aws

開發人員指南

Amazon S3 Glacier

API 版本 2012-06-01

Amazon S3 Glacier: 開發人員指南

Table of Contents

此頁面僅適用於使用保管庫和 2012 年起原始 REST API 的 S3 冰川服務的現有客戶。

如果您正在尋找存檔儲存解決方案，我們建議您使用 Amazon S3 中的 S3 Glacier 儲存類別、S3 冰川即時擷取、S3 冰川彈性擷取和 S3 Glacier Deep Archive。若要進一步了解這些儲存選項，請參閱 Amazon S3 使用者指南中的 S3 Glacier 儲存類別和使用 S3 Glacier 儲存類別的長期資料儲存。這些儲存類別使用 Amazon S3 API，可在所有區域使用，並且可以在 Amazon S3 主控台中管理。它們提供了諸如存儲成本分析，存儲鏡頭，包括多種加密選項的安全功能等功能。

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。

Amazon S3 Glacier 是什麼？

如果您目前正在使用 Amazon S3 Glacier (S3 Glacier) 服務，而且想要進一步了解，您可以在本指南中找到所需的資訊。S3 Glacier 是一種安全、耐用的服務，可使用保存庫進行低成本的資料封存和長期備份。如需 S3 Glacier 服務定價的相關資訊，請參閱 S3 Glacier 定價。

主題

您目前使用的是 S3 Glacier 嗎？

> **Note**
>
> 本節是關於 S3 Glacier 服務。如果您目前使用 S3 Glacier 儲存類別 (S3 冰川即時擷取、S3 冰川彈性擷取和 S3 Glacier Deep Archive)，請參閱 Amazon S3 使用者指南中的存檔物件的儲存類別。

如果您目前使用 S3 Glacier 服務，並想進一步了解，我們建議您先閱讀下列章節來開始：

- 什麼是 Amazon S3 Glacier：本節的其他部分說明基礎資料模型、支援的作業，以及您可來與服務互動的 AWS 開發套件。
- 入門：Amazon S3 Glacier 入門 一節會逐步解說如何建立保存庫、上傳封存、建立工作以下載封存、擷取工作輸出和刪除封存。

> **Important**
>
> S3 Glacier 確實會提供主控台。不過，任何封存作業 (例如上傳、下載或刪除) 都需要您使用 AWS Command Line Interface (AWS CLI) 或撰寫程式碼。沒有主控台支援封存操作。例如，若要上傳相片、影片和其他文件等資料，您必須直接使用 REST API AWS CLI 或使用 AWS SDK，使用或撰寫程式碼來提出要求。

若要安裝 AWS CLI，請參閱AWS Command Line Interface。如需使用 S3 Glacier 搭配 AWS CLI的詳細資訊，請參閱 S3 Glacier的AWS CLI 參考。如需使用將存檔上傳 AWS CLI 至 S3 冰川的範例，請參閱搭配使用 S3 冰川 AWS Command Line Interface。

除了入門章節的內容，您可能會想要進一步了解 S3 Glacier 作業。下列各節提供有關使用 REST API 和 Java 和 Microsoft .NET AWS 開發套件使用 S3 冰川的詳細資訊：

- 使用 AWS 軟體開發套件搭配 Amazon S3 冰川

  本節提供本指南中各種程式碼範例中所使用之 AWS SDK 的概觀。檢閱本節有助於閱讀以下章節。它包含這些開發套件所提供高階和低階 API 的概觀，使用時機以及用於執行本指南中所提供程式碼範例的一般步驟。

- 在 Amazon S3 Glacier 中使用文件庫

  本節提供各種保存庫作業的詳細資訊，例如建立保存庫、擷取保存庫中繼資料、使用工作來擷取保存庫庫存，以及設定保存庫通知等。除了使用 S3 Glacier 主控台之外，您還可以將 AWS 開發套件用於各種文件庫操作。本節說明 API 並使用和提供工作範例 AWS SDK for .NET。 AWS SDK for Java

- 在 Amazon S3 Glacier 中使用封存

  本節提供封存作業的詳細資訊，例如在單一請求中上傳封存，或使用分段上傳作業來上傳分段中的大型封存。本節也說明如何建立工作以非同步方式下載封存。本節提供使用 AWS SDK for Java 與 AWS SDK for .NET的範例。

- Amazon S3 Glacier API 參考

  S3 Glacier 是 RESTful 的服務。本節說明 REST 操作，包括語法和範例請求以及所有操作的回應。AWS SDK 程式庫包裝此 API，簡化您的程式設計工作。

Amazon S3 Glacier 資料模型

Amazon S3 Glacier 資料模型核心元件包括保存庫和封存。S3 Glacier 是以 REST 為基礎的 Web 服務。以 REST 而言，保存庫和封存是資源。此外，S3 Glacier 資料模型包含工作和通知設定資源。這些資源可以補足核心資源。

主題

- 保存庫
- 存檔

- 任務
- 通知組態

保存庫

在 S3 Glacier 中，保存庫是儲存封存的一種容器。保存庫類似於 Amazon S3 儲存貯體。當您建立儲存庫時，您可以指定名稱並選擇您 AWS 區域 要建立資料保險箱的位置。

每個保存庫資源都有唯一的地址。一般形式為：

```
https://region-specific-endpoint/account-id/vaults/vault-name
```

例如，假設您在美國西部 (奧勒岡) 區域中，在帳戶中使用 ID 111122223333 建立保存庫 (examplevault)。您可使用以下 URI 定址此保存庫：

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/examplevault
```

以下是 URI 各種元件的含義：

- glacier.us-west-2.amazonaws.com 識別美國西部 (奧勒岡) 區域。
- 111122223333是擁有儲存庫的 AWS 帳戶 ID。
- vaults 是指 AWS 帳戶所擁有的保存庫集合。
- examplevault 識別保存庫集合中的特定保存庫。

AWS 帳戶 可以在任何受支援 AWS 區域的儲存庫中建立儲存庫。如需支援的清單 AWS 區域，請參閱存取 Amazon S3 Glacier。在區域內，帳戶必須使用唯一保存庫名稱。 AWS 帳戶 可以在不同的區域中建立相同名稱的儲存庫。

您可以在保存庫中存放無限數量的封存。根據您的業務或應用程式需求，您可以將這些封存存放在一個保存庫或多個保存庫。

S3 Glacier 支援各種保存庫作業。保存庫作業為區域特定。例如，當您建立保存庫時，您可以在特定區域建立。當您請求儲存庫清單時，您會從特定的資料保險箱清單中提出要求 AWS 區域，而產生的清單僅包括在該特定區域中建立的 Vault。

存檔

封存可以是任何資料，例如照片、影片或文件。封存類似於 Amazon S3 物件，並且是 S3 Glacier 中的儲存基本單位。每個封存都有唯一的 ID 以及選擇性說明。您僅可在上傳封存期間指定選擇性說明。S3 Glacier 會為歸檔指派一個 ID，該 ID 在存放歸檔 AWS 區域 中是唯一的。

每個封存都有唯一的地址。一般形式如下所示：

```
https://region-specific-endpoint/account-id/vaults/vault-name/archives/archive-id
```

以下是存放在帳戶為 111122223333 之美國西部 (奧勒岡) 區域中 `examplevault` 保存庫之封存的範例 URI：

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/
examplevault/archives/NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

您可以在保存庫中存放無限數量的封存。

任務

S3 Glacier 工作可以擷取封存，或取得保存庫的庫存。

擷取封存和保存庫庫存 (封存的清單) 是 S3 Glacier 中的非同步作業，您必須先啟動工作，然後在 S3 Glacier 完成工作後才能下載工作輸出。

> **Note**
>
> S3 Glacier 提供冷儲存資料封存解決方案。如果您的應用程式需要儲存解決方案，需要即時擷取資料，您可以考慮使用 Amazon S3。如需詳細資訊，請參閱 Amazon Simple Storage Service (Amazon S3)。

若要起始保存庫工作，您要提供保存庫名稱。封存擷取工作需要保存庫名稱和封存 ID 二者。您也可以提供選擇性任務描述，以協助識別任務。

封存擷取以及保存庫庫存工作會與保存庫相關聯。保存庫可以在任何時間點有多個進行中的工作。當您傳送工作請求 (起始工作) 時，S3 Glacier 會傳回工作 ID 以追蹤工作。每個任務都是由 URI 的形式來唯一識別：

```
https://region-specific-endpoint/account-id/vaults/vault-name/jobs/job-id
```

下列是與帳戶為 111122223333 的美國西部 (奧勒岡) 區域中 examplevault 保存庫相關聯的工作範例。

```
https://glacier.us-west-2.amazonaws.com/111122223333/vaults/examplevault/jobs/
HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

對於每個工作，S3 Glacier 會維護工作類型、描述、建立日期、完成日期和工作狀態之類的資訊。您可以取得有關特定工作的資訊，或取得與保存庫關聯的所有工作清單。S3 Glacier 傳回的工作清單包含所有進行中和最近已完成的工作。

通知組態

由於工作需要時間才能執行，所以 S3 Glacier 支援通知機制，可在工作完成時通知您。您可以設定保存庫，以在工作完成時將通知傳送到 Amazon Simple Notification Service (Amazon SNS) 主題。您可以在通知設定中每個保存庫指定 Amazon SNS 主題。

S3 Glacier 會將通知設定儲存為 JSON 文件。以下是範例保存庫通知組態：

```
{
   "Topic": "arn:aws:sns:us-west-2:111122223333:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

請注意，通知設定會與保存庫相關聯；每個保存庫都可以有一個設定。每個通知組態資源都是由 URI 的形式來唯一識別：

```
https://region-specific-endpoint/account-id/vaults/vault-name/notification-
configuration
```

S3 Glacier 支援設定、取得和刪除通知設定的作業。在您刪除通知設定後，當保存庫上的資料擷取作業完成時不會傳送任何通知。

S3 Glacier 中受支援的作業

若要使用保存庫和封存 (請參閱 Amazon S3 Glacier 資料模型)，Amazon S3 Glacier 支援一組作業。在所有支援的操作中，只有以下操作是非同步：

- 擷取封存
- 擷取保存庫庫存 (保存庫的清單)

這些操作要求您要先起始任務，然後下載任務輸出。以下章節總結 S3 Glacier 作業。

保存庫作業

S3 Glacier 提供建立和刪除保存庫的作業。您可以取得特定保存庫或 AWS 區域中所有保存庫的保存庫說明。保存庫說明提供資訊，例如建立日期、保存庫中的封存數、保存庫中所有封存所使用的總大小 (以位元組為單位)，以及 S3 Glacier 產生保存庫庫存的日期。S3 Glacier 也提供設定、擷取和刪除保存庫的通知設定的作業。如需詳細資訊，請參閱 在 Amazon S3 Glacier 中使用文件庫。

封存操作

S3 Glacier 為您提供上傳和刪除封存的作業。您無法更新現有的封存，您必須刪除現有的封存並上傳新的封存。請注意，每次上傳封存時，S3 Glacier 都會產生新的封存 ID。如需詳細資訊，請參閱 在 Amazon S3 Glacier 中使用封存。

任務

您可以啟動 S3 Glacier 工作，來對封存執行擷取，或取得保存庫的庫存。

以下是 S3 Glacier 工作的類型：

- `archive-retrieval`：擷取封存。

  如需詳細資訊，請參閱 在 S3 Glacier 中下載封存。
- `inventory-retrieval`：清點保存庫。

  如需詳細資訊，請參閱 在 Amazon S3 Glacier 中下載文件庫庫存。

存取 Amazon S3 Glacier

Amazon S3 冰川是一種 REST 風格的網絡服務，它使用 HTTP 和 HTTPS 作為傳輸協議，並使用 JavaScript 對象符號（JSON）作為消息序列化格式。應用程式程式碼能直接發出請求到 S3 Glacier Web 服務 API。直接使用 REST API 時，您必須編寫必要的程式碼，以簽署和驗證您的請求。如需 API (匯入 API) 的詳細資訊，請參閱「Amazon S3 Glacier API 參考」。

或者，您也可以使用包裝 S3 Glacier REST API 呼叫的開發 AWS 套件來簡化應用程式開發。您提供您的登入資料，而這些程式庫負責身分驗證和請求簽署。如需使用 AWS 開發套件的詳細資訊，請參閱使用 AWS 軟體開發套件搭配 Amazon S3 冰川。

S3 Glacier 也會提供主控台。但是，所有封存和工作作業都需要您直接使用 REST API 或 AWS SDK 包裝函式庫來撰寫程式碼並提出要求。若要存取 S3 Glacier 主控台，請前往 S3 Glacier 主控台。

區域與終端節點

您可以在特定的情況下建立儲存庫 AWS 區域。您一律會將 S3 Glacier 請求傳送到 AWS 區域的特定端點。如需 S3 Glacier 支援的 AWS 區域 清單，請參閱《AWS 一般參考》中的 Amazon S3 Glacier 端點和配額。

Amazon S3 Glacier 入門

您可以使用保存庫和封存，來開始使用 Amazon S3 Glacier (S3 Glacier)。保存庫是用於儲存封存的容器，封存是存放在保存庫中的任何物件 (如相片、影片或文件)。封存是 S3 Glacier 中儲存基本單位。本入門練習提供的指導，可供您在保存庫和封存上探索基本 S3 Glacier 作業。如需有關這些資源的詳細資訊，請參閱 Amazon S3 Glacier 資料模型一節。

在入門練習中，您將建立保存庫、上傳和下載封存，最後刪除封存和保存庫。您可以以程式設計方式執行所有這些操作。但是，入門練習會使用 S3 Glacier 管理主控台來建立和刪除保存庫。若要上傳和下載歸檔，此入門章節會針對 AWS SDK for Java 和使用高階 API AWS SDK for .NET。高階 API 在使用 S3 Glacier 時提供簡化的程式設計體驗。如需將高階 API 與 AWS SDK 搭配使用的詳細資訊，請參閱使用 AWS 軟體開發套件搭配 Amazon S3 冰川。

> **Important**
>
> S3 Glacier 確實會提供主控台。不過，任何封存作業 (例如上傳、下載或刪除) 都需要您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。沒有主控台支援封存操作。例如，若要上傳相片、影片和其他文件等資料，您必須直接使用 REST API AWS CLI 或使用 AWS SDK，使用或撰寫程式碼來提出要求。
>
> 若要安裝 AWS CLI，請參閱AWS Command Line Interface。如需搭配使用 S3 冰川的詳細資訊 AWS CLI，請參閱 S3 冰川的AWS CLI 參考資料。如需使用將存檔上傳 AWS CLI 至 S3 冰川的範例，請參閱搭配使用 S3 冰川 AWS Command Line Interface。

此入門練習提供 Java 和 C# 程式碼範例，供您上傳和下載封存。入門練習的最後一節提供一些步驟，您可以透過這些步驟了解有關 S3 Glacier 開發人員體驗的詳細資訊。

主題

步驟 1：開始 S3 Glacier 之前

在開始本練習之前，您必須先註冊 AWS 帳戶 (如果您還沒有)，然後下載其中一個 AWS SDK。如需詳細資訊，請參閱下節。

主題

設定 AWS 帳戶 和管理員使用者

如果您尚未這麼做，您必須註冊 AWS 帳戶 並在帳戶中建立系統管理員使用者。

若要完成設定，請遵循以下主題的指示。

設定 AWS 帳戶 並建立管理員使用者

註冊成為 AWS

當您註冊 Amazon Web Services (AWS) 時，系統會自動註冊您 AWS 帳戶 的所有服務 AWS，包括 S3 Glacier。您只需支付實際使用服務的費用。如需 S3 Glacier 用量費率的詳細資訊，請參閱 Amazon S3 Glacier 定價頁面。

如果您已經擁有 AWS 帳戶，請跳至下載適當的 AWS SDK。如果您沒有 AWS 帳戶，請使用下列程序來建立一個。

如果您沒有 AWS 帳戶，請完成以下步驟來建立一個。

若要註冊成為 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電，並在電話鍵盤輸入驗證碼。

   當您註冊一個時 AWS 帳戶，將創建AWS 帳戶根使用者一個。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。安全性最佳做法是將管理存取權指派給使用者，並僅使用 root 使用者來執行需要 root 使用者存取權的工作。

若要建立管理員使用者，請選擇下列其中一個選項。

選擇一種管理管理員的方式	到	By	您也可以
在 IAM Identity Center (建議)	使用短期憑證存取 AWS。 這與安全性最佳實務一致。有關最佳實務的資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。	請遵循 AWS IAM Identity Center 使用者指南的入門中的說明。	AWS IAM Identity Center 在《使用AWS Command Line Interface 者指南》中設定 AWS CLI 要使用的，以設定程式設計方式存取。
在 IAM 中 (不建議使用)	使用長期憑證存取 AWS。	請遵循 IAM 使用者指南中建立您的第一個 IAM 管理員使用者和使用者群組的說明。	請參閱 IAM 使用者指南中的管理 IAM 使用者的存取金鑰，設定程式設計存取。

下載適當的 AWS SDK

若要嘗試入門練習，您必須決定要使用哪種程式設計語言，然後下載適用於您的開發平台的 AWS SDK。

入門練習提供 Java 和 C# 範例。

下載 AWS SDK for Java

若要測試此開發人員指南中的 Java 範例，您需要 AWS SDK for Java。您有以下下載選項：

- 如果您使用的是 Eclipse，您可以使用更新網站下載並安裝 http://aws.amazon.com/eclipse/。 AWS Toolkit for Eclipse 如需詳細資訊，請參閱 AWS Toolkit for Eclipse。
- 如果您使用任何其他 IDE 來建立應用程式，請下載 AWS SDK for Java。

下載 AWS SDK for .NET

若要測試此開發人員指南中的 C# 範例，您需要 AWS SDK for .NET。您有以下下載選項：

- 如果您使用的是視覺工作室，您可以同時安裝 AWS SDK for .NET 和 AWS Toolkit for Visual Studio. 此工具組提供適用於 Visual Studio 的 AWS 資源管理器，以及可用於開發的專案範本。若要下載 AWS SDK for .NET，請前往 http://aws.amazon.com/sdkfornet。依預設，安裝指令碼會同時安裝 AWS SDK 和 AWS Toolkit for Visual Studio. 若要進一步了解工具組，請參閱《AWS Toolkit for Visual Studio 使用者指南》。
- 如果您使用任何其他 IDE 來建立應用程式，則可以使用前面步驟中提供的相同連結並僅安裝 AWS SDK for .NET。

步驟 2：在 S3 Glacier 中建立保存庫

文件庫是儲存封存的一種容器。您的第一步是在其中一個受支持的文件庫中創建一個保管庫 AWS 區域。如需 Amazon S3 Glacier 支援的清單，請參閱AWS 一般參考中的 Amazon S3 冰川端點和配額。AWS 區域

您可以用程式設計方式或使用 S3 Glacier 主控台建立保存庫。本節使用主控台來建立文件庫。

建立文件庫

1. 登入 AWS Management Console 並開啟 S3 冰川主控台，網址為 https://console.aws.amazon.com/glacier/home。
2. 在左側導覽窗格中，選擇保存庫。
3. 選擇建立保存庫。

   即會開啟建立保存庫頁面。
4. 在「選取區域」下， AWS 區域 從「區域」選取器中選取。保存庫將位於您所選的區域中。
5. 在保存庫名稱中，輸入保存庫的名稱。

   以下是保存庫命名要求：

   - 資料保險箱名稱在建立資料保險箱的 AWS 帳戶 和 AWS 區域 中必須是唯一的。
   - 保存庫名稱長度必須介於 1 到 255 個字元之間。
   - 保存庫名稱只能包含下列字元：a-z、A-Z、0-9、_ (底線)、- (連字號) 和 . (句號)。
6. 在事件通知下，若要在工作完成時開啟或關閉保存庫上的通知，請選擇以下其中一個設定：

   - 關閉通知：通知會在指定的工作完成時關閉，且不會將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題。

- 開啟通知：當指定的工作完成時，通知會開啟，並會將通知傳送至提供的 Amazon SNS 主題。

  如果您選擇開啟通知，請參閱使用 Amazon S3 Glacier 主控台設定保存庫通知。

7. 如果 AWS 區域 和資料保險箱名稱正確，請選擇「建立儲存庫」。

S3 Glacier 主控台的保存庫頁面中現在會列出新的保存庫。

步驟 3：將封存上傳至 S3 Glacier 中的保存庫

在此步驟中，將範例封存上傳到在前面步驟中所建立的保存庫中 (請參閱 步驟 2：在 S3 Glacier 中建立保存庫)。根據所使用的開發平台，選擇本節結尾的其中一個連結。

> **Important**
>
> 任何封存作業 (例如上傳、下載或刪除) 都要求您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。沒有主控台支援封存操作。例如，若要上傳相片、影片和其他文件等資料，您必須直接使用 REST API AWS CLI 或使用 AWS SDK，使用或撰寫程式碼來提出要求。
> 若要安裝 AWS CLI，請參閱AWS Command Line Interface。如需搭配使用 S3 冰川的詳細資訊 AWS CLI，請AWS CLI 參閱 S3 冰川的參考資料。如需使用將存檔上傳 AWS CLI 至 S3 冰川的範例，請參閱搭配使用 S3 冰川 AWS Command Line Interface。

封存是存放在保存庫中的任何物件 (如相片、影片或文件)。封存是 S3 Glacier 中儲存基本單位。您可以在單一請求中上傳封存。若是大型封存，S3 Glacier 提供分段上傳 API 作業，可讓您分成幾個部分上傳封存。

在此入門部分，您在單一請求中上傳範例封存。本練習中，指定一個較小的檔案。對於較大的檔案，分段上傳是合適的。如需詳細資訊，請參閱 上傳分段中的大型封存 (分段上傳)。

主題

- 使用 AWS SDK for Java 將封存上傳至 S3 Glacier 中的文件庫
- 使用 AWS SDK for .NET 將封存上傳至 S3 Glacier 中的文件庫

使用 AWS SDK for Java 將封存上傳至 S3 Glacier 中的文件庫

以下 Java 程式碼範例使用 AWS SDK for Java 的高階 API 將範例封存上傳到文件庫。在程式碼範例中，請注意下列事項：

- 範例會建立 `AmazonGlacierClient` 類別的執行個體。
- 此範例使用 AWS SDK for Java 高階 API 的 `ArchiveTransferManager` 類別的 `upload` API 操作。
- 此範例使用美國西部 (奧勒岡) 區域 (`us-west-2`)。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您必須如所示，使用要更新之封存檔的名稱更新程式碼。

> **Note**
>
> Amazon S3 Glacier 會在文件庫中保存所有封存的庫存。當您上傳下列範例中的存檔時，它將不會顯示管理主控台的文件庫中，直到文件庫清查已更新。此更新通常一天執行一次。

適用於 Java 2.x 的開發套件

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.UploadArchiveRequest;
import software.amazon.awssdk.services.glacier.model.UploadArchiveResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.io.File;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
```

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

    static final int ONE_MB = 1024 * 1024;

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

                Usage:   <strPath> <vaultName>\s

                Where:
                   strPath - The path to the archive to upload (for example, C:\\AWS
\\test.pdf).
                   vaultName - The name of the vault.
                """;

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

        String strPath = args[0];
        String vaultName = args[1];
        File myFile = new File(strPath);
        Path path = Paths.get(strPath);
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String archiveId = uploadContent(glacier, path, vaultName, myFile);
        System.out.println("The ID of the archived item is " + archiveId);
        glacier.close();
    }

    public static String uploadContent(GlacierClient glacier, Path path, String
 vaultName, File myFile) {
```

```
        // Get an SHA-256 tree hash value.
        String checkVal = computeSHA256(myFile);
        try {
            UploadArchiveRequest uploadRequest = UploadArchiveRequest.builder()
                    .vaultName(vaultName)
                    .checksum(checkVal)
                    .build();

            UploadArchiveResponse res = glacier.uploadArchive(uploadRequest, path);
            return res.archiveId();

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

    private static String computeSHA256(File inputFile) {
        try {
            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 tree hash = %s\n", toHex(treeHash));
            return toHex(treeHash);

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s", inputFile,
ioe.getMessage());
            System.exit(-1);

        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256:
%s", nsae.getMessage());
            System.exit(-1);
        }
        return "";
    }

    public static byte[] computeSHA256TreeHash(File inputFile) throws IOException,
            NoSuchAlgorithmException {

        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }
```

```
    /**
     * Computes an SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk, even if it's smaller than 1 MB.
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
            int idx = 0;

            while ((bytesRead = fileStream.read(buff, 0, ONE_MB)) > 0) {
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
            }

            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
```

```
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.length > 1) {
            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];
            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining.
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes.
                    md.reset();
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();

                } else { // Take care of the remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }

        return prevLvlHashes[0];
    }

    /**
```

```
     * Returns the hexadecimal representation of the input byte array
     */
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);
        for (byte datum : data) {
            String hex = Integer.toHexString(datum & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

- 如需 API 的詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》的 UploadArchive。

使用 AWS SDK for .NET 將封存上傳至 S3 Glacier 中的文件庫

以下 C# 程式碼範例使用 AWS SDK for .NET 的高階 API 將範例存檔上傳到文件庫。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon S3 Glacier 區域端點建立 `ArchiveTransferManager` 類別的執行個體。
- 此程式碼範例使用美國西部 (奧勒岡) 區域 (`us-west-2`)。
- 本範例使用 `ArchiveTransferManager` 類別的 `Upload` API 操作以上傳封存。對於較小的封存，這個操作會將封存直接上傳到 S3 Glacier。對於較大的封存，此操作使用 S3 Glacier 的分段上傳 API 操作，將上傳分割成多個部分，一旦在將資料串流到 S3 Glacier 時遇到任何錯誤，可以更好地進行錯誤復原。

