S3 Express One Zone 儲存貯體使用包含可用區域 ID 的特定命名格式，例如 `bucket-name--usw2-az1--x-s3`。當 SDK 偵測到此命名模式時，會自動將請求路由到適當的 S3 Express One Zone 端點，並套用最佳化的身分驗證流程。工作階段身分驗證會建立暫時性的儲存貯體特定登入資料，提供對儲存貯體的低延遲存取，並由 SDK 自動快取和重新整理。如需進一步了解，請參閱《Amazon [S3 使用者指南](#)》中的 [S3 Express One Zone](#)。Amazon S3

預設會針對 S3 Express One Zone 儲存貯體啟用工作階段身分驗證。

使用下列項目設定此功能：

**s3\_disable\_express\_session\_auth** - 共用 AWS **config** 檔案設定,  
**AWS\_S3\_DISABLE\_EXPRESS\_SESSION\_AUTH** - 環境變數, **aws.disableS3ExpressAuth** - JVM  
系統屬性：僅限 Java/Kotlin

控制是否停用 S3 Express One Zone 工作階段身分驗證。設定為 `true`，開發套件會對 S3 Express One Zone 儲存貯體使用標準 SigV4 身分驗證，而非工作階段身分驗證。S3

預設值：`false`

有效值：

- **true** – 停用 S3 Express One Zone 工作階段身分驗證。
- **false** – 啟用 S3 Express One Zone 工作階段身分驗證。

在 `config` 檔案中設定此值的範例：

```
[default]
s3_disable_express_session_auth=true
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_S3_DISABLE_EXPRESS_SESSION_AUTH=true
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_S3_DISABLE_EXPRESS_SESSION_AUTH true
```

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">AWS CLI v1</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	JVM 系統屬性為 <code>aws.s3DisableExpressSessionAuth</code> 。
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## 身分驗證機制

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

AWS 服務支援多個身分驗證機制，例如 AWS Signature 第 4 版 (SigV4) 和 AWS Signature 第 4a 版 (SigV4a)。根據預設，SDKs會根據服務模型定義選取身分驗證機制，並優先考慮提供最佳相容性的機制。不過，您可以設定偏好的身分驗證機制，以針對特定需求進行最佳化。

與 SigV4 不同，使用 SigV4a 簽署的請求在多個 中有效 AWS 區域。SigV4a 透過跨區域請求簽署提供增強的可用性，可在區域中斷期間自動容錯移轉至備份區域。這對 AWS Identity and Access Management 或 Amazon CloudFront 等全球服務特別有用。

如需這兩個身分驗證機制的詳細資訊，請參閱《IAM 使用者指南》中的 [AWS API 請求的簽章版本 4](#)。

使用下列項目設定此功能：

**auth\_scheme\_preference** - 共用 AWS `config` 檔案設定, **AWS\_AUTH\_SCHEME\_PREFERENCE** - 環境變數, **aws.authSchemePreference** - JVM 系統屬性：僅限 Java/Kotlin

依優先順序指定以逗號分隔的偏好身分驗證機制清單。當服務支援多個身分驗證方案時，開發套件會嘗試以指定的順序使用此清單中的方案，如果沒有可用的偏好方案，則會回復為預設行為。

預設值：無。

有效值：下列一或多個項目的逗號分隔清單：

- **sigv4** – Signature 第 4 版（最高效能，單一區域）

- **sigv4a** – Signature 第 4a 版 ( 增強可用性、跨區域支援，簽署效能比 SigV4 慢 )
- **httpBearerAuth** – HTTP 承載字符身分驗證

配置名稱之間的空格和標籤字元會被忽略。

在 config 檔案中設定此值以偏好 SigV4a 的範例：

```
[default]
auth_scheme_preference=sigv4a,sigv4
```

### **sigv4a\_signing\_region\_set** - 共用 AWS config 檔案設定, **AWS\_SIGV4A\_SIGNING\_REGION\_SET** - 環境變數

為 SigV4a 多區域簽署指定以逗號分隔 AWS 區域的清單。如果 SigV4a 是選取的身分驗證機制，則會將其用作請求的預設區域集。

預設值：由請求決定。

有效值：以逗號分隔的清單 AWS 區域。區域之間的空格和標籤字元會被忽略。

## 支援 AWS SDKs 和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援 備註或更多資訊
<a href="#">AWS CLI v2</a>	是
<a href="#">適用於 C++ 的 SDK</a>	是
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否
<a href="#">適用於 Java 的 SDK 2.x</a>	是
<a href="#">適用於 Java 的 SDK 1.x</a>	否
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是

SDK	支援	備註或更多資訊
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	否	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	否	

## AWS 區域

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

AWS 區域 是使用時需要了解的重要概念 AWS 服務。

使用 AWS 區域，您可以存取實際位於特定地理區域的 AWS 服務。這有助於讓您的資料和應用程式在靠近您和您的使用者將存取它們的位置執行。區域提供容錯能力、穩定性和恢復能力，也可降低延遲。使用 區域，您可以建立備援資源，以保持可用且不受區域中斷影響。



大多數 AWS 服務 請求都與特定地理區域相關聯。除非您明確使用 提供的複寫功能，否則您在一個區域中建立的資源不會存在於任何其他區域中 AWS 服務。例如，Amazon S3 和 Amazon EC2 支援跨區域複寫。有些服務，例如 IAM，沒有區域資源。

AWS 一般參考 包含下列資訊：

- 若要了解區域和端點之間的關係，以及檢視現有區域端點的清單，請參閱 [AWS 服務端點](#)。
- 若要檢視每個 所有支援的區域和端點的目前清單 AWS 服務，請參閱 [服務端點和配額](#)。

## 建立服務用戶端

若要以程式設計方式存取 AWS 服務，SDKs會為每個 使用用戶端類別/物件 AWS 服務。例如，如果您的應用程式需要存取 Amazon EC2，您的應用程式會建立 Amazon EC2 用戶端物件以與該服務連接。

如果程式碼本身未明確為用戶端指定區域，用戶端會預設為使用透過下列設定設定的 區域region。不過，可以針對任何個別用戶端物件明確設定用戶端的作用中區域。以這種方式設定區域優先於該特定服務用戶端的任何全域設定。替代區域是在該用戶端的執行個體化期間指定，專屬於您的 SDK（請參閱您的特定 SDK 指南或軟體開發套件的程式碼基底）。

使用下列項目設定此功能：

**region** - 共用 AWS **config**檔案設定, **AWS\_REGION** - 環境變數, **aws.region** - JVM 系統屬性：僅限 Java/Kotlin

指定 AWS 區域 要用於 AWS 請求的預設值。此區域用於未隨附要使用之特定區域的 SDK 服務請求。

預設值：無。您必須明確指定此值。

有效值：

- 適用於所選服務的任何區域代碼，如 AWS 一般參考中的 [AWS 服務端點](#) 所列。例如，值 `us-east-1` 會將端點設定為 AWS 區域 美國東部（維吉尼亞北部）。
- `aws-global` 指定 服務的全域端點，除了區域端點之外，還支援個別的全域端點，例如 AWS Security Token Service (AWS STS) 和 Amazon Simple Storage Service (Amazon S3)。

在 `config` 檔案中設定此值的範例：

```
[default]
region = us-west-2
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_REGION=us-west-2
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_REGION us-west-2
```

大多數 SDKs 都有一個「組態」物件，可用於從應用程式程式碼內設定預設區域。如需詳細資訊，請參閱您的特定 AWS SDK 開發人員指南。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	AWS CLI v2 在 中的任何值 <code>AWS_REGION</code> 之前，會先使用 中的任何值 <code>AWS_DEFAULT_REGION</code> （會檢查兩個變數）。
<a href="#">AWS CLI v1</a>	是	AWS CLI v1 會將名為 的環境變數 <code>AWS_DEFAULT_REGION</code> 用於此目的。
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用 config 檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	此 SDK 會將名為 <code>AWS_DEFAULT_REGION</code> 的環境變數用於此目的。
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## AWS STS 區域端點

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

AWS Security Token Service (AWS STS) 提供全球和區域服務。有些 AWS SDKs 和 CLIs 預設使用全域服務端點 (<https://sts.amazonaws.com>)，有些則使用區域服務端點 ([https://sts.{region\\_identifier}.{partition\\_domain}](https://sts.{region_identifier}.{partition_domain}))。在[預設啟用](#)的區域中，對 AWS STS 全域端點的請求會在請求產生的相同區域中自動提供。在選擇加入區域中，全域 AWS STS 端點的請求由單一 AWS 區域美國東部（維吉尼亞北部）提供。如需 AWS STS 端點的詳細資訊，請參閱 AWS Security Token Service 《API 參考》中的[端點](#)或《AWS Identity and Access Management 使用者指南 [AWS STS](#)》中的[管理 AWS 區域](#)。

AWS 最佳實務是盡可能使用區域端點並設定您的 [AWS 區域](#)。商業 以外 [分割區](#) 中的客戶必須使用區域端點。並非所有 SDKs 和工具都支援此設定，但所有 都已定義全球和區域端點的行為。如需詳細資訊，請參閱下一節。

### Note

AWS 已對 [依預設啟用](#) 的區域中的 AWS Security Token Service (AWS STS) 全域端點 (<https://sts.amazonaws.com>) 進行變更，以增強其彈性和效能。對全域端點的 AWS STS 請求會在與您的工作負載 AWS 區域 相同的中自動提供。這些變更不會部署至選擇加入區域。我們建議您使用適當的 AWS STS 區域端點。如需詳細資訊，請參閱 [AWS Identity and Access Management 《使用者指南》](#) 中的 [AWS STS 全域端點變更](#)。

對於支援此設定的 SDKs 和工具，客戶可以使用下列項目來設定功能：

**sts\_regional\_endpoints** - 共用 AWS **config** 檔案設定, **AWS\_STS\_REGIONAL\_ENDPOINTS** - 環境變數

此設定指定 SDK 或工具如何決定用來與 AWS 服務 AWS Security Token Service () 通訊的端點 AWS STS。

預設值：`regional`，請參閱下表中的例外狀況。

### Note

2022 年 7 月之後發行的所有新 SDK 主要版本都將預設為 `regional`。新的 SDK 主要版本可能會移除此設定並使用 `regional` 行為。為了降低此變更的未來影響，我們建議您盡可能 `regional` 在應用程式中開始使用。

有效值： (建議值：`regional`)

- **legacy** – 使用全域 AWS STS 端點 `sts.amazonaws.com`。
- **regional** – SDK 或工具一律使用目前設定區域的 AWS STS 端點。例如，如果用戶端設定為使用 `us-west-2`，則對的所有呼叫 AWS STS 都會對區域端點 進行 `sts.us-west-2.amazonaws.com`，而不是全域 `sts.amazonaws.com` 端點。若要在啟用此設定的同時，將請求傳送至全域端點，您可以將區域設為 `aws-global`。

在 `config` 檔案中設定這些值的範例：

```
[default]
sts_regional_endpoints = regional
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_STS_REGIONAL_ENDPOINTS=regional
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_STS_REGIONAL_ENDPOINTS regional
```

## 支援 AWS SDKs和工具

### Note

AWS 最佳實務是盡可能使用區域端點並設定您的 [AWS 區域](#)。

以下資料表摘要說明 SDK 或工具：

- 支援設定：是否支援 STS 區域端點的共用config檔案變數和環境變數。
- 預設設定值：如果支援，則設定的預設值。
- 預設服務用戶端目標 STS 端點：用戶端使用什麼預設端點，即使變更它的設定不可用。
- 服務用戶端備用行為：軟體開發套件應該使用區域端點，但未設定區域時所執行的動作。這是行為，無論它是否因為預設值而使用區域端點，還是因為 regional 已由 設定選取。

資料表也會使用下列值：

- 全域端點：<https://sts.amazonaws.com>。
- 區域端點：根據您應用程式[AWS 區域](#)所使用的設定。
- **us-east-1**（區域）：使用us-east-1區域端點，但具有比一般全域請求更長的工作階段字符。

SDK	預設設定值	預設服務用戶端目標 STS 端點	服務用戶端備用行為	備註或更多資訊
<a href="#">AWS CLI v2</a>	否 N/A	區域 (Region) 端點	全域端點	
<a href="#">AWS CLI v1</a>	是 legacy	全域端點	全域端點	
<a href="#">適用於 C++ 的 SDK</a>	否 N/A	區域 (Region) 端點	us-east-1 (區域)	
<a href="#">SDK for Go V2 (1.x)</a>	否 N/A	區域 (Region) 端點	請求失敗	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是 legacy	全域端點	全域端點	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	否 N/A	區域 (Region) 端點	請求失敗	如果未設定區域，AssumeRole 和 AssumeRoleWithWebIdentity 將使用全域 STS 端點。
<a href="#">適用於 Java 的 SDK 1.x</a>	是 legacy	全域端點	全域端點	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	否 N/A	區域 (Region) 端點	us-east-1 (區域)	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是 legacy	全域端點	全域端點	

SDK	預設設定值	預設服務用戶端目標 STS 端點	服務用戶端備用行為	備註或更多資訊
<a href="#">適用於 Kotlin 的 SDK</a>	否 N/A	區域 (Region) 端點	全域端點	
<a href="#">適用於 .NET 4.x 的 SDK</a>	否 N/A	區域 (Region) 端點	us-east-1 (區域)	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是 regional	全域端點	全域端點	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是 regional	全域端點	請求失敗	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是 regional	全域端點	全域端點	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是 regional	區域 (Region) 端點	請求失敗	
<a href="#">適用於 Rust 的 SDK</a>	否 N/A	區域 (Region) 端點	請求失敗	
<a href="#">適用於 Swift 的 SDK</a>	否 N/A	區域 (Region) 端點	請求失敗	
<a href="#">PowerShell V5 的工具</a>	是 regional	全域端點	全域端點	

SDK	預設設定值	預設服務用戶端目標 STS 端點	服務用戶端備用行為	備註或更多資訊
<a href="#">PowerShell V4 的工具</a>	是 regional	全域端點	全域端點	

## Amazon S3 的資料完整性保護

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

一段時間內，AWS SDKs將資料上傳至 Amazon Simple Storage Service 或從 Amazon Simple Storage Service 下載資料時支援資料完整性檢查。先前，這些檢查是選擇加入。現在，我們預設已使用 CRC 型演算法啟用這些檢查，例如 CRC32 或 CRC64NVME。雖然每個 SDK 或工具都有預設演算法，但您可以選擇不同的演算法。您也可以視需要繼續手動提供上傳的預先計算檢查總和。跨上傳、分段上傳、下載和加密模式的一致行為可簡化用戶端完整性檢查。

最新版本的 AWS SDKs，並 AWS CLI 自動計算每次上傳的[循環備援檢查 \(CRC\) 型檢查總和](#)，並將其傳送至 Amazon S3。Amazon S3 會獨立計算伺服器端的檢查總和，並根據提供的值進行驗證，然後再將物件及其檢查總和永久存放在物件的中繼資料中。透過將檢查總和與物件一起存放在中繼資料中，當物件下載時，也可以自動傳回相同的檢查總和，並用於驗證下載。您也可以隨時驗證存放在物件中繼資料中的檢查總和。

若要進一步了解檢查總和操作、分段上傳或支援的檢查總和演算法清單，請參閱《[Amazon Simple Storage Service 使用者指南](#)》中的[在 Amazon S3 中檢查物件完整性](#)。

分段上傳：

Amazon S3 也為開發人員提供跨單一部分和分段上傳的一致完整物件檢查總和。

以多個部分上傳檔案時，SDKs會計算每個部分的檢查總和。Amazon S3 使用這些檢查總和透過 UploadPart API 驗證每個部分的完整性。此外，當您呼叫 CompleteMultipartUpload API 時，Amazon S3 會驗證整個檔案的大小和檢查總和。



如果您的 SDK 具有 Amazon S3 Transfer Manager 來協助分段上傳，則檢查總和會使用[支援 AWS SDKs和工具](#)資料表中找到的開發套件特定預設演算法來驗證組件。您可以選擇加入完整的物件檢查總和，方法是checksum\_type將設定為 `FULL_OBJECT`或選擇使用 `CRC64NVME` 演算法。

如果您使用的是較舊版本的 SDK 或 AWS CLI：

如果您的應用程式使用 SDK 或工具 2024 年 12 月之前的版本，Amazon S3 仍會在新物件上計算 `CRC64NVME` 檢查總和，並將其存放在物件中繼資料中以供日後參考。您稍後可以將存放的 CRC 與在端計算的 CRC 進行比較，並確認網路傳輸正確。此外，您仍然可以透過使用 [PutObject](#)或 [UploadPart](#)請求提供您自己的預先計算檢查總和來手動擴展完整性保護，這是在較舊版本中解決此問題的標準技術。

使用下列項目設定此功能：

**request\_checksum\_calculation** - 共用 AWS **config**檔案設定,  
**AWS\_REQUEST\_CHECKSUM\_CALCULATION** - 環境變數, **aws.requestChecksumCalculation** - JVM 系統屬性：僅限 Java/Kotlin

根據預設，使用者在傳送請求時可以選擇加入計算請求檢查總和。使用者可以選擇任何[可用的檢查總和演算法](#)，做為建置請求的一部分。否則，會使用 SDK 特定的預設演算法。如需每個 SDK 或工具的預設演算法，請參閱 [支援 AWS SDKs和工具](#)資料表。

預設值：WHEN\_SUPPORTED

有效值：

- **WHEN\_SUPPORTED** – 當 API 操作支援時，對所有請求承載執行檢查總和驗證，例如資料傳輸到 Amazon S3。
- **WHEN\_REQUIRED** – 只有在 API 操作需要時，才會執行檢查總和驗證。

**response\_checksum\_validation** - 共用 AWS **config**檔案設定,  
**AWS\_RESPONSE\_CHECKSUM\_VALIDATION** - 環境變數, **aws.responseChecksumValidation** - JVM 系統屬性：僅限 Java/Kotlin

根據預設，使用者在傳送請求時可以選擇加入回應檢查總和驗證。系統會計算回應承載的檢查總和，並與檢查總和回應標頭進行比較。如果檢查總和驗證失敗，會在讀取承載時向使用者引發錯誤。

檢查總和回應標頭也會指出檢查總和的演算法。Amazon S3 用戶端會嘗試驗證支援檢查總和之所有 Amazon S3 API 操作的回應檢查總和。不過，如果軟體開發套件未實作指定的檢查總和演算法，則會略過此驗證。

預設值：WHEN\_SUPPORTED

有效值：

- **WHEN\_SUPPORTED** – 當 API 操作支援時，檢查總和驗證會在所有回應承載上執行，例如資料傳輸到 Amazon S3。
- **WHEN\_REQUIRED** – 只有在 API 操作支援且發起人已明確啟用操作的檢查總和時，才會執行檢查總和驗證。例如，當呼叫 Amazon S3 GetObject API 且 ChecksumMode 參數設定為啟用時。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

### Note

在下表中，「CRT」是指 [AWS 通用執行期 \(CRT\) 程式庫](#)，可能需要為您的專案新增額外的相依性。

SDK	支援	預設檢查總和演算法	支援的檢查總和演算法	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	CRC64NVME	CRC64NVME, CRC32, CRC32C, SHA1, SHA256	對於 AWS CLI v1，預設演算法和支援的演算法將與 Python (Boto3) 相同。
<a href="#">適用於 C++ 的 SDK</a>	是	CRC64NVME	CRC64NVME, CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	CRC32	CRC64NVME, CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否			

SDK	支援	預設檢查總和演算法	支援的檢查總和演算法	備註或更多資訊
<a href="#">適用於 Java 的 SDK 2.x</a>	是	CRC32	CRC64NVME ( 僅透過 CRT)、CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 Java 的 SDK 1.x</a>	否			
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否			
<a href="#">適用於 Kotlin 的 SDK</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	CRC32	CRC32, CRC32C ( 僅透過 CRT)、SHA1, SHA256	awscli 需要擴充功能才能使用 CRC32C。
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	CRC32	CRC64NVME ( 僅透過 CRT)、CRC32, CRC32C ( 僅透過 CRT)、SHA1, SHA256	

SDK	支援	預設檢查總和演算法	支援的檢查總和演算法	備註或更多資訊
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	CRC32	CRC64NVME ( 僅透過 CRT)、CRC32, CRC32C ( 僅透過 CRT)、SHA1, SHA256	
<a href="#">適用於 Rust 的 SDK</a>	是	CRC32	CRC64NVME, CRC32, CRC32C, SHA1, SHA256	
<a href="#">適用於 Swift 的 SDK</a>	是	CRC32	CRC64NVME, CRC32, CRC32C, SHA1, SHA256	所有演算法都需要 CRT 相依性。
<a href="#">PowerShell V5 的工具</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	
<a href="#">PowerShell V4 的工具</a>	是	CRC32	CRC32, CRC32C, SHA1, SHA256	

## 雙堆疊和 FIPS 端點

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

使用下列項目設定此功能：

**use\_dualstack\_endpoint** - 共用 AWS **config**檔案設定, **AWS\_USE\_DUALSTACK\_ENDPOINT** - 環境變數, **aws.useDualstackEndpoint** - JVM 系統屬性：僅限 Java/Kotlin

開啟或關閉軟體開發套件是否將請求傳送至雙堆疊端點。若要進一步了解支援 IPv4 和 IPv6 流量的雙堆疊端點，請參閱《Amazon Simple Storage Service 使用者指南》中的[使用 Amazon S3 雙堆疊端點](#)。雙堆疊端點適用於部分區域的某些服務。

預設值：false

有效值：

- **true** – SDK 或工具會嘗試使用雙堆疊端點來提出網路請求。如果服務和/或不存在雙堆疊端點 AWS 區域，請求將會失敗。
- **false** – SDK 或工具不會使用雙堆疊端點提出網路請求。

**use\_fips\_endpoint** - 共用 AWS **config**檔案設定, **AWS\_USE\_FIPS\_ENDPOINT** - 環境變數, **aws.useFipsEndpoint** - JVM 系統屬性：僅限 Java/Kotlin

開啟或關閉軟體開發套件或工具是否將請求傳送至符合 FIPS 標準的端點。聯邦資訊處理標準 (FIPS) 是一組美國政府對資料及其加密的安全要求。政府機構、合作夥伴以及想要與聯邦政府進行業務合作的人員必須遵守 FIPS 準則。與標準 AWS 端點不同，FIPS 端點使用經過 FIPS 140 驗證的 TLS 軟體程式庫。如果啟用此設定，且服務不存在 FIPS 端點 AWS 區域，則 AWS 呼叫可能會失敗。[服務特定的端點](#)和 `--endpoint-url`選項會 AWS Command Line Interface 覆寫此設定。

若要進一步了解依指定 FIPS 端點的其他方式 AWS 區域，請參閱[依服務的 FIPS 端點](#)。如需 Amazon Elastic Compute Cloud 服務端點的詳細資訊，請參閱《Amazon EC2 API 參考》中的[雙堆疊 \(IPv4 和 IPv6\) 端點](#)。

預設值：false

有效值：

- **true** – SDK 或工具會將請求傳送至符合 FIPS 規範的端點。
- **false** – SDK 或工具不會將請求傳送至符合 FIPS 規範的端點。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## 端點探索

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

SDKs使用端點探索來存取服務端點 (URLs存取各種資源) AWS，同時仍保有視需要變更URLs彈性。如此一來，您的程式碼就會自動偵測新的端點。某些服務沒有固定端點。相反地，您可以透過請求先取得端點，在執行時間期間取得可用的端點。擷取可用的端點之後，程式碼接著會使用端點來存取其他操作。例如，對於 Amazon Timestream，軟體開發套件會發出擷取可用端點的DescribeEndpoints請求，然後使用這些端點來完成特定操作，例如 CreateDatabase或 CreateTable。

使用下列項目設定此功能：

**endpoint\_discovery\_enabled** - 共用 AWS config檔案設定，  
**AWS\_ENABLE\_ENDPOINT\_DISCOVERY** - 環境變數，**aws.endpointDiscoveryEnabled** - JVM 系統屬性：僅限 Java/Kotlin，若要直接在程式碼中設定值，請直接參閱您的特定 SDK。

開啟或關閉 DynamoDB 的端點探索。

Timestream 需要端點探索，Amazon DynamoDB 則為選用。此設定預設為 true或，false取決於服務是否需要端點探索。Timestream 請求預設為 true，Amazon DynamoDB 請求預設為 false。

有效值：

- **true** – SDK 應自動嘗試探索服務的端點，其中端點探索是選用的。
- **false** – SDK 不應自動嘗試探索服務的端點，其中端點探索是選用的。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	若要使用共用config檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	是	適用於 Java 的 SDK 2.x <code>AWS_ENDPOINT_DISCOVERY_ENABLED</code> 用於環境變數名稱。
<a href="#">適用於 Java 的 SDK 1.x</a>	部分	不支援 JVM 系統屬性。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	部分	僅支援 Timestream。
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	是	



SDK	支援 備註或更多資訊
<a href="#">PowerShell V4 的工具</a>	是

## 一般組態設定

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

SDKs支援一些設定整體軟體開發套件行為的一般設定。

使用下列項目設定此功能：

### **api\_versions** - 共用 AWS **config**檔案設定

有些 AWS 服務會維護多個 API 版本，以支援回溯相容性。根據預設，SDK 和 AWS CLI 操作會使用最新的可用 API 版本。若要要求特定 API 版本用於您的請求，請在設定檔中包含 `api_versions` 設定。

預設值：無。（開發套件會使用最新 API 版本。）

有效值：這是一個巢狀設定，後面接著一或多個縮排行，每個行都會識別一個 AWS 服務和要使用的 API 版本。請參閱 AWS 服務的文件，以了解可用的 API 版本。

此範例會為 `config` 檔案中的兩個 AWS 服務設定特定的 API 版本。這些 API 版本僅適用於在包含這些設定的描述檔下執行的命令。任何其他服務的命令會使用該服務 API 的最新版本。

```
api_versions =  
  ec2 = 2015-03-01  
  cloudfront = 2015-09-017
```

### **ca\_bundle** - 共用 AWS **config**檔案設定, **AWS\_CA\_BUNDLE** - 環境變數

指定建立 SSL/TLS 連線時要使用的自訂憑證套件（具有 `.pem` 副檔名的檔案）路徑。

預設值：無

有效值：指定完整路徑或基本檔案名稱。如果有基本檔案名稱，系統會嘗試在PATH環境變數指定的資料夾中尋找程式。

在 config 檔案中設定此值的範例：

```
[default]
ca_bundle = dev/apps/ca-certs/cabundle-2019mar05.pem
```

由於作業系統如何處理路徑和路徑字元逸出的差異，以下是在 Windows 的 config 檔案中設定此值的範例：

```
[default]
ca_bundle = C:\\Users\\username\\.aws\\aws-custom-bundle.pem
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_CA_BUNDLE=/dev/apps/ca-certs/cabundle-2019mar05.pem
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_CA_BUNDLE C:\dev\apps\ca-certs\cabundle-2019mar05.pem
```

## output - 共用 AWS config 檔案設定

指定 AWS CLI 和其他 AWS SDKs和工具中結果的格式。

預設值：json

有效值：

- **json** – 輸出的格式為 [JSON](#) 字串。
- **yaml** – 輸出的格式為 [YAML](#) 字串。
- **yaml-stream** – 輸出採用串流方式且格式為 [YAML](#) 字串。串流可加速處理大型資料類型。
- **text** – 輸出的格式是多行以 Tab 分隔的字串值。這對於將輸出傳遞給文字處理器 (如 `grep`、`sed` 或 `awk`) 非常有用。
- **table** – 輸出的格式為使用字元 `+|-` 形成儲存格框線的表格。它通常以「方便人類使用」的格式來呈現資訊，這種格式比其他格式更容易閱讀，但在編寫程式方面較不有用。