如需如何執行下列範例的逐步說明，請參閱 執行程式碼範例。您必須更新程式碼，如所示的文件庫名稱和要上傳的封存檔案的名稱。

> **Note**
>
> S3 Glacier 會在文件庫中保存所有封存的庫存。當您上傳下列範例中的封存時，直到文件庫庫存更新前，封存都不會顯示在管理主控台的文件庫中。此更新通常一天執行一次。

Example ：使用 AWS SDK for .NET 的高階 API 上傳封存

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
    class ArchiveUploadHighLevel_GettingStarted
    {
        static string vaultName = "examplevault";
        static string archiveToUpload = "*** Provide file name (with full path) to
 upload ***";

        public static void Main(string[] args)
        {
            try
            {
                var manager = new
 ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
                // Upload an archive.
                string archiveId = manager.Upload(vaultName, "getting started archive
 test", archiveToUpload).ArchiveId;
                Console.WriteLine("Copy and save the following Archive ID for the next
 step.");
                Console.WriteLine("Archive ID: {0}", archiveId);
                Console.WriteLine("To continue, press Enter");
                Console.ReadKey();
            }
            catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
            catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
            catch (Exception e) { Console.WriteLine(e.Message); }
            Console.WriteLine("To continue, press Enter");
            Console.ReadKey();
        }
    }
```

```
}
```

步驟 4：從 S3 Glacier 中的保存庫下載封存

在此步驟中，您將下載之前在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳的範例封存。

> **Important**
>
> Amazon S3 Glacier 確實會提供主控台。不過，任何封存作業 (例如上傳、下載或刪除) 都需要您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。沒有主控台支援封存操作。例如，若要上傳相片、影片和其他文件等資料，您必須直接使用 REST API AWS CLI 或使用 AWS SDK，使用或撰寫程式碼來提出要求。
>
> 若要安裝 AWS CLI，請參閱AWS Command Line Interface。如需搭配使用 S3 冰川的詳細資訊 AWS CLI，請AWS CLI 參閱 S3 冰川的參考資料。如需使用將存檔上傳 AWS CLI 至 S3 冰川的範例，請參閱搭配使用 S3 冰川 AWS Command Line Interface。

一般而言，從 S3 Glacier 擷取資料包含兩步驟程序：

1. 啟動擷取任務。
2. 工作完成後，下載資料的位元組。

若要從 S3 Glacier 擷取封存，首先要啟動工作。在工作完成後下載資料。如需封存擷取的詳細資訊，請參閱 使用 AWS 主控台擷取 S3 Glacier 封存。

請求的存取時間取決於您選擇的擷取選項：快速、標準或大量擷取。用於規模幾乎最大的封存 (250 MB 以上) 時，使用快速擷取所存取的資料，通常會在 1-5 分鐘內即可使用。使用標準擷取而擷取的封存通常在 3-5 小時之間即可使用。大量擷取通常會於 5-12 小時內即可使用。如需各種擷取選項的詳細資訊，請參閱 S3 Glacier 常見問答集。如需有關資料擷取費用的詳細資訊，請參閱 S3 Glacier 定價頁面。

下列主題中顯示的程式碼範例啟動工作，請等待其完成，然後下載封存的資料。

主題

- 使用 AWS SDK for Java 從 S3 Glacier 中的文件庫下載封存
- 使用 AWS SDK for .NET 從 S3 Glacier 中的文件庫下載封存

使用 AWS SDK for Java 從 S3 Glacier 中的文件庫下載封存

以下 Java 程式碼範例使用 AWS SDK for Java 的高階 API，下載您在之前步驟中上傳的封存。在程式碼範例中，請注意下列事項：

- 範例會建立 `AmazonGlacierClient` 類別的執行個體。
- 此程式碼使用美國西部 (奧勒岡) 區域 (`us-west-2`) 比對在 步驟 2：在 S3 Glacier 中建立保存庫 中建立文件庫的位置。
- 此範例使用 AWS SDK for Java 高階 API 的 `ArchiveTransferManager` 類別的 `download` API 操作。此範例會建立 Amazon Simple Notification Service (Amazon SNS) 主題，以及訂閱該主題的 Amazon Simple Queue Service (Amazon SQS) 佇列。若您依照 步驟 1：開始 S3 Glacier 之前 的指示建立 AWS Identity and Access Management (IAM) 管理員使用者，使用者可擁有需要的 IAM 許可，以建立並使用 Amazon SNS 主題和 Amazon SQS 佇列。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要按照在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳之檔案的封存 ID 來更新程式碼。

Example ：使用 AWS SDK for Java 下載封存

```
import java.io.File;
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sqs.AmazonSQSClient;

public class AmazonGlacierDownloadArchive_GettingStarted {
    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID ***";
    public static String downloadFilePath  = "*** provide location to download archive
 ***";

    public static AmazonGlacierClient glacierClient;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

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

```
     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        glacierClient = new AmazonGlacierClient(credentials);
        sqsClient = new AmazonSQSClient(credentials);
        snsClient = new AmazonSNSClient(credentials);

        glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
        sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
        snsClient.setEndpoint("sns.us-west-2.amazonaws.com");

        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient,
 sqsClient, snsClient);

            atm.download(vaultName, archiveId, new File(downloadFilePath));

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

使用 AWS SDK for .NET 從 S3 Glacier 中的文件庫下載封存

以下 C# 程式碼範例使用 AWS SDK for .NET 高階 API，來下載您之前在 使用 AWS SDK for .NET 將封存上傳至 S3 Glacier 中的文件庫 中上傳的封存。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon S3 Glacier 區域端點建立 `ArchiveTransferManager` 類別的執行個體。
- 此程式碼範例使用美國西部 (奧勒岡) 區域 (`us-west-2`)，比對之前在 步驟 2：在 S3 Glacier 中建立保存庫 中建立文件庫的位置。
- 本範例使用 `ArchiveTransferManager` 類別的 `Download` API 操作以下載封存。此範例會建立 Amazon Simple Notification Service (Amazon SNS) 主題，以及訂閱該主題的 Amazon Simple Queue Service (Amazon SQS) 佇列。若您依照 步驟 1：開始 S3 Glacier 之前 的指示建立 AWS Identity and Access Management (IAM) 管理員使用者，使用者可擁有需要的 IAM 許可，以建立並使用 Amazon SNS 主題和 Amazon SQS 佇列。
- 此範例接著啟動封存擷取任務，輪詢佇列以使封存可用。封存可用時，就會開始下載。如需封存擷取時間的詳細資訊，請參閱 封存擷取選項。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您需要按照在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳之檔案的封存 ID 來更新程式碼。

Example ：使用 AWS SDK for .NET 的高階 API 下載封存

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
    class ArchiveDownloadHighLevel_GettingStarted
    {
        static string vaultName = "examplevault";
        static string archiveId = "*** Provide archive ID ***";
        static string downloadFilePath = "*** Provide the file name and path to where
 to store the download ***";

        public static void Main(string[] args)
        {
            try
            {
                var manager = new
 ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);

                var options = new DownloadOptions();
                options.StreamTransferProgress +=
 ArchiveDownloadHighLevel_GettingStarted.progress;
                // Download an archive.
                Console.WriteLine("Intiating the archive retrieval job and then polling
 SQS queue for the archive to be available.");
                Console.WriteLine("Once the archive is available, downloading will
 begin.");
                manager.Download(vaultName, archiveId, downloadFilePath, options);
                Console.WriteLine("To continue, press Enter");
                Console.ReadKey();
            }
            catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
            catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
            catch (Exception e) { Console.WriteLine(e.Message); }
            Console.WriteLine("To continue, press Enter");
            Console.ReadKey();
        }
```

```
        static int currentPercentage = -1;
        static void progress(object sender, StreamTransferProgressArgs args)
        {
            if (args.PercentDone != currentPercentage)
            {
                currentPercentage = args.PercentDone;
                Console.WriteLine("Downloaded {0}%", args.PercentDone);
            }
        }
    }
}
```

步驟 5：從 S3 Glacier 中的保存庫刪除封存

在此步驟中，您將刪除在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳的範例封存。

> **Important**
>
> 您無法使用 Amazon S3 Glacier 主控台刪除封存。任何封存作業 (例如上傳、下載或刪除) 都需要您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。要上傳數據 (例如照片，視頻和其他文檔) ，您必須直接使用 REST API AWS CLI 或使用 AWS SDK 來使用或編寫代碼來提出請求。
>
> 若要安裝 AWS CLI，請參閱AWS Command Line Interface。如需搭配使用 S3 冰川的詳細資訊 AWS CLI，請AWS CLI 參閱 S3 冰川的參考資料。如需使用將存檔上傳 AWS CLI 至 S3 冰川的範例，請參閱搭配使用 S3 冰川 AWS Command Line Interface。

刪除範例封存檔，請遵循下列其中一個開發套件或 AWS CLI：

- 使用 AWS SDK for Java 從 S3 Glacier 中的文件庫刪除封存
- 使用 AWS SDK for .NET 從 S3 Glacier 中的文件庫刪除封存
- 使用 AWS CLI刪除 S3 Glacier 中的封存

相關章節

- 步驟 3：將封存上傳至 S3 Glacier 中的保存庫

- 刪除 Amazon S3 Glacier 中的封存

使用 AWS SDK for Java 從 S3 Glacier 中的文件庫刪除封存

以下程式碼範例使用 AWS SDK for Java 來刪除封存。在程式碼中，請注意下列事項：

- `DeleteArchiveRequest` 物件描述刪除請求，包括封存歸檔所在的文件庫名稱和封存 ID。
- `deleteArchive` API 操作會將請求傳送至 Amazon S3 Glacier，以刪除封存。
- 此範例使用美國西部 (奧勒岡) 區域 (`us-west-2`)。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要按照在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳之檔案的封存 ID 來更新程式碼。

Example ：使用 AWS SDK for Java 刪除封存

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteArchiveRequest;

public class AmazonGlacierDeleteArchive_GettingStarted {

    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID***";
    public static AmazonGlacierClient client;

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


     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {

            // Delete the archive.
            client.deleteArchive(new DeleteArchiveRequest()
                .withVaultName(vaultName)
```

```
                .withArchiveId(archiveId));

            System.out.println("Deleted archive successfully.");

        } catch (Exception e) {
            System.err.println("Archive not deleted.");
            System.err.println(e);
        }
    }
}
```

使用 AWS SDK for .NET 從 S3 Glacier 中的文件庫刪除封存

以下 C# 程式碼範例使用 AWS SDK for .NET 的高階 API，刪除在先前步驟中上傳的封存。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon S3 Glacier 區域端點建立 `ArchiveTransferManager` 類別的執行個體。
- 此程式碼範例使用美國西部 (奧勒岡) 區域 (`us-west-2`)。
- 此範例使用所提供 `ArchiveTransferManager` 類別的 `Delete` API 操作作為 AWS SDK for .NET 的高階 API。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您需要按照在 步驟 3：將封存上傳至 S3 Glacier 中的保存庫 中上傳之檔案的封存 ID 來更新程式碼。

Example ：使用 AWS SDK for .NET 的高階 API 刪除封存

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteHighLevel_GettingStarted
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
```

```
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
        manager.DeleteArchive(vaultName, archiveId);
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

使用 AWS CLI刪除 S3 Glacier 中的封存

您可以使用 AWS Command Line Interface （ AWS CLI ）刪除 Amazon S3 冰川中的存檔。

主題

- (先決條件) 設定 AWS CLI
- 範例：使用刪除歸檔 AWS CLI

(先決條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需相關指示，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   配置 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證您的 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 保存庫的清單，請使用 `list-vaults` 命令。用您的身份*### 1234567890* 12。 AWS 帳戶

```
aws glacier list-vaults --account-id 123456789012
```

- 若要查看的目前規劃資料 AWS CLI，請使用`aws configure list`指令。

```
aws configure list
```

範例：使用刪除歸檔 AWS CLI

1. 使用 initiate-job 命令啟動清查擷取任務。如需 initiate-job 命令的詳細資訊，請參閱啟動工作。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters "{\"Type\": \"inventory-retrieval\"}"
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. 使用 describe-job 命令檢查先前擷取工作的狀態。如需有關 describe-job 命令的詳細資訊，請參閱描述工作。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

預期的輸出結果：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
```

```
}
```

3. 等候工作完成。

   您必須等到任務輸出準備好供您下載。如果您在保存庫上設定通知設定，或者在起始工作時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成工作後向該主題傳送訊息。

   您可以為文件庫中的特定事件設定通知組態。如需詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 Amazon SNS 主題。

4. 工作完成時，請使用 `get-job-output` 命令將擷取工作下載至檔案 `output.json`。如需 `get-job-output` 命令的詳細資訊，請參閱取得工作輸出。

   ```
   aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
    --job-id *** jobid *** output.json
   ```

   這個命令會產生一個包含下列欄位的檔案。

   ```
   {
   "VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
   "InventoryDate":"*** job completion date ***",
   "ArchiveList":[{
   {"ArchiveId":"*** archiveid ***",
   "ArchiveDescription":"*** archive description (if set) ***",
   "CreationDate":"*** archive creation date ***",
   "Size":"*** archive size (in bytes) ***",
   "SHA256TreeHash":"*** archive hash ***"
   }],
   "ArchiveId": 123456789

   }
   ```

5. 使用 `delete-archive` 命令從文件庫中刪除每個存檔，直到沒有存檔為止。

   ```
   aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
    --archive-id="*** archiveid ***"
   ```

如需有關 `delete-archive` 命令的詳細資訊，請參閱刪除封存。

步驟 6：在 S3 Glacier 中刪除保存庫

文件庫是儲存封存的一種容器。若要刪除 Amazon S3 Glacier 保存庫，您必須先刪除保存庫中的所有現有封存，直到 S3 Glacier 運算最後庫存為止。

您可以用程式設計方式或使用 S3 Glacier 主控台刪除保存庫。如需以程式設計方式刪除文件庫的資訊，請參閱 刪除 Amazon S3 Glacier 中的文件庫。

> **Important**
>
> 如果您在最近 24 小時內將封存上傳至保存庫，或從保存庫中刪除封存，您必須等到最後一個保存庫庫存更新，才能反映最新資訊。S3 Glacier 會每 24 小時定期為每個保存庫準備好庫存。

刪除空白文件庫的步驟

1. 登入 AWS Management Console 並開啟 S3 冰川主控台，網址為 https://console.aws.amazon.com/glacier/home。
2. 從 「選取區域」功能表中 AWS 區域 ，選擇您要刪除的 Vault。

   在此入門練習中，範例保存庫位於美國西部 (奧勒岡) 區域。
3. 選取您要刪除之空白保存庫旁的選項按鈕。如果保存庫不是空白，必須先刪除所有封存，然後再刪除保存庫。如需詳細資訊，請參閱 刪除 Amazon S3 Glacier 中的封存。

   > **Important**
   >
   > 刪除保存庫無法復原。
4. 選擇刪除。
5. 刪除保存庫對話方塊即會出現。選擇刪除。

刪除非空白文件庫的步驟

1. 若要刪除非空白保存庫，您必須先刪除所有現有的封存，然後再刪除保存庫。您可以透過撰寫程式碼來執行此操作，使用 REST API AWS SDK for Java、 AWS SDK for .NET 或 AWS CLI. 如需刪除封存的資訊，請參閱 步驟 5：從 S3 Glacier 中的保存庫刪除封存。
2. 保存庫為空之後，請依照上述程序中的步驟刪除空的保存庫。

接下來做些什麼？

既然您已嘗試入門練習，您可以瀏覽以下章節，以進一步了解 Amazon S3 Glacier。

- 在 Amazon S3 Glacier 中使用文件庫
- 在 Amazon S3 Glacier 中使用封存

在 Amazon S3 Glacier 中使用文件庫

文件庫是儲存封存的一種容器。建立文件庫時，指定文件庫名稱，以及要建立文件庫的 AWS 區域。如需 S3 Glacier 支援的 AWS 區域 清單，請參閱《AWS 一般參考》中的 Amazon S3 Glacier 端點和配額。

您可以在文件庫中存放無限數量的封存。

> **Important**
>
> S3 Glacier 確實會提供主控台。然而，任何封存操作，例如上傳、下載或刪除，都要求您使用 AWS Command Line Interface (AWS CLI) 或撰寫程式碼。沒有主控台支援封存操作。例如，若要上傳資料 (例如照片、影片和其他文件)，您必須使用 AWS CLI 或撰寫程式碼來發出請求，方法是直接使用 REST API 或使用 AWS 開發套件。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需使用 S3 Glacier 搭配 AWS CLI 的詳細資訊，請參閱 S3 Glacier的 AWS CLI 參考。如需使用 AWS CLI 將封存上傳到 S3 Glacier 的範例，請參閱使用 S3 Glacier 搭配 AWS Command Line Interface。

主題

S3 Glacier 中的文件庫操作

S3 Glacier 支援各種文件庫操作。文件庫操作專用於特定 AWS 區域。換句話說，當您建立文件庫時，會在特定 AWS 區域 中建立文件庫。當您列出文件庫時，S3 Glacier 會從您在請求中指定的 AWS 區域傳回文件庫清單。

建立和刪除文件庫

AWS 帳戶 每個 AWS 區域 最多可建立 1,000 個文件庫。如需 S3 Glacier 支援的 AWS 區域 清單，請參閱《AWS 一般參考》中的 Amazon S3 Glacier 端點和配額。

只有在文件庫中沒有封存的情況下，才可以刪除文件庫，因為在 S3 Glacier 計算的最後一個庫存中沒有封存，而且自上次清點以來，沒有寫入文件庫。

> **Note**
>
> S3 Glacier 會每 24 小時定期為每個文件庫準備好庫存。由於庫存可能不會反映最新資訊，S3 Glacier 可透過檢查上次文件庫清點以來是否有任何寫入操作，來確保文件庫確實是空的。

如需詳細資訊，請參閱在 Amazon S3 Glacier 中建立文件庫及刪除 Amazon S3 Glacier 中的文件庫。

擷取文件庫中繼資料

您可以擷取文件庫資訊，例如文件庫建立日期、文件庫中的封存數和文件庫中所有封存的總大小。S3 Glacier 為您提供了 API 呼叫，以擷取帳戶中特定 AWS 區域 區域的特定文件庫或所有文件庫的資訊。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中擷取文件庫中繼資料。

下載文件庫庫存

文件庫庫存是指文件庫中的封存清單。對於清單中的每個封存，庫存提供封存資訊，例如封存 ID、建立日期和大小。從第一次將封存上傳到文件庫的那一天起，S3 Glacier 大約每天更新一次文件庫庫存。文件庫清查都必須存在，您才能夠下載。

下載文件庫清查是一種非同步操作。您必須先起始任務以下載庫存。收到任務請求後，S3 Glacier 會備妥庫存以供下載。任務完成後，您即可下載庫存資料。

若由於任務的非同步本質，您可以使用 Amazon Simple Notification Service (Amazon SNS) 通知，以在任務完成時通知您。您可以為每個個別任務請求指定一個 Amazon SNS 主題，或者將文件庫設定為在特定文件庫事件發生時傳送通知。

S3 Glacier 會每 24 小時定期為每個文件庫準備好庫存。如果從上次清查以來，沒有新增或刪除文件庫的存檔，則清查日期不會更新。

當您為文件庫庫存啟動任務時，S3 Glacier 會傳回其產生的最後一個庫存，這是一個時間點快照，而不是即時資料。您可能沒有發現為每個存檔上傳擷取文件庫清查的好處。但是，假設您在用戶端上維護資

料庫，該用戶端包含與上傳到 S3 Glacier 的封存相關聯的中繼資料。然後，您可能會發現文件庫庫存的好處，可以在資料庫中使用實際的文件庫庫存來調節資訊。

如需有關擷取庫存的詳細資訊，請參閱在 Amazon S3 Glacier 中下載文件庫庫存。

設定文件庫通知

從 S3 Glacier 中擷取任何內容 (例如從文件庫或文件庫庫存中封存) 都是兩步驟程序。請先啟動任務。任務完成後，即可下載輸出。若要了解任務何時完成，您可以使用 S3 Glacier 通知。S3 Glacier 會將通知訊息傳送至您提供的 Amazon Simple Notification Service (Amazon SNS) 主題。

您可以設定文件庫通知，以及識別文件庫事件和事件發生時要通知的 Amazon SNS 主題。無論何時發生文件庫事件，S3 Glacier 都會將通知傳送到指定的 Amazon SNS 主題。如需更多詳細資訊，請參閱在 Amazon S3 Glacier 中設定文件庫通知。

在 Amazon S3 Glacier 中建立文件庫

建立文件庫可將文件庫新增到您帳戶中的一組文件庫中。AWS 帳戶 每個 AWS 區域最多可建立 1,000 個文件庫。如需 Amazon S3 Glacier (S3 Glacier) 支援的 AWS 區域清單，請參閱《AWS 一般參考》中的區域與端點。

建立文件庫時，必須提供文件庫名稱。以下是文件庫命名要求：

- 名稱長度可介於 1 到 255 個字元之間。
- 有效字元為 a-z、A-Z、0-9、'_' (底線)、'-' (連字號) 和 '.' (句號)。

文件庫名稱在帳戶和建立文件庫的 AWS 區域內必須是唯一的。也就是說，一個帳戶可以在 AWS 區域建立具有相同名稱的文件庫，但不能在相同 AWS 區域。

主題

- 使用 AWS SDK for Java 在 Amazon S3 Glacier 中建立文件庫
- 使用 AWS SDK for .NET 在 Amazon S3 Glacier 中建立文件庫
- 在 Amazon S3 Glacier 中使用 REST API 建立文件庫
- 使用 Amazon S3 Glacier 主控台建立文件庫
- 使用 AWS Command Line Interface 在 Amazon S3 Glacier 中建立文件庫

使用 AWS SDK for Java 在 Amazon S3 Glacier 中建立文件庫

低階 API 為所有文件庫操作提供了方法，包括建立和刪除文件庫、取得文件庫描述以及取得在特定 AWS 區域 中建立的文件庫清單。以下是使用 AWS SDK for Java 建立文件庫的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要在其中建立文件庫的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 您可以透過建立 `CreateVaultRequest` 類別的執行個體來提供請求資訊。

   Amazon S3 Glacier (S3 Glacier) 要求您提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用]。

3. 以參數形式提供請求物件，以便執行 `createVault` 方法。

   S3 Glacier 傳回的回應在 `CreateVaultResult` 物件中是可用的。

下列 Java 程式碼片段描述前述步驟。該程式碼片段會在 `us-west-2` 區域中建立文件庫。列印的 `Location` 是文件庫的相對 URI，其中包含帳戶 ID、AWS 區域 和文件庫名稱的。

```
AmazonGlacierClient client = new AmazonGlacierClient(credentials);
client.setEndpoint("https://glacier.us-west-2.amazonaws.com");

CreateVaultRequest request = new CreateVaultRequest()
    .withVaultName("*** provide vault name ***");
CreateVaultResult result = client.createVault(request);

System.out.println("Created vault successfully: " + result.getLocation());
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 [建立文件庫 (PUT 文件庫)]。

範例：使用 AWS SDK for Java 建立文件庫

以下 Java 程式碼範例在 us-west-2 區域中建立文件庫 (如需關於 AWS 區域 的詳細資訊，請參閱存取 Amazon S3 Glacier)。此外，程式碼範例擷取文件庫資訊, 列出同一 AWS 區域 中的所有文件庫，然後刪除所建立的文件庫。

如需如何執行下列範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。

Example

```
import java.io.IOException;
import java.util.List;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.CreateVaultRequest;
import com.amazonaws.services.glacier.model.CreateVaultResult;
import com.amazonaws.services.glacier.model.DeleteVaultRequest;
import com.amazonaws.services.glacier.model.DescribeVaultOutput;
import com.amazonaws.services.glacier.model.DescribeVaultRequest;
import com.amazonaws.services.glacier.model.DescribeVaultResult;
import com.amazonaws.services.glacier.model.ListVaultsRequest;
import com.amazonaws.services.glacier.model.ListVaultsResult;


public class AmazonGlacierVaultOperations {

    public static AmazonGlacierClient client;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        String vaultName = "examplevaultfordelete";

        try {
            createVault(client, vaultName);
            describeVault(client, vaultName);
            listVaults(client);
            deleteVault(client, vaultName);
```

```
        } catch (Exception e) {
            System.err.println("Vault operation failed." + e.getMessage());
        }
    }

    private static void createVault(AmazonGlacierClient client, String vaultName) {
        CreateVaultRequest createVaultRequest = new CreateVaultRequest()
            .withVaultName(vaultName);
        CreateVaultResult createVaultResult = client.createVault(createVaultRequest);

        System.out.println("Created vault successfully: " +
createVaultResult.getLocation());
    }

    private static void describeVault(AmazonGlacierClient client, String vaultName) {
        DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
            .withVaultName(vaultName);
        DescribeVaultResult describeVaultResult  =
client.describeVault(describeVaultRequest);

        System.out.println("Describing the vault: " + vaultName);
        System.out.print(
                "CreationDate: " + describeVaultResult.getCreationDate() +
                "\nLastInventoryDate: " + describeVaultResult.getLastInventoryDate() +
                "\nNumberOfArchives: " + describeVaultResult.getNumberOfArchives() +
                "\nSizeInBytes: " + describeVaultResult.getSizeInBytes() +
                "\nVaultARN: " + describeVaultResult.getVaultARN() +
                "\nVaultName: " + describeVaultResult.getVaultName());
    }

    private static void listVaults(AmazonGlacierClient client) {
        ListVaultsRequest listVaultsRequest = new ListVaultsRequest();
        ListVaultsResult listVaultsResult = client.listVaults(listVaultsRequest);