## parameter\_validation - 共用 AWS config 檔案設定

指定軟體開發套件或工具是否在將 AWS 命令列參數傳送至服務端點之前嘗試驗證命令列參數。

預設值：`true`

有效值：

- **true** – 預設值。SDK 或工具會執行命令列參數的用戶端驗證。這有助於 SDK 或工具確認參數有效，並擷取一些錯誤。開發套件或工具可以在傳送請求至 AWS 服務端點之前，拒絕無效的請求。
- **false** – SDK 或工具不會在將命令列參數傳送到 AWS 服務端點之前驗證命令列參數。AWS 服務端點負責驗證所有請求，並拒絕無效的請求。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	部分	<code>api_versions</code> 不支援。
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	部分	<code>api_versions</code> <code>parameter_validation</code> 不支援和。
<a href="#">適用於 Go 1.x (V1) 的 SDK</a>	部分	<code>api_versions</code> <code>parameter_validation</code> 不支援和。若要使用共用 config 檔案設定，您必須開啟從組態檔案載入；請參閱 <a href="#">工作階段</a> 。
<a href="#">適用於 Java 的 SDK 2.x</a>	否	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	否	
<a href="#">適用於 .NET 4.x 的 SDK</a>	否	
<a href="#">適用於 .NET 3.x 的 SDK</a>	否	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	否	
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	否	
<a href="#">PowerShell V4 的工具</a>	否	

## 主機字首注入

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

主機字首插入是一項功能，其中 AWS SDKs會自動在特定 API 操作的服務端點主機名稱前面加上字首。此字首可以是靜態字串或動態值，其中包含來自請求參數的資料。

例如，使用 Amazon Simple Storage Service 在 Amazon S3 物件或儲存貯體上執行動作時，軟體開發套件會取代最終 API 端點中的儲存貯體名稱和 AWS 帳戶 ID。

雖然一般 AWS 服務端點需要此行為，但在使用 VPC 端點或本機測試工具等自訂端點時，可能會導致問題。在這些情況下，您可能需要停用主機字首注入。

使用下列項目設定此功能：

**disable\_host\_prefix\_injection** - 共用 AWS config 檔案設定,  
**AWS\_DISABLE\_HOST\_PREFIX\_INJECTION** - 環境變數, **aws.disableHostPrefixInjection** - JVM 系統屬性：僅限 Java/Kotlin

此設定會控制軟體開發套件或工具是否透過在軟體開發套件的用戶端物件或變數中定義的主機字首前加上來修改端點主機名稱。

預設值：`false`

有效值：

- **true** – 停用主機字首注入。SDK 不會修改端點主機名稱。
- **false** – 啟用主機字首注入。SDK 會在端點主機名稱的主機字首前面加上。

在 config 檔案中設定此值的範例：

```
[default]
disable_host_prefix_injection = true
```

透過命令列設定環境變數的 Linux/macOS 範例：

```
export AWS_DISABLE_HOST_PREFIX_INJECTION=true
```

透過命令列設定環境變數的 Windows 範例：

```
setx AWS_DISABLE_HOST_PREFIX_INJECTION true
```

## 主機字首注入的範例

下表範例顯示 SDKs 如何在啟用和停用主機字首注入時修改最終端點。

- 主機字首：在 SDK 的用戶端物件或程式碼中的變數上設定的主機字首屬性字串範本。
- 輸入：在 SDK 的用戶端物件或程式碼中的變數上設定的其他輸入。
- 用戶端端點：用戶端的衍生端點。
- 設定值：上一個設定的解析值。

- 產生的端點：SDK 用戶端用來進行 API 呼叫的產生端點。

主機字首	輸入	用戶端端點	設定值	產生的端點
"data."	{}	"https://service.us-west-2.amazonaws.com"	false	"https://data.service.us-west-2.amazonaws.com"
"{Bucket}-{AccountId}."	Bucket: "amzn-s3-demo-bucket1", AccountId: "123456789012"	"https://service.us-west-2.amazonaws.com"	false	"https://amzn-s3-demo-bucket1-123456789012.service.us-west-2.amazonaws.com"
"data."	{}	"https://override.us-west-2.amazonaws.com" (as an override endpoint)	true	"https://override.us-west-2.amazonaws.com"

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">enableHostPrefixInjection</a> 。

SDK	支援	備註或更多資訊
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	否	可以使用 <a href="#">中介軟體</a> 停用。
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">SdkAdvancedClientOption.DISABLE_HOST_PREFIX_INJECTION</a> 。
<a href="#">適用於 Java 的 SDK 1.x</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">withDisableHostPrefixInjection</a> 。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">disableHostPrefix</a> 。
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">hostPrefixEnabled</a> 。
<a href="#">適用於 Kotlin 的 SDK</a>	否	
<a href="#">適用於 .NET 4.x 的 SDK</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">DisableHostPrefixInjection</a> 。
<a href="#">適用於 .NET 3.x 的 SDK</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">DisableHostPrefixInjection</a> 。
<a href="#">適用於 PHP 的 SDK 3.x</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">disable_host_prefix_injection</a> 。
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	您可以使用，在用戶端的程式碼中設定 <a href="#">inject_host_prefix</a> 。
<a href="#">適用於 Ruby 的 SDK 3.x</a>	否	不支援設定，但可以使用在用戶端的程式碼中設定： <a href="#">disable_host_prefix_injection</a> 。
<a href="#">適用於 Rust 的 SDK</a>	否	
<a href="#">適用於 Swift 的 SDK</a>	否	

SDK	支援	備註或更多資訊
<a href="#">PowerShell V5 的工具</a>	否	不支援設定，但可以使用參數 包含在特定 cmdlet 中 - ClientConfig @{DisableHostPrefixInjection = \$true}。
<a href="#">PowerShell V4 的工具</a>	否	不支援設定，但可以使用參數 包含在特定 cmdlet 中 - ClientConfig @{DisableHostPrefixInjection = \$true}。

## IMDS 用戶端

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

SDKs使用工作階段導向請求實作執行個體中繼資料服務第 2 版 (IMDSv2) 用戶端。如需 IMDSv2 的詳細資訊，請參閱《Amazon EC2 使用者指南》中的[使用 IMDSv2](#)。Amazon EC2 IMDS 用戶端可透過 SDK 程式碼庫中提供的用戶端組態物件進行設定。

使用下列項目設定此功能：

### **retries** - 用戶端組態物件成員

任何失敗請求的額外重試嘗試次數。

預設值：3

有效值：大於 0 的數字。

### **port** - 用戶端組態物件成員

端點的連接埠。

預設值：80

有效值：數字。



**token\_ttl** - 用戶端組態物件成員

字符的 TTL。

預設值：21,600 秒 (6 小時，最大分配時間)。

有效值：數字。

**endpoint** - 用戶端組態物件成員

IMDS 的端點。

預設值：如果 `endpoint_mode` 等於 IPv4，則預設端點為 `http://169.254.169.254`。如果 `endpoint_mode` 等於 IPv6，則預設端點為 `http://[fd00:ec2::254]`。

有效值：有效 URI。

大多數 SDKs 支援下列選項。如需詳細資訊，請參閱您的特定 SDK 程式碼庫。

**endpoint\_mode** - 用戶端組態物件成員

IMDS 的端點模式。

預設值：IPv4

有效值：IPv4、IPv6

**http\_open\_timeout** - 用戶端組態物件成員 (名稱可能有所不同)

等待連線開啟的秒數。

預設值：1 秒。

有效值：大於 0 的數字。

**http\_read\_timeout** - 用戶端組態物件成員 (名稱可能有所不同)

要讀取的資料區塊的秒數。

預設值：1 秒。

有效值：大於 0 的數字。

**http\_debug\_output** - 用戶端組態物件成員 (名稱可能有所不同)

設定用於偵錯的輸出串流。

預設值：無。

有效值：有效的 I/O 串流，例如 STDOUT。

### **backoff** - 用戶端組態物件成員（名稱可能會有所不同）

重試或客戶提供的退避函數之間要呼叫的休眠秒數。這會覆寫預設指數退避策略。

預設值：依 SDK 而異。

有效值：依 SDK 而異。可以是數值或呼叫自訂函數。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	否	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	是	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	是	
<a href="#">適用於 Kotlin 的 SDK</a>	否	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## 重試行為

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

重試行為包括 SDKs 如何嘗試從對提出的請求所產生的失敗中復原的設定 AWS 服務。

使用下列項目設定此功能：

**retry\_mode** - 共用 AWS **config**檔案設定, **AWS\_RETRY\_MODE** - 環境變數, **aws.retryMode** - JVM 系統屬性：僅限 Java/Kotlin

指定 SDK 或開發人員工具嘗試重試的方式。

預設值：此值專屬於您的 SDK。檢查您的特定軟體開發套件指南或軟體開發套件的程式碼庫，了解其預設的 `retry_mode`。

有效值：

- **standard** – (建議) 建議的一組跨 AWS SDKs 重試規則。此模式包含一組標準錯誤，這些錯誤會重試，並自動調整重試次數，以最大化可用性和穩定性。此模式可在多租用戶應用程式中安全使用。除非`max_attempts`明確設定，否則使用此模式的預設嘗試次數上限為三次。
- **adaptive** – 一種重試模式，僅適用於專門的使用案例，其中包含標準模式的功能以及自動用戶端速率限制。除非您小心隔離應用程式租用戶，否則不建議多租用戶應用程式使用此重試模式。如需詳細資訊，請參閱[選擇 standard和 adaptive 重試模式](#)。此模式是實驗性的，未來可能會變更行為。
- **legacy** – (不建議) 專屬於您的 SDK (請參閱您的特定 SDK 指南或軟體開發套件的程式碼庫)。

**max\_attempts** - 共用 AWS **config**檔案設定, **AWS\_MAX\_ATTEMPTS** - 環境變數,

**aws.maxAttempts** - JVM 系統屬性：僅限 Java/Kotlin

指定對請求進行的最大嘗試次數。

預設值：如果未指定此值，其預設值取決於 `retry_mode`設定的值：

- 如果 `retry_mode`為 `legacy`- 使用 SDK 特有的預設值 (請參閱特定 SDK 指南或 SDK 的程式碼基底以取得`max_attempts`預設值)。
- 如果 `retry_mode`是 `standard` – 進行三次嘗試。
- 如果 `retry_mode`是 `adaptive` – 進行三次嘗試。

有效值：大於 0 的數字。

## 選擇 **standard**和 **adaptive** 重試模式

建議您使用`standard`重試模式，除非您確定您的用量更適合 `adaptive`。

### Note

`adaptive` 模式假設您要根據後端服務可能調節請求的範圍來集區用戶端。如果您不這樣做，如果您對兩個資源使用相同的用戶端，則一個資源中的調節可能會延遲對不相關資源的請求。

標準	自適應
應用程式使用案例：全部。	應用程式使用案例：

標準	自適應
	<ol style="list-style-type: none"> <li>1. 對延遲不敏感。</li> <li>2. 用戶端只會存取單一資源，或者，您提供的邏輯是由正在存取的服務資源分別集區您的用戶端。</li> </ol>
支援中斷電路，以防止 SDK 在中斷期間重試。  在失敗時使用抖動指數退避。	支援中斷電路，以防止 SDK 在中斷期間重試。  使用動態退避持續時間來嘗試將失敗的請求數量降至最低，以換取增加延遲的可能性。
永遠不要延遲第一次的請求嘗試，只延遲重試。	可以調節或延遲初始請求嘗試。

如果您選擇使用 adaptive 模式，您的應用程式必須建構以每個可能節流的資源為中心的用戶端。在這種情況下，資源的微調比只考慮每個資源更精細 AWS 服務。AWS 服務 可以具有用於調節請求的其他維度。讓我們使用 Amazon DynamoDB 服務做為範例。DynamoDB 使用 AWS 區域 加上要存取的資料表來調節請求。這表示您的程式碼存取的一個資料表可能會受到比其他資料表更多的調節。如果您的程式碼使用相同的用戶端存取所有資料表，且對其中一個資料表的請求受到調節，則適應性重試模式將降低所有資料表的請求率。您的程式碼應設計成每個Region-and-table對有一個用戶端。如果您在使用 adaptive 模式時遇到意外延遲，請參閱您正在使用服務的特定 AWS 文件指南。

## 重試模式實作詳細資訊

AWS SDKs 會使用 [字符儲存貯體](#) 來決定是否應重試請求，以及（在 adaptive 重試模式下）應傳送請求的速度。軟體開發套件會使用兩個字符儲存貯體：重試字符儲存貯體和請求速率字符儲存貯體。

- 重試字符儲存貯體用於判斷 SDK 是否應暫時停用重試，以便在中斷期間保護上游和下游服務。在嘗試重試之前，會從儲存貯體取得權杖，並在請求成功時將權杖傳回至儲存貯體。如果嘗試重試時儲存貯體是空的，則 SDK 不會重試請求。
- 請求率字符儲存貯體僅用於 adaptive 重試模式，以判斷傳送請求的速率。在傳送請求之前，會從儲存貯體取得字符，並根據服務傳回的限流回應，以動態決定的速率將字符傳回至儲存貯體。

以下是 standard 和 adaptive 重試模式的高階虛擬程式碼：

```
MakeSDKRequest() {
  attempts = 0
  loop {
```

```
GetSendToken()
response = SendHTTPRequest()
RequestBookkeeping(response)
if not Retryable(response)
    return response
attempts += 1
if attempts >= MAX_ATTEMPTS:
    return response
if not HasRetryQuota(response)
    return response
delay = ExponentialBackoff(attempts)
sleep(delay)
}
}
```

以下是有關虛擬程式碼中所用元件的更多詳細資訊：

### **GetSendToken:**

此步驟僅用於adaptive重試模式。此步驟會從請求率字符儲存貯體取得字符。如果字符不可用，它會等待一個字符變成可用。您的 SDK 可能有組態選項，可讓請求失敗，而不是等待。儲存貯體中的字符會根據用戶端收到的限流回應數量，以動態決定的速率重新填充。

### **SendHTTPRequest:**

此步驟會將請求傳送至 AWS。AWS SDKs使用 HTTP 程式庫，該程式庫使用連線集區在提出 HTTP 請求時重複使用現有的連線。一般而言，如果請求因限流錯誤而失敗，但請求因暫時性錯誤而失敗，則會重複使用連線。

### **RequestBookkeeping:**

如果請求成功，權杖會新增至權杖儲存貯體。僅針對adaptive重試模式，請求速率字符儲存貯體的填充速率會根據收到的回應類型更新。

### **Retryable:**

此步驟會根據下列項目決定是否可以重試回應：

- HTTP 狀態碼。
- 從服務傳回的錯誤碼。
- 連線錯誤，定義為軟體開發套件收到的任何錯誤，其中未收到服務的 HTTP 回應。

暫時性錯誤 (HTTP 狀態碼 400、408、500、502、503 和 504) 和限流錯誤 (HTTP 狀態碼 400、403、429、502、503 和 509) 都可以重試。SDK 重試行為是結合錯誤代碼或服務的其他資料來決定。

### MAX\_ATTEMPTS:

除非由 `retry_mode` 設定覆寫，否則預設的嘗試次數上限會由 `max_attempts` 設定設定。

### HasRetryQuota

此步驟會從重試權杖儲存貯體取得權杖。如果重試權杖儲存貯體是空的，則不會重試請求。

### ExponentialBackoff

對於可以重試的錯誤，系統會使用截斷的指數退避來計算重試延遲。SDKs 使用截斷的二進位指數退避與抖動。下列演算法顯示如何為請求的回應定義睡眠時間，以秒為單位  $i$ ：

```
seconds_to_sleep_i = min(b*r^i, MAX_BACKOFF)
```

在上述演算法中，適用下列值：

$b$  = random number within the range of:  $0 \leq b \leq 1$

$r$  = 2

`MAX_BACKOFF` = 20 seconds 適用於大多數 SDKs。請參閱您的特定 SDK 指南或原始程式碼進行確認。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援 備註或更多資訊
<a href="#">AWS CLI v2</a>	是
<a href="#">適用於 C++ 的 SDK</a>	是
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是

SDK	支援	備註或更多資訊
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	是	JVM 系統屬性：使用 <code>com.amazonaws.sdk.maxAttempts</code> 而非 <code>aws.maxAttempts</code> ；使用 <code>com.amazonaws.sdk.retryMode</code> 而非 <code>aws.retryMode</code> 。
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	支援重試次數上限、具有抖動的指數退避，以及用於重試退避的自訂方法選項。
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	



## 請求壓縮

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

傳送請求到支援接收壓縮承載的時 AWS 服務，AWS SDKs和工具可以自動壓縮承載。將承載傳送至服務之前在用戶端上壓縮承載，可能會減少將資料傳送至服務所需的整體請求數和頻寬，並減少由於承載大小的服務限制而導致的失敗請求。針對壓縮，軟體開發套件或工具會選取服務和軟體開發套件都支援的編碼演算法。不過，目前可能的編碼清單僅包含 gzip，但未來可能會擴展。

如果您的應用程式使用 [Amazon CloudWatch](#)，請求壓縮特別有用。CloudWatch 是一種監控和可觀測性服務，以日誌、指標和事件的形式收集監控和操作資料。支援壓縮的服務操作的一個範例是 CloudWatch 的 [PutMetricDataAPI](#) 方法。

使用下列項目設定此功能：

**disable\_request\_compression** - 共用 AWS **config** 檔案設定,  
**AWS\_DISABLE\_REQUEST\_COMPRESSION** - 環境變數, **aws.disableRequestCompression** - JVM  
系統屬性：僅限 Java/Kotlin

開啟或關閉軟體開發套件或工具是否會在傳送請求之前壓縮承載。

預設值：false

有效值：

- **true** – 關閉請求壓縮。
- **false** – 盡可能使用請求壓縮。

**request\_min\_compression\_size\_bytes** - 共用 AWS **config** 檔案設定, **AWS\_REQUEST\_MIN\_COMPRESSION\_SIZE\_BYTES** - 環境變數,  
**aws.requestMinCompressionSizeBytes** - JVM 系統屬性：僅限 Java/Kotlin

設定 SDK 或工具應壓縮之請求內文的位元組大小下限。壓縮時，小型承載可能會變長，因此有較低的限制可執行壓縮。此值包含在內，會壓縮大於或等於值的請求大小。

預設值：10240 位元組

有效值：包含介於 0 和 10485760 位元組之間的整數值。

## 支援 AWS SDKs和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	

SDK	支援	備註或更多資訊
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## 服務特定的端點

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

服務特定的端點組態提供選項，讓您針對 API 請求使用您選擇的端點，並讓該選擇持續存在。這些設定提供支援本機端點、VPC 端點和第三方本機 AWS 開發環境的彈性。不同的端點可用於測試和生產環境。您可以為個人 AWS 服務指定端點 URL。

使用下列項目設定此功能：

**endpoint\_url** - 共用 AWS config檔案設定, **AWS\_ENDPOINT\_URL** - 環境變數,  
**aws.endpointUrl** - JVM 系統屬性：僅限 Java/Kotlin

當直接在設定檔中或做為環境變數指定時，此設定會指定用於所有服務請求的端點。此端點會由任何已設定的服務特定端點覆寫。

您也可以使用共用 AWS config檔案的 `services` 區段中使用此設定，為特定服務設定自訂端點。如需用於 `services` 區段中子區段的所有服務識別符索引鍵清單，請參閱 [服務特定端點的識別符](#)。

預設值：none

有效值：包含端點方案和主機的 URL。URL 可以選擇性地包含包含一或多個路徑區段的路徑元件。

**AWS\_ENDPOINT\_URL\_<SERVICE>** - 環境變數, **aws.endpointUrl<ServiceName>** - JVM 系統屬性：僅限 Java/Kotlin

**AWS\_ENDPOINT\_URL\_<SERVICE>**，其中 **<SERVICE>** 是 AWS 服務 識別符，會為特定服務設定自訂端點。如需所有服務特定環境變數的清單，請參閱 [服務特定端點的識別符](#)。

此服務特定的端點會覆寫 `aws.endpointUrl` 中設定的任何全域端點 **AWS\_ENDPOINT\_URL**。

預設值：none

有效值：包含端點方案和主機的 URL。URL 可以選擇性地包含包含一或多個路徑區段的路徑元件。

**ignore\_configured\_endpoint\_urls** - 共用 AWS **config** 檔案設定, **AWS\_IGNORE\_CONFIGURED\_ENDPOINT\_URLS** - 環境變數, **aws.ignoreConfiguredEndpointUrls** - JVM 系統屬性：僅限 Java/Kotlin

此設定用於忽略所有自訂端點組態。

請注意，無論此設定為何，都會使用程式碼中或服務用戶端本身上設定的任何明確端點。例如，包含具有 AWS CLI 命令的 `--endpoint-url` 命令列參數，或將端點 URL 傳遞至用戶端建構函式，一律會生效。

預設值：false

有效值：

- **true** – SDK 或工具不會從共用 **config** 檔案或環境變數讀取任何自訂組態選項，以設定端點 URL。
- **false** – SDK 或工具使用共用 **config** 檔案或環境變數中的任何可用使用者提供的端點。

## 使用環境變數設定端點

若要將所有服務的請求路由到自訂端點 URL，請設定 **AWS\_ENDPOINT\_URL** 全域環境變數。

```
export AWS_ENDPOINT_URL=http://localhost:4567
```

若要將特定的請求路由 AWS 服務 至自訂端點 URL，請使用 **AWS\_ENDPOINT\_URL\_<SERVICE>** 環境變數。Amazon DynamoDB 具有 `serviceId` 的 [DynamoDB](#)。對於此服務，端點 URL 環境變數為 **AWS\_ENDPOINT\_URL\_DYNAMODB**。此端點優先於 **AWS\_ENDPOINT\_URL** 針對此服務在 `aws.endpointUrl` 中設定的全域端點。

```
export AWS_ENDPOINT_URL_DYNAMODB=http://localhost:5678
```

另一個範例 AWS Elastic Beanstalk 是 `serviceId` 的 [Elastic Beanstalk](#)。AWS 服務 識別符是以 API 模型的 為基礎 `serviceId`，方法是將所有空格替換為底線和大寫所有字母。若要設定此服務的端點，對應的環境變數為 `AWS_ENDPOINT_URL_ELASTIC_BEANSTALK`。如需所有服務特定環境變數的清單，請參閱 [服務特定端點的識別符](#)。

```
export AWS_ENDPOINT_URL_ELASTIC_BEANSTALK=http://localhost:5567
```

## 使用共用 `config` 檔案設定端點

在共用 `config` 檔案中，`endpoint_url` 會在不同位置用於不同功能。

- `endpoint_url` 會直接在 內指定，`profile` 使該端點成為全域端點。
- `endpoint_url` 巢狀在 `services` 區段中的服務識別符索引鍵下，可讓該端點僅適用於對該服務提出的請求。如需在共用 `config` 檔案中定義 `services` 區段的詳細資訊，請參閱 [組態檔案的格式](#)。

下列範例使用 `services` 定義來設定要用於 Amazon S3 的服務特定端點 URL，以及要用於所有其他服務的自訂全域端點：

```
[profile dev-s3-specific-and-global]  
endpoint_url = http://localhost:1234  
services = s3-specific  
  
[services s3-specific]  
s3 =  
  endpoint_url = https://play.min.io:9000
```

單一設定檔可以設定多個服務的端點。此範例說明如何在相同的設定檔中設定 Amazon S3 和 AWS Elastic Beanstalk 的服務特定端點 URLs。AWS Elastic Beanstalk 具有 `serviceId` 的 [Elastic Beanstalk](#)。AWS 服務 識別符是以 API 模型的 為基礎 `serviceId`，方法是將所有空格替換為底線，並縮小所有字母大小寫。因此，服務識別符金鑰會變成 `elastic_beanstalk`，且此服務的設定會從第 行開始 `elastic_beanstalk =`。如需要在 `services` 區段中使用的所有服務識別碼金鑰的清單，請參閱 [服務特定端點的識別符](#)。

```
[services testing-s3-and-eb]  
s3 =  
  endpoint_url = http://localhost:4567
```

```
elastic_beanstalk =
    endpoint_url = http://localhost:8000

[profile dev]
services = testing-s3-and-eb
```

服務組態區段可從多個設定檔使用。例如，兩個設定檔可以使用相同的services定義，同時變更其他設定檔屬性：

```
[services testing-s3]
s3 =
    endpoint_url = https://localhost:4567

[profile testing-json]
output = json
services = testing-s3

[profile testing-text]
output = text
services = testing-s3
```

## 使用角色型登入資料在設定檔中設定端點

如果您的設定檔具有透過 `source_profile` 參數設定用於 IAM 假設角色功能，以角色為基礎的憑證，則 SDK 只會使用指定設定檔的服務組態。它不會使用與其連結的角色的設定檔。例如，使用下列共用 config 檔案：

```
[profile A]
credential_source = Ec2InstanceMetadata
endpoint_url = https://profile-a-endpoint.aws/

[profile B]
source_profile = A
role_arn = arn:aws:iam::123456789012:role/roleB
services = profileB

[services profileB]
ec2 =
    endpoint_url = https://profile-b-ec2-endpoint.aws
```

如果您使用設定檔 B 並在程式碼中呼叫 Amazon EC2，則端點會解析為 `https://profile-b-ec2-endpoint.aws`。如果您的程式碼向任何其他服務發出要求，端點解析將不會遵循任何自訂邏輯。端

點未解析為設定檔 A 中定義的全域端點。若要讓全域端點對設定檔 B 生效，您需要直接在 B 設定檔中設定 `endpoint_url`。如需 `source_profile` 設定的詳細資訊，請參閱[擔任角色登入資料提供者](#)。

## 設定的優先順序

此功能的設定可以同時使用，但每個服務只會優先使用一個值。對於對指定進行的 API 呼叫 AWS 服務，以下順序用於選取值：

1. 程式碼中或服務用戶端本身上設定的任何明確設定，都優先於任何其他設定。
  - 對於 AWS CLI，這是 `--endpoint-url` 命令列參數提供的值。對於 SDK，明確指派可以採用您在執行個體化 AWS 服務用戶端或組態物件時設定的參數形式。
2. 服務特定環境變數提供的值，例如 `AWS_ENDPOINT_URL_DYNAMODB`。
3. `AWS_ENDPOINT_URL` 全域端點環境變數所提供的值。
4. 由 `endpoint_url` 設定所提供的值，巢狀在共用 config 檔案的 `services` 區段中的服務識別符索引鍵下。
5. 直接在共用 config 檔案 `profile` 的內指定的 `endpoint_url` 設定所提供的值。
6. 最後 AWS 服務 會使用個別的任何預設端點 URL。

## 支援 AWS SDKs 和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	是	
<a href="#">適用於 C++ 的 SDK</a>	是	
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	
<a href="#">適用於 Java 的 SDK 1.x</a>	否	

SDK	支援	備註或更多資訊
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	是	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	是	
<a href="#">適用於 Swift 的 SDK</a>	是	
<a href="#">PowerShell V5 的工具</a>	是	
<a href="#">PowerShell V4 的工具</a>	是	

## 服務特定端點的識別符

如需如何以及在何處使用下表中識別符的資訊，請參閱 [服務特定的端點](#)。



<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數
AccessAnalyzer	aws_endpoint_url_accessanalyzer
Account	aws_endpoint_url_account
ACM	aws_endpoint_url_acm
ACM PCA	aws_endpoint_url_acm_pca
Alexa For Business	aws_endpoint_url_alexa_for_business
amp	aws_endpoint_url_amp
Amplify	aws_endpoint_url_amplify
AmplifyBackend	aws_endpoint_url_amplifybackend
AmplifyUIBuilder	aws_endpoint_url_amplifyuibuilder

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API Gateway 服務識別符金鑰
API Gateway	api AWS_ENDPOINT_URL_API_GATEWAY
ApiGatewayManagementApi	api AWS_ENDPOINT_URL_APIGATEWAYMANAGEMENTAPI
ApiGatewayV2	api AWS_ENDPOINT_URL_APIGATEWAYV2
AppConfig	api AWS_ENDPOINT_URL_APPCONFIG
AppConfigData	api AWS_ENDPOINT_URL_APPCONFIGDATA
AppFabric	api AWS_ENDPOINT_URL_APPFABRIC
Appflow	api AWS_ENDPOINT_URL_APPFLOW
AppIntegrations	api AWS_ENDPOINT_URL_APPINTEGRATIONS