        List<DescribeVaultOutput> vaultList = listVaultsResult.getVaultList();
        System.out.println("\nDescribing all vaults (vault list):");
        for (DescribeVaultOutput vault : vaultList) {
            System.out.println(
                    "\nCreationDate: " + vault.getCreationDate() +
                    "\nLastInventoryDate: " + vault.getLastInventoryDate() +
                    "\nNumberOfArchives: " + vault.getNumberOfArchives() +
                    "\nSizeInBytes: " + vault.getSizeInBytes() +
                    "\nVaultARN: " + vault.getVaultARN() +
```

```
                    "\nVaultName: " + vault.getVaultName());
        }
    }

    private static void deleteVault(AmazonGlacierClient client, String vaultName) {
        DeleteVaultRequest request = new DeleteVaultRequest()
            .withVaultName(vaultName);
        client.deleteVault(request);
        System.out.println("Deleted vault: " + vaultName);
    }

}
```

使用 AWS SDK for .NET 在 Amazon S3 Glacier 中建立文件庫

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了文件庫的建立方法。

主題

- 使用 AWS SDK for .NET 的高階 API 建立文件庫
- 使用 AWS SDK for .NET 的低階 API 建立文件庫

使用 AWS SDK for .NET 的高階 API 建立文件庫

高階 API 的 `ArchiveTransferManager` 類別提供可用於在 AWS 區域中建立文件庫的 `CreateVault` 方法。

範例：使用 AWS SDK for .NET 的高階 API 的文件庫操作

以下 C# 程式碼範例建立和刪除美國西部 (奧勒岡) 區域中的文件庫。如需您可在其中建立文件庫的 AWS 區域 清單，請參閱存取 Amazon S3 Glacier。

如需如何執行下列範例的逐步說明，請參閱 執行程式碼範例。您需要更新所示具有保管庫名稱的程式碼。

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;
```

```
namespace glacier.amazon.com.docsamples
{
  class VaultCreateDescribeListVaultsDeleteHighLevel
  {
    static string vaultName = "*** Provide vault name ***";

    public static void Main(string[] args)
    {
      try
      {
          var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
          manager.CreateVault(vaultName);
          Console.WriteLine("Vault created. To delete the vault, press Enter");
          Console.ReadKey();
          manager.DeleteVault(vaultName);
          Console.WriteLine("\nVault deleted. To continue, press Enter");
          Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

使用 AWS SDK for .NET 的低階 API 建立文件庫

低階 API 為所有文件庫操作提供了方法，包括建立和刪除文件庫、取得文件庫描述以及取得在特定 AWS 區域 中建立的文件庫清單。以下是使用 AWS SDK for .NET 建立文件庫的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要在其中建立文件庫的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 您可以透過建立 `CreateVaultRequest` 類別的執行個體來提供請求資訊。

   Amazon S3 Glacier (S3 Glacier) 要求您提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用](#)。

3. 以參數形式提供請求物件，以便執行 `CreateVault` 方法。

   S3 Glacier 傳回的回應在 `CreateVaultResponse` 物件中是可用的。

範例：使用 AWS SDK for .NET 的低階 API 的文件庫操作

下列 C# 範例描述前述步驟。此範例在美國西部 (奧勒岡) 區域建立文件庫。此外，程式碼範例擷取文件庫資訊, 列出同一 AWS 區域 中的所有文件庫，然後刪除所建立的文件庫。列印的 `Location` 是文件庫的相對 URI，其中包含帳戶 ID、AWS 區域 和文件庫名稱。

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 建立文件庫 (PUT 文件庫)。

如需如何執行下列範例的逐步說明，請參閱 執行程式碼範例。您需要更新所示具有保管庫名稱的程式碼。

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class VaultCreateDescribeListVaultsDelete
  {
    static string vaultName = "*** Provide vault name ***";
    static AmazonGlacierClient client;

    public static void Main(string[] args)
    {
       try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Creating a vault.");
          CreateAVault();
          DescribeVault();
          GetVaultsList();
```

```
          Console.WriteLine("\nVault created. Now press Enter to delete the vault...");
          Console.ReadKey();
          DeleteVault();
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void CreateAVault()
    {
      CreateVaultRequest request = new CreateVaultRequest()
      {
        VaultName = vaultName
      };
      CreateVaultResponse response = client.CreateVault(request);
      Console.WriteLine("Vault created: {0}\n", response.Location);
    }

    static void DescribeVault()
    {
      DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
      {
        VaultName = vaultName
      };

      DescribeVaultResponse describeVaultResponse =
client.DescribeVault(describeVaultRequest);
      Console.WriteLine("\nVault description...");
      Console.WriteLine(
        "\nVaultName: " + describeVaultResponse.VaultName +
        "\nVaultARN: " + describeVaultResponse.VaultARN +
        "\nVaultCreationDate: " + describeVaultResponse.CreationDate +
        "\nNumberOfArchives: " + describeVaultResponse.NumberOfArchives +
        "\nSizeInBytes: " + describeVaultResponse.SizeInBytes +
        "\nLastInventoryDate: " + describeVaultResponse.LastInventoryDate
        );
    }

    static void GetVaultsList()
    {
```

```
      string lastMarker = null;
      Console.WriteLine("\n List of vaults in your account in the specific
 region ...");
      do
      {
        ListVaultsRequest request = new ListVaultsRequest()
        {
          Marker = lastMarker
        };
        ListVaultsResponse response = client.ListVaults(request);

        foreach (DescribeVaultOutput output in response.VaultList)
        {
          Console.WriteLine("Vault Name: {0} \tCreation Date: {1} \t #of archives:
 {2}",
                            output.VaultName, output.CreationDate,
 output.NumberOfArchives);
        }
        lastMarker = response.Marker;
      } while (lastMarker != null);
    }

    static void DeleteVault()
    {
      DeleteVaultRequest request = new DeleteVaultRequest()
      {
        VaultName = vaultName
      };
      DeleteVaultResponse response = client.DeleteVault(request);
    }
  }
}
```

在 Amazon S3 Glacier 中使用 REST API 建立文件庫

若要使用 REST API 建立文件庫，請參閱 建立文件庫 (PUT 文件庫)。

使用 Amazon S3 Glacier 主控台建立文件庫

若要使用 Amazon S3 Glacier (S3 Glacier) 主控台建立文件庫的更多資訊，請參閱入門教學中的步驟 2：在 S3 Glacier 中建立保存庫。

使用 AWS Command Line Interface 在 Amazon S3 Glacier 中建立文件庫

請依照下列步驟，使用 AWS Command Line Interface (AWS CLI)，在 Amazon S3 Glacier (S3 Glacier) 中建立文件庫。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 建立文件庫

1. 使用 `create-vault` 命令，在帳戶 *111122223333* 下建立一個名為 *awsexamplevault* 的保管庫。

```
aws glacier create-vault --vault-name awsexamplevault --account-id 111122223333
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault"
}
```

2. 使用 `describe-vault` 指令驗證建立。

```
aws glacier describe-vault --vault-name awsexamplevault --account-id 111122223333
```

在 Amazon S3 Glacier 中擷取文件庫中繼資料

您可以擷取文件庫資訊，例如文件庫建立日期、文件庫中的封存數和文件庫中所有封存的總大小。Amazon S3 Glacier (S3 Glacier) 為您提供了 API 呼叫，以擷取帳戶中特定 AWS 區域的特定文件庫或所有文件庫的資訊。

如果您擷取文件庫清單，S3 Glacier 將傳回按文件庫名稱之 ASCII 值排序的清單。該清單最多可包含 1,000 個文件庫。您應該一直檢查標記的回應以在此繼續列表；如果沒有更多項目，標記欄位為 `null`。您可以選擇限制 回應中傳回的文件庫數量。如果回應中的文件庫比所傳回的更多，則會對結果進行分頁。您需要傳送其他請求，以擷取下一組文件庫。

主題

- 使用 AWS SDK for Java 擷取 Amazon S3 Glacier 中的文件庫中繼資料
- 使用 AWS SDK for .NET 擷取 Amazon S3 Glacier 中的文件庫中繼資料
- 使用 REST API 擷取文件庫中繼資料
- 使用 AWS Command Line Interface 擷取 Amazon S3 Glacier 中的文件庫中繼資料

使用 AWS SDK for Java 擷取 Amazon S3 Glacier 中的文件庫中繼資料

主題

- 擷取文件庫的文件庫中繼資料

- 擷取區域中所有文件庫的文件庫中繼資料
- 範例：使用適用於 Java 的 Amazon 開發套件擷取文件庫中繼資料

擷取文件庫的文件庫中繼資料

您可以針對特定 AWS 區域中的特定文件庫或所有文件庫擷取中繼資料。以下的步驟說明如何使用適用於 Java 的 Amazon 開發套件的低階 API，為特定文件庫擷取文件庫中繼資料。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `DescribeVaultRequest` 類別的執行個體來提供請求資訊。

   Amazon S3 Glacier (S3 Glacier) 要求您提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用。
3. 以參數形式提供請求物件，以便執行 `describeVault` 方法。

   S3 Glacier 傳回的文件庫中繼資料資訊在 `DescribeVaultResult` 物件中是可用的。

下列 Java 程式碼片段描述前述步驟。

```
DescribeVaultRequest request = new DescribeVaultRequest()
 .withVaultName("*** provide vault name***");

DescribeVaultResult result = client.describeVault(request);

System.out.print(
        "\nCreationDate: " + result.getCreationDate() +
        "\nLastInventoryDate: " + result.getLastInventoryDate() +
        "\nNumberOfArchives: " + result.getNumberOfArchives() +
        "\nSizeInBytes: " + result.getSizeInBytes() +
        "\nVaultARN: " + result.getVaultARN() +
        "\nVaultName: " + result.getVaultName());
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 描述文件庫 (GET 文件庫)。

擷取區域中所有文件庫的文件庫中繼資料

您也可以使用 `listVaults` 方法，針對特定 AWS 區域中的所有文件庫擷取中繼資料。

以下 Java 程式碼片段擷取 `us-west-2` 區域中的文件庫清單。請求會限制回應中傳回的文件庫數量為 5 個。然後，程式碼片段將進行一系列 `listVaults` 呼叫，用以從該 AWS 區域擷取整個文件庫清單。

```
AmazonGlacierClient client;
client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

String marker = null;
do {
    ListVaultsRequest request = new ListVaultsRequest()
        .withLimit("5")
        .withMarker(marker);
    ListVaultsResult listVaultsResult = client.listVaults(request);

    List<DescribeVaultOutput> vaultList = listVaultsResult.getVaultList();
    marker = listVaultsResult.getMarker();
    for (DescribeVaultOutput vault : vaultList) {
        System.out.println(
                "\nCreationDate: " + vault.getCreationDate() +
                "\nLastInventoryDate: " + vault.getLastInventoryDate() +
                "\nNumberOfArchives: " + vault.getNumberOfArchives() +
                "\nSizeInBytes: " + vault.getSizeInBytes() +
                "\nVaultARN: " + vault.getVaultARN() +
                "\nVaultName: " + vault.getVaultName());
    }
} while (marker != null);
```

在前面的程式碼片段中，如果您沒有在請求中指定 `Limit` 值，則 S3 Glacier 將傳回多達 10 個文件庫，這些文件庫是由 S3 Glacier API 設定的。如果要列出更多的文件庫，則回應 `marker` 欄位包含文件庫 Amazon Resource Name (ARN)，以便以新的請求繼續列出，否則 `marker` 欄位為 null。

請注意，清單中為每個文件庫傳回的資訊與透過呼叫特定文件庫的 `describeVault` 方法所獲得的資訊相同。

> **Note**
>
> `listVaults` 方法呼叫底層 REST API (請參閱 「列出文件庫」(GET 文件庫))。

範例：使用適用於 Java 的 Amazon 開發套件擷取文件庫中繼資料

如需運作中程式碼範例，請參閱「範例：使用 AWS SDK for Java 建立文件庫」。Java 程式碼範例建立文件庫和擷取文件庫中繼資料。

使用 AWS SDK for .NET 擷取 Amazon S3 Glacier 中的文件庫中繼資料

主題

- 擷取文件庫的文件庫中繼資料
- 擷取區域中所有文件庫的文件庫中繼資料
- 範例：使用 AWS SDK for .NET 的低階 API 擷取文件庫中繼資料

擷取文件庫的文件庫中繼資料

您可以針對特定 AWS 區域中的特定文件庫或所有文件庫擷取中繼資料。以下是使用 AWS SDK for .NET 的低階 API 為特定文件庫擷取文件庫中繼資料的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `DescribeVaultRequest` 類別的執行個體來提供請求資訊。

   Amazon S3 Glacier (S3 Glacier) 要求您提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。
3. 以參數形式提供請求物件，以便執行 `DescribeVault` 方法。

   S3 Glacier 傳回的文件庫中繼資料資訊在 `DescribeVaultResult` 物件中是可用的。

下列 C# 程式碼片段描述前述步驟。程式碼片段擷取美國西部 (奧勒岡) 區域中現有文件庫的中繼資料資訊。

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

DescribeVaultRequest describeVaultRequest = new DescribeVaultRequest()
{
  VaultName = "*** Provide vault name ***"
};
DescribeVaultResponse describeVaultResponse =
 client.DescribeVault(describeVaultRequest);
Console.WriteLine("\nVault description...");
Console.WriteLine(
   "\nVaultName: " + describeVaultResponse.VaultName +
   "\nVaultARN: " + describeVaultResponse.VaultARN +
   "\nVaultCreationDate: " + describeVaultResponse.CreationDate +
   "\nNumberOfArchives: " + describeVaultResponse.NumberOfArchives +
   "\nSizeInBytes: " + describeVaultResponse.SizeInBytes +
   "\nLastInventoryDate: " + describeVaultResponse.LastInventoryDate
   );
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 描述文件庫 (GET 文件庫)。

擷取區域中所有文件庫的文件庫中繼資料

您也可以使用 `ListVaults` 方法，針對特定 AWS 區域中的所有文件庫擷取中繼資料。

以下 C# 程式碼片段擷取美國西部 (奧勒岡) 區域中的文件庫清單。請求會限制回應中傳回的文件庫數量為 5 個。然後，程式碼片段將進行一系列 `ListVaults` 呼叫，用以從該 AWS 區域擷取整個文件庫清單。

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);
string lastMarker = null;
Console.WriteLine("\n List of vaults in your account in the specific AWS Region ...");
```

```
do
{
  ListVaultsRequest request = new ListVaultsRequest()
  {
    Limit = 5,
    Marker = lastMarker
  };
  ListVaultsResponse response = client.ListVaults(request);

  foreach (DescribeVaultOutput output in response.VaultList)
  {
    Console.WriteLine("Vault Name: {0} \tCreation Date: {1} \t #of archives: {2}",
                      output.VaultName, output.CreationDate, output.NumberOfArchives);
  }
  lastMarker = response.Marker;
} while (lastMarker != null);
```

在前面的程式碼片段中，如果您沒有在請求中指定 `Limit` 值，則 S3 Glacier 將傳回多達 10 個文件庫，這些文件庫是由 S3 Glacier API 設定的。

請注意，清單中為每個文件庫傳回的資訊與透過呼叫特定文件庫的 `DescribeVault` 方法所獲得的資訊相同。

> **Note**
>
> `ListVaults` 方法呼叫底層 REST API (請參閱 「列出文件庫」(GET 文件庫))。

範例：使用 AWS SDK for .NET 的低階 API 擷取文件庫中繼資料

如需運作中程式碼範例，請參閱「範例：使用 AWS SDK for .NET 的低階 API 的文件庫操作」。C# 程式碼範例建立文件庫和擷取文件庫中繼資料。

使用 REST API 擷取文件庫中繼資料

若要使用 REST API 列出文件庫的更多資訊，請參閱 「列出文件庫」(GET 文件庫)。若要描述一個文件庫的詳細資訊，請參閱 描述文件庫 (GET 文件庫)。

使用 AWS Command Line Interface 擷取 Amazon S3 Glacier 中的文件庫中繼資料

此範例示範如何使用 AWS Command Line Interface (AWS CLI) 擷取 Amazon S3 Glacier (S3 Glacier) 中的文件庫資訊和中繼資料。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 擷取文件庫中繼資料

- 使用 `describe-vault` 命令，在帳戶 *111122223333* 下描述一個名為 *awsexamplevault* 的文件庫。

```
aws glacier describe-vault --vault-name awsexamplevault --account-id 111122223333
```

在 Amazon S3 Glacier 中下載文件庫庫存

將第一個封存上傳到文件庫後，Amazon S3 Glacier (S3 Glacier) 會自動建立文件庫庫存，然後每天大約更新一次。S3 Glacier 建立第一個庫存後，通常需要半天到一天的時間才可以擷取庫存。您可以透過以下兩個步驟從 S3 Glacier 中擷取文件庫庫存：

1. 使用 啟動任務 (POST 任務) 操作啟動庫存擷取任務。

> **Important**
>
> 資料擷取政策可能導致啟動擷取作業請求失敗，並出現 `PolicyEnforcedException` 例外狀況。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。如需 `PolicyEnforcedException` 例外狀況的詳細資訊，請參閱 錯誤回應。

2. 任務完成後，使用 「取得任務輸出」 (GET 輸出) 操作下載位元組。

例如，擷取存檔或文件庫清查要求您先啟動擷取作業。以非同步方式執行任務請求。當您開始擷取任務時，S3 Glacier 會建立任務並在回應中傳回任務 ID。當 S3 Glacier 完成任務時，您可以取得任務輸出、封存位元組或文件庫庫存的資料。

必須完成任務，才能取得其輸出。若要判斷任務的狀態，您有下列選項：

- 等待任務完成通知：您可以指定在任務完成後，S3 Glacier 可以發佈通知的 Amazon Simple Notification Service (Amazon SNS) 主題。您可以使用以下方法指定此 Amazon SNS 主題：
  - 為每個任務指定 Amazon SNS 主題。

    啟動任務時，需選擇指定 Amazon SNS 主題。
  - 在文件庫上設定通知組態。

您可以為文件庫中的特定事件設定通知設定 (請參閱 在 Amazon S3 Glacier 中設定文件庫通知)。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 SNS 主題。

如果您在文件庫上已設定通知設定，而且您在啟動任務時也指定 Amazon SNS 主題，當您啟動任務時，S3 Glacier 會將任務完成訊息傳送到這兩個主題。

您可以將 SNS 主題設定為透過電子郵件通知您，或者將訊息儲存在應用程式可以輪詢的 Amazon Simple Queue Service (Amazon SQS) 中。當訊息出現在佇列中時，您可以檢查任務是否順利完成，然後下載任務的輸出。

- 明確地請求任務資訊：S3 Glacier 也提供描述任務操作 (描述任務 (GET JobID))，可讓您輪詢任務資訊。您可以定期發送此請求以獲取任務資訊。但是，使用 Amazon SNS 通知是建議的選項。

> **Note**
>
> 您透過 SNS 通知取得的資訊，與您呼叫描述任務所取得的資訊相同。

主題

關於庫存

從您第一次將封存上傳到文件庫的那一天起，S3 Glacier 大約每天更新一次文件庫庫存。如果從上次清查以來，沒有新增或刪除文件庫的存檔，則清查日期不會更新。當您為文件庫庫存啟動任務時，S3 Glacier 會傳回其產生的最後一個庫存，這是一個時間點快照，而不是即時資料。請注意，S3 Glacier 為文件庫建立第一個庫存後，通常需要半天到一天的時間才可以擷取庫存。

您可能沒有發現為每個存檔上傳擷取文件庫清查的好處。不過，假設您在用戶端上維護一個資料庫，該用戶端將上傳到 S3 Glacier 之封存的相關中繼資料建立關聯。然後，您可能會發現文件庫清查的好處，可以視需要在資料庫中使用實際的文件庫清查來調節資訊。您可以透過篩選存檔建立日期或設定配額來限制擷取到的清查項目數量。如需有關限制庫存擷取的詳細資訊，請參閱 庫存擷取範圍。

您可以以逗號分隔值 (CSV) 或 JSON 兩種格式傳回庫存。您可以選擇指定啟動庫存任務時的格式。預設格式是 JSON。如需有關庫存任務輸出中傳回的資料欄位的詳細資訊，請參閱回應內文取得任務輸出 API 的 。

使用 AWS SDK for Java 在 Amazon S3 Glacier 中下載文件庫庫存

以下是使用 AWS SDK for Java 低階 API 來擷取文件庫庫存的步驟。高階的 API 不支援擷取文件庫庫存。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 透過執行 `initiateJob` 方法來起始庫存擷取任務。

   透過提供 `initiateJob` 物件中的任務資訊來執行 `InitiateJobRequest`。

   > **Note**
   >
   > 請注意，如果文件庫庫存尚未完成，便會傳回錯誤。Amazon S3 Glacier (S3 Glacier) 會每 24 小時定期為每個文件庫準備好庫存。

   S3 Glacier 會在回應中傳回任務 ID。該回應在 `InitiateJobResult` 類別的執行個體中可用。

```
InitiateJobRequest initJobRequest = new InitiateJobRequest()
    .withVaultName("*** provide vault name ***")
    .withJobParameters(
            new JobParameters()
                .withType("inventory-retrieval")
                .withSNSTopic("*** provide SNS topic ARN ****")
      );

InitiateJobResult initJobResult = client.initiateJob(initJobRequest);
String jobId = initJobResult.getJobId();
```

3. 等候任務完成。

您必須等到任務輸出準備好供您下載。如果您在文件庫上設定通知設定，或者在起始任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

您也可以透過呼叫 `describeJob` 方法來輪詢 S3 Glacier，以判斷任務完成狀態。不過，使用 Amazon SNS 主題進行通知是建議的方法。以下章節提供的程式碼使用適用於 S3 Glacier 的 Amazon SNS 來發布訊息。

4. 透過執行 `getJobOutput` 方法下載任務輸出 (文件庫庫存資料)。

   您透過建立 `GetJobOutputRequest` 類別的執行個體來提供帳戶 ID、任務 ID 和文件庫名稱。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用]。

   S3 Glacier 傳回的輸出在 `GetJobOutputResult` 物件中可用。

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withVaultName("*** provide vault name ***")
        .withJobId("*** provide job ID ***");
GetJobOutputResult jobOutputResult = client.getJobOutput(jobOutputRequest);
// jobOutputResult.getBody(); provides the output stream.
```

> **Note**
>
> 如需與任務相關的底層 REST API 的資訊，請參閱 [任務操作]。

範例：使用適用於 Java 的 Amazon 開發套件擷取文件庫庫存

以下 Java 程式碼範例將擷取指定文件庫的文件庫清查。

範例會執行下列任務：

- 建立 Amazon Simple Notification Service (Amazon SNS) 主題。

  S3 Glacier 會在完成任務後向該主題傳送通知。
- 建立 Amazon Simple Queue Service (Amazon SQS) 佇列。

此範例將政策連接至佇列，以使 Amazon SNS 主題能夠發佈訊息到佇列。

- 起始任務以下載指定的封存。

  在任務請求中，會指定所建立的 Amazon SNS 主題，因此，S3 Glacier 可以在完成任務後發布通知到主題。

- 檢查 Amazon SQS 佇列中是否有包含任務 ID 的訊息。

  如果有訊息，剖析 JSON 並檢查任務是否順利完成。如果是，請下載封存。

- 透過刪除 Amazon SNS 主題和其建立的 Amazon SQS 佇列來清除。

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.fasterxml.jackson.core.JsonFactory;
import com.fasterxml.jackson.core.JsonParseException;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
```

```
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class AmazonGlacierDownloadInventoryWithSQSPolling {

    public static String vaultName = "*** provide vault name ***";
    public static String snsTopicName = "*** provide topic name ***";
    public static String sqsQueueName = "*** provide queue name ***";
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name ***";
    public static String region = "*** region ***";
    public static long sleepTime = 600;
    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");
```

```
        try {
            setupSQS();

            setupSNS();

            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            Boolean success = waitForJobToComplete(jobId, sqsQueueURL);
            if (!success) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId);

            cleanUp();

        } catch (Exception e) {
            System.err.println("Inventory retrieval failed.");
            System.err.println(e);
        }
    }

    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
```

```
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);

        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {

        JobParameters jobParameters = new JobParameters()
            .withType("inventory-retrieval")
            .withSNSTopic(snsTopicARN);

        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }

    private static Boolean waitForJobToComplete(String jobId, String sqsQueueUrl)
throws InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
```

```
                new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").textValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").textValue();
                    String statusCode = jobDescNode.get("StatusCode").textValue();
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
          }
        return (messageFound && jobSuccessful);
    }

    private static void downloadJobOutput(String jobId) throws IOException {

        GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
            .withVaultName(vaultName)
            .withJobId(jobId);
        GetJobOutputResult getJobOutputResult =
client.getJobOutput(getJobOutputRequest);

        FileWriter fstream = new FileWriter(fileName);
        BufferedWriter out = new BufferedWriter(fstream);
        BufferedReader in = new BufferedReader(new
InputStreamReader(getJobOutputResult.getBody()));
        String inputLine;
        try {
            while ((inputLine = in.readLine()) != null) {
                out.write(inputLine);
```

```
            }
        }catch(IOException e) {
            throw new AmazonClientException("Unable to save archive", e);
        }finally{
            try {in.close();}  catch (Exception e) {}
            try {out.close();}  catch (Exception e) {}
        }
        System.out.println("Retrieved inventory to " + fileName);
    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
}
```

使用 AWS SDK for .NET 在 Amazon S3 Glacier 中下載文件庫庫存

以下是使用 AWS SDK for .NET 低階 API 來擷取文件庫庫存的步驟。高階的 API 不支援擷取文件庫庫存。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 透過執行 `InitiateJob` 方法來起始庫存擷取任務。

   您可以在 `InitiateJobRequest` 物件中提供任務資訊。Amazon S3 Glacier (S3 Glacier) 會在回應中傳回任務 ID。該回應在 `InitiateJobResponse` 類別的執行個體中可用。

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
  VaultName = vaultName,
  JobParameters = new JobParameters()
  {
    Type = "inventory-retrieval",
    SNSTopic = "*** Provide Amazon SNS topic arn ***",
  }
```

```
};
InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

3. 等候任務完成。

   您必須等到任務輸出準備好供您下載。如果您在文件庫上設定識別 Amazon Simple Notification Service (Amazon SNS) 主題的通知設定，或者在開始任務時指定 Amazon SNS 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。以下章節提供的程式碼使用適用於 S3 Glacier 的 Amazon SNS 來發布訊息。

   您也可以透過呼叫 `DescribeJob` 方法來輪詢 S3 Glacier，以判斷任務完成狀態。雖然，使用 Amazon SNS 主題進行通知是建議的方法。

4. 透過執行 `GetJobOutput` 方法下載任務輸出 (文件庫庫存資料)。

   透過建立 `GetJobOutputRequest` 類別的執行個體來提供帳戶 ID、文件庫名稱和任務 ID 資訊。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用]。

   S3 Glacier 傳回的輸出在 `GetJobOutputResponse` 物件中可用。

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};

GetJobOutputResponse getJobOutputResponse =
 client.GetJobOutput(getJobOutputRequest);
using (Stream webStream = getJobOutputResponse.Body)
{
   using (Stream fileToSave = File.OpenWrite(fileName))
   {
     CopyStream(webStream, fileToSave);
   }
}
```

> **Note**
>
> 如需與任務相關的底層 REST API 的資訊，請參閱 任務操作。

範例：使用 AWS SDK for .NET 的低階 API 擷取文件庫庫存

以下 C# 程式碼範例將擷取指定文件庫的文件庫庫存。

範例會執行下列任務：

- 設定 Amazon SNS 主題。

  S3 Glacier 會在完成任務後向該主題傳送通知。
- 設定 Amazon SQS 佇列。

  此範例將政策連接至佇列，以使 Amazon SNS 主題能夠發佈訊息。
- 啟動任務以下載指定的封存。

  在任務請求中，該範例指定 Amazon SNS 主題，以便 S3 Glacier 在任務完成後傳送訊息。
- 定期檢查 Amazon SQS 佇列中的訊息。

  如果有訊息，剖析 JSON 並檢查任務是否順利完成。如果是，請下載封存。該代碼範例使用 JSON.NET 程式庫 (請參閱 JSON.NET) 來剖析 JSON。
- 透過刪除 Amazon SNS 主題和其建立的 Amazon SQS 佇列來清除。

Example

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
```

```
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;

namespace glacier.amazon.com.docsamples
{
  class VaultInventoryJobLowLevelUsingSNSSQS
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string fileName  = "*** Provide file name and path where to store inventory
 ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
    const string SQS_POLICY =
        "{" +
        "    \"Version\" : \"2012-10-17\"," +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"AWS\" : \"arn:aws:iam::123456789012:root\" }," 
 +
        "            \"Action\"    : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QuernArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
        "            }" +
        "        }" +
        "    ]" +
        "}";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
            Console.WriteLine("Setup SNS topic and SQS queue.");
```

```
            SetupTopicAndQueue();
            Console.WriteLine("To continue, press Enter"); Console.ReadKey();

            Console.WriteLine("Retrieve Inventory List");
            GetVaultInventory(client);
        }
        Console.WriteLine("Operations successful.");
        Console.WriteLine("To continue, press Enter"); Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
       // Delete SNS topic and SQS queue.
       snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
       sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }

    static void SetupTopicAndQueue()
    {
      long ticks = DateTime.Now.Ticks;

      // Setup SNS topic.
      snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
      sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);

      topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
      Console.Write("topicArn: "); Console.WriteLine(topicArn);

      CreateQueueRequest createQueueRequest =  new CreateQueueRequest();
      createQueueRequest.QueueName = "GlacierDownload-" + ticks;
      CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
      queueUrl = createQueueResponse.QueueUrl;
      Console.Write("QueueURL: "); Console.WriteLine(queueUrl);

      GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
      getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
      getQueueAttributesRequest.QueueUrl = queueUrl;
```

```
      GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
      queueArn = response.QueueARN;
      Console.Write("QueueArn: ");Console.WriteLine(queueArn);

      // Setup the Amazon SNS topic to publish to the SQS queue.
      snsClient.Subscribe(new SubscribeRequest()
      {
        Protocol = "sqs",
        Endpoint = queueArn,
        TopicArn = topicArn
      });

      // Add the policy to the queue so SNS can send messages to the queue.
      var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QuernArn}",
queueArn);

      sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
      {
          QueueUrl = queueUrl,
          Attributes = new Dictionary<string, string>
          {
              { QueueAttributeName.Policy, policy }
          }
      });

   }

   static void GetVaultInventory(AmazonGlacierClient client)
   {
      // Initiate job.
      InitiateJobRequest initJobRequest = new InitiateJobRequest()
      {
        VaultName = vaultName,
        JobParameters = new JobParameters()
        {
          Type = "inventory-retrieval",
          Description = "This job is to download a vault inventory.",
          SNSTopic = topicArn,
        }
      };

      InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
      string jobId = initJobResponse.JobId;
```

```
     // Check queue for a message and if job completed successfully, download
inventory.
     ProcessQueue(jobId, client);
   }

   private static void ProcessQueue(string jobId, AmazonGlacierClient client)
   {
     ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest()
{ QueueUrl = queueUrl, MaxNumberOfMessages = 1 };
     bool jobDone = false;
     while (!jobDone)
     {
       Console.WriteLine("Poll SQS queue");
       ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
       if (receiveMessageResponse.Messages.Count == 0)
       {
         Thread.Sleep(10000 * 60);
         continue;
       }
       Console.WriteLine("Got message");
       Message message = receiveMessageResponse.Messages[0];
       Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
       Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
       string statusCode = fields["StatusCode"] as string;

       if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
       {
         Console.WriteLine("Downloading job output");
         DownloadOutput(jobId, client); // Save job output to the specified file
location.
       }
       else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
         Console.WriteLine("Job failed... cannot download the inventory.");

       jobDone = true;
       sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
ReceiptHandle = message.ReceiptHandle });
     }
```

```
    }

    private static void DownloadOutput(string jobId, AmazonGlacierClient client)
    {
      GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
      {
        JobId = jobId,
        VaultName = vaultName
      };

      GetJobOutputResponse getJobOutputResponse =
  client.GetJobOutput(getJobOutputRequest);
      using (Stream webStream = getJobOutputResponse.Body)
      {
        using (Stream fileToSave = File.OpenWrite(fileName))
        {
          CopyStream(webStream, fileToSave);
        }
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

使用 REST API下載文件庫清查

使用 REST API下載文件庫清查

下載文件庫清查程序包含兩個步驟。

1. 啟動 `inventory-retrieval` 類型的任務。如需更多詳細資訊，請參閱 [啟動任務 (POST 任務)]。
2. 任務完成後，下載庫存資料。如需更多詳細資訊，請參閱 [「取得任務輸出」 (GET 輸出)]。

使用 AWS Command Line Interface 在 Amazon S3 Glacier 中下載文件庫庫存

請遵循以下步驟，使用 AWS Command Line Interface (AWS CLI) 下載 Amazon S3 Glacier (S3 Glacier) 中的文件庫庫存。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 下載文件庫清查

1. 使用 `initiate-job` 命令啟動清查擷取任務。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters='{"Type": "inventory-retrieval"}'
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. 使用 describe-job 命令檢查先前擷取任務的狀態。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

預期的輸出結果：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. 等候任務完成。

您必須等到任務輸出準備好供您下載。S3 Glacier 完成任務後，任務 ID 至少在 24 小時內不會過期。如果您在文件庫上設定通知設定，或者在起始任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

您可以為文件庫中的特定事件設定通知組態。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 SNS 主題。

4. 完成時，請使用 `get-job-output` 命令將擷取任務下載至檔案 `output.json`。

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

這個命令會產生一個包含下列欄位的檔案。

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[
{"ArchiveId":"*** archiveid ***",
"ArchiveDescription":"*** archive description (if set) ***",
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}
{"ArchiveId":
...
]}
```

在 Amazon S3 Glacier 中設定文件庫通知

從 Amazon S3 Glacier 中擷取任何內容 (例如從文件庫或文件庫庫存中封存) 都是兩步驟程序。

1. 啟動擷取任務。
2. 任務完成後，下載任務輸出。

可以在文件庫中設定通知設定，以便在任務完成後，將訊息傳送到 Amazon Simple Notification Service (Amazon SNS) 主題。

主題

- 在 S3 Glacier 中設定文件庫通知：一般概念

- 使用 AWS SDK for Java 在 Amazon S3 Glacier 中設定文件庫通知
- 使用 AWS SDK for .NET 在 Amazon S3 Glacier 中設定文件庫通知
- 使用 REST API，在 S3 Glacier 中設定文件庫通知
- 使用 S3 Glacier 主控台設定文件庫通知
- 使用 AWS Command Line Interface 設定文件庫通知

在 S3 Glacier 中設定文件庫通知：一般概念

S3 Glacier 擷取任務請求是以非同步方式執行的。您必須等到 S3 Glacier 完成任務後才能取得其輸出。您可以定期輪詢 S3 Glacier 來判斷任務狀態，但那不是最佳的方法。S3 Glacier 也支援通知。任務完成後，該任務就可將訊息張貼到 Amazon Simple Notification Service (Amazon SNS) 主題。使用此功能需要您在文件庫上設定通知設定。您可以在設定中識別一或多個事件，以及您希望 S3 Glacier 在事件發生時向其傳送訊息的 Amazon SNS 主題。

S3 Glacier 定義與任務完成 (`ArchiveRetrievalCompleted`、`InventoryRetrievalCompleted`) 具體相關的事件，您可以將其新增至文件庫的通知設定中。當特定任務完成後，S3 Glacier 會將通知訊息發布到 SNS 主題。

通知組態是 JSON 文件，如以下範例所示。

```
{
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

請注意，您只能為文件庫設定一個 Amazon SNS 主題。

> **Note**
>
> 將通知設定新增到文件庫，會導致 S3 Glacier 在每次發生通知設定中所指定的事件時傳送通知。您還可以選擇在每個任務啟動請求中指定 Amazon SNS 主題。如果在文件庫中同時新增通知設定，並在啟動任務請求中指定 Amazon SNS 主題，則 S3 Glacier 會同時傳送這兩個通知。

任務完成訊息 S3 Glacier 傳送包括以下資訊：任務類型 (`InventoryRetrieval`、`ArchiveRetrieval`)、任務完成狀態，SNS 主題名稱、任務狀態碼，以

及文件庫 ARN。以下是 InventoryRetrieval 任務完成後，傳送到 SNS 主題的通知 S3 Glacier 範例。

```
{
 "Action": "InventoryRetrieval",
 "ArchiveId": null,
 "ArchiveSizeInBytes": null,
 "Completed": true,
 "CompletionDate": "2012-06-12T22:20:40.790Z",
 "CreationDate": "2012-06-12T22:20:36.814Z",
 "InventorySizeInBytes":11693,
 "JobDescription": "my retrieval job",
 "JobId":"HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
 "SHA256TreeHash":null,
 "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
 "StatusCode":"Succeeded",
 "StatusMessage": "Succeeded",
 "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

如果 Completed 欄位為 true，您還必須檢查 StatusCode 來確認任務順利完成還是失敗。

> **Note**
>
> 請注意，Amazon SNS 主題必須允許文件庫發布通知。在預設情況下，只有 Amazon SNS 主題擁有者可以向該主題發布訊息。不過，如果 Amazon SNS 主題和文件庫屬於不同的 AWS 帳戶，則必須將 Amazon SNS 主題設定為接受由文件庫的發布。您可以在 Amazon SNS 主控台設定 Amazon SNS 主題政策。

如需 Amazon SNS 的詳細資訊，請參閱 Amazon SNS 入門。

使用 AWS SDK for Java 在 Amazon S3 Glacier 中設定文件庫通知

以下是使用 AWS SDK for Java 低階 API 設定文件庫通知的步驟。

1. 建立 AmazonGlacierClient 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 SetVaultNotificationsRequest 類別的執行個體，來提供通知組態資訊。

您需要提供文件庫名稱、通知組態資訊和帳戶 ID。在指定通知設定時，您提供現有 Amazon SNS 主題的 Amazon Resource Name (ARN) 和要進行通知的一或多個事件。如需支援的事件清單，請參閱設定文件庫通知組態 (PUT 通知的組態)。

3. 以參數形式提供請求物件，以便執行 `setVaultNotifications` 方法。

下列 Java 程式碼片段描述前述步驟。程式碼片段在文件庫上設定通知組態。當 `ArchiveRetrievalCompleted` 事件或 `InventoryRetrievalCompleted` 事件發生時，設定請求 Amazon S3 Glacier (S3 Glacier) 將通知傳送到指定的 Amazon SNS 主題。

```
SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
        .withAccountId("-")
        .withVaultName("*** provide vault name ***")
        .withVaultNotificationConfig(
                new VaultNotificationConfig()
                .withSNSTopic("*** provide SNS topic ARN ***")
                .withEvents("ArchiveRetrievalCompleted", "InventoryRetrievalCompleted")
         );
client.setVaultNotifications(request);
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 文件庫操作。

範例：使用 AWS SDK for Java 設定文件庫的通知組態

以下 Java 程式碼範例設定文件庫的通知組態，刪除組態，然後復原組態。如需如何執行下列範例的逐步說明，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用。

Example

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.GetVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.GetVaultNotificationsResult;
```

```
import com.amazonaws.services.glacier.model.SetVaultNotificationsRequest;
import com.amazonaws.services.glacier.model.VaultNotificationConfig;


public class AmazonGlacierVaultNotifications {

    public static AmazonGlacierClient client;
    public static String vaultName = "*** provide vault name ****";
    public static String snsTopicARN = "*** provide sns topic ARN ***";

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {

            System.out.println("Adding notification configuration to the vault.");
            setVaultNotifications();
            getVaultNotifications();
            deleteVaultNotifications();

        } catch (Exception e) {
            System.err.println("Vault operations failed." + e.getMessage());
        }
    }

    private static void setVaultNotifications() {
        VaultNotificationConfig config = new VaultNotificationConfig()
            .withSNSTopic(snsTopicARN)
            .withEvents("ArchiveRetrievalCompleted", "InventoryRetrievalCompleted");

        SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
                .withVaultName(vaultName)
                .withVaultNotificationConfig(config);

        client.setVaultNotifications(request);
        System.out.println("Notification configured for vault: " + vaultName);
    }

    private static void getVaultNotifications() {
        VaultNotificationConfig notificationConfig = null;
```

```
        GetVaultNotificationsRequest request = new GetVaultNotificationsRequest()
                .withVaultName(vaultName);
        GetVaultNotificationsResult result = client.getVaultNotifications(request);
        notificationConfig = result.getVaultNotificationConfig();

        System.out.println("Notifications configuration for vault: "
                + vaultName);
        System.out.println("Topic: " + notificationConfig.getSNSTopic());
        System.out.println("Events: " + notificationConfig.getEvents());
    }

    private static void deleteVaultNotifications() {
            DeleteVaultNotificationsRequest request = new
 DeleteVaultNotificationsRequest()
                .withVaultName(vaultName);
            client.deleteVaultNotifications(request);
            System.out.println("Notifications configuration deleted for vault: " +
 vaultName);
    }
}
```

使用 AWS SDK for .NET 在 Amazon S3 Glacier 中設定文件庫通知

以下是使用 AWS SDK for .NET 低階 API 設定文件庫通知的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定文件庫所在的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `SetVaultNotificationsRequest` 類別的執行個體，來提供通知組態資訊。

   您需要提供文件庫名稱、通知組態資訊和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用]。

   在指定通知設定時，您提供現有 Amazon SNS 主題的 Amazon Resource Name (ARN) 和要進行通知的一或多個事件。如需支援的事件清單，請參閱[設定文件庫通知組態 (PUT 通知的組態)]。
3. 以參數形式提供請求物件，以便執行 `SetVaultNotifications` 方法。
4. 在設定有關文件庫的通知組態之後，您可以透過呼叫 `GetVaultNotifications` 方法擷取組態資訊，以及透過呼叫用戶端所提供的 `DeleteVaultNotifications` 方法將其移除。

範例：使用 AWS SDK for .NET 設定文件庫的通知組態

下列 C# 程式碼範例描述前述步驟。此範例在美國西部 (奧勒岡) 區域設定有關文件庫 (「examplevault」) 的通知設定、擷取設定，然後將其刪除。當 ArchiveRetrievalCompleted 事件或 InventoryRetrievalCompleted 事件發生時，設定請求 Amazon S3 Glacier (S3 Glacier) 將通知傳送到指定的 Amazon SNS 主題。

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 文件庫操作。

如需執行下列範例的逐步說明，請參閱「執行程式碼範例」。您需要如所示更新程式碼，並提供現有的文件庫名稱和 Amazon SNS 主題。

Example

```
using System;
using System.Collections.Generic;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class VaultNotificationSetGetDelete
  {
    static string vaultName   = "examplevault";
    static string snsTopicARN = "*** Provide Amazon SNS topic ARN ***";

    static IAmazonGlacier client;

    public static void Main(string[] args)
    {
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Adding notification configuration to the vault.");
          SetVaultNotificationConfig();
          GetVaultNotificationConfig();
          Console.WriteLine("To delete vault notification configuration, press Enter");
```

```
          Console.ReadKey();
          DeleteVaultNotificationConfig();
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void SetVaultNotificationConfig()
    {

      SetVaultNotificationsRequest request = new SetVaultNotificationsRequest()
      {
        VaultName = vaultName,
        VaultNotificationConfig = new VaultNotificationConfig()
        {
          Events   = new List<string>() { "ArchiveRetrievalCompleted",
"InventoryRetrievalCompleted" },
          SNSTopic = snsTopicARN
        }
      };
      SetVaultNotificationsResponse response = client.SetVaultNotifications(request);
    }

    static void GetVaultNotificationConfig()
    {
      GetVaultNotificationsRequest request = new GetVaultNotificationsRequest()
      {
        VaultName = vaultName,
        AccountId = "-"
      };
      GetVaultNotificationsResponse response = client.GetVaultNotifications(request);
      Console.WriteLine("SNS Topic ARN: {0}",
response.VaultNotificationConfig.SNSTopic);
      foreach (string s in response.VaultNotificationConfig.Events)
        Console.WriteLine("Event : {0}", s);
    }

    static void DeleteVaultNotificationConfig()
    {
      DeleteVaultNotificationsRequest request = new DeleteVaultNotificationsRequest()
```

```
      {
        VaultName = vaultName
      };
      DeleteVaultNotificationsResponse response =
 client.DeleteVaultNotifications(request);
    }
  }
}
```

使用 REST API，在 S3 Glacier 中設定文件庫通知

若要使用 REST API 設定文件庫通知，請參閱 設定文件庫通知組態 (PUT 通知的組態)。此外，您還可以取得文件庫通知 (取得文件庫通知 (GET 通知的組態)) 和刪除文件庫通知 (刪除文件庫通知 (DELETE 通知的組態))。

使用 S3 Glacier 主控台設定文件庫通知

本節說明如何使用 Amazon S3 Glacier 主控台，設定文件庫通知。設定通知時，您可以指定任務完成事件，其會將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題。除了設定文件庫通知外，您也可以指定在啟動任務時發布通知的主題。如果將保存件庫設為通知特定事件，並且在任務啟動請求中設定通知，則會傳送兩個通知。

要設定文件庫通知

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在左側導覽窗格中，選擇文件庫。
3. 在文件庫清單中，選擇文件庫。
4. 在通知區段中，選擇編輯。
5. 在事件通知頁面，選擇開啟通知。
6. 在通知區段中，選擇下列其中一個 Amazon Simple Notification Service (Amazon SNS) 選項，然後按照對應的步驟進行操作：

Amazon SNS 選項	動作
建立新的 SNS 主題	1. 選擇建立新的 SNS 主題。

Amazon SNS 選項	動作
	2. 對於主題名稱，輸入新主題的名稱。 主題名稱長度上限為 256 個字元。允許英數字母、連字號 (-) 和底線 (_)。主題名稱在帳戶和 AWS 區域中必須是獨一無二的。 3. (選用) 如果您要使用 SMS 訊息訂閱主題，請輸入顯示名稱的名稱。 顯示名稱的長度上限為 100 個字元。
選擇現有的 SNS 主題	1. 選擇選擇現有的 SNS 主題。 2. 在指定 SNS 主題下，選擇以下其中一個選項： • 從您的 SNS 主題中選擇 此時會顯示 SNS 主題下拉式清單。 請從下拉式清單中選擇現有主題。 • 輸入 SNS 主題 ARN 此時會顯示 Amazon SNS 主題 ARN 文字方塊。 輸入 SNS 主題的 Amazon Resource Name (ARN)。SNS 主題 ARN 使用下列格式： `arn:aws:sns:region:account-id:topic-name` 您可以在 Amazon SNS 主控台中找到 SNS 主題 ARN。

7. 在事件下，選取您要傳送通知的一或兩個事件：

   • 若要僅在封存擷取任務完成時傳送通知，請選取封存擷取任務完成。

- 若要僅在文件庫庫存任務完成時傳送通知，請選取文件庫庫存擷取任務完成。

使用 AWS Command Line Interface 設定文件庫通知

本節說明如何使用 AWS Command Line Interface 設定文件庫通知。設定通知時，您要指定任務完成事件，其會觸發對 Amazon Simple Notification Service (Amazon SNS) 主題的通知。除了設定文件庫通知外，您也可以指定在啟動作業時發佈通知的主題。如果您的文件庫設定為通知特定事件，並且在作業啟動請求中指定通知，則會發送兩個通知。

請遵循下列步驟，使用 AWS CLI 設定文件庫通知。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 設定文件庫通知

1. 使用 `set-vault-notifications` 命令，設定文件庫發生特定事件時將傳送的通知。根據預設，您不會收到任何通知。

```
aws glacier set-vault-notifications --vault-name examplevault --account-
id 111122223333 --vault-notification-config file://notificationconfig.json
```

2. 通知組態是 JSON 文件，如以下範例所示。

```
{
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

   如需在 S3 Glacier 使用 Amazon SNS 主題的詳細資訊，請參閱在 S3 Glacier 中設定文件庫通知：一般概念

   如需 Amazon SNS 的詳細資訊，請參閱 Amazon SNS 入門。

刪除 Amazon S3 Glacier 中的文件庫

只有在截至上次計算的庫存為止，文件庫中沒有任何封存，而且自上次清點以來，未對文件庫進行任何寫入時，Amazon S3 Glacier (S3 Glacier) 才會刪除文件庫。如需刪除封存的資訊，請參閱 刪除 Amazon S3 Glacier 中的封存。如需下載文件庫清查的詳細資訊，請參閱在 Amazon S3 Glacier 中下載文件庫庫存。

> **Note**
>
> S3 Glacier 會每 24 小時定期為每個文件庫準備好庫存。由於庫存可能不會反映最新資訊，S3 Glacier 可透過檢查上次文件庫清點以來是否有任何寫入操作，來確保文件庫確實是空的。

主題

- 使用 AWS SDK for Java 刪除 Amazon S3 Glacier 中的文件庫
- 使用 AWS SDK for .NET 刪除 Amazon S3 Glacier 中的文件庫
- 在 S3 Glacier 中使用 REST API 刪除文件庫

- 範例：使用 S3 Glacier 主控台刪除空白文件庫
- 使用 AWS Command Line Interface 刪除 Amazon S3 Glacier 中的文件庫

使用 AWS SDK for Java 刪除 Amazon S3 Glacier 中的文件庫

以下是使用 AWS SDK for Java 低階 API 來刪除文件庫的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要從要刪除文件庫的位置指定 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 您可以透過建立 `DeleteVaultRequest` 類別的執行個體來提供請求資訊。

   您需要提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用。

3. 以參數形式提供請求物件，以便執行 `deleteVault` 方法。

   Amazon S3 Glacier (S3 Glacier) 只會在文件庫為空時才會將其刪除。如需更多詳細資訊，請參閱 刪除文件庫 (DELETE 文件庫)。

下列 Java 程式碼片段描述前述步驟。

```
try {
    DeleteVaultRequest request = new DeleteVaultRequest()
        .withVaultName("*** provide vault name ***");

    client.deleteVault(request);
    System.out.println("Deleted vault: " + vaultName);
} catch (Exception e) {
    System.err.println(e.getMessage());
}
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 刪除文件庫 (DELETE 文件庫)。

範例：使用 AWS SDK for Java 刪除文件庫

如需運作中程式碼範例，請參閱「範例：使用 AWS SDK for Java 建立文件庫」。Java 程式碼範例顯示基本文件庫操作，包括建立和刪除文件庫。

使用 AWS SDK for .NET 刪除 Amazon S3 Glacier 中的文件庫

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了建立文件庫的方法。

主題

- 使用 AWS SDK for .NET 的高階 API 刪除文件庫
- 使用 AWS SDK for .NET 的低階 API 刪除文件庫

使用 AWS SDK for .NET 的高階 API 刪除文件庫

高階 API 的 `ArchiveTransferManager` 類別提供可用來刪除文件庫的 `DeleteVault` 方法。

範例：使用 AWS SDK for .NET 的高階 API 刪除文件庫

如需運作中程式碼範例，請參閱「範例：使用 AWS SDK for .NET 的高階 API 的文件庫操作」。C# 程式碼範例顯示基本文件庫操作，包括建立和刪除文件庫。

使用 AWS SDK for .NET 的低階 API 刪除文件庫

以下是使用 AWS SDK for .NET 刪除文件庫的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要從要刪除文件庫的位置指定 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `DeleteVaultRequest` 類別的執行個體來提供請求資訊。