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
Application Auto Scaling	a  AWS_ENDPOINT_URL_APPLICATION_AUTO_SCALING o  c
Application Insights	a  AWS_ENDPOINT_URL_APPLICATION_INSIGHTS o  t
ApplicationCostProfiler	a  AWS_ENDPOINT_URL_APPLICATIONCOSTPROFILER o  f
App Mesh	a  AWS_ENDPOINT_URL_APP_MESH
AppRunner	a  AWS_ENDPOINT_URL_APPRUNNER
AppStream	a  AWS_ENDPOINT_URL_APPSTREAM
AppSync	a  AWS_ENDPOINT_URL_APPSYNC
ARC Zonal Shift	a: AWS_ENDPOINT_URL_ARC_ZONAL_SHIFT _:

<b>serviceId</b>	共用 API 方案的 服務 識別 符 金 鑰	<b>AWS_ENDPOINT_URL_&lt;SERVICE&gt;</b> 環境變數
Artifact	a:	AWS_ENDPOINT_URL_ARTIFACT
Athena	a:	AWS_ENDPOINT_URL_ATHENA
AuditManager	a: g:	AWS_ENDPOINT_URL_AUDITMANAGER
Auto Scaling	a: i:	AWS_ENDPOINT_URL_AUTO_SCALING
Auto Scaling Plans	a: i:	AWS_ENDPOINT_URL_AUTO_SCALING_PLANS
b2bi	b:	AWS_ENDPOINT_URL_B2BI
Backup	b:	AWS_ENDPOINT_URL_BACKUP
Backup Gateway	b: t:	AWS_ENDPOINT_URL_BACKUP_GATEWAY
BackupStorage	b: r:	AWS_ENDPOINT_URL_BACKUPSTORAGE
Batch	b:	AWS_ENDPOINT_URL_BATCH

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API 案例的服務識別符金鑰
BCM Data Exports	b: AWS_ENDPOINT_URL_BCM_DATA_EXPORTS e:
Bedrock	b: AWS_ENDPOINT_URL_BEDROCK
Bedrock Agent	b: AWS_ENDPOINT_URL_BEDROCK_AGENT g:
Bedrock Agent Runtime	b: AWS_ENDPOINT_URL_BEDROCK_AGENT_RUNTIME g: ir
Bedrock Runtime	b: AWS_ENDPOINT_URL_BEDROCK_RUNTIME ur
billingconductor	b: AWS_ENDPOINT_URL_BILLINGCONDUCTOR n:
Braket	b: AWS_ENDPOINT_URL_BRAKET
Budgets	b: AWS_ENDPOINT_URL_BUDGETS
Cost Explorer	c: AWS_ENDPOINT_URL_COST_EXPLORER o:

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 AWS 方案的服務識別符金鑰
chatbot	cli AWS_ENDPOINT_URL_CHATBOT
Chime	cli AWS_ENDPOINT_URL_CHIME
Chime SDK Identity	cli AWS_ENDPOINT_URL_CHIME_SDK_IDENTITY _i
Chime SDK Media Pipelines	cli AWS_ENDPOINT_URL_CHIME_SDK_MEDIA_PIPELINES _r p
Chime SDK Meetings	cli AWS_ENDPOINT_URL_CHIME_SDK_MEETINGS _r
Chime SDK Messaging	cli AWS_ENDPOINT_URL_CHIME_SDK_MESSAGING _r g
Chime SDK Voice	cli AWS_ENDPOINT_URL_CHIME_SDK_VOICE _v

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰	環境變數
CleanRooms	c: AWS_ENDPOINT_URL_CLEANROOMS	
CleanRoomsML	c: AWS_ENDPOINT_URL_CLEANROOMSML	
Cloud9	c: AWS_ENDPOINT_URL_CLOUD9	
CloudControl	c: AWS_ENDPOINT_URL_CLOUDCONTROL	
CloudDirectory	c: AWS_ENDPOINT_URL_CLOUDDIRECTORY	
CloudFormation	c: AWS_ENDPOINT_URL_CLOUDFORMATION	
CloudFront	c: AWS_ENDPOINT_URL_CLOUDFRONT	
CloudFront KeyValuesStore	c: AWS_ENDPOINT_URL_CLOUDFRONT_KEYVALUESTORE	

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案 的 服務 識別 符 金 鑰
CloudHSM	c: AWS_ENDPOINT_URL_CLOUDHSM
CloudHSM V2	c: AWS_ENDPOINT_URL_CLOUDHSM_V2 v:
CloudSearch	c: AWS_ENDPOINT_URL_CLOUDSEARCH cl
CloudSearch Domain	c: AWS_ENDPOINT_URL_CLOUDSEARCH_DOMAIN cl
CloudTrail	c: AWS_ENDPOINT_URL_CLOUDTRAIL l
CloudTrail Data	c: AWS_ENDPOINT_URL_CLOUDTRAIL_DATA l_
CloudWatch	c: AWS_ENDPOINT_URL_CLOUDWATCH h
codeartifact	c: AWS_ENDPOINT_URL_CODEARTIFACT a



serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
CodeBuild	codebuild AWS_ENDPOINT_URL_CODEBUILD
CodeCatalyst	codecatalyst AWS_ENDPOINT_URL_CODECATALYST
CodeCommit	codecommit AWS_ENDPOINT_URL_CODECOMMIT
CodeDeploy	codedeploy AWS_ENDPOINT_URL_CODEDEPLOY
CodeGuru Reviewer	codeguru-reviewer AWS_ENDPOINT_URL_CODEGURU_REVIEWER
CodeGuru Security	codeguru-security AWS_ENDPOINT_URL_CODEGURU_SECURITY
CodeGuruProfiler	codeguru-profiler AWS_ENDPOINT_URL_CODEGURUPROFILER
CodePipeline	codepipeline AWS_ENDPOINT_URL_CODEPIPELINE

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數
CodeStar	code AWS_ENDPOINT_URL_CODESTAR
CodeStar connections	code AWS_ENDPOINT_URL_CODESTAR_CONNECTIONS code
codestar notifications	code AWS_ENDPOINT_URL_CODESTAR_NOTIFICATIONS code
Cognito Identity	code AWS_ENDPOINT_URL_COGNITO_IDENTITY code
Cognito Identity Provider	code AWS_ENDPOINT_URL_COGNITO_IDENTITY_PROVIDER code
Cognito Sync	code AWS_ENDPOINT_URL_COGNITO_SYNC code
Comprehend	code AWS_ENDPOINT_URL_COMPREHEND code

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API 方案的服務識別符金鑰
ComprehendMedical	AWS_ENDPOINT_URL_COMPREHENDMEDICAL
Compute Optimizer	AWS_ENDPOINT_URL_COMPUTE_OPTIMIZER
Config Service	AWS_ENDPOINT_URL_CONFIG_SERVICE
Connect	AWS_ENDPOINT_URL_CONNECT
Connect Contact Lens	AWS_ENDPOINT_URL_CONNECT_CONTACT_LENS
ConnectCampaigns	AWS_ENDPOINT_URL_CONNECTCAMPAIGNS
ConnectCases	AWS_ENDPOINT_URL_CONNECTCASES
ConnectParticipant	AWS_ENDPOINT_URL_CONNECTPARTICIPANT

serviceId	共用 AWS 方案 的 服務 識別 符 金 鑰 環境變數
ControlTower	c: AWS_ENDPOINT_URL_CONTROLTOWER w:
Cost Optimization Hub	c: AWS_ENDPOINT_URL_COST_OPTIMIZATION_HUB m: h:
Cost and Usage Report Service	c: AWS_ENDPOINT_URL_COST_AND_USAGE_REPO u: RT_SERVICE o: c:
Customer Profiles	c: AWS_ENDPOINT_URL_CUSTOMER_PROFILES p:
DataBrew	d: AWS_ENDPOINT_URL_DATABREW
DataExchange	d: AWS_ENDPOINT_URL_DATAEXCHANGE n:
Data Pipeline	d: AWS_ENDPOINT_URL_DATA_PIPELINE l:
DataSync	d: AWS_ENDPOINT_URL_DATASYNC

<b>serviceId</b>	<b>共用 AWS_ENDPOINT_URL_&lt;SERVICE&gt; 環境變數</b>
DataZone	d: AWS_ENDPOINT_URL_DATAZONE
DAX	d: AWS_ENDPOINT_URL_DAX
Detective	d: AWS_ENDPOINT_URL_DETECTIVE
Device Farm	d: AWS_ENDPOINT_URL_DEVICE_FARM
DevOps Guru	d: AWS_ENDPOINT_URL_DEVOPS_GURU
Direct Connect	d: AWS_ENDPOINT_URL_DIRECT_CONNECT
Application Discovery Service	a: AWS_ENDPOINT_URL_APPLICATION_DISCOVERY_SERVICE
DLM	d: AWS_ENDPOINT_URL_DLM

<b>serviceId</b>	<b>共用 AWS_ENDPOINT_URL_&lt;SERVICE&gt; 環境變數</b>
Database Migration Service	d: AWS_ENDPOINT_URL_DATABASE_MIGRATION_ m: SERVICE _:
DocDB	d: AWS_ENDPOINT_URL_DOCDB
DocDB Elastic	d: AWS_ENDPOINT_URL_DOCDB_ELASTIC s:
drs	d: AWS_ENDPOINT_URL_DRS
Directory Service	d: AWS_ENDPOINT_URL_DIRECTORY_SERVICE _:
DynamoDB	d: AWS_ENDPOINT_URL_DYNAMODB
DynamoDB Streams	d: AWS_ENDPOINT_URL_DYNAMODB_STREAMS s:
EBS	e: AWS_ENDPOINT_URL_EBS
EC2	e: AWS_ENDPOINT_URL_EC2

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
EC2 Instance Connect	e: AWS_ENDPOINT_URL_EC2_INSTANCE_CONNECT
ECR	e: AWS_ENDPOINT_URL_ECR
ECR PUBLIC	e: AWS_ENDPOINT_URL_ECR_PUBLIC
ECS	e: AWS_ENDPOINT_URL_ECS
EFS	e: AWS_ENDPOINT_URL_EFS
EKS	e: AWS_ENDPOINT_URL_EKS
EKS Auth	e: AWS_ENDPOINT_URL_EKS_AUTH
Elastic Inference	e: AWS_ENDPOINT_URL_ELASTIC_INFERENCE
ElastiCache	e: AWS_ENDPOINT_URL_ELASTICACHE
Elastic Beanstalk	e: AWS_ENDPOINT_URL_ELASTIC_BEANSTALK

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
Elastic Transcoder	e: AWS_ENDPOINT_URL_ELASTIC_TRANSCODER r:
Elastic Load Balancing	e: AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING o: c:
Elastic Load Balancing v2	e: AWS_ENDPOINT_URL_ELASTIC_LOAD_BALANCING_V2 o: c:
EMR	er AWS_ENDPOINT_URL_EMR
EMR containers	er AWS_ENDPOINT_URL_EMR_CONTAINERS i:
EMR Serverless	er AWS_ENDPOINT_URL_EMR_SERVERLESS r:
EntityResolution	er AWS_ENDPOINT_URL_ENTITYRESOLUTION o:



<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API 方案的服務識別符金鑰
Elasticsearch Service	e: AWS_ENDPOINT_URL_ELASTICSEARCH_SERVICE a: i:
EventBridge	e: AWS_ENDPOINT_URL_EVENTBRIDGE g:
Evidently	e: AWS_ENDPOINT_URL_EVIDENTLY
finspace	f: AWS_ENDPOINT_URL_FINSPLACE
finspace data	f: AWS_ENDPOINT_URL_FINSPLACE_DATA d:
Firehose	f: AWS_ENDPOINT_URL_FIREHOSE
fis	f: AWS_ENDPOINT_URL_FIS
FMS	fr: AWS_ENDPOINT_URL_FMS
forecast	f: AWS_ENDPOINT_URL_FORECAST
forecastquery	f: AWS_ENDPOINT_URL_FORECASTQUERY u:

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
FraudDetector	f: AWS_ENDPOINT_URL_FRAUDETECTOR c1
FreeTier	f: AWS_ENDPOINT_URL_FREETIER
FSx	f: AWS_ENDPOINT_URL_FSX
GameLift	g: AWS_ENDPOINT_URL_GAMELIFT
Glacier	g: AWS_ENDPOINT_URL_GLACIER
Global Accelerator	g: AWS_ENDPOINT_URL_GLOBAL_ACCELERATOR c6
Glue	g: AWS_ENDPOINT_URL_GLUE
grafana	g: AWS_ENDPOINT_URL_GRAFANA
Greengrass	g: AWS_ENDPOINT_URL_GREENGRASS s
GreengrassV2	g: AWS_ENDPOINT_URL_GREENGRASSV2 s\

<b>serviceId</b>	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 <b>AWS_ENDPOINT_URL_&lt;SERVICE&gt;</b> 環境變數
GroundStation	g: AWS_ENDPOINT_URL_GROUNDSTATION t:
GuardDuty	g: AWS_ENDPOINT_URL_GUARDDUTY
Health	h: AWS_ENDPOINT_URL_HEALTH
HealthLake	h: AWS_ENDPOINT_URL_HEALTHLAKE e
Honeycode	h: AWS_ENDPOINT_URL_HONEYCODE
IAM	i: AWS_ENDPOINT_URL_IAM
identitystore	i: AWS_ENDPOINT_URL_IDENTITYSTORE t:
imagebuilder	i: AWS_ENDPOINT_URL_IMAGEBUILDER d:
ImportExport	i: AWS_ENDPOINT_URL_IMPORTEXPORT o:

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API 方案的服務識別符金鑰
Inspector	i AWS_ENDPOINT_URL_INSPECTOR
Inspector Scan	i AWS_ENDPOINT_URL_INSPECTOR_SCAN_
Inspector2	i AWS_ENDPOINT_URL_INSPECTOR2 2
InternetMonitor	i AWS_ENDPOINT_URL_INTERNETMONITOR o
IoT	i AWS_ENDPOINT_URL_IOT
IoT Data Plane	i AWS_ENDPOINT_URL_IOT_DATA_PLANE p
IoT Jobs Data Plane	i AWS_ENDPOINT_URL_IOT_JOBS_DATA_PLANE d e
IoT 1Click Devices Service	i AWS_ENDPOINT_URL_IOT_1CLICK_DEVICES_ k_SERVICE_