   您需要提供文件庫名稱和帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。
3. 以參數形式提供請求物件，以便執行 `DeleteVault` 方法。

   Amazon S3 Glacier (S3 Glacier) 只會在文件庫為空時才會將其刪除。如需更多詳細資訊，請參閱 刪除文件庫 (DELETE 文件庫)。

下列 C# 程式碼片段描述前述步驟。程式碼片段擷取在預設 AWS 區域中存在的文件庫中繼資料資訊。

```
AmazonGlacier client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USEast1);

DeleteVaultRequest request = new DeleteVaultRequest()
{
  VaultName = "*** provide vault name ***"
};

DeleteVaultResponse response = client.DeleteVault(request);
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 刪除文件庫 (DELETE 文件庫)。

範例：使用 AWS SDK for .NET 的低階 API 刪除文件庫

如需運作中程式碼範例，請參閱「範例：使用 AWS SDK for .NET 的低階 API 的文件庫操作」。C# 程式碼範例顯示基本文件庫操作，包括建立和刪除文件庫。

在 S3 Glacier 中使用 REST API 刪除文件庫

若要使用 REST API 刪除文件庫，請參閱 刪除文件庫 (DELETE 文件庫)。

範例：使用 S3 Glacier 主控台刪除空白文件庫

> **Note**
>
> 若要刪除文件庫，您必須先刪除該文件庫內所有現有的封存。您可以透過編寫程式碼來發出刪除封存請求，方法是使用 REST API、AWS SDK for Java、AWS SDK for .NET 或使用 AWS Command Line Interface (AWS CLI)。如需刪除封存的資訊，請參閱 步驟 5：從 S3 Glacier 中的保存庫刪除封存。

文件庫清空後，您就可以使用下列步驟將其刪除。

使用 Amazon S3 Glacier 主控台刪除空白文件庫

1. 登入 AWS Management Console，並開啟位於 S3 Glacier 主控台上的 S3 Glacier 主控台。

2. 在選取區域下，選擇文件庫所在的 AWS 區域 位置。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫清單中，選取要刪除之文件庫名稱旁的選項，然後選擇頁面頂端的刪除。
5. 在刪除文件庫對話方塊中，透過選擇刪除來確認您要刪除文件庫。

> **Important**
>
> 刪除文件庫無法復原。

6. 若要確認您是否已刪除文件庫，請開啟文件庫清單，然後輸入您已刪除的文件庫名稱。如果找不到文件庫，就表示已成功刪除。

使用 AWS Command Line Interface 刪除 Amazon S3 Glacier 中的文件庫

您可以使用 AWS Command Line Interface (AWS CLI) 刪除 Amazon S3 Glacier (S3 Glacier) 中的空和不為空的文件庫。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

   ```
   aws help
   ```

- 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

```
aws glacier list-vaults --account-id 123456789012
```

- 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

```
aws configure list
```

範例：使用 AWS CLI 刪除空的文件庫

- 使用 `delete-vault` 指令刪除不包含存檔的文件庫。

```
aws glacier delete-vault --vault-name awsexamplevault --account-id 111122223333
```

範例：使用 AWS CLI 刪除非空的文件庫

只有在截至上次計算的庫存為止，文件庫中沒有任何封存，而且自上次清點以來，未對文件庫進行任何寫入時，S3 Glacier 才會刪除文件庫。刪除非空的文件庫有三個步驟：從文件庫的清查報告擷取存檔 ID、刪除每個存檔，然後刪除文件庫。

1. 使用 `initiate-job` 命令啟動清查擷取任務。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters='{"Type": "inventory-retrieval"}'
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. 使用 `describe-job` 命令檢查先前擷取任務的狀態。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

預期的輸出結果：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. 等候任務完成。

   您必須等到任務輸出準備好供您下載。如果您在文件庫上設定通知設定，或者在起始任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

   您可以為文件庫中的特定事件設定通知組態。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 SNS 主題。

4. 完成時，請使用 `get-job-output` 命令將擷取任務下載至檔案 `output.json`。

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

這個命令會產生一個包含下列欄位的檔案。

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[
{"ArchiveId":"*** archiveid ***",
```

```
"ArchiveDescription":*** archive description (if set) ***,
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}
{"ArchiveId":
...
]}
```

5. 使用 `delete-archive` 命令從文件庫中刪除每個存檔，直到沒有存檔為止。

```
aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
 --archive-id="*** archiveid ***"
```

> **Note**
>
> 如果封存 ID 的開頭為連字號或其他特殊字元，您必須將其放在引號中，才能執行此命令。

6. 使用 `initiate-job` 命令來啟動新的清查擷取任務。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters='{"Type": "inventory-retrieval"}'
```

7. 完成後，請使用 `delete-vault` 命令刪除沒有存檔的文件庫。

```
aws glacier delete-vault --vault-name awsexamplevault --account-id 111122223333
```

標記 S3 Glacier 文件庫

您可以用標籤的形式，將自己的中繼資料指派給 Amazon S3 Glacier 文件庫。標籤是您為文件庫定義的金鑰值對。如需有關標籤的基本資訊，包括標籤的限制，請參閱 標記 Amazon S3 Glacier 資源。

下列主題說明如何新增、列出和移除文件庫的標籤。

主題

- 使用 Amazon S3 Glacier 主控台標記文件庫
- 使用 AWS CLI 標記文件庫

- 使用 Amazon S3 Glacier API 標記文件庫
- 相關章節

使用 Amazon S3 Glacier 主控台標記文件庫

您可以如下列程序所述，使用 S3 Glacier 主控台新增、列出和移除標籤。

查看文件庫的標籤

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在選取區域下，從「區域」選擇器中選取 AWS 區域。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫清單中，選擇文件庫。
5. 選擇文件庫屬性索引標籤。捲動至標籤區段以檢視與文件庫相關聯的標籤。

若要新增標籤到文件庫

您最多可以將 50 個標籤與文件庫建立關聯。與文件庫相關聯的標籤，必須具備唯一的標籤索引鍵。

如需標籤限制的詳細資訊，請參閱標記 Amazon S3 Glacier 資源。

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在選取區域下，從「區域」選擇器中選取 AWS 區域。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫清單中，選擇要新增標籤的文件庫名稱。
5. 選擇文件庫屬性索引標籤。
6. 在 Tags (標籤) 區段中，選擇 Add (新增)。Add tags (新增標籤) 頁面隨即出現。
7. 在新增標籤頁面上，在索引鍵欄位內指定標籤索引鍵，(選用) 在值欄位內指定標籤值。
8. 選擇 Save changes (儲存變更)。

編輯標籤

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在選取區域下，從「區域」選擇器中選取 AWS 區域。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫清單中，選擇文件庫名稱。
5. 選擇文件庫屬性索引標籤，然後向下捲動至標籤區段。
6. 在標籤下，選取要變更之標籤旁邊的核取方塊，然後選擇編輯。即會出現編輯標籤頁面。
7. 更新索引鍵欄位中的標籤索引鍵，並選擇性地更新值欄位中的標籤值。
8. 選擇 Save changes (儲存變更)。

若要從文件庫中移除標籤

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在選取區域下，從「區域」選擇器中選取 AWS 區域。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫清單中，選擇要從中移除標籤的文件庫名稱。
5. 選擇文件庫屬性索引標籤。向下捲動至標籤區段。
6. 在標籤下，選取要移除之標籤旁邊的核取方塊，然後選擇刪除。
7. 刪除標籤對話方塊隨即開啟。如要確認刪除所選標籤，請選擇刪除。

使用 AWS CLI 標記文件庫

請遵循下列步驟，使用 AWS Command Line Interface (AWS CLI) 新增、列出或移除標籤。

每個標籤皆包含鍵與值。每個文件庫最多可擁有 50 個標籤。

1. 若要將標籤新增至文件庫，請使用 `add-tags-to-vault` 命令。

```
aws glacier add-tags-to-vault --vault-name examplevault --account-id 111122223333
 --tags id=1234,date=2020
```

   若要取得此文件庫操作的更多資訊，請參閱將標籤新增至文件庫。

2. 若要列出連接至文件庫的所有標籤，請使用 `list-tags-for-vault` 命令。

```
aws glacier list-tags-for-vault --vault-name examplevault --account-id 111122223333
```

若要取得此文件庫操作的更多資訊，請參閱列出文件庫的標籤。

3. 若要從連接至文件庫的一組標籤移除一或多個標籤，請使用 `remove-tags-from-vault` 命令。

```
aws glacier remove-tags-from-vault --vault-name examplevault --account-
id 111122223333 --tag-keys date
```

若要取得此文件庫操作的更多資訊，請參閱將標籤從文件庫中移除。

使用 Amazon S3 Glacier API 標記文件庫

您可以使用 S3 Glacier API 來新增、列出和移除標籤。如需範例，請參閱下列文件：

新增標籤至文件庫 (POST 標籤新增)

新增或更新所指定文件庫的標籤。

列出文件庫的標籤 (GET 標籤)

列出所指定文件庫的標籤。

從文件庫移除標籤 (POST tags remove)

從指定的文件庫中移除標籤。

相關章節

- 標記 Amazon S3 Glacier 資源

S3 Glacier 文件庫鎖定

下列主題說明如何鎖定 Amazon S3 Glacier 中的文件庫以及如何使用文件庫鎖定政策。

主題

- 文件庫鎖定概觀

- 使用 S3 Glacier API 鎖定文件庫
- 使用 AWS Command Line Interface 鎖定文件庫
- 使用 S3 Glacier 主控台鎖定文件庫

文件庫鎖定概觀

S3 Glacier 文件庫鎖定可讓您使用文件庫鎖定政策，輕鬆部署並強制執行各個 S3 Glacier 文件庫的合規控制。您可以在文件庫鎖定政策中指定控制功能，例如「單寫多讀」(WORM)，並鎖定該政策以防未來進行編輯。

> **Important**
>
> 鎖定文件庫鎖定政策後，將無法再變更或刪除該政策。

S3 Glacier 會強制執行文件庫鎖定政策中的控制集，以協助達成您的合規目標。例如，您可以使用文件庫鎖定政策強制執行資料保留。您可以使用 AWS Identity and Access Management (IAM) 政策語言，在文件庫鎖定政策中部署各種合規控制。如需文件庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

文件庫鎖定政策不同於文件庫存取政策。兩個政策都可管理文件庫的存取控制。不過，文件庫鎖定政策是可鎖定的，以防止未來進行變更，為合規控制提供強而有力的執行力。您可以使用文件庫鎖定政策定期部署法規和合規性控制，這通常需要對資料的存取進行緊密的控制。

> **Important**
>
> 我們建議您先建立文件庫、完成文件庫鎖定政策，然後將封存上傳至文件庫，以便將政策套用至其中。

相對的，您使用文件庫存取政策來對不是與何規性相關、暫時和需要頻繁修改，施行存取控制。您可以將文件庫鎖定和文件庫存取政策一起搭配使用。例如，您可以在文件庫鎖定政策 (拒絕刪除) 中實作以時間為基礎的資料保留規則，以及在文件庫存取政策中授予讀取權限給指定的第三方或您的業務合作夥伴 (允許讀取)。

鎖定文件庫需要採取兩個步驟：

1. 透過將文件庫鎖定政策連接至文件庫來啟動鎖定，這會將鎖定設定為進行中的狀態，並傳回鎖定 ID。政策處於進行中狀態時，您有 24 小時可在鎖定 ID 過期之前驗證文件庫鎖定政策。若要防止文

件庫結束進行中狀態，您必須在這 24 小時內完成文件庫鎖定程序。否則，文件庫鎖定政策將遭到刪除。

2. 使用鎖定 ID 來完成鎖定程序。如果文件庫鎖定政策運作未如預期，您可以停止鎖定程序並從頭重新開始。如需有關如何使用 S3 Glacier API 鎖定文件庫的詳細資訊，請參閱使用 S3 Glacier API 鎖定文件庫。

使用 S3 Glacier API 鎖定文件庫

若要使用 Amazon S3 Glacier API 鎖定文件庫，您先使用文件庫鎖定政策呼叫 啟動文件庫鎖定 (POST 鎖定政策)，此政策會指定您要部署的控制項。`Initiate Vault Lock` 操作會將政策連接至文件庫，將文件庫鎖定轉為進行中狀態，並傳回唯一的鎖定 ID。文件庫鎖定進入進行中狀態後，您有 24 小時可透過使用 完成文件庫鎖定 (POST lockId) 呼叫傳回的鎖定 ID 呼叫 `Initiate Vault Lock` 來完成鎖定。

> **Important**
>
> - 我們建議您先建立文件庫、完成文件庫鎖定政策，然後將封存上傳至文件庫，以便將政策套用至其中。
> - 在文件庫鎖定政策遭鎖定之後，再也無法變更或刪除該政策。

如果您未在進入進行中狀態後的 24 小時內完成文件庫鎖定程序，文件庫會自動結束進行中狀態，並且文件庫鎖定政策會遭到移除。您可以再次呼叫 `Initiate Vault Lock` 安裝新的文件庫鎖定政策，並轉換到進行中狀態。

進行中狀態提供機會讓您在鎖定前測試文件庫鎖定政策。文件庫鎖定政策在進行中狀態會充分發揮作用，宛如文件庫已經鎖定，不過您也可以透過呼叫 「中止文件庫鎖定」(DELETE 鎖定政策) 來移除政策。若要調整政策，您可以依需要多次重複 `Abort Vault Lock`/`Initiate Vault Lock` 組合，來驗證文件庫鎖定政策變更。

在您驗證文件庫鎖定政策後，您可以使用最近的鎖定 ID 呼叫 完成文件庫鎖定 (POST lockId)，來完成文件庫鎖定程序。文件庫會轉換到鎖定狀態，其中文件庫鎖定政策將不可變更，並且無法再透過呼叫 `Abort Vault Lock` 移除。

相關章節

- 保存庫鎖定政策

- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 完成文件庫鎖定 (POST lockId)
- 取得文件庫鎖定 (GET 鎖定政策)
- 啟動文件庫鎖定 (POST 鎖定政策)

使用 AWS Command Line Interface 鎖定文件庫

您可以使用 AWS Command Line Interface 鎖定文件庫。這會在指定的文件庫上安裝文件庫鎖定政策，並傳回鎖定 ID。您必須在 24 小時內完成文件庫鎖定程序，否則會將文件庫鎖定政策從文件庫中移除。

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

1. 使用 `initiate-vault-lock` 安裝文件庫鎖定政策，並將文件庫鎖定的鎖定狀態設為 `InProgress`。

```
aws glacier initiate-vault-lock --vault-name examplevault --account-id 111122223333
 --policy file://lockconfig.json
```

2. 通知設定是 JSON 文件，如以下範例所示。在使用此命令之前，請將 *VAULT_ARN* 和 *Principal* 替換為適合您使用案例的值。

   若要尋找要鎖定的文件庫 ARN，您可以使用 `list-vaults` 命令。

```
{"Policy":"{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-lock
\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam::111122223333:root\"},
\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"VAULT_ARN\",\"Condition\":
{\"NumericLessThanEquals\":{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}"}
```

3. 啟動文件庫鎖定之後，您應該會看到傳回的 `lockId`。

```
{
    "lockId": "LOCK_ID"
}
```

若要完成文件庫鎖定，您必須在 24 小時內執行 `complete-vault-lock`，否則會將文件庫鎖定政策從文件庫中移除。

```
aws glacier complete-vault-lock --vault-name examplevault --account-id 111122223333 --
lock-id LOCK_ID
```

相關章節

- 在 AWS CLI 命令參考中的 initiate-vault-lock
- 在 AWS CLI 命令參考中的 list-vaults
- 在 AWS CLI 命令參考中的 complete-vault-lock
- 保存庫鎖定政策
- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 完成文件庫鎖定 (POST lockId)
- 取得文件庫鎖定 (GET 鎖定政策)
- 啟動文件庫鎖定 (POST 鎖定政策)

使用 S3 Glacier 主控台鎖定文件庫

Amazon S3 Glacier 文件庫鎖定可讓您使用文件庫鎖定政策，輕鬆部署並強制執行各個 S3 Glacier 文件庫的合規控制。如需 S3 Glacier 文件庫鎖定的詳細資訊，請參閱使用文件庫鎖定政策的 Amazon S3 Glacier 存取控制。

> **Important**
>
> - 我們建議您先建立文件庫、完成文件庫鎖定政策，然後將封存上傳至文件庫，以便將政策套用至其中。
> - 在文件庫鎖定政策遭鎖定之後，再也無法變更或刪除該政策。

使用 S3 Glacier 主控台在文件庫上啟動文件庫鎖定政策

您透過將文件庫鎖定政策連接至文件庫來啟動鎖定，這會將鎖定設為進行中的狀態，並傳回鎖定 ID。政策處於進行中狀態時，您有 24 小時可在鎖定 ID 過期之前驗證文件庫鎖定政策。

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在選取區域下，從「區域」選擇器中選取 AWS 區域。
3. 在左側導覽窗格中，選擇文件庫。
4. 在文件庫頁面上，選擇建立文件庫。
5. 建立新的文件庫。

   > **Important**
   >
   > 我們建議您先建立文件庫、完成文件庫鎖定政策，然後將封存上傳至文件庫，以便將政策套用至其中。

6. 從文件庫清單中選擇新的文件庫。
7. 選擇文件庫政策索引標籤。
8. 在文件庫鎖定政策區段中，選擇啟動文件庫鎖定政策。
9. 在啟動文件庫鎖定政策頁面上，在標準文字方塊中，以文字格式指定文件庫鎖定政策中的記錄保留控制項。

> **Note**
>
> 您能夠以文字格式指定文件庫鎖定政策中的記錄保留控制項，並透過呼叫 `Initiate Vault Lock` API 操作或透過 S3 Glacier 主控台中的互動式 UI 來啟動文件庫鎖定。如需格式化文件庫鎖定政策的相關資訊，請參閱 Amazon S3 Glacier 文件庫鎖定政策範例。

10. 選擇 Save changes (儲存變更)。
11. 在記錄文件庫鎖定 ID 對話方塊中，複製鎖定 ID 並將其儲存在安全的位置。

> **Important**
>
> 啟動文件庫鎖定政策後，您有 24 小時的時間驗證政策並完成鎖定程序。若要完成鎖定程序，您必須提供鎖定 ID。如果未在 24 小時內提供，鎖定 ID 就會過期，而且進行中的政策也會遭到刪除。

12. 將鎖定 ID 儲存在安全的地方後，選擇關閉。
13. 在接下來的 24 小時內測試文件庫鎖定政策。如果政策如預期運作，請選擇完成文件庫鎖定政策。
14. 在完成文件庫鎖定對話方塊中，選取該核取方塊以確認「文件庫鎖定」政策程序的完成是不可還原的。
15. 在文字方塊中輸入您提供的鎖定 ID。
16. 選擇完成文件庫鎖定。

在 Amazon S3 Glacier 中使用封存

封存是存放在文件庫中的任何物件 (如相片、影片或文件)。其是在 Amazon S3 Glacier (S3 Glacier) 中的基本儲存單位。每個封存都有唯一的 ID 以及選擇性說明。當您上傳封存時，S3 Glacier 會傳回回應，其中包含封存 ID。這個封存 ID 在儲存封存的 AWS 區域中是唯一的。以下是封存 ID 範例。

```
TJgHcrOSfAkV6hdPqOATYfp_0ZaxL1pIBOc02iZ0gDPMr2ig-
nhwd_PafstsdIf6HSrjHnP-3p6LCJClYytFT_CBhT9CwNxbRaM5MetS3I-
GqwxI3Y8QtgbJbhEQPs0mJ3KExample
```

封存 ID 長度為 138 位元組。當您上傳封存，可以提供可選的說明。您可以使用其 ID 而不是其說明來擷取封存。

> **Important**
>
> S3 Glacier 提供管理主控台。您可以使用主控台來建立及刪除文件庫。不過，所有與 S3 Glacier 的其他互動，都需要您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。例如，若要上傳資料 (例如照片、影片和其他文件)，您必須使用 AWS CLI 或撰寫程式碼來發出請求，方法是直接使用 REST API 或使用 Amazon 開發套件。如需使用 S3 Glacier 搭配 AWS CLI 的詳細資訊，請參閱 S3 Glacier 的 AWS CLI 參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。

主題

Amazon S3 Glacier 中的封存操作

S3 Glacier 支援以下基本封存操作：上傳、下載和刪除。下載封存是一種非同步操作。

在 Amazon S3 Glacier 中上傳封存

您可以透過單一操作上傳封存，也可以分段上傳封存。您用來上傳部分封存的 API 呼叫稱為分段上傳。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中上傳封存。

> **Important**
>
> S3 Glacier 提供管理主控台。您可以使用主控台來建立及刪除文件庫。不過，所有與 S3 Glacier 的其他互動，都需要您使用 AWS Command Line Interface (CLI) 或撰寫程式碼。例如，若要上傳資料 (例如照片、影片和其他文件)，您必須使用 AWS CLI 或撰寫程式碼來發出請求，方法是直接使用 REST API 或使用 Amazon 開發套件。如需使用 S3 Glacier 搭配 AWS CLI 的詳細資訊，請參閱 S3 Glacier 的 AWS CLI 參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。

在 Amazon S3 Glacier 中尋找封存 ID

您可以透過為包含封存的文件庫下載文件庫庫存來取得封存 ID。如需下載文件庫庫存的詳細資訊，請參閱在 Amazon S3 Glacier 中下載文件庫庫存。

在 Amazon S3 Glacier 下載封存

下載封存是一種非同步操作。您必須先啟動任務來下載特定的封存。在收到任務請求後，S3 Glacier 會準備封存以供下載。任務完成後，下載封存資料。因為任務的非同步本質，您可以請求 S3 Glacier 在任務完成時，向 Amazon Simple Notification Service (Amazon SNS) 主題傳送通知。您可以為每個個別任務請求指定一個 SNS 主題，或者設定您的文件庫在特定事件發生時傳送通知。如需下載封存的詳細資訊，請參閱 在 S3 Glacier 中下載封存。

刪除 Amazon S3 Glacier 中的封存

S3 Glacier 提供 API 呼叫，您可以使用此呼叫，一次刪除一個封存。如需更多詳細資訊，請參閱 刪除 Amazon S3 Glacier 中的封存。

在 S3 Glacier 中更新封存

在您上傳封存後，您不能更新內容或其說明。您可以更新封存內容或其說明的唯一方法是刪除封存並上傳另一個封存。請注意，每次上傳封存時，S3 Glacier 都會傳回唯一的封存 ID。

維護用戶端封存中繼資料

除了選填的封存說明外，S3 Glacier 不支援封存的任何額外中繼資料。上傳封存時，S3 Glacier 會指派一個 ID (不透明的字元序列)，您無法從中推斷出封存的任何含義。您可以在用戶端維護封存的中繼資料。中繼資料可以包括封存名稱和有關封存的其他有意義的資訊。

> **Note**
>
> 如果您是 Amazon Simple Storage Service (Amazon S3) 的客戶，您就會知道，當物件上傳到儲存貯體時，您可以將物件指定為物件金鑰，例如 `MyDocument.txt` 或 `SomePhoto.jpg`。在 S3 Glacier 中，不能為上傳的封存指定物件索引鍵。

如果您維護用戶端封存中繼資料，請注意，S3 Glacier 會維護文件庫庫存，其中包括封存 ID 以及在封存上傳期間提供的任何說明。您可能偶爾會下載文件庫庫存，以協調為封存中繼資料維護的用戶端資料庫中的任何問題。但是，S3 Glacier 幾乎每天都需要文件庫庫存。當您請求文件庫庫存時，S3 Glacier 會傳回其所準備的最後一個庫存，是某個時間點快照。

在 Amazon S3 Glacier 中上傳封存

Amazon S3 Glacier (S3 Glacier) 提供的管理主控台，可用來建立和刪除文件庫。但是，您無法藉由使用管理主控台，將封存上傳到 S3 Glacier。若要上傳資料 (例如照片、影片和其他文件)，您必須使用 AWS CLI 或撰寫程式碼來發出請求，方法是直接使用 REST API 或使用 Amazon 開發套件。

如需使用 S3 Glacier 搭配 AWS CLI 的詳細資訊，請參閱 S3 Glacier 的 AWS CLI 參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。以下上傳主題說明如何使用適用於 Java 的 Amazon 開發套件、適用於 .NET 的 Amazon 開發套件和 REST API，將封存上傳到 S3 Glacier。

主題

- 將封存上傳到 Amazon S3 Glacier 的選項
- 在單一操作中上傳封存
- 上傳分段中的大型封存 (分段上傳)

將封存上傳到 Amazon S3 Glacier 的選項

視上傳資料大小之不同，S3 Glacier 提供下列選項：

- 在單一操作中上傳封存：在單一操作中，您可以上傳大小為 1 位元組到最大 4 GB 的封存。但是，我們還是鼓勵 S3 Glacier 客戶使用分段上傳，來上傳大於 100 MB 的封存。如需更多詳細資訊，請參閱 在單一操作中上傳封存 。
- 上傳部分封存 - 使用分段上傳 API，您可以上傳大型封存，最高可達 40,000 GB (10,000* 4 GB)。

  分段上傳 API 呼叫是專為改善較大型封存上傳體驗所設計。您可以分段上傳封存。這些封存部分可個別、依任何順序以及同時上傳。如果某個部分上傳失敗，您只需要再次上傳該部分，而不是整個封存。您可以為大小介於 1 位元組到 40,000 GB 之間的封存使用分段上傳。如需更多詳細資訊，請參閱 上傳分段中的大型封存 (分段上傳)。

> **Important**
>
> 此 S3 Glacier 文件庫庫存每天只會更新一次。當上傳封存時，您將不會立即看到新增到文件庫的新存檔 (在主控台中或在下載的文件庫清查清單中)，直到文件庫清查已更新。

使用 AWS Snowball 服務

AWS Snowball 藉由使用 Amazon 擁有的裝置，繞過網際網路，加速將大量資料移入和移出 AWS。如需詳細資訊，請參閱 AWS Snowball 的詳細資訊頁面。

若要將現有資料上傳到 Amazon S3 Glacier (S3 Glacier)，您可能會考慮使用其中一個 AWS Snowball 裝置類型，將資料匯入 Amazon S3，然後使用生命週期規則，將其移至 S3 Glacier 儲存類別，以供封存使用。當您將 Amazon S3 物件轉換為 S3 Glacier 儲存類別時，Amazon S3 會在內部使用 S3 Glacier 以較低成本擁有耐久的儲存空間。雖然物件存放在 S3 Glacier 中，他們會保有您在 Amazon S3 中管理的 Amazon S3 物件，且您無法透過 S3 Glacier 直接存取。

如需有關 Amazon S3 生命週期設定以及將物件轉換到 S3 Glacier 儲存類別的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件生命週期管理和轉換物件。

在單一操作中上傳封存

如 在 Amazon S3 Glacier 中上傳封存 中所述，您可以在單一操作中上傳較小的封存。但是，我們還是鼓勵 Amazon S3 Glacier (S3 Glacier) 客戶使用分段上傳，來上傳大於 100 MB 的封存。

主題

- 使用 AWS Command Line Interface 在單一操作中上傳封存

- 使用 AWS SDK for Java 在單一操作中上傳封存
- 在 Amazon S3 Glacier 中使用 AWS SDK for .NET 在單一操作中上傳封存
- 使用 REST API 在單一操作中上傳封存

使用 AWS Command Line Interface 在單一操作中上傳封存

您可以使用 AWS Command Line Interface (AWS CLI)，在 Amazon S3 Glacier (S3 Glacier) 中上傳封存。

主題

- (必要條件) 設定 AWS CLI
- 範例：使用 AWS CLI 上傳封存

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 上傳封存

若要上傳封存，您必須建立文件庫。如需有關建立文件庫的詳細資訊，請參閱在 Amazon S3 Glacier 中建立文件庫。

1. 使用 upload-archive 命令將封存新增至現有文件庫。在下面的範例中替換 vault name 和 account ID。對於 body 參數，指定您要上傳之檔案的路徑。