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
IoT 1Click Projects	iots AWS_ENDPOINT_URL_IOT_1CLICK_PROJECTS
IoTAnalytics	iots AWS_ENDPOINT_URL_IOTANALYTICS
IotDeviceAdvisor	iots AWS_ENDPOINT_URL_IOTDEVICEADVISOR
IoT Events	iots AWS_ENDPOINT_URL_IOT_EVENTS
IoT Events Data	iots AWS_ENDPOINT_URL_IOT_EVENTS_DATA
IoTFleetHub	iots AWS_ENDPOINT_URL_IOTFLEETHUB
IoTFleetWise	iots AWS_ENDPOINT_URL_IOTFLEETWISE

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
IoTSecureTunneling	i\ AWS_ENDPOINT_URL_IOTSECURETUNNELING t\
IoTSiteWise	i\ AWS_ENDPOINT_URL_IOTSITWISE s\
IoTThingsGraph	i\ AWS_ENDPOINT_URL_IOTTHINGSGRAPH g\
IoTTwinMaker	i\ AWS_ENDPOINT_URL_IOTTWINMAKER k\
IoT Wireless	i\ AWS_ENDPOINT_URL_IOT_WIRELESS e\
ivs	i\ AWS_ENDPOINT_URL_IVS
IVS RealTime	i\ AWS_ENDPOINT_URL_IVS_REALTIME i\
ivschat	i\ AWS_ENDPOINT_URL_IVSCHAT
Kafka	k\ AWS_ENDPOINT_URL_KAFKA

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
KafkaConnect	k: AWS_ENDPOINT_URL_KAFKACONNECT e:
kendra	k: AWS_ENDPOINT_URL_KENDRA
Kendra Ranking	k: AWS_ENDPOINT_URL_KENDRA_RANKING n:
Keyspaces	k: AWS_ENDPOINT_URL_KEYSPACES
Kinesis	k: AWS_ENDPOINT_URL_KINESIS
Kinesis Video Archived Media	k: AWS_ENDPOINT_URL_KINESIS_VIDEO_ARCHIVED_MEDIA i:
Kinesis Video Media	k: AWS_ENDPOINT_URL_KINESIS_VIDEO_MEDIA i:

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
Kinesis Video Signaling	k: AWS_ENDPOINT_URL_KINESIS_VIDEO_SIGNALING i: a:
Kinesis Video WebRTC Storage	k: AWS_ENDPOINT_URL_KINESIS_VIDEO_WEBRT i: C_STORAGE t: e:
Kinesis Analytics	k: AWS_ENDPOINT_URL_KINESIS_ANALYTICS n:
Kinesis Analytics V2	k: AWS_ENDPOINT_URL_KINESIS_ANALYTICS_V2 n: v:
Kinesis Video	k: AWS_ENDPOINT_URL_KINESIS_VIDEO i:
KMS	k: AWS_ENDPOINT_URL_KMS
LakeFormation	l: AWS_ENDPOINT_URL_LAKEFORMATION t:



serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
Lambda	1: AWS_ENDPOINT_URL_LAMBDA
Launch Wizard	1: AWS_ENDPOINT_URL_LAUNCH_WIZARD 2:
Lex Model Building Service	1: AWS_ENDPOINT_URL_LEX_MODEL_BUILDING_SERVICE 2:
Lex Runtime Service	1: AWS_ENDPOINT_URL_LEX_RUNTIME_SERVICE me
Lex Models V2	1: AWS_ENDPOINT_URL_LEX_MODELS_V2 S_
Lex Runtime V2	1: AWS_ENDPOINT_URL_LEX_RUNTIME_V2 me
License Manager	1: AWS_ENDPOINT_URL_LICENSE_MANAGER at

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 AWS 方案的服務識別符金鑰
License Manager Linux Subscriptions	l: AWS_ENDPOINT_URL_LICENSE_MANAGER_LINUX_SUBSCRIPTIONS
License Manager User Subscriptions	l: AWS_ENDPOINT_URL_LICENSE_MANAGER_USER_SUBSCRIPTIONS
Lightsail	l: AWS_ENDPOINT_URL_LIGHTSAIL
Location	l: AWS_ENDPOINT_URL_LOCATION
CloudWatch Logs	c: AWS_ENDPOINT_URL_CLOUDWATCH_LOGS
LookoutEquipment	l: AWS_ENDPOINT_URL_LOOKOUTEQUIPMENT
LookoutMetrics	l: AWS_ENDPOINT_URL_LOOKOUTMETRICS

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 API 方案的服務識別符金鑰
LookoutVision	LookoutVision: AWS_ENDPOINT_URL_LOOKOUTVISION
m2	m2: AWS_ENDPOINT_URL_M2
Machine Learning	machine-learning: AWS_ENDPOINT_URL_MACHINE_LEARNING
Macie2	macie2: AWS_ENDPOINT_URL_MACIE2
ManagedBlockchain	managed-blockchain: AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN
ManagedBlockchain Query	managed-blockchain-query: AWS_ENDPOINT_URL_MANAGEDBLOCKCHAIN_QUERY
Marketplace Agreement	marketplace-agreement: AWS_ENDPOINT_URL_MARKETPLACE_AGREEMENT
Marketplace Catalog	marketplace-catalog: AWS_ENDPOINT_URL_MARKETPLACE_CATALOG

<b>serviceId</b>	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
Marketplace Deployment	m: AWS_ENDPOINT_URL_MARKETPLACE_DEPLOYMENT c: m:
Marketplace Entitlement Service	m: AWS_ENDPOINT_URL_MARKETPLACE_ENTITLE c: MENT_SERVICE e: v:
Marketplace Commerce Analytics	m: AWS_ENDPOINT_URL_MARKETPLACE_COMMERC c: E_ANALYTICS c: i:
MediaConnect	m: AWS_ENDPOINT_URL_MEDIACONNECT e:
MediaConvert	m: AWS_ENDPOINT_URL_MEDIACONVERT e:
MediaLive	m: AWS_ENDPOINT_URL_MEDIALIVE

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
MediaPackage	media_aws_endpoint_url_mediapackage
MediaPackage Vod	media_aws_endpoint_url_mediapackage_vod
MediaPackageV2	media_aws_endpoint_url_mediapackagev2
MediaStore	media_aws_endpoint_url_mediastore
MediaStore Data	media_aws_endpoint_url_mediastore_data
MediaTailor	media_aws_endpoint_url_mediatailor
Medical Imaging	media_aws_endpoint_url_medical_imaging
MemoryDB	media_aws_endpoint_url_memorydb

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 環境變數
Marketplace Metering	m: AWS_ENDPOINT_URL_MARKETPLACE_METERING cli
Migration Hub	m: AWS_ENDPOINT_URL_MIGRATION_HUB cli
mgn	m: AWS_ENDPOINT_URL_MGN
Migration Hub Refactor Spaces	m: AWS_ENDPOINT_URL_MIGRATION_HUB_REFACTOR_SPACES cli
MigrationHub Config	m: AWS_ENDPOINT_URL_MIGRATIONHUB_CONFIG cli
MigrationHubOrchestrator	m: AWS_ENDPOINT_URL_MIGRATIONHUBORCHESTRATOR cli

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數
MigrationHubStrategy	m: AWS_ENDPOINT_URL_MIGRATIONHUBSTRATEGY h: g:
Mobile	m: AWS_ENDPOINT_URL_MOBILE
mq	m: AWS_ENDPOINT_URL_MQ
MTurk	m: AWS_ENDPOINT_URL_MTURK
MWAA	m: AWS_ENDPOINT_URL_MWAA
Neptune	n: AWS_ENDPOINT_URL_NEPTUNE
Neptune Graph	n: AWS_ENDPOINT_URL_NEPTUNE_GRAPH r:
neptunedata	n: AWS_ENDPOINT_URL_NEPTUNEDATA t:
Network Firewall	n: AWS_ENDPOINT_URL_NETWORK_FIREWALL i:
NetworkManager	n: AWS_ENDPOINT_URL_NETWORKMANAGER n:

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
NetworkMonitor	network: AWS_ENDPOINT_URL_NETWORKMONITOR
nimble	network: AWS_ENDPOINT_URL_NIMBLE
OAM	oam: AWS_ENDPOINT_URL_OAM
Omics	omics: AWS_ENDPOINT_URL_OMICS
OpenSearch	opensearch: AWS_ENDPOINT_URL_OPENSEARCH
OpenSearchServerless	opensearchserverless: AWS_ENDPOINT_URL_OPENSEARCHSERVERLESS
OpsWorks	opsworks: AWS_ENDPOINT_URL_OPSWORKS
OpsWorksCM	opsworkscm: AWS_ENDPOINT_URL_OPSWORKSCM
Organizations	organizations: AWS_ENDPOINT_URL_ORGANIZATIONS
OSIS	osis: AWS_ENDPOINT_URL_OSIS



serviceId	AWS_ENDPOINT_URL_<SERVICE> 環境變數
Outposts	AWS_ENDPOINT_URL_OUTPOSTS
p8data	AWS_ENDPOINT_URL_P8DATA
p8data	AWS_ENDPOINT_URL_P8DATA
Panorama	AWS_ENDPOINT_URL_PANORAMA
Payment Cryptography	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY
Payment Cryptography Data	AWS_ENDPOINT_URL_PAYMENT_CRYPTOGRAPHY_DATA
Pca Connector Ad	AWS_ENDPOINT_URL_PCA_CONNECTOR_AD
Personalize	AWS_ENDPOINT_URL_PERSONALIZE

<b>serviceId</b>	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
Personalize Events	p: AWS_ENDPOINT_URL_PERSONALIZE_EVENTS z:
Personalize Runtime	p: AWS_ENDPOINT_URL_PERSONALIZE_RUNTIME z: e
PI	p: AWS_ENDPOINT_URL_PI
Pinpoint	p: AWS_ENDPOINT_URL_PINPOINT
Pinpoint Email	p: AWS_ENDPOINT_URL_PINPOINT_EMAIL er
Pinpoint SMS Voice	p: AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE sr
Pinpoint SMS Voice V2	p: AWS_ENDPOINT_URL_PINPOINT_SMS_VOICE_V2 sr _`
Pipes	p: AWS_ENDPOINT_URL_PIPES

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
Polly	p: AWS_ENDPOINT_URL_POLLY
Pricing	p: AWS_ENDPOINT_URL_PRICING
PrivateNetworks	p: AWS_ENDPOINT_URL_PRIVATENETWORKS t:
Proton	p: AWS_ENDPOINT_URL_PROTON
QBusiness	q: AWS_ENDPOINT_URL_QBUSINESS
QConnect	q: AWS_ENDPOINT_URL_QCONNECT
QLDB	q: AWS_ENDPOINT_URL_QLDB
QLDB Session	q: AWS_ENDPOINT_URL_QLDB_SESSION i:
QuickSight	q: AWS_ENDPOINT_URL_QUICKSIGHT t:
RAM	r: AWS_ENDPOINT_URL_RAM
rbin	r: AWS_ENDPOINT_URL_RBIN

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 案例的服務識別符金鑰
RDS	AWS_ENDPOINT_URL_RDS
RDS Data	AWS_ENDPOINT_URL_RDS_DATA
Redshift	AWS_ENDPOINT_URL_REDSHIFT
Redshift Data	AWS_ENDPOINT_URL_REDSHIFT_DATA
Redshift Serverless	AWS_ENDPOINT_URL_REDSHIFT_SERVERLESS
Rekognition	AWS_ENDPOINT_URL_REKOGNITION
repostspace	AWS_ENDPOINT_URL_REPOSTSPACE
resiliencehub	AWS_ENDPOINT_URL_RESILIENCEHUB

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
Resource Explorer 2	<pre> AWS_ENDPOINT_URL_RESOURCE_EXPLORER_2 </pre>
Resource Groups	<pre> AWS_ENDPOINT_URL_RESOURCE_GROUPS </pre>
Resource Groups Tagging API	<pre> AWS_ENDPOINT_URL_RESOURCE_GROUPS_TAGGING_API </pre>
RoboMaker	<pre> AWS_ENDPOINT_URL_ROBOMAKER </pre>
RolesAnywhere	<pre> AWS_ENDPOINT_URL_ROLESEANYWHERE </pre>
Route 53	<pre> AWS_ENDPOINT_URL_ROUTE_53 </pre>
Route53 Recovery Cluster	<pre> AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CLUSTER </pre>

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 環境變數
Route53 Recovery Control Config	r: AWS_ENDPOINT_URL_ROUTE53_RECOVERY_CO n: NTRONL_CONFIG
Route53 Recovery Readiness	r: AWS_ENDPOINT_URL_ROUTE53_RECOVERY_RE e: ADINESS
Route 53 Domains	r: AWS_ENDPOINT_URL_ROUTE_53_DOMAINS d:
Route53Resolver	r: AWS_ENDPOINT_URL_ROUTE53RESOLVER s:
RUM	r: AWS_ENDPOINT_URL_RUM
S3	s: AWS_ENDPOINT_URL_S3
S3 Control	s: AWS_ENDPOINT_URL_S3_CONTROL l
S3Outposts	s: AWS_ENDPOINT_URL_S3OUTPOSTS s

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
SageMaker	s: AWS_ENDPOINT_URL_SAGEMAKER
SageMaker A2I Runtime	s: AWS_ENDPOINT_URL_SAGEMAKER_A2I_RUNTIME _i ir
Sagemaker Edge	s: AWS_ENDPOINT_URL_SAGEMAKER_EDGE _e
SageMaker FeatureStore Runtime	s: AWS_ENDPOINT_URL_SAGEMAKER_FEATURESTORE_RUNTIME _f t ir
SageMaker Geospatial	s: AWS_ENDPOINT_URL_SAGEMAKER_GEOSPATIAL _g a:
SageMaker Metrics	s: AWS_ENDPOINT_URL_SAGEMAKER_METRICS _m
SageMaker Runtime	s: AWS_ENDPOINT_URL_SAGEMAKER_RUNTIME _r

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
savingsplans	s: AWS_ENDPOINT_URL_SAVINGSPLANS a:
Scheduler	s: AWS_ENDPOINT_URL_SCHEDULER
schemas	s: AWS_ENDPOINT_URL_SCHEMAS
SimpleDB	s: AWS_ENDPOINT_URL_SIMPLEDB
Secrets Manager	s: AWS_ENDPOINT_URL_SECRETS_MANAGER a:
SecurityHub	s: AWS_ENDPOINT_URL_SECURITYHUB u:
SecurityLake	s: AWS_ENDPOINT_URL_SECURITYLAKE a:
ServerlessApplicationRepository	s: AWS_ENDPOINT_URL_SERVERLESSAPPLICATI s: ONREPOSITORY i: t:



serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
Service Quotas	s: AWS_ENDPOINT_URL_SERVICE_QUOTAS u:
Service Catalog	s: AWS_ENDPOINT_URL_SERVICE_CATALOG a:
Service Catalog AppRegistry	s: AWS_ENDPOINT_URL_SERVICE_CATALOG_APP a: REGISTRY p:
ServiceDiscovery	s: AWS_ENDPOINT_URL_SERVICEDISCOVERY s:
SES	s: AWS_ENDPOINT_URL_SES
SESV2	s: AWS_ENDPOINT_URL_SESV2
Shield	s: AWS_ENDPOINT_URL_SHIELD
signer	s: AWS_ENDPOINT_URL_SIGNER
SimSpaceWeaver	s: AWS_ENDPOINT_URL_SIMSPACEWEAVER e:

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 AWS_ENDPOINT_URL_<SERVICE> 環境變數
SMS	s: AWS_ENDPOINT_URL_SMS
Snow Device Management	s: AWS_ENDPOINT_URL_SNOW_DEVICE_MANAGEMENT
Snowball	s: AWS_ENDPOINT_URL_SNOWBALL
SNS	s: AWS_ENDPOINT_URL_SNS
SQS	s: AWS_ENDPOINT_URL_SQS
SSM	s: AWS_ENDPOINT_URL_SSM
SSM Contacts	s: AWS_ENDPOINT_URL_SSM_CONTACTS
SSM Incidents	s: AWS_ENDPOINT_URL_SSM_INCIDENTS
Ssm Sap	s: AWS_ENDPOINT_URL_SSM_SAP
SSO	s: AWS_ENDPOINT_URL_SSO

serviceId	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數 方案的服務識別符金鑰
SSO Admin	s: AWS_ENDPOINT_URL_SSO_ADMIN
SSO OIDC	s: AWS_ENDPOINT_URL_SSO_OIDC
SFN	s: AWS_ENDPOINT_URL_SFN
Storage Gateway	s: AWS_ENDPOINT_URL_STORAGE_GATEWAY
STS	s: AWS_ENDPOINT_URL_STS
SupplyChain	s: AWS_ENDPOINT_URL_SUPPLYCHAIN
Support	s: AWS_ENDPOINT_URL_SUPPORT
Support App	s: AWS_ENDPOINT_URL_SUPPORT_APP
SWF	s: AWS_ENDPOINT_URL_SWF
synthetics	s: AWS_ENDPOINT_URL_SYNTHETICS

serviceId	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 環境變數
Textract	t: AWS_ENDPOINT_URL_TEXTRACT
Timestream InfluxDB	t: AWS_ENDPOINT_URL_TIMESTREAM_INFLUXDB m_ b
Timestream Query	t: AWS_ENDPOINT_URL_TIMESTREAM_QUERY m_
Timestream Write	t: AWS_ENDPOINT_URL_TIMESTREAM_WRITE m_
tnb	t: AWS_ENDPOINT_URL_TNB
Transcribe	t: AWS_ENDPOINT_URL_TRANSCRIBE e
Transfer	t: AWS_ENDPOINT_URL_TRANSFER
Translate	t: AWS_ENDPOINT_URL_TRANSLATE
TrustedAdvisor	t: AWS_ENDPOINT_URL_TRUSTEDADVISOR v:

<b>serviceId</b>	共用 AWS CLI 案 的 服 務 識 別 符 金 鑰 環境變數
VerifiedPermissions	v: AWS_ENDPOINT_URL_VERIFIEDPERMISSIONS e: s
Voice ID	v: AWS_ENDPOINT_URL_VOICE_ID
VPC Lattice	v: AWS_ENDPOINT_URL_VPC_LATTICE c:
WAF	w: AWS_ENDPOINT_URL_WAF
WAF Regional	w: AWS_ENDPOINT_URL_WAF_REGIONAL n:
WAFV2	w: AWS_ENDPOINT_URL_WAFV2
WellArchitected	w: AWS_ENDPOINT_URL_WELLARCHITECTED t:
Wisdom	w: AWS_ENDPOINT_URL_WISDOM
WorkDocs	w: AWS_ENDPOINT_URL_WORKDOCS
WorkLink	w: AWS_ENDPOINT_URL_WORKLINK

<b>serviceId</b>	共用 AWS_ENDPOINT_URL_<SERVICE> 環境變數
WorkMail	wc AWS_ENDPOINT_URL_WORKMAIL
WorkMailMessageFlow	wc AWS_ENDPOINT_URL_WORKMAILMESSAGEFLOW e: w
WorkSpaces	wc AWS_ENDPOINT_URL_WORKSPACES S
WorkSpaces Thin Client	wc AWS_ENDPOINT_URL_WORKSPACES_THIN_CLIENT S_ i
WorkSpaces Web	wc AWS_ENDPOINT_URL_WORKSPACES_WEB S_
XRay	x: AWS_ENDPOINT_URL_XRAY

## 智慧組態預設值

### Note

如需了解設定頁面配置或解譯以下 Support AWS SDKs和工具資料表的說明，請參閱 [了解本指南的設定頁面](#)。

透過智慧型組態預設值功能，AWS SDKs可以為其他組態設定提供預先定義的最佳化預設值。

使用下列項目設定此功能：

**defaults\_mode** - 共用 AWS **config**檔案設定, **AWS\_DEFAULTS\_MODE** - 環境變數,  
**aws.defaultsMode** - JVM 系統屬性：僅限 Java/Kotlin

使用此設定，您可以選擇與您的應用程式架構相符的模式，然後為您的應用程式提供最佳化的預設值。如果 AWS SDK 設定有明確設定的值，則該值一律優先。如果 AWS SDK 設定沒有明確設定的值，而且 **defaults\_mode** 不等於舊版，則此功能可以為您的應用程式最佳化的各種設定提供不同的預設值。設定可能包括下列項目：HTTP 通訊設定、重試行為、服務區域端點設定，以及可能的任何 SDK 相關組態。使用此功能的客戶可以取得針對常見使用案例量身打造的新組態預設值。如果您的 **defaults\_mode** 不等於 **legacy**，我們建議您在升級 SDK 時執行應用程式的測試，因為提供的預設值可能會隨著最佳實務的演進而變更。

預設值：legacy

注意：軟體 SDKs 的新主要版本預設為 **standard**。

有效值：

- **legacy** – 提供預設設定，這些設定會因 SDK 而異，並在建立之前已存在 **defaults\_mode**。
- **standard** – 提供應可在大多數情況下安全執行的最新建議預設值。
- **in-region** – 以標準模式為基礎，包括針對 AWS 服務在相同內呼叫的應用程式量身打造的最佳化 AWS 區域。
- **cross-region** – 以標準模式為基礎，包括為 AWS 服務在不同區域中呼叫的應用程式量身打造的最佳化。
- **mobile** – 以標準模式建置，並包含專為行動應用程式量身打造的最佳化。
- **auto** – 以標準模式為基礎，並包含實驗性功能。SDK 會嘗試探索執行時期環境，自動判斷適當的設定。自動偵測是以啟發式為基礎，不提供 100% 的準確性。如果無法判斷執行時間環境，則

會使用 `standard` 模式。自動偵測可能會查詢[執行個體中繼資料](#)，這可能會導致延遲。如果啟動延遲對您的應用程式至關重要，我們建議您改為明確選擇 `defaults_mode`。

在 `config` 檔案中設定此值的範例：

```
[default]
defaults_mode = standard
```

下列參數可能會根據的選擇進行最佳化`defaults_mode`：

- `retryMode` – 指定 SDK 嘗試重試的方式。請參閱 [重試行為](#)。
- `stsRegionalEndpoints` – 指定 SDK 如何決定用來與 AWS Security Token Service () 通訊的 AWS 服務 端點AWS STS。請參閱 [AWS STS 區域端點](#)。
- `s3UsEast1RegionalEndpoints` – 指定 SDK 如何決定用來與 `us-east-1` 區域的 Amazon S3 通訊 AWS 的服務端點。
- `connectTimeoutInMillis` – 在通訊端上進行初始連線嘗試之後，逾時之前的時間量。如果用戶端未收到連線交握完成，用戶端會放棄操作並失敗。
- `tlsNegotiationTimeoutInMillis` – 從傳送 CLIENT HELLO 訊息到用戶端和伺服器完全交涉密碼和交換金鑰的時間，TLS 交握可以花費的時間上限。

每個設定的預設值會根據為您的應用程式`defaults_mode`選取的而變更。這些值目前設定如下（可能會有所變更）：

參數	<b>standard</b> 模式	<b>in-region</b> 模式	<b>cross-region</b> 模式	<b>mobile</b> 模式
<code>retryMode</code>	<code>standard</code>	<code>standard</code>	<code>standard</code>	<code>standard</code>
<code>stsRegionalEndpoints</code>	<code>regional</code>	<code>regional</code>	<code>regional</code>	<code>regional</code>
<code>s3UsEast1RegionalEndpoints</code>	<code>regional</code>	<code>regional</code>	<code>regional</code>	<code>regional</code>



參數	standard 模式	in-region 模式	cross-region 模式	mobile 模式
connectTimeoutInMillis	3100	1100	3100	30000
tlsNegotiationTimeoutInMillis	3100	1100	3100	30000

例如，如果您 `defaults_mode` 選取的是 `standard`，則 `standard` 的值會指派給 `retry_mode`（從有效 `retry_mode` 選項），而 `regional` 的值會指派給 `stsRegionalEndpoints`（從有效 `stsRegionalEndpoints` 選項）。

## 支援 AWS SDKs 和工具

下列 SDKs 支援本主題中所述的功能和設定。會記下任何部分例外狀況。適用於 Java 的 AWS SDK 和適用於 Kotlin 的 AWS SDK 僅支援任何 JVM 系統屬性設定。

SDK	支援	備註或更多資訊
<a href="#">AWS CLI v2</a>	否	
<a href="#">適用於 C++ 的 SDK</a>	是	參數未最佳化： <code>stsRegionalEndpoints</code> 、 <code>s3UsEast1RegionalEndpoints</code> 、 <code>tlsNegotiationTimeoutInMillis</code> 。
<a href="#">適用於 Go V2 的 SDK (1.x)</a>	是	參數未最佳化： <code>retryMode</code> 、 <code>stsRegionalEndpoints</code> 、 <code>s3UsEast1RegionalEndpoints</code> 。

SDK	支援	備註或更多資訊
<a href="#">適用於 Go 的 SDK 1.x (V1)</a>	否	
<a href="#">適用於 Java 的 SDK 2.x</a>	是	參數未最佳化：stsRegionalEndpoints。
<a href="#">適用於 Java 的 SDK 1.x</a>	否	
<a href="#">適用於 JavaScript 3.x 的 SDK</a>	是	未最佳化的參數：stsRegionalEndpoints、s3UsEast1RegionalEndpoints、tlsNegotiationTimeoutInMillis。connectTimeoutInMillis 稱為 connectionTimeout。
<a href="#">適用於 JavaScript 2.x 的 SDK</a>	否	
<a href="#">適用於 Kotlin 的 SDK</a>	否	
<a href="#">適用於 .NET 4.x 的 SDK</a>	是	參數未最佳化：connectTimeoutInMillis、tlsNegotiationTimeoutInMillis。
<a href="#">適用於 .NET 3.x 的 SDK</a>	是	參數未最佳化：connectTimeoutInMillis、tlsNegotiationTimeoutInMillis。
<a href="#">適用於 PHP 的 SDK 3.x</a>	是	參數未最佳化：tlsNegotiationTimeoutInMillis。

SDK	支援	備註或更多資訊
<a href="#">適用於 Python 的 SDK (Boto3)</a>	是	參數未最佳化：tlsNegotiationTimeoutInMilliseconds。
<a href="#">適用於 Ruby 的 SDK 3.x</a>	是	
<a href="#">適用於 Rust 的 SDK</a>	否	
<a href="#">適用於 Swift 的 SDK</a>	否	
<a href="#">PowerShell V5 的工具</a>	是	參數未最佳化：connectTimeoutInMilliseconds、tlsNegotiationTimeoutInMilliseconds。
<a href="#">PowerShell V4 的工具</a>	是	參數未最佳化：connectTimeoutInMilliseconds、tlsNegotiationTimeoutInMilliseconds。

# AWS 通用執行期 (CRT) 程式庫

AWS Common Runtime (CRT) 程式庫是 SDKs的基本程式庫。CRT 是以 C 撰寫的獨立套件模組化系列。每個套件都為不同的必要功能提供良好的效能和最少的佔用空間。這些功能是常見且跨所有 SDKs 共用的，可提供更好的程式碼重複使用、最佳化和準確性。套件包括：

- [awslabs/aws-c-auth](#)：AWS 用戶端身分驗證（標準登入資料提供者和簽署 (sigv4)）
- [awslabs/aws-c-cal](#)：密碼編譯基本類型、雜湊 (MD5、SHA256, SHA256 HMAC)、簽署者、AES
- [awslabs/aws-c-common](#)：基本資料結構、執行緒/同步基本類型、緩衝區管理、stdlib 相關函數
- [awslabs/aws-c-compression](#)：壓縮演算法 (Huffman 編碼/解碼)
- [awslabs/aws-c-event-stream](#)：事件串流訊息處理（標頭、排除、承載、crc/trailer）、透過事件串流的遠端程序呼叫 (RPC) 實作
- [awslabs/aws-c-http](#)：HTTP/1.1 和 HTTP/2 規格的 C99 實作
- [awslabs/aws-c-io](#)：通訊端 (TCP、UDP)、DNS、管道、事件迴圈、頻道、SSL/TLS
- [awslabs/aws-c-iot](#)：與裝置整合 AWS IoT 雲端服務的 C99 實作
- [awslabs/aws-c-mqtt](#)：物聯網 (IoT) 的標準輕量型傳訊通訊協定
- [awslabs/aws-c-s3](#)：用於與 Amazon S3 服務通訊的 C99 程式庫實作，旨在將高頻寬 Amazon EC2 執行個體的輸送量最大化
- [awslabs/aws-c-sdkutils](#)：用於剖析和管理 AWS 設定檔的公用程式程式庫
- [awslabs/aws-checksums](#)：跨平台硬體加速 CRC32c 和 CRC32，可恢復至高效的軟體實作
- [awslabs/aws-lc](#)：由 AWS 密碼編譯團隊為 AWS 及其客戶維護的一般用途密碼編譯程式庫，以 Google BoringSSL 專案和 OpenSSL 專案的程式碼為基礎
- [awslabs/s2n](#)：TLS/SSL 通訊協定的 C99 實作，設計為小型且快速，以安全性為優先