```
aws glacier upload-archive --vault-name awsexamplevault --account-id 123456789012
 --body archive.zip
```

2. 預期的輸出結果：

```
{
    "archiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
    "checksum": "969fb39823836d81f0cc028195fcdbcbbe76cdde932d4646fa7de5f21e18aa67",
    "location": "/123456789012/vaults/awsexamplevault/archives/
kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw"
}
```

完成後，此命令將輸出 S3 Glacier 中的封存 ID、檢查總和與位置。如需有關 upload-archive 命令的詳細資訊，請參閱《AWS CLI 命令參考》中的 upload-archive。

使用 AWS SDK for Java 在單一操作中上傳封存

適用於 Java 的 Amazon 開發套件提供的高階和低階 API 都提供了上傳封存的方法。

主題

- 使用 AWS SDK for Java 的高階 API 上傳封存
- 使用 AWS SDK for Java 的低階 API 在單一操作中上傳封存

使用 AWS SDK for Java 的高階 API 上傳封存

高階 API 的 ArchiveTransferManager 類別提供 upload 方法，可以使用該方法將存檔上傳到文件庫。

> **Note**
>
> 您可以使用 upload 方法上傳小型或大型封存。根據您要上傳的封存大小，此方法會判斷在單一操作上傳，或使用分段上傳 API 將封存分成部分上傳。

範例：使用 AWS SDK for Java 的高階 API 上傳封存

以下 Java 程式碼範例將封存上傳到美國西部 (奧勒岡) 區域 (us-west-2) 中的文件庫 (examplevault)。如需支援 AWS 區域和端點的清單，請參閱存取 Amazon S3 Glacier。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要更新程式碼，如所示的要上傳的文件庫名稱和要上傳的檔案名稱。

Example

```
import java.io.File;
import java.io.IOException;
import java.util.Date;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.glacier.transfer.UploadResult;


public class ArchiveUploadHighLevel {
    public static String vaultName = "*** provide vault name ***";
    public static String archiveToUpload = "*** provide name of file to upload ***";

    public static AmazonGlacierClient client;

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


     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(client,
 credentials);
```

```
            UploadResult result = atm.upload(vaultName, "my archive " + (new Date()),
 new File(archiveToUpload));
            System.out.println("Archive ID: " + result.getArchiveId());

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

使用 AWS SDK for Java 的低階 API 在單一操作中上傳封存

低階 API 提供所有封存操作的方法。以下是使用 AWS SDK for Java 上傳封存的步驟。

1. 建立 AmazonGlacierClient 類別的執行個體 (用戶端)。

   您需要指定要上傳封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 UploadArchiveRequest 類別的執行個體來提供請求資訊。

   除了要上傳的資料外，還需要提供承載的檢查總和 (SHA-256 樹狀雜湊)、文件庫名稱、資料的內容長度和帳戶 ID。

   如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用]。
3. 以參數形式提供請求物件，以便執行 uploadArchive 方法。

   Amazon S3 Glacier (S3 Glacier) 會在回應中，傳回剛上傳封存的封存 ID。

下列 Java 程式碼片段描述前述步驟。

```
AmazonGlacierClient client;

UploadArchiveRequest request = new UploadArchiveRequest()
    .withVaultName("*** provide vault name ***")
    .withChecksum(checksum)
    .withBody(new ByteArrayInputStream(body))
    .withContentLength((long)body.length);

UploadArchiveResult uploadArchiveResult = client.uploadArchive(request);
```

```
System.out.println("Location (includes ArchiveID): " +
 uploadArchiveResult.getLocation());
```

範例：使用 AWS SDK for Java 的低階 API 在單一操作中上傳封存

以下 Java 程式碼範例使用 AWS SDK for Java 將存檔上傳到文件庫 (examplevault)。如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要更新程式碼，如所示的要上傳的文件庫名稱和要上傳的檔案名稱。

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.UploadArchiveRequest;
import com.amazonaws.services.glacier.model.UploadArchiveResult;
public class ArchiveUploadLowLevel {

    public static String vaultName = "*** provide vault name ****";
    public static String archiveFilePath = "*** provide to file upload ****";
    public static AmazonGlacierClient client;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {
            // First open file and read.
            File file = new File(archiveFilePath);
            InputStream is = new FileInputStream(file);
            byte[] body = new byte[(int) file.length()];
            is.read(body);

            // Send request.
            UploadArchiveRequest request = new UploadArchiveRequest()
```

```
                .withVaultName(vaultName)
                .withChecksum(TreeHashGenerator.calculateTreeHash(new
 File(archiveFilePath)))
                .withBody(new ByteArrayInputStream(body))
                .withContentLength((long)body.length);

            UploadArchiveResult uploadArchiveResult = client.uploadArchive(request);

            System.out.println("ArchiveID: " + uploadArchiveResult.getArchiveId());

        } catch (Exception e)
        {
            System.err.println("Archive not uploaded.");
            System.err.println(e);
        }
    }
}
```

在 Amazon S3 Glacier 中使用 AWS SDK for .NET 在單一操作中上傳封存

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了在單一操作中上傳封存的方法。

主題

- 使用 AWS SDK for .NET 的高階 API 上傳封存
- 使用 AWS SDK for .NET 的低階 API 在單一操作中上傳封存

使用 AWS SDK for .NET 的高階 API 上傳封存

高階 API 的 `ArchiveTransferManager` 類別提供 `Upload` 方法，您可以使用該方法將存檔上傳到文件庫。

> **Note**
>
> 您可以使用 `Upload` 方法上傳小型或大型檔案。根據您要上傳的檔案大小，此方法會判斷在單一操作上傳，或使用分段上傳 API 以將檔案分段上傳。

範例：使用 AWS SDK for .NET 的高階 API 上傳封存

以下 C# 程式碼範例將封存上傳到美國西部 (奧勒岡) 區域中的文件庫 (`examplevault`)。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您必須如所示，使用要上傳的檔案名稱更新程式碼。

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadHighLevel
  {
    static string vaultName = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";

    public static void Main(string[] args)
    {
       try
      {
         var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
         // Upload an archive.
         string archiveId = manager.Upload(vaultName, "upload archive test",
 archiveToUpload).ArchiveId;
         Console.WriteLine("Archive ID: (Copy and save this ID for use in other
 examples.) : {0}", archiveId);
         Console.WriteLine("To continue, press Enter");
         Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

使用 AWS SDK for .NET 的低階 API 在單一操作中上傳封存

低階 API 提供所有封存操作的方法。以下是使用 AWS SDK for .NET 上傳封存的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要上傳封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `UploadArchiveRequest` 類別的執行個體來提供請求資訊。

   除了要上傳的資料外，還需要提供承載的檢查總和 (SHA-256 樹狀雜湊)、文件庫名稱和帳戶 ID。

   如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。
3. 以參數形式提供請求物件，以便執行 `UploadArchive` 方法。

   S3 Glacier 會在回應中傳回剛上傳封存的封存 ID。

範例：使用 AWS SDK for .NET 的低階 API 在單一操作中上傳封存

下列 C# 程式碼範例描述前述步驟。此範例使用 AWS SDK for .NET 將封存上傳到文件庫 (`examplevault`)。

> **Note**
>
> 如需有關底層 REST API 在單一請求中上傳封存的詳細資訊，請參閱 上傳封存 (POST 封存)。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您必須如所示，使用要上傳的檔案名稱更新程式碼。

Example

```
using System;
using System.IO;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadSingleOpLowLevel
  {
    static string vaultName       = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";
```

```
    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Uploading an archive.");
          string archiveId = UploadAnArchive(client);
          Console.WriteLine("Archive ID: {0}", archiveId);
        }
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static string UploadAnArchive(AmazonGlacierClient client)
    {
      using (FileStream fileStream = new FileStream(archiveToUpload, FileMode.Open,
 FileAccess.Read))
      {
        string treeHash = TreeHashGenerator.CalculateTreeHash(fileStream);
        UploadArchiveRequest request = new UploadArchiveRequest()
        {
          VaultName = vaultName,
          Body = fileStream,
          Checksum = treeHash
        };
        UploadArchiveResponse response = client.UploadArchive(request);
        string archiveID = response.ArchiveId;
        return archiveID;
      }
    }
  }
}
```

使用 REST API 在單一操作中上傳封存

您可以使用上傳封存 API 呼叫以在單一操作中上傳封存。如需更多詳細資訊，請參閱 上傳封存 (POST 封存)。

上傳分段中的大型封存 (分段上傳)

主題

- 分段上傳程序
- 現況
- 使用 AWS CLI 上傳大型封存
- 範例：使用適用於 Java 的 Amazon 開發套件，以部分形式上傳大型封存
- 使用 AWS SDK for .NET 上傳大型封存
- 使用 REST API 上傳分段中的大型封存

分段上傳程序

如 在 Amazon S3 Glacier 中上傳封存 中所述，我們鼓勵 Amazon S3 Glacier (S3 Glacier) 客戶使用分段上傳，來上傳大於 100 MiB 的封存。

1. 啟動分段上傳

   當您傳送要求以起始分段上傳時，S3 Glacier 會傳回分段上傳 ID，其為分段上傳的唯一識別碼。任何後續分段上傳操作中都需要此 ID。S3 Glacier 完成任務後，此 ID 至少在 24 小時內不會過期。

   當您要求啟動分段上傳時，請指定分段大小 (以位元組為單位)。您上傳的每個分段，除了最後一個分段，都必須為這個大小。

   > **Note**
   >
   > 您不需要了解使用分段上傳時的整體存檔大小。這表示您在開始上傳存檔時，可以在不知道存檔大小的情況下，使用分段上傳。您只需要在啟動分段上傳時，決定分段大小。

   在起始分段上傳請求時，您也可以提供選用的封存描述。

2. 分段上傳

對於每個分段上傳請求，您必須包含在步驟 1 取得的分段上傳 ID。在請求中，您還必須指定內容範圍 (以位元組為單位)，識別分段在最終封存中的位置。S3 Glacier 稍後會使用內容範圍資訊，以適當的順序組合封存。由於您提供內容範圍給上傳的每個分段，它會在封存的最終組件中決定分段的位置，因此您可以任何順序上傳分段。您也可以平行上傳這些分段。如果您使用和之前上傳分段相同的內容範圍上傳新的分段，將會覆寫之前上傳的分段。

3. 完成 (或停止) 分段上傳

   上傳所有封存分段之後，您可以完整的操作。同樣地，您必須在請求中指定上傳 ID。S3 Glacier 會根據您提供的內容範圍，透過以遞增順序串連部分來建立封存。S3 Glacier 對完成分段上傳請求的回應包括新建立封存的封存 ID。如果您在起始分段上傳請求中提供選填的封存描述，S3 Glacier 會將其與組合的封存建立關聯。在您成功完成分段上傳後，您無法參照分段上傳 ID。這表示您無法存取與該分段上傳 ID 關聯的分段。

   如果停止分段上傳，您即無法使用該分段上傳 ID 上傳更多的分段。與已停止分段上傳關聯之任何部分耗用的所有儲存體都會釋出。如有任何部分上傳正在進行，則會在停止後仍會成功或失敗。

其他分段上傳操作

Amazon S3 Glacier (S3 Glacier) 提供下列額外的分段上傳 API 呼叫。

- 列出部分：使用此操作時，您可以列出特定分段上傳的部分。它會傳回有關已針對分段上傳上傳之分段的資訊。對於每個列出部分的請求，S3 Glacier 會傳回最多 1,000 個部分的資訊。如果有更多分段要針對分段上傳列出，結果會分頁並在繼續列出的回應中傳回標記。您需要傳送額外的請求，以擷取後續分段。請注意，傳回的組件清單不包含尚未完成之上傳的組件。
- 列出分段上傳：使用此操作，您即可取得進行中之分段上傳的清單。進行中的分段上傳是您已啟動但尚未完成或已停止的上傳。對於每個列出分段上傳請求，S3 Glacier 會傳回最多 1,000 個分段上傳。如果有更多分段要列出，則結果會分頁並在繼續列出的回應中傳回標記。您需要傳送額外的請求，以擷取剩餘的分段上傳。

現況

下表提供分段上傳核心規格。

項目	規格
最大封存大小	10,000 x 4 gibibytes (GiB)

項目	規格
每次上傳的組件數目上限	10,000
組件大小	1 MiB 至 4 GiB，最後一個部分可以是 < 1 MiB。您可以位元組指定大小值。 部分大小必須是 1 MiB (1024 kibibytes [KiB]) 乘以 2 的乘方。例如，1048576 (1 MiB)、2097152 (2 MiB)、4194304 (4 MiB)、8388608 (8 MiB)。
列出組件要求的傳回組件數上限	1,000
列出分段上傳要求所傳回的分段上傳數目上限	1,000

使用 AWS CLI 上傳大型封存

您可以使用 AWS Command Line Interface (AWS CLI)，在 Amazon S3 Glacier (S3 Glacier) 中上傳封存。為了改善大型封存的上傳體驗，S3 Glacier 提供了數個 API 操作來支援分段上傳。透過使用這些 API 操作，您能夠以部分形式上傳封存。這些封存部分可個別、依任何順序以及同時上傳。如果某個部分上傳失敗，您需要再次上傳該部分，而不是整個封存。您可以為大小介於 1 位元組到 40,000 GiB 之間的封存使用分段上傳。

如需 S3 Glacier 分段上傳的詳細資訊，請參閱上傳分段中的大型封存 (分段上傳)。

主題

- (必要條件) 設定 AWS CLI
- (先決條件) 安裝 Python
- (先決條件) 建立 S3 Glacier 文件庫
- 範例：使用 AWS CLI 以部分形式上傳大型封存

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

(先決條件) 安裝 Python

若要完成分段上傳，您必須計算要上傳之封存的 SHA256 樹雜湊。這樣做與計算要上傳之檔案的 SHA256 樹雜湊不同。若要計算您要上傳之封存的 SHA256 樹雜湊，您可以使用 Java、C# (使用 .NET) 或 Python。在此範例中，您將使用 Python。如需使用 Java 或 C# 的指示，請參閱運算檢查總和。

如需安裝 Python 的詳細資訊，請參閱《Boto3 開發人員指南》中的安裝或更新 Python。

(先決條件) 建立 S3 Glacier 文件庫

若要使用下列範例，您必須建立至少一個 S3 Glacier 文件庫。如需有關建立文件庫的詳細資訊，請參閱在 Amazon S3 Glacier 中建立文件庫。

範例：使用 AWS CLI 以部分形式上傳大型封存

在此範例中，您將建立檔案並使用分段上傳 API 操作，將此檔案以部分的形式上傳到 Amazon S3 Glacier。

> **Important**
>
> 開始此程序之前，請確認您已執行所有必要步驟。若要上傳封存，您必須建立文件庫、設定 AWS CLI 並準備好使用 Java、C# 或 Python 來計算 SHA256 樹雜湊。

下列程序使用 `initiate-multipart-upload`、`upload-multipart-part`、和 `complete-multipart-upload` AWS CLI 命令。

如需有關這些命令之一的詳細資訊，請參閱《AWS CLI 命令參考》中 initiate-multipart-upload、upload-multipart-part 和 complete-multipart-upload。

1. 使用 initiate-multipart-upload 命令以建立分段上傳資源。請指定在請求中部分大小 (以位元組為單位)。您上傳的每個部分，除了最後一個部分，都將是這個大小。您不需要了解啟動上傳時的整體封存大小。但是，在完成最後一步的上傳時，您將需要每個部分的總大小 (以位元組為單位)。

   在下列命令中，將 `--vault-name` 和 `--account-ID` 參數的值替換為您自己的資訊。此命令指定您將上傳每個檔案部分大小為 1 MiB (1024 x 1024 位元組) 的封存。如有需要，請替換此 `--part-size` 參數值。

   ```
   aws glacier initiate-multipart-upload --vault-name awsexamplevault --part-
   size 1048576 --account-id 123456789012
   ```

   預期的輸出結果：

   ```
   {
   "location": "/123456789012/vaults/awsexamplevault/multipart-uploads/uploadId",
   "uploadId": "uploadId"
   }
   ```

   完成後，此命令會在 S3 Glacier 中輸出分段上傳資源的上傳 ID 和位置。於後續步驟中，您將會使用此上傳 ID。

2. 在此範例中，您可以使用下列命令建立 4.4 MiB 檔案，將其分割成 1 MiB 區塊，然後上傳每個區塊。若要上傳自己的檔案，您可以按照類似的程序，將資料分區並上傳每個部分。

   Linux 或 macOS

   下列命令會在 Linux 或 macOS 上建立名為 `file_to_upload` 的 4.4 MiB 檔案。

```
mkfile -n 9000b file_to_upload
```

Windows

下列命令會在 Windows 上建立名為 file_to_upload 的 4.4 MiB 檔案。

```
fsutil file createnew file_to_upload 4608000
```

3. 接下來，您將這個檔案分割成 1 MiB 區塊。

```
split -b 1048576 file_to_upload chunk
```

您現在擁有以下五個區塊。前四個是 1 MiB，最後一個大約是 400 KiB。

```
chunkaa
chunkab
chunkac
chunkad
chunkae
```

4. 使用 upload-multipart-part 命令以上傳部分封存。您可以依任何順序上傳封存部分。您也可以平行上傳這些部分。您可以上傳多達 10,000 個部分的分段上傳。

在下列命令中，替換 --vault-name、--account-ID 和 --upload-id 參數的值。上傳 ID 必須與 initiate-multipart-upload 命令輸出的 ID 相符。此 --range 參數指定您將上傳大小為 1 MiB (1024 x 1024 位元組) 的部分。此大小必須符合您在 initiate-multipart-upload 命令中指定的大小。如有需要，請調整此大小值。此 --body 參數會指定您要上傳之部分的名稱。

```
aws glacier upload-multipart-part --body chunkaa --range='bytes 0-1048575/*' --
vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

如果成功，該命令將產生輸出，其中內含上傳部分的檢查總和。

5. 再次執行 upload-multipart-part 命令，以上傳分段上傳的剩餘部分。更新每個命令的 --range 和 --body 參數值，以符合您要上傳的部分。

```
aws glacier upload-multipart-part --body chunkab --range='bytes 1048576-2097151/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkac --range='bytes 2097152-3145727/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkad --range='bytes 3145728-4194303/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

```
aws glacier upload-multipart-part --body chunkae --range='bytes 4194304-4607999/*'
 --vault-name awsexamplevault --account-id 123456789012 --upload-id upload_ID
```

> **Note**
>
> 最終命令的 `--range` 參數值較小，因為上傳的最後一部分小於 1 MiB。如果成功，每個命令都會產生輸出，內含每個上傳部分的檢查總和。

6. 接下來，您將組合封存並完成上傳。您必須包含封存的總大小和 SHA256 樹雜湊。

   若要計算封存的 SHA256 樹雜湊，您可以使用 Java、C# 或 Python。在此範例中，您將使用 Python。如需使用 Java 或 C# 的指示，請參閱運算檢查總和。

   建立 Python 檔案 `checksum.py` 並插入下列程式碼。如果需要，請替換原始檔案的名稱。

   ```
   from botocore.utils import calculate_tree_hash

   checksum = calculate_tree_hash(open('file_to_upload', 'rb'))
   print(checksum)
   ```

7. 執行 `checksum.py` 以計算 SHA256 樹雜湊。以下雜湊可能與輸出不相符。

   ```
   $ python3 checksum.py
   $ 3d760edb291bfc9d90d35809243de092aea4c47b308290ad12d084f69988ae0c
   ```

8. 使用 complete-multipart-upload 命令完成封存上傳。替換 `--vault-name`、`--account-ID`、`--upload-ID` 和 `--checksum` 參數的值。`--archive` 參數值指定封存的總大小 (以位元組為單位)。這個值必須是您上傳的個別部分之所有大小的總和。如有需要，請替換此值。

   ```
   aws glacier complete-multipart-upload --archive-size 4608000 --vault-
   name awsexamplevault --account-id 123456789012 --upload-id upload_ID --
   checksum checksum
   ```

完成後，此命令會在 S3 Glacier 中輸出封存的 ID、檢查總和與位置。

範例：使用適用於 Java 的 Amazon 開發套件，以部分形式上傳大型封存

適用於 Java 的 Amazon 開發套件提供的高階和低階 API 都提供了上傳大型封存的方法 (請參閱 在 Amazon S3 Glacier 中上傳封存)。

- 高階的 API 提供了一種可用來上傳任何大小的封存的方法。根據您要上傳的檔案，該方法可以透過單一操作上傳封存，或者使用 Amazon S3 Glacier (S3 Glacier) 中的分段上傳支援以部分形式上傳封存。
- 低階 API 對應接近底層 REST 實作。因此，它提供一個方法，在一個操作中上傳較小的封存，以及一組方法，可支援分段上傳以上傳較大封存。本節說明使用低階 API 以部分形式上傳大型封存。

如需高階和低階的 API 的更多資訊，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用。

主題

- 使用 AWS SDK for Java 的高階 API 以部分形式上傳大型封存
- 使用 AWS SDK for Java 的低階 API 以部分形式上傳大型封存

使用 AWS SDK for Java 的高階 API 以部分形式上傳大型封存

您可以使用高階 API 的相同方法來上傳小型或大型封存。高階 API 方法會根據封存大小，決定透過在單一操作，還是使用由 S3 Glacier 提供的分段上傳 API 來上傳封存。如需更多詳細資訊，請參閱 使用 AWS SDK for Java 的高階 API 上傳封存。

使用 AWS SDK for Java 的低階 API 以部分形式上傳大型封存

對於精細控制上傳，您可以使用低階 API，您可以設定請求和處理回應。以下是使用 AWS SDK for Java 以部分形式上傳大型封存的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要儲存封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 呼叫 `initiateMultipartUpload` 方法以啟動分段上傳。

您需要提供要上傳存檔的文件庫名稱、要用於上傳存檔部分的部分大小以及可選說明。您可以透過建立 `InitiateMultipartUploadRequest` 類別的執行個體，來提供這項資訊。S3 Glacier 會在回應中傳回上傳 ID。

3. 透過呼叫 `uploadMultipartPart` 方法上傳部分。

   對於您上載的每個部分，您需要提供文件庫名稱、在該部分中上傳的最終組合的存檔中的位元組範圍、部分資料的檢查總和和上傳 ID。

4. 呼叫 `completeMultipartUpload` 方法以計算分段上傳。

   您需要提供上傳 ID、整個封存的檢查總和、封存大小 (您上傳的所有部分的組合大小) 和文件庫名稱。S3 Glacier 從上傳部分建構封存並傳回封存 ID。

範例：使用 AWS SDK for Java 以部分形式上傳大型封存

以下 Java 程式碼範例使用 AWS SDK for Java 將存檔上傳到文件庫 (`examplevault`)。如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您必須如所示，使用要上傳的檔案名稱更新程式碼。

> **Note**
>
> 此範例對從 1 MB 到 1 GB 的部分大小有效。不過，S3 Glacier 支援最多 4 GB 的部分大小。

Example

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.NoSuchAlgorithmException;
import java.util.Arrays;
import java.util.Date;
import java.util.LinkedList;
import java.util.List;

import com.amazonaws.AmazonClientException;
import com.amazonaws.AmazonServiceException;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
```

```
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.CompleteMultipartUploadRequest;
import com.amazonaws.services.glacier.model.CompleteMultipartUploadResult;
import com.amazonaws.services.glacier.model.InitiateMultipartUploadRequest;
import com.amazonaws.services.glacier.model.InitiateMultipartUploadResult;
import com.amazonaws.services.glacier.model.UploadMultipartPartRequest;
import com.amazonaws.services.glacier.model.UploadMultipartPartResult;
import com.amazonaws.util.BinaryUtils;

public class ArchiveMPU {

    public static String vaultName = "examplevault";
    // This example works for part sizes up to 1 GB.
    public static String partSize = "1048576"; // 1 MB.
    public static String archiveFilePath = "*** provide archive file path ***";
    public static AmazonGlacierClient client;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

        try {
            System.out.println("Uploading an archive.");
            String uploadId = initiateMultipartUpload();
            String checksum = uploadParts(uploadId);
            String archiveId = CompleteMultiPartUpload(uploadId, checksum);
            System.out.println("Completed an archive. ArchiveId: " + archiveId);

        } catch (Exception e) {
            System.err.println(e);
        }

    }

    private static String initiateMultipartUpload() {
        // Initiate
        InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest()
            .withVaultName(vaultName)
            .withArchiveDescription("my archive " + (new Date()))
            .withPartSize(partSize);
```

```
        InitiateMultipartUploadResult result = client.initiateMultipartUpload(request);

        System.out.println("ArchiveID: " + result.getUploadId());
        return result.getUploadId();
    }

    private static String uploadParts(String uploadId) throws AmazonServiceException,
NoSuchAlgorithmException, AmazonClientException, IOException {

        int filePosition = 0;
        long currentPosition = 0;
        byte[] buffer = new byte[Integer.valueOf(partSize)];
        List<byte[]> binaryChecksums = new LinkedList<byte[]>();

        File file = new File(archiveFilePath);
        FileInputStream fileToUpload = new FileInputStream(file);
        String contentRange;
        int read = 0;
        while (currentPosition < file.length())
        {
            read = fileToUpload.read(buffer, filePosition, buffer.length);
            if (read == -1) { break; }
            byte[] bytesRead = Arrays.copyOf(buffer, read);

            contentRange = String.format("bytes %s-%s/*", currentPosition,
currentPosition + read - 1);
            String checksum = TreeHashGenerator.calculateTreeHash(new
ByteArrayInputStream(bytesRead));
            byte[] binaryChecksum = BinaryUtils.fromHex(checksum);
            binaryChecksums.add(binaryChecksum);
            System.out.println(contentRange);

            //Upload part.
            UploadMultipartPartRequest partRequest = new UploadMultipartPartRequest()
            .withVaultName(vaultName)
            .withBody(new ByteArrayInputStream(bytesRead))
            .withChecksum(checksum)
            .withRange(contentRange)
            .withUploadId(uploadId);

            UploadMultipartPartResult partResult =
client.uploadMultipartPart(partRequest);
            System.out.println("Part uploaded, checksum: " + partResult.getChecksum());
```

```
            currentPosition = currentPosition + read;
        }
        fileToUpload.close();
        String checksum = TreeHashGenerator.calculateTreeHash(binaryChecksums);
        return checksum;
    }

    private static String CompleteMultiPartUpload(String uploadId, String checksum)
 throws NoSuchAlgorithmException, IOException {

        File file = new File(archiveFilePath);

        CompleteMultipartUploadRequest compRequest = new
 CompleteMultipartUploadRequest()
            .withVaultName(vaultName)
            .withUploadId(uploadId)
            .withChecksum(checksum)
            .withArchiveSize(String.valueOf(file.length()));

        CompleteMultipartUploadResult compResult =
 client.completeMultipartUpload(compRequest);
        return compResult.getLocation();
    }
}
```

使用 AWS SDK for .NET 上傳大型封存

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了以部分形式上傳大型封存的方法 (請參閱在 Amazon S3 Glacier 中上傳封存)。

- 高階的 API 提供了一種可用來上傳任何大小的封存的方法。根據您要上傳的檔案，該方法可以透過單一操作上傳封存，或者使用 Amazon S3 Glacier (S3 Glacier) 中的分段上傳支援以部分形式上傳封存。
- 低階 API 對應接近底層 REST 實作。因此，它提供一個方法，在一個操作中上傳較小的封存，以及一組方法，可支援分段上傳以上傳較大封存。本節說明使用低階 API 以部分形式上傳大型封存。

如需高階和低階的 API 的更多資訊，請參閱 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。

主題

- 使用 AWS SDK for .NET 的高階 API 以部分形式上傳大型封存

- 使用 AWS SDK for .NET 的低階 API 以部分形式上傳大型封存

使用 AWS SDK for .NET 的高階 API 以部分形式上傳大型封存

您可以使用高階 API 的相同方法來上傳小型或大型封存。高階 API 方法會根據封存大小，決定透過在單一操作，還是使用由 S3 Glacier 提供的分段上傳 API 來上傳封存。如需更多詳細資訊，請參閱 使用 AWS SDK for .NET 的高階 API 上傳封存。

使用 AWS SDK for .NET 的低階 API 以部分形式上傳大型封存

對於精細控制上傳，您可以使用低階 API，您可以設定請求和處理回應。以下是使用 AWS SDK for .NET 以部分形式上傳大型封存的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要儲存封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 呼叫 `InitiateMultipartUpload` 方法以啟動分段上傳。

   您需要提供要上傳封存的文件庫名稱、要用於上傳封存部分的部分大小以及可選說明。您可以透過建立 `InitiateMultipartUploadRequest` 類別的執行個體，來提供這項資訊。S3 Glacier 會在回應中傳回上傳 ID。
3. 透過呼叫 `UploadMultipartPart` 方法上傳部分。

   對於您上載的每個部分，您需要提供文件庫名稱、在該部分中上傳的最終組合的存檔中的位元組範圍、部分資料的檢查總和和上傳 ID。
4. 呼叫 `CompleteMultipartUpload` 方法以計算分段上傳。

   您需要提供上傳 ID、整個封存的檢查總和、封存大小 (您上傳的所有部分的組合大小) 和文件庫名稱。S3 Glacier 從上傳部分建構封存並傳回封存 ID。

範例：使用適用於 .NET 的 Amazon 開發套件，以部分形式上傳大型封存

以下 C# 程式碼範例使用 AWS SDK for .NET 將封存上傳到文件庫 (`examplevault`)。如需執行此範例的逐步說明，請參閱 執行程式碼範例。您必須如所示，使用要上傳的檔案名稱更新程式碼。

Example

```
using System;
using System.Collections.Generic;
```

```
using System.IO;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveUploadMPU
  {
    static string vaultName       = "examplevault";
    static string archiveToUpload = "*** Provide file name (with full path) to upload
 ***";
    static long partSize          = 4194304; // 4 MB.

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      List<string> partChecksumList = new List<string>();
      try
      {
         using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Uploading an archive.");
          string uploadId  = InitiateMultipartUpload(client);
          partChecksumList = UploadParts(uploadId, client);
          string archiveId = CompleteMPU(uploadId, client, partChecksumList);
          Console.WriteLine("Archive ID: {0}", archiveId);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static string InitiateMultipartUpload(AmazonGlacierClient client)
    {
      InitiateMultipartUploadRequest initiateMPUrequest = new
 InitiateMultipartUploadRequest()
      {
```

```
        VaultName = vaultName,
        PartSize = partSize,
        ArchiveDescription = "Test doc uploaded using MPU."
      };

      InitiateMultipartUploadResponse initiateMPUresponse =
client.InitiateMultipartUpload(initiateMPUrequest);

      return initiateMPUresponse.UploadId;
    }

    static List<string> UploadParts(string uploadID, AmazonGlacierClient client)
    {
      List<string> partChecksumList = new List<string>();
      long currentPosition = 0;
      var buffer = new byte[Convert.ToInt32(partSize)];

      long fileLength = new FileInfo(archiveToUpload).Length;
      using (FileStream fileToUpload = new FileStream(archiveToUpload, FileMode.Open,
FileAccess.Read))
      {
        while (fileToUpload.Position < fileLength)
        {
          Stream uploadPartStream = GlacierUtils.CreatePartStream(fileToUpload,
partSize);
          string checksum = TreeHashGenerator.CalculateTreeHash(uploadPartStream);
          partChecksumList.Add(checksum);
          // Upload part.
          UploadMultipartPartRequest uploadMPUrequest = new
UploadMultipartPartRequest()
          {

            VaultName = vaultName,
            Body = uploadPartStream,
            Checksum = checksum,
            UploadId = uploadID
          };
          uploadMPUrequest.SetRange(currentPosition, currentPosition +
uploadPartStream.Length - 1);
          client.UploadMultipartPart(uploadMPUrequest);

          currentPosition = currentPosition + uploadPartStream.Length;
        }
      }
```

```
      return partChecksumList;
    }

    static string CompleteMPU(string uploadID, AmazonGlacierClient client, List<string>
 partChecksumList)
    {
      long fileLength = new FileInfo(archiveToUpload).Length;
      CompleteMultipartUploadRequest completeMPUrequest = new
 CompleteMultipartUploadRequest()
      {
        UploadId = uploadID,
        ArchiveSize = fileLength.ToString(),
        Checksum = TreeHashGenerator.CalculateTreeHash(partChecksumList),
        VaultName = vaultName
      };

      CompleteMultipartUploadResponse completeMPUresponse =
 client.CompleteMultipartUpload(completeMPUrequest);
      return completeMPUresponse.ArchiveId;
    }
  }
}
```

使用 REST API 上傳分段中的大型封存

如上傳分段中的大型封存 (分段上傳)中所述，分段上傳是指一組操作，可讓您分段上傳封存，並執行相關的操作。如需這些操作的詳細資訊，請參閱下列 API 參考主題：

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 中止分段上傳 (DELETE uploadID)
- 清單部分 (GET uploadID)
- 列出分段上傳 (GET 分段 - 上傳)

在 S3 Glacier 中下載封存

Amazon S3 Glacier 提供的管理主控台，可用來建立和刪除文件庫。但是，您無法藉由使用管理主控台，從 S3 Glacier 下載封存。若要下載資料 (例如照片、影片和其他文件)，您必須使用 AWS

Command Line Interface (AWS CLI) 或撰寫程式碼來發出請求，方法是直接使用 REST API 或使用 AWS 開發套件。

如需使用 S3 Glacier 搭配 AWS CLI 的詳細資訊，請參閱 S3 Glacier 的 AWS CLI 參考。若要安裝 AWS CLI，請參閱 AWS Command Line Interface。下列主題說明如何使用 AWS SDK for Java、AWS SDK for .NET 和 Amazon S3 Glacier REST API，將封存下載到 S3 Glacier。

主題

使用 AWS 主控台擷取 S3 Glacier 封存

從 Amazon S3 Glacier 擷取封存是一種非同步操作，首先您會啟動任務，然後在任務完成後下載輸出。若要啟動封存擷取任務，請使用 啟動任務 (POST 任務) REST API 操作或 AWS CLI 或 AWS 開發套件中的等同操作。

主題

從 S3 Glacier 擷取封存是兩步驟程序。

擷取封存

1. 啟動封存擷取任務。

    a. 取得您想要擷取的封存 ID。您可以從文件庫的清查取得存檔 ID。您可以使用 REST API、AWS CLI 或 AWS 開發套件，取得封存 ID。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中下載文件庫庫存。

    b. 啟動請求 S3 Glacier 的任務，透過使用 啟動任務 (POST 任務) 操作，以便為後續的下載準備整個封存或部分封存。

當您啟動任務時，S3 Glacier 會在回應中傳回任務 ID，並以非同步的方式執行任務。(如步驟 2 所述，直到任務完成後，您才能下載任務輸出。)

> **Important**
>
> 僅適用於標準擷取的資料擷取政策，可能導致 `Initiate Job` 請求失敗，並出現 `PolicyEnforcedException` 例外狀況。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。如需 `PolicyEnforcedException` 例外狀況的詳細資訊，請參閱 錯誤回應。

必要時，在 S3 Glacier 存放資料的大型區段也可供您還原。如需從 S3 Glacier 儲存類別還原資料的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的封存物件的儲存類別。

2. 任務完成後，使用 「取得任務輸出」 (GET 輸出) 操作下載位元組。

   您可以下載所有位元組，或指定位元組範圍，以僅下載任務輸出的一部分。對於較大的輸出，下載區塊形式的輸出對於發生下載失敗的情況有所幫助，例如網路失敗。如果您在單一請求中取得任務輸出，並且發生網路失敗，此時您必須重頭開始重新下載輸出。不過，如果您以區塊形式下載輸出，當發生任何失敗情況時，您只需重新開始下載較小部分，而不是整個輸出。

S3 Glacier 必須先完成任務，您才能取得其輸出。完成任務之後至少 24 小時內，任務不會過期，意思是您可以在任務完成後 24 小時內下載輸出。若要判斷您的任務是否已完成後，您可使用以下其中一個選項來檢查其狀態：

- 等待任務完成通知：您可以指定在任務完成後，S3 Glacier 可以發佈通知的 Amazon Simple Notification Service (Amazon SNS) 主題。S3 Glacier 僅在完成任務後傳送通知。

  啟動任務時，您可指定任務的 Amazon SNS 主題。在任務請求中除了指定 Amazon SNS 主題，如果文件庫已針對封存擷取事件設定通知設定，則 S3 Glacier 也會將通知發布至該 SNS 主題。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。
- 明確地請求任務資訊：您也可使用 S3 Glacier `Describe Job` API 操作 (描述任務 (GET JobID))，以定期輪詢任務資訊。不過，我們建議使用 Amazon SNS 通知。

> **Note**
>
> 您透過使用 Amazon SNS 通知所取得的資訊，與您呼叫 `Describe Job` API 操作所取得的資訊相同。

封存擷取選項

啟動任務來擷取封存時，您可以根據存取時間和成本需求，指定以下其中一項擷取選項。如需擷取定價的相關資訊，請參閱 Amazon S3 Glacier 定價。

- 快速：快速擷取可讓您在偶爾需要緊急請求還原封存時，能快速存取在 S3 Glacier Flexible Retrieval 儲存體類別或 S3 Intelligent-Tiering Archive Access 層中存放的資料。用於規模幾乎最大的封存 (250 MB 以上) 時，使用快速擷取所存取的資料，通常在 1-5 分鐘內即可使用。佈建容量可確保快速擷取在需要時有可用的擷取容量。如需更多詳細資訊，請參閱 佈建的容量。
- 標準：標準擷取可讓您在幾小時內存取任何封存。標準擷取通常會於 3-5 小時內完成。未指定擷取選項時，擷取請求的預設選項會是「標準」。
- 大量：大量擷取是成本最低廉的 S3 Glacier 擷取選項，可用於在一天內擷取大量資料 (甚至達到數 PB)，而無需太多費用。大量擷取通常會於 5-12 小時內完成。

下表摘要說明封存擷取選項。如需定價的詳細資訊，請參閱 Amazon S3 Glacier 定價。

若要進行 `Expedited`、`Standard` 或 `Bulk` 擷取，請將 RestoreObject REST API 操作請求中的 `Tier` 請求元素設定為所需的選項，或是設定為 AWS Command Line Interface (AWS CLI) 或 AWS SDK 中的相等選項。若已購買佈建的容量，則所有快速擷取都會自動透過佈建的容量提供服務。

佈建的容量

佈建容量有助於確保快速擷取在需要時有可用的擷取容量。每個容量單位都提供每 5 分鐘至少可以執行三次「快速」擷取，並提供最多每秒 150 MB (MBps) 的擷取輸送量。

如果工作負載需要非常穩定且可預測的資料子集即時存取，我們建議您購買佈建的擷取容量。但即使沒有佈建的容量，通常仍可進行快速擷取，除非您要擷取的需求量不尋常地高，但此情況很罕見。但是若您需要無論情況如何，皆能存取快速擷取，則必須購買佈建的擷取容量。

購買佈建容量

您可以使用 S3 Glacier 主控台、購買佈建容量 (POST 佈建的容量) REST API 操作、AWS 開發套件或 AWS CLI，來購買已佈建的容量單位。如需佈建的容量定價資訊，請參閱 Amazon S3 Glacier 定價。

佈建容量單位會持續一個月，從購買的日期和時間開始。

如果開始日期是在某個月的第 31 天，過期日期則是下個月的最後一天。例如，如果開始日期是 8 月 31 日，過期日期則是 9 月 30 日。如果開始日期是 1 月 31 日，過期日期則是 2 月 28 日。

使用 Amazon S3 Glacier 主控台購買佈建的容量

1. 於 https://console.aws.amazon.com/glacier/home 登入 AWS Management Console 並開啟 S3 Glacier 主控台。
2. 在左側的導覽窗格中，選擇資料擷取設定。
3. 在佈建容量單位 (PCU) 下，選擇購買 PCU。這時會顯示購買 PCU 對話方塊。
4. 如果您想要購買佈建的容量，請在確認購買方塊中輸入 **`confirm`**。
5. 選擇購買 PCU。

遠端封存擷取

當您從 S3 Glacier 擷取封存時，您可以選擇性指定要擷取之封存的範圍或部分。預設值是擷取整個封存。指定位元組範圍，可在您執行下列動作時派上用場：

- 管理您的資料下載 ：S3 Glacier 允許在完成擷取請求後 24 小時下載擷取的資料。因此，您可能希望只擷取封存部分，讓您可以在指定下載時段內管理下載的排程。
- 擷取大型檔案的目標部分：例如，假設您之前已彙總多個檔案，並以單一封存形式將檔案上傳，而現在您想要擷取幾個檔案。在這種情況下，您可以使用一個擷取請求，指定包含您需要的檔案的封存範圍。或者，您可以啟動多個擷取請求，其中每一個都附帶有一或多個檔案的範圍。

使用範圍擷取啟動擷取任務時，您必須提供符合百萬位元組範圍。換言之，位元組範圍以零開始 (檔案的開頭)，或在之後以 1 MB 為間隔 (1 MB、2 MB、3 MB 等)。

範圍結尾可以是封存的結尾，或任何大於您的範圍開頭的 1 MB 間隔。此外，如果您要在下載資料後取得檢查總和值 (擷取任務完成後)，您在任務啟動請求的範圍，也必須符合樹雜湊。您可以使用檢查總和，來協助確保資料在傳輸期間未遭到損毀。如需有關符合百萬位元組與符合樹雜湊的詳細資訊，請參閱 下載資料時接收檢查總和。

使用 AWS SDK for Java 下載 Amazon S3 Glacier 中的封存

適用於 Java 的 Amazon 開發套件提供的高階和低階 API 都提供了下載封存的方法。

主題

- 使用 AWS SDK for Java 的高階 API 下載封存
- 使用 AWS SDK for Java 的低階 API 下載封存

使用 AWS SDK for Java 的高階 API 下載封存

高階 API 的 `ArchiveTransferManager` 類別提供可用來下載封存的 `download` 方法。

> **Important**
>
> 此 `ArchiveTransferManager` 類別會建立 Amazon Simple Notification Service (Amazon SNS) 主題，以及訂閱該主題的 Amazon Simple Queue Service (Amazon SQS) 佇列。然後啟動封存擷取任務並輪詢佇列以使封存可用。封存可用時，就會開始下載。如需封存擷取時間的詳細資訊，請參閱 封存擷取選項。

範例：使用 AWS SDK for Java 的高階 API 下載封存

以下 Java 程式碼範例從美國西部 (奧勒岡) 區域 (`us-west-2`) 中的文件庫 (`examplevault`) 下載封存。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要更新程式碼，如所示之在現有的封存 ID 以及欲儲存下載封存的本機檔案路徑 。

Example

```
import java.io.File;
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.transfer.ArchiveTransferManager;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sqs.AmazonSQSClient;


public class ArchiveDownloadHighLevel {
    public static String vaultName = "examplevault";
    public static String archiveId = "*** provide archive ID ***";
    public static String downloadFilePath  = "*** provide location to download archive
 ***";
```

```
    public static AmazonGlacierClient glacierClient;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        glacierClient = new AmazonGlacierClient(credentials);

        sqsClient = new AmazonSQSClient(credentials);
        snsClient = new AmazonSNSClient(credentials);
        glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
        sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
        snsClient.setEndpoint("sns.us-west-2.amazonaws.com");

        try {
            ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient,
 sqsClient, snsClient);

            atm.download(vaultName, archiveId, new File(downloadFilePath));
            System.out.println("Downloaded file to " + downloadFilePath);

        } catch (Exception e)
        {
            System.err.println(e);
        }
    }
}
```

使用 AWS SDK for Java 的低階 API 下載封存

以下是使用 AWS SDK for Java 低階 API 來擷取文件庫庫存的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要從中下載封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 執行 `archive-retrieval` 方法以啟動 `initiateJob` 任務。

透過建立 InitiateJobRequest 類別的執行個體來提供任務資訊，例如要下載之封存的封存 ID，以及希望 Amazon S3 Glacier (S3 Glacier) 發佈任務完成訊息的選用 Amazon SNS 主題。S3 Glacier 會在回應中傳回任務 ID。該回應在 InitiateJobResult 類別的執行個體中可用。

```
JobParameters jobParameters = new JobParameters()
    .withArchiveId("*** provide an archive id ***")
    .withDescription("archive retrieval")
    .withRetrievalByteRange("*** provide a retrieval range***") // optional
    .withType("archive-retrieval");

InitiateJobResult initiateJobResult = client.initiateJob(new InitiateJobRequest()
    .withJobParameters(jobParameters)
    .withVaultName(vaultName));

String jobId = initiateJobResult.getJobId();
```

您可以選擇指定位元組範圍來請求 S3 Glacier，以便僅準備封存的一部分。例如，您可以透過新增以下陳述式，來請求 S3 Glacier 僅準備封存的 1 MB 到 2 MB 部分，從而更新上述請求。

```
int ONE_MEG = 1048576;
String retrievalByteRange = String.format("%s-%s", ONE_MEG, 2*ONE_MEG -1);

JobParameters jobParameters = new JobParameters()
    .withType("archive-retrieval")
    .withArchiveId(archiveId)
    .withRetrievalByteRange(retrievalByteRange)
    .withSNSTopic(snsTopicARN);

InitiateJobResult initiateJobResult = client.initiateJob(new InitiateJobRequest()
    .withJobParameters(jobParameters)
    .withVaultName(vaultName));

String jobId = initiateJobResult.getJobId();
```

3. 等候任務完成。

您必須等到任務輸出準備好供您下載。如果您在文件庫上設定識別 Amazon Simple Notification Service (Amazon SNS) 主題的通知設定，或者在啟動任務時指定 Amazon SNS 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

您也可以透過呼叫 `describeJob` 方法來輪詢 S3 Glacier，以判斷任務完成狀態。雖然，使用 Amazon SNS 主題進行通知是建議的方法。

4. 透過執行 `getJobOutput` 方法下載任務輸出 (封存資料)。

   透過建立 `GetJobOutputRequest` 類別的執行個體來提供請求資訊，如任務 ID 和文件庫名稱。S3 Glacier 傳回的輸出在 `GetJobOutputResult` 物件中可用。

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withJobId("*** provide a job ID ***")
        .withVaultName("*** provide a vault name ****");
GetJobOutputResult jobOutputResult = client.getJobOutput(jobOutputRequest);

// jobOutputResult.getBody() // Provides the input stream.
```

   上述程式碼片段下載整個任務的輸出。您可以選擇只擷取輸出的一部分，或透過指定 `GetJobOutputRequest` 中的位元組範圍以較小的區塊下載整個輸出。

```
GetJobOutputRequest jobOutputRequest = new GetJobOutputRequest()
        .withJobId("*** provide a job ID ***")
        .withRange("bytes=0-1048575")   // Download only the first 1 MB of the
 output.
        .withVaultName("*** provide a vault name ****");
```

   為了回應 `GetJobOutput` 呼叫，如果滿足某些條件，S3 Glacier 將傳回下載資料部分的檢查總和。如需更多詳細資訊，請參閱 下載資料時接收檢查總和。

   為了驗證下載沒有錯誤，您可以計算用戶端的檢查總和，並將其與在回應中傳送的檢查總和 S3 Glacier 相比較。

   對於指定可選範圍的封存擷取任務，當您取得任務說明時，它將包含您正在擷取的範圍的檢查總和 (SHA256TreeHash)。您可以使用此值進一步驗證稍後下載的整個位元組範圍的準確性。例如，如果啟動任務以擷取樹狀雜湊符合的封存範圍，然後以區塊的方式下載輸出，以便每個 `GetJobOutput` 請求傳回檢查總和，則可以計算在用戶端上下載的每個部分的檢查總和，然後計算樹狀雜湊。您可

以將它與 S3 Glacier 傳回的檢查總和進行比較，以回應您的描述任務請求，以驗證您下載的整個位元組範圍與存放在 S3 Glacier 中的位元組範圍相同。

如需運作範例，請參閱 範例 2：使用區塊中的 AWS SDK for Java下載輸出的低階 API 擷取封存 。

範例 1：使用 AWS SDK for Java 的低階 API 擷取封存

以下 Java 程式碼範例從指定的文件庫下載封存。任務完成後，該範例將在單一 `getJobOutput` 呼叫中下載整個輸出。如需有關以區塊形式下載輸出的範例，請參閱 範例 2：使用區塊中的 AWS SDK for Java下載輸出的低階 API 擷取封存 。

範例會執行下列任務：

- 建立 Amazon Simple Notification Service (Amazon SNS) 主題。

  S3 Glacier 會在完成任務後向該主題傳送通知。
- 建立 Amazon Simple Queue Service (Amazon SQS) 佇列。

  此範例將政策連接至佇列，以使 Amazon SNS 主題能夠發佈訊息到佇列。
- 起始任務以下載指定的封存。

  在任務請求中，會指定所建立的 Amazon SNS 主題，因此，S3 Glacier 可以在完成任務後發布通知到主題。
- 定期檢查 Amazon SQS 佇列中是否有包含任務 ID 的訊息。

  如果有訊息，剖析 JSON 並檢查任務是否順利完成。如果是，請下載封存。
- 透過刪除 Amazon SNS 主題和其建立的 Amazon SQS 佇列來清除。