CRT 可透過 Go 和 Rust 以外的所有 SDKs 使用。

## CRT 相依性

CRT 程式庫形成複雜的關係和相依性網。如果您需要直接從來源建置 CRT，了解這些關係會很有幫助。不過，大多數使用者會透過其語言 SDK（例如 AWS 適用於 C++ 的 SDK 或適用於 Java 的 AWS SDK）或其語言 IoT 裝置 SDK（例如適用於 C++ 的 AWS IoT 開發套件或適用於 Java 的 AWS IoT 開發套件）存取 CRT 功能。在下圖中，語言 CRT 繫結方塊是指包裝特定語言 SDK CRT 程式庫的

套件。這是形式的套件集合aws-crt-\*，其中 '\*' 是 SDK 語言（例如 [aws-crt-cpp](#)或 [aws-crt-java](#)）。

以下是 CRT 程式庫的階層相依性圖例。

CRT 相依性圖表顯示個別 CRT 程式庫如何相互關聯。

# AWS SDKs和工具維護政策

## 概觀

本文件概述 AWS 軟體開發套件 (SDKs) 和工具的維護政策，包括 Mobile 和 IoT SDKs及其基礎相依性。AWS 會定期向 AWS SDKs 和工具提供更新，其中可能包含對新 API AWS APIs、新功能、增強功能、錯誤修正、安全修補程式或文件更新的支援。更新也可能解決相依性、語言執行期和作業系統的變更。AWS SDK 版本會發佈至套件管理員 (例如 Maven、NuGet、PyPI)，並在 GitHub 上以原始碼的形式提供。

我們建議使用者隨時掌握 SDK up-to-date以掌握最新功能、安全性更新和基礎相依性。不建議繼續使用不支援的 SDK 版本，並且由使用者自行決定是否繼續使用。

## 版本控制

AWS SDK 發行版本採用 X.Y.Z 格式，其中 X 代表主要版本。增加 SDK 的主要版本表示此 SDK 進行了重大且重大的變更，以支援語言中的新慣用語和模式。主要版本會在公有界面 (例如類別、方法、類型等)、行為或語意變更時推出。應用程式需要更新，才能使用最新的 SDK 版本。請務必根據提供的升級準則，仔細更新主要版本 AWS。

## SDK 主要版本生命週期

主要 SDKs 和工具版本的生命週期包含 5 個階段，如下所述。

- 開發人員預覽 (階段 0) - 在此階段期間，不支援 SDKs，不應在生產環境中使用，且僅用於早期存取和意見回饋目的。未來版本可能會引入重大變更。一旦將發行版本 AWS 識別為穩定產品，它可能會將其標記為發行候選。除非出現重大錯誤，否則發行候選項目已準備好進行 GA 發行，並將獲得完整 AWS 支援。
- 一般可用性 (GA) (階段 1) - 在此階段，完全支援 SDKs。AWS 將提供定期的 SDK 版本，包括對新服務的支援、現有服務的 API 更新，以及錯誤和安全性修正。對於工具，AWS 將提供包含新功能更新和錯誤修正的定期版本。AWS 將支援 SDK 的 GA 版本至少 24 個月。
- 維護公告 (階段 2) - AWS 將在 SDK 進入維護模式前至少 6 個月公告。在此期間，軟體開發套件將繼續獲得完全支援。一般而言，維護模式會在下一個主要版本轉換為 GA 時同時宣布。
- 維護 (階段 3) - 在維護模式中，AWS 會限制 SDK 版本，以僅解決重大錯誤修正和安全問題。軟體開發套件不會接收新服務或現有服務的 API 更新，或更新以支援新區域。除非另有說明，否則維護模式的預設持續時間為 12 個月。

- 終止End-of-Support ( 階段 4) - 當 SDK 達到終止支援時，將不再接收更新或版本。先前發佈的版本將繼續透過公有套件管理員提供，且程式碼將保留在 GitHub 上。GitHub 儲存庫可能已封存。使用者可自行決定是否使用已end-of-support 開發套件。我們建議使用者升級至新的主要版本。

以下是 SDK 主要版本生命週期的視覺化說明。請注意，以下顯示的時間表具有說明性且不繫結。  
維護政策時間表

## 相依性生命週期

AWS SDKs具有基礎相依性，例如語言執行時間、作業系統或第三方程式庫和架構。這些相依性通常與語言社群或擁有該特定元件的廠商相關聯。每個社群或廠商都會為其產品發佈自己的end-of-support 排程。

下列術語用於對基礎第三方相依性進行分類：

- 作業系統 (OS)：範例包括 Amazon Linux AMI、Amazon Linux 2、Windows 2008、Windows 2012、Windows 2016 等。
- 語言執行時間：範例包括 Java 7、Java 8、Java 11、.NET Core、.NET Standard、.NET PCL 等。
- 第三方程式庫/架構：範例包括 OpenSSL、.NET Framework 4.5、Java EE 等。

我們的政策是在社群或廠商結束對相依性的支援後，持續支援 SDK 相依性至少 6 個月。不過，此政策可能會因特定相依性而有所不同。

### Note

AWS 保留在不增加主要 SDK 版本的情況下停止支援基礎相依性的權利

## 通訊方法

維護公告會以多種方式進行通訊：

- 電子郵件公告會傳送至受影響的帳戶，宣布我們的計劃結束對特定 SDK 版本的支援。電子郵件將概述end-of-support路徑、指定行銷活動時間表，並提供升級指引。
- AWS 軟體開發套件文件（例如 API 參考文件）、使用者指南、軟體開發套件產品行銷頁面和 GitHub readme) 已更新，以指出行銷活動時間表並提供有關升級受影響應用程式的指引。

- 發佈 AWS 部落格文章，概述end-of-support路徑，並重複行銷活動時間表。
- 棄用警告會新增至 SDKs，概述end-of-support路徑，並連結至 SDK 文件。

若要查看可用主要版本的 AWS SDKs和工具清單及其維護生命週期中的位置，請參閱 [版本生命週期](#)。



## AWS SDKs和工具版本生命週期

下表顯示可用的 AWS 軟體開發套件 (SDK) 主要版本的清單，以及它們在維護生命週期中與相關時間表的位置。如需主要版本 AWS SDKs 和工具及其基礎相依性生命週期的詳細資訊，請參閱 [維護政策](#)。

SDK	主要版本	目前階段	一般可用日期	備註
<a href="#">AWS CLI</a>	1.x	維護公告	9/2/2013	如需詳細資訊和日期，請參閱 <a href="#">公告</a>
<a href="#">AWS CLI</a>	2.x	一般可用性	2/10/2020	
<a href="#">適用於 C++ 的 SDK</a>	1.x	一般可用性	9/2/2015	
<a href="#">適用於 Go V2 的 SDK</a>	V2 1.x	一般可用性	1/19/2021	
<a href="#">適用於 Go 的 SDK</a>	1.x	End-of-Support	11/19/2015	
<a href="#">適用於 Java 的開發套件</a>	1.x	End-of-Support	3/25/2010	
<a href="#">適用於 Java 的開發套件</a>	2.x	一般可用性	11/20/2018	
<a href="#">適用於 JavaScript 的 SDK</a>	1.x	End-of-Support	5/6/2013	
<a href="#">適用於 JavaScript 的 SDK</a>	2.x	End-of-Support	6/19/2014	
<a href="#">適用於 JavaScript 的 SDK</a>	3.x	一般可用性	12/15/2020	
<a href="#">適用於 Kotlin 的 SDK</a>	1.x	一般可用性	11/27/2023	

SDK	主要版本	目前階段	一般可用日期	備註
<a href="#">適用於 .NET 的 SDK</a>	1.x	End-of-Support	2009 年 11 月	
<a href="#">適用於 .NET 的 SDK</a>	2.x	End-of-Support	11/8/2013	
<a href="#">適用於 .NET 的 SDK</a>	3.x	一般可用性	7/28/2015	
<a href="#">適用於 .NET 的 SDK</a>	4.x	一般可用性	4/28/2025	
<a href="#">適用於 PHP 的 SDK</a>	2.x	End-of-Support	11/2/2012	
<a href="#">適用於 PHP 的 SDK</a>	3.x	一般可用性	5/27/2015	
<a href="#">適用於 Python 的 SDK (Boto2)</a>	1.x	End-of-Support	7/13/2011	
<a href="#">適用於 Python 的 SDK (Boto3)</a>	1.x	一般可用性	6/22/2015	
<a href="#">適用於 Python 的 SDK (Botocore)</a>	1.x	一般可用性	6/22/2015	
<a href="#">適用於 Ruby 的 SDK</a>	1.x	End-of-Support	7/14/2011	
<a href="#">適用於 Ruby 的 SDK</a>	2.x	End-of-Support	2/15/2015	
<a href="#">適用於 Ruby 的 SDK</a>	3.x	一般可用性	8/29/2017	
<a href="#">適用於 Rust 的 SDK</a>	1.x	一般可用性	11/27/2023	

SDK	主要版本	目前階段	一般可用日期	備註
<a href="#">適用於 Swift 的 SDK</a>	1.x	一般可用性	9/17/2024	
Tools for PowerShell	2.x	End-of-Support	11/8/2013	
Tools for PowerShell	3.x	End-of-Support	7/29/2015	
<a href="#">PowerShell 的工具</a>	4.x	一般可用性	11/21/2019	
<a href="#">PowerShell 的工具</a>	5.x	一般可用性	6/23/2025	

搜尋未提及的 SDK 或工具？例如，加密 SDKs、IoT 裝置 SDKs 和 Mobile SDKs 不包含在本指南中。若要尋找這些其他工具的文件，請參閱[在上建置的工具 AWS](#)。

## 適用於 AWS SDKs和工具的文件歷史記錄參考指南

下表說明 AWS SDKs 和工具參考指南的重要新增和更新。如需有關此文件更新的通知，您可以訂閱 RSS 摘要。

變更	描述	日期
<a href="#">新增 S3 Express One Zone 設定</a>	新增 S3 Express One Zone 設定以停用工作階段身分驗證。	2025 年 10 月 13 日
<a href="#">新增身分驗證決策樹</a>	新增決策樹，以協助在選項之間進行身分驗證決策。	2025 年 9 月 23 日
<a href="#">新增身分驗證機制功能</a>	新增身分驗證機制功能。AWS STS 區域端點的更新。	2025 年 8 月 18 日
<a href="#">新增適用於 PowerShell 的工具版本</a>	將最新版本的 Tools for PowerShell 支援新增至所有設定與 AWS SDKs的參考相容性資料表。新增主機字首注入功能。	2025 年 6 月 23 日
<a href="#">頁面標題更新</a>	更多標題、資料表標題、摘要和 SEO 更新。	2025 年 3 月 5 日
<a href="#">頁面標題更新</a>	更新內容以使用更描述性的標題。	2025 年 2 月 24 日
<a href="#">將 Swift SDK 新增至設定參考</a>	將 Swift SDK 支援新增至所有設定參考與 AWS SDKs的相容性資料表。	2024 年 9 月 17 日
<a href="#">適用於 Java 的 SDK 1.x 系統屬性</a>	新增 適用於 Java 的 AWS SDK 1.x 所支援 JVM 系統組態設定的詳細資訊。	2024 年 5 月 30 日
<a href="#">設定更新</a>	新增 JVM 系統組態設定。	2024 年 3 月 27 日

<a href="#">相容性資料表更新</a>	更新 SDK 支援的相容性、更新 IAM Identity Center 程序。	2024 年 2 月 20 日
<a href="#">容器憑證更新。IMDS 更新。</a>	新增對 Amazon EKS 的支援。 新增設定以停用 IMDSv1 備用。	2023 年 12 月 29 日
<a href="#">請求壓縮</a>	新增請求壓縮功能的設定。	2023 年 12 月 27 日
<a href="#">相容性資料表</a>	更新 SDK 和工具功能的相容性資料表，以包含適用於 Kotlin 的 SDK、適用於 Rust 的 SDK 和 AWS Tools for PowerShell。	2023 年 12 月 10 日
<a href="#">身分驗證更新</a>	更新 SDKs 和工具支援的身分驗證方法。	2023 年 7 月 1 日
<a href="#">IAM 最佳實務更新</a>	更新了指南以符合 IAM 最佳實務。如需更多詳細資訊，請參閱 <a href="#">IAM 中的安全最佳實務</a> 。	2023 年 2 月 27 日
<a href="#">SSO 更新</a>	更新新的 SSO 字符組態的 SSO 登入資料。	2022 年 11 月 19 日
<a href="#">設定更新</a>	更新以支援一般組態和 Amazon S3 多區域存取點的資料表。	2022 年 11 月 17 日
<a href="#">設定更新</a>	更新以釐清 IMDS 用戶端和 IMDS 登入資料。環境變數的更新。	2022 年 11 月 4 日
<a href="#">更新歡迎頁面</a>	宣布 Amazon CodeWhisperer。	2022 年 9 月 22 日
<a href="#">單一登入的服務名稱變更</a>	更新以反映現在稱為 AWS SSO AWS IAM Identity Center。	2022 年 7 月 26 日

---

<a href="#">設定更新</a>	對組態檔案詳細資訊和支援的設定進行次要更新。	2022 年 6 月 15 日
<a href="#">更新</a>	本指南幾乎所有部分的大量更新。	2022 年 2 月 1 日
<a href="#">初始版本</a>	本指南的第一個版本會發佈給大眾。	2020 年 3 月 13 日

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。