```
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileOutputStream;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
```

```
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.codehaus.jackson.JsonFactory;
import org.codehaus.jackson.JsonNode;
import org.codehaus.jackson.JsonParseException;
import org.codehaus.jackson.JsonParser;
import org.codehaus.jackson.map.ObjectMapper;

import com.amazonaws.AmazonClientException;
import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class AmazonGlacierDownloadArchiveWithSQSPolling {
```

```
    public static String archiveId = "*** provide archive ID ****";
    public static String vaultName = "*** provide vault name ***";
    public static String snsTopicName = "*** provide topic name ***";
    public static String sqsQueueName = "*** provide queue name ***";
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name ***";
    public static String region = "*** region ***";
    public static long sleepTime = 600;
    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");

        try {
            setupSQS();

            setupSNS();

            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            Boolean success = waitForJobToComplete(jobId, sqsQueueURL);
            if (!success) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId);

            cleanUp();

        } catch (Exception e) {
            System.err.println("Archive retrieval failed.");
```

```
            System.err.println(e);
        }
    }

    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);

        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {
```

```
        JobParameters jobParameters = new JobParameters()
            .withType("archive-retrieval")
            .withArchiveId(archiveId)
            .withSNSTopic(snsTopicARN);

        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }

    private static Boolean waitForJobToComplete(String jobId, String sqsQueueUrl)
throws InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getJsonFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
               new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").getTextValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").getTextValue();
                    String statusCode = jobDescNode.get("StatusCode").getTextValue();
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
```

```
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
          }
        return (messageFound && jobSuccessful);
    }

    private static void downloadJobOutput(String jobId) throws IOException {

        GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
            .withVaultName(vaultName)
            .withJobId(jobId);
        GetJobOutputResult getJobOutputResult =
client.getJobOutput(getJobOutputRequest);

        InputStream input = new BufferedInputStream(getJobOutputResult.getBody());
        OutputStream output = null;
        try {
            output = new BufferedOutputStream(new FileOutputStream(fileName));

            byte[] buffer = new byte[1024 * 1024];

            int bytesRead = 0;
            do {
                bytesRead = input.read(buffer);
                if (bytesRead <= 0) break;
                output.write(buffer, 0, bytesRead);
            } while (bytesRead > 0);
        } catch (IOException e) {
            throw new AmazonClientException("Unable to save archive", e);
        } finally {
            try {input.close();}  catch (Exception e) {}
            try {output.close();} catch (Exception e) {}
        }
        System.out.println("Retrieved archive to " + fileName);
    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
```

```
}
```

範例 2：使用區塊中的 AWS SDK for Java下載輸出的低階 API 擷取封存

下列 Java 程式碼範例會從 S3 Glacier 擷取封存。該程式碼範例透過在 `GetJobOutputRequest` 物件中指定位元組範圍來下載區塊中的任務輸出。

```
import java.io.BufferedInputStream;
import java.io.ByteArrayInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import com.fasterxml.jackson.core.JsonFactory;
import com.fasterxml.jackson.core.JsonParseException;
import com.fasterxml.jackson.core.JsonParser;
import com.fasterxml.jackson.databind.JsonNode;
import com.fasterxml.jackson.databind.ObjectMapper;

import com.amazonaws.auth.policy.Policy;
import com.amazonaws.auth.policy.Principal;
import com.amazonaws.auth.policy.Resource;
import com.amazonaws.auth.policy.Statement;
import com.amazonaws.auth.policy.Statement.Effect;
import com.amazonaws.auth.policy.actions.SQSActions;
import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.TreeHashGenerator;
import com.amazonaws.services.glacier.model.GetJobOutputRequest;
import com.amazonaws.services.glacier.model.GetJobOutputResult;
import com.amazonaws.services.glacier.model.InitiateJobRequest;
import com.amazonaws.services.glacier.model.InitiateJobResult;
import com.amazonaws.services.glacier.model.JobParameters;
import com.amazonaws.services.sns.AmazonSNSClient;
import com.amazonaws.services.sns.model.CreateTopicRequest;
import com.amazonaws.services.sns.model.CreateTopicResult;
import com.amazonaws.services.sns.model.DeleteTopicRequest;
import com.amazonaws.services.sns.model.SubscribeRequest;
import com.amazonaws.services.sns.model.SubscribeResult;
import com.amazonaws.services.sns.model.UnsubscribeRequest;
import com.amazonaws.services.sqs.AmazonSQSClient;
```

```
import com.amazonaws.services.sqs.model.CreateQueueRequest;
import com.amazonaws.services.sqs.model.CreateQueueResult;
import com.amazonaws.services.sqs.model.DeleteQueueRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesRequest;
import com.amazonaws.services.sqs.model.GetQueueAttributesResult;
import com.amazonaws.services.sqs.model.Message;
import com.amazonaws.services.sqs.model.ReceiveMessageRequest;
import com.amazonaws.services.sqs.model.SetQueueAttributesRequest;


public class ArchiveDownloadLowLevelWithRange {

    public static String vaultName = "*** provide vault name ***";
    public static String archiveId = "*** provide archive id ***";
    public static String snsTopicName = "glacier-temp-sns-topic";
    public static String sqsQueueName = "glacier-temp-sqs-queue";
    public static long downloadChunkSize = 4194304; // 4 MB
    public static String sqsQueueARN;
    public static String sqsQueueURL;
    public static String snsTopicARN;
    public static String snsSubscriptionARN;
    public static String fileName = "*** provide file name to save archive to ***";
    public static String region   = "*** region ***";
    public static long sleepTime  = 600;

    public static AmazonGlacierClient client;
    public static AmazonSQSClient sqsClient;
    public static AmazonSNSClient snsClient;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier." + region + ".amazonaws.com");
        sqsClient = new AmazonSQSClient(credentials);
        sqsClient.setEndpoint("https://sqs." + region + ".amazonaws.com");
        snsClient = new AmazonSNSClient(credentials);
        snsClient.setEndpoint("https://sns." + region + ".amazonaws.com");

        try {
            setupSQS();

            setupSNS();
```

```
            String jobId = initiateJobRequest();
            System.out.println("Jobid = " + jobId);

            long archiveSizeInBytes = waitForJobToComplete(jobId, sqsQueueURL);
            if (archiveSizeInBytes==-1) { throw new Exception("Job did not complete
successfully."); }

            downloadJobOutput(jobId, archiveSizeInBytes);

            cleanUp();

        } catch (Exception e) {
            System.err.println("Archive retrieval failed.");
            System.err.println(e);
        }
    }

    private static void setupSQS() {
        CreateQueueRequest request = new CreateQueueRequest()
            .withQueueName(sqsQueueName);
        CreateQueueResult result = sqsClient.createQueue(request);
        sqsQueueURL = result.getQueueUrl();

        GetQueueAttributesRequest qRequest = new GetQueueAttributesRequest()
            .withQueueUrl(sqsQueueURL)
            .withAttributeNames("QueueArn");

        GetQueueAttributesResult qResult = sqsClient.getQueueAttributes(qRequest);
        sqsQueueARN = qResult.getAttributes().get("QueueArn");

        Policy sqsPolicy =
            new Policy().withStatements(
                    new Statement(Effect.Allow)
                    .withPrincipals(Principal.AllUsers)
                    .withActions(SQSActions.SendMessage)
                    .withResources(new Resource(sqsQueueARN)));
        Map<String, String> queueAttributes = new HashMap<String, String>();
        queueAttributes.put("Policy", sqsPolicy.toJson());
        sqsClient.setQueueAttributes(new SetQueueAttributesRequest(sqsQueueURL,
queueAttributes));

    }
    private static void setupSNS() {
```

```
        CreateTopicRequest request = new CreateTopicRequest()
            .withName(snsTopicName);
        CreateTopicResult result = snsClient.createTopic(request);
        snsTopicARN = result.getTopicArn();

        SubscribeRequest request2 = new SubscribeRequest()
            .withTopicArn(snsTopicARN)
            .withEndpoint(sqsQueueARN)
            .withProtocol("sqs");
        SubscribeResult result2 = snsClient.subscribe(request2);

        snsSubscriptionARN = result2.getSubscriptionArn();
    }
    private static String initiateJobRequest() {

        JobParameters jobParameters = new JobParameters()
            .withType("archive-retrieval")
            .withArchiveId(archiveId)
            .withSNSTopic(snsTopicARN);

        InitiateJobRequest request = new InitiateJobRequest()
            .withVaultName(vaultName)
            .withJobParameters(jobParameters);

        InitiateJobResult response = client.initiateJob(request);

        return response.getJobId();
    }

    private static long waitForJobToComplete(String jobId, String sqsQueueUrl) throws
InterruptedException, JsonParseException, IOException {

        Boolean messageFound = false;
        Boolean jobSuccessful = false;
        long archiveSizeInBytes = -1;
        ObjectMapper mapper = new ObjectMapper();
        JsonFactory factory = mapper.getFactory();

        while (!messageFound) {
            List<Message> msgs = sqsClient.receiveMessage(
               new
ReceiveMessageRequest(sqsQueueUrl).withMaxNumberOfMessages(10)).getMessages();

            if (msgs.size() > 0) {
```

```
                for (Message m : msgs) {
                    JsonParser jpMessage = factory.createJsonParser(m.getBody());
                    JsonNode jobMessageNode = mapper.readTree(jpMessage);
                    String jobMessage = jobMessageNode.get("Message").textValue();

                    JsonParser jpDesc = factory.createJsonParser(jobMessage);
                    JsonNode jobDescNode = mapper.readTree(jpDesc);
                    String retrievedJobId = jobDescNode.get("JobId").textValue();
                    String statusCode = jobDescNode.get("StatusCode").textValue();
                    archiveSizeInBytes =
jobDescNode.get("ArchiveSizeInBytes").longValue();
                    if (retrievedJobId.equals(jobId)) {
                        messageFound = true;
                        if (statusCode.equals("Succeeded")) {
                            jobSuccessful = true;
                        }
                    }
                }

            } else {
              Thread.sleep(sleepTime * 1000);
            }
          }
        return (messageFound && jobSuccessful) ? archiveSizeInBytes : -1;
    }

    private static void downloadJobOutput(String jobId, long archiveSizeInBytes) throws
IOException {

        if (archiveSizeInBytes < 0) {
            System.err.println("Nothing to download.");
            return;
        }

        System.out.println("archiveSizeInBytes: " + archiveSizeInBytes);
        FileOutputStream fstream = new FileOutputStream(fileName);
        long startRange = 0;
        long endRange = (downloadChunkSize > archiveSizeInBytes) ? archiveSizeInBytes
-1 : downloadChunkSize - 1;

        do {

            GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
                .withVaultName(vaultName)
```

```
                .withRange("bytes=" + startRange + "-" + endRange)
                .withJobId(jobId);
            GetJobOutputResult getJobOutputResult =
client.getJobOutput(getJobOutputRequest);

            BufferedInputStream is = new
BufferedInputStream(getJobOutputResult.getBody());
            byte[] buffer = new byte[(int)(endRange - startRange + 1)];

            System.out.println("Checksum received: " +
getJobOutputResult.getChecksum());
            System.out.println("Content range " +
getJobOutputResult.getContentRange());


            int totalRead = 0;
            while (totalRead < buffer.length) {
                int bytesRemaining = buffer.length - totalRead;
                int read = is.read(buffer, totalRead, bytesRemaining);
                if (read > 0) {
                    totalRead = totalRead + read;
                } else {
                    break;
                }

            }
            System.out.println("Calculated checksum: " +
TreeHashGenerator.calculateTreeHash(new ByteArrayInputStream(buffer)));
            System.out.println("read = " + totalRead);
            fstream.write(buffer);

            startRange = startRange + (long)totalRead;
            endRange = ((endRange + downloadChunkSize) >  archiveSizeInBytes) ?
archiveSizeInBytes : (endRange + downloadChunkSize);
            is.close();
        } while (endRange <= archiveSizeInBytes  && startRange < archiveSizeInBytes);

        fstream.close();
        System.out.println("Retrieved file to " + fileName);

    }

    private static void cleanUp() {
        snsClient.unsubscribe(new UnsubscribeRequest(snsSubscriptionARN));
```

```
        snsClient.deleteTopic(new DeleteTopicRequest(snsTopicARN));
        sqsClient.deleteQueue(new DeleteQueueRequest(sqsQueueURL));
    }
}
```

使用 AWS SDK for .NET 下載 Amazon S3 Glacier 中的封存

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了下載封存的方法。

主題

- 使用 AWS SDK for .NET 的高階 API 下載封存
- 使用 AWS SDK for .NET 的低階 API 下載封存

使用 AWS SDK for .NET 的高階 API 下載封存

高階 API 的 `ArchiveTransferManager` 類別提供可用來下載封存的 `Download` 方法。

> **Important**
>
> 此 `ArchiveTransferManager` 類別會建立 Amazon Simple Notification Service (Amazon SNS) 主題，以及訂閱該主題的 Amazon Simple Queue Service (Amazon SQS) 佇列。然後啟動封存擷取任務並輪詢佇列以使封存可用。封存可用時，就會開始下載。如需封存擷取時間的詳細資訊，請參閱 封存擷取選項。

範例：使用 AWS SDK for .NET 的高階 API 下載封存

以下 C# 程式碼範例從美國西部 (奧勒岡) 區域中的文件庫 (`examplevault`) 下載封存。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您需要更新程式碼，如所示之在現有的封存 ID 以及欲儲存下載封存的本機檔案路徑 。

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadHighLevel
```

```
  {
    static string vaultName        = "examplevault";
    static string archiveId        = "*** Provide archive ID ***";
    static string downloadFilePath = "*** Provide the file name and path to where to
 store the download ***";

    public static void Main(string[] args)
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);

        var options = new DownloadOptions();
        options.StreamTransferProgress += ArchiveDownloadHighLevel.progress;
        // Download an archive.
        Console.WriteLine("Intiating the archive retrieval job and then polling SQS
 queue for the archive to be available.");
        Console.WriteLine("Once the archive is available, downloading will begin.");
        manager.Download(vaultName, archiveId, downloadFilePath, options);
        Console.WriteLine("To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static int currentPercentage = -1;
    static void progress(object sender, StreamTransferProgressArgs args)
    {
      if (args.PercentDone != currentPercentage)
      {
        currentPercentage = args.PercentDone;
        Console.WriteLine("Downloaded {0}%", args.PercentDone);
      }
    }
  }
}
```

使用 AWS SDK for .NET 的低階 API 下載封存

以下這些步驟說明，如何使用 AWS SDK for .NET 的低階 API，下載 Amazon S3 Glacier (S3 Glacier) 封存。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要從中下載封存的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 執行 `archive-retrieval` 方法以啟動 `InitiateJob` 任務。

   透過建立 `InitiateJobRequest` 類別的執行個體來提供任務資訊，例如要下載的封存的封存 ID，以及希望 S3 Glacier 發佈任務完成訊息的選用 Amazon SNS 主題。S3 Glacier 會在回應中傳回任務 ID。該回應在 `InitiateJobResponse` 類別的執行個體中可用。

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);

InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
  VaultName = vaultName,
  JobParameters = new JobParameters()
  {
    Type      = "archive-retrieval",
    ArchiveId = "*** Provide archive id ***",
    SNSTopic  = "*** Provide Amazon SNS topic ARN ***",
  }
};

InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

   您可以選擇性指定位元組範圍，以請求 S3 Glacier 僅準備封存的一部分，如以下請求所示。該請求指定 S3 Glacier，以便僅準備封存的 1 MB 到 2 MB 部分。

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);


InitiateJobRequest initJobRequest = new InitiateJobRequest()
{
```

```
  VaultName = vaultName,
  JobParameters = new JobParameters()
  {
    Type      = "archive-retrieval",
    ArchiveId = "*** Provide archive id ***",
    SNSTopic  = "*** Provide Amazon SNS topic ARN ***",
  }
};
// Specify byte range.
int ONE_MEG = 1048576;
initJobRequest.JobParameters.RetrievalByteRange = string.Format("{0}-{1}", ONE_MEG, 2
 * ONE_MEG -1);

InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
string jobId = initJobResponse.JobId;
```

3. 等候任務完成。

   您必須等到任務輸出準備好供您下載。如果您在文件庫上設定識別 Amazon Simple Notification Service (Amazon SNS) 主題的通知設定，或者在啟動任務時指定 Amazon SNS 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。以下章節提供的程式碼使用適用於 S3 Glacier 的 Amazon SNS 來發布訊息。

   您也可以透過呼叫 `DescribeJob` 方法來輪詢 S3 Glacier，以判斷任務完成狀態。雖然，使用 Amazon SNS 主題進行通知是建議的方法。

4. 透過執行 `GetJobOutput` 方法下載任務輸出 (封存資料)。

   透過建立 `GetJobOutputRequest` 類別的執行個體來提供請求資訊，如任務 ID 和文件庫名稱。S3 Glacier 傳回的輸出在 `GetJobOutputResponse` 物件中可用。

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};

GetJobOutputResponse getJobOutputResponse = client.GetJobOutput(getJobOutputRequest);
using (Stream webStream = getJobOutputResponse.Body)
{
  using (Stream fileToSave = File.OpenWrite(fileName))
  {
     CopyStream(webStream, fileToSave);
```

```
  }
}
```

上述程式碼片段下載整個任務的輸出。您可以選擇只擷取輸出的一部分，或透過指定 GetJobOutputRequest 中的位元組範圍以較小的區塊下載整個輸出。

```
GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
{
  JobId = jobId,
  VaultName = vaultName
};
getJobOutputRequest.SetRange(0, 1048575); // Download only the first 1 MB chunk of
 the output.
```

為了回應 GetJobOutput 呼叫，如果滿足某些條件，S3 Glacier 將傳回下載資料部分的檢查總和。如需更多詳細資訊，請參閱 下載資料時接收檢查總和。

為了驗證下載沒有錯誤，您可以計算用戶端的檢查總和，並將其與在回應中傳送的檢查總和 S3 Glacier 相比較。

對於指定的可選範圍的封存擷取任務，當您取得任務描述時，它包括要擷取的範圍的檢查總和 (SHA256TreeHash)。您可以使用此值進一步驗證您稍後下載的整個位元組範圍的準確性。例如，如果啟動任務以擷取樹狀雜湊符合的封存範圍，然後以區塊的方式下載輸出，以便每個 GetJobOutput 請求傳回檢查總和，則可以計算在用戶端上下載的每個部分的檢查總和，然後計算樹狀雜湊。您可以將它與 S3 Glacier 傳回的檢查總和進行比較，以回應您的描述任務請求，以驗證您下載的整個位元組範圍與存放在 S3 Glacier 中的位元組範圍相同。

如需運作範例，請參閱 範例 2：使用區塊中的 AWS SDK for .NET下載輸出的低階 API 擷取封存。

範例 1：使用 AWS SDK for .NET 的低階 API 擷取封存

以下 C# 程式碼範例從指定的文件庫下載封存。任務完成後，該範例將在單一 GetJobOutput 呼叫中下載整個輸出。如需有關以區塊形式下載輸出的範例，請參閱 範例 2：使用區塊中的 AWS SDK for .NET下載輸出的低階 API 擷取封存。

範例會執行下列任務：

- 設定 Amazon Simple Notification Service (Amazon SNS) 主題

S3 Glacier 會在完成任務後向該主題傳送通知。

- 設定 Amazon Simple Queue Service (Amazon SQS) 佇列。

  此範例將政策連接至佇列，以使 Amazon SNS 主題能夠發佈訊息。
- 起始任務以下載指定的封存。

  在任務請求中，該範例指定 Amazon SNS 主題，以便 S3 Glacier 在任務完成後傳送訊息。
- 定期檢查 Amazon SQS 佇列中的訊息。

  如果有訊息，剖析 JSON 並檢查任務是否順利完成。如果是，請下載封存。該代碼範例使用 JSON.NET 程式庫 (請參閱 JSON.NET) 來剖析 JSON。
- 透過刪除 Amazon SNS 主題和其建立的 Amazon SQS 佇列來清除。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadLowLevelUsingSNSSQS
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string archiveID = "*** Provide archive ID ***";
    static string fileName  = "*** Provide the file name and path to where to store
 downloaded archive ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
    const string SQS_POLICY =
```

```
        "{" +
        "    \"Version\" : \"2012-10-17\"," +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"Service\" : \"sns.amazonaws.com\" }," +
        "            \"Action\"    : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QueueArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
        "            }" +
        "        }" +
        "    ]" +
        "}";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Setup SNS topic and SQS queue.");
          SetupTopicAndQueue();
          Console.WriteLine("To continue, press Enter"); Console.ReadKey();
          Console.WriteLine("Retrieving...");
          RetrieveArchive(client);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
        // Delete SNS topic and SQS queue.
        snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
        sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }
```

```
   static void SetupTopicAndQueue()
   {
     snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
     sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);

     long ticks = DateTime.Now.Ticks;
     topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
     Console.Write("topicArn: "); Console.WriteLine(topicArn);

     CreateQueueRequest createQueueRequest = new CreateQueueRequest();
     createQueueRequest.QueueName = "GlacierDownload-" + ticks;
     CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
     queueUrl = createQueueResponse.QueueUrl;
     Console.Write("QueueURL: "); Console.WriteLine(queueUrl);

     GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
     getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
     getQueueAttributesRequest.QueueUrl = queueUrl;
     GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
     queueArn = response.QueueARN;
     Console.Write("QueueArn: "); Console.WriteLine(queueArn);

     // Setup the Amazon SNS topic to publish to the SQS queue.
     snsClient.Subscribe(new SubscribeRequest()
     {
       Protocol = "sqs",
       Endpoint = queueArn,
       TopicArn = topicArn
     });

     // Add policy to the queue so SNS can send messages to the queue.
     var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QueueArn}",
queueArn);

     sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
     {
         QueueUrl = queueUrl,
         Attributes = new Dictionary<string, string>
```

```
          {
              { QueueAttributeName.Policy, policy }
          }
      });
    }

    static void RetrieveArchive(AmazonGlacierClient client)
    {
      // Initiate job.
      InitiateJobRequest initJobRequest = new InitiateJobRequest()
      {
        VaultName = vaultName,
        JobParameters = new JobParameters()
        {
          Type = "archive-retrieval",
          ArchiveId = archiveID,
          Description = "This job is to download archive.",
          SNSTopic = topicArn,
        }
      };
      InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
      string jobId = initJobResponse.JobId;

      // Check queue for a message and if job completed successfully, download archive.
      ProcessQueue(jobId, client);
    }

    private static void ProcessQueue(string jobId, AmazonGlacierClient client)
    {
      ReceiveMessageRequest receiveMessageRequest = new ReceiveMessageRequest()
{ QueueUrl = queueUrl, MaxNumberOfMessages = 1 };
      bool jobDone = false;
      while (!jobDone)
      {
        Console.WriteLine("Poll SQS queue");
        ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
        if (receiveMessageResponse.Messages.Count == 0)
        {
          Thread.Sleep(10000 * 60);
          continue;
        }
        Console.WriteLine("Got message");
        Message message = receiveMessageResponse.Messages[0];
```

```
        Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
        Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
        string statusCode = fields["StatusCode"] as string;

        if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
        {
          Console.WriteLine("Downloading job output");
          DownloadOutput(jobId, client); // Save job output to the specified file
location.
        }
        else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
          Console.WriteLine("Job failed... cannot download the archive.");

        jobDone = true;
        sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
ReceiptHandle = message.ReceiptHandle });
      }
    }

    private static void DownloadOutput(string jobId, AmazonGlacierClient client)
    {
      GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
      {
        JobId = jobId,
        VaultName = vaultName
      };

      GetJobOutputResponse getJobOutputResponse =
client.GetJobOutput(getJobOutputRequest);
      using (Stream webStream = getJobOutputResponse.Body)
      {
          using (Stream fileToSave = File.OpenWrite(fileName))
          {
              CopyStream(webStream, fileToSave);
          }
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
```

```
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

範例 2：使用區塊中的 AWS SDK for .NET下載輸出的低階 API 擷取封存

下列 C# 程式碼範例會從 S3 Glacier 擷取封存。該程式碼範例透過在 `GetJobOutputRequest` 物件中指定位元組範圍來下載區塊中的任務輸出。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;
using Amazon.SimpleNotificationService;
using Amazon.SimpleNotificationService.Model;
using Amazon.SQS;
using Amazon.SQS.Model;
using Newtonsoft.Json;
using System.Collections.Specialized;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDownloadLowLevelUsingSQLSNSOutputUsingRange
  {
    static string topicArn;
    static string queueUrl;
    static string queueArn;
    static string vaultName = "*** Provide vault name ***";
    static string archiveId = "*** Provide archive ID ***";
    static string fileName  = "*** Provide the file name and path to where to store
 downloaded archive ***";
    static AmazonSimpleNotificationServiceClient snsClient;
    static AmazonSQSClient sqsClient;
```

```
    const string SQS_POLICY =
        "{" +
        "    \"Version\" : \"2012-10-17\"," +
        "    \"Statement\" : [" +
        "        {" +
        "            \"Sid\" : \"sns-rule\"," +
        "            \"Effect\" : \"Allow\"," +
        "            \"Principal\" : {\"AWS\" : \"arn:aws:iam::123456789012:root\" }," +
        "            \"Action\"  : \"sqs:SendMessage\"," +
        "            \"Resource\"  : \"{QuernArn}\"," +
        "            \"Condition\" : {" +
        "                \"ArnLike\" : {" +
        "                    \"aws:SourceArn\" : \"{TopicArn}\"" +
        "                }" +
        "            }" +
        "        }" +
        "    ]" +
        "}";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;

      try
      {
          using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
          {
              Console.WriteLine("Setup SNS topic and SQS queue.");
              SetupTopicAndQueue();
              Console.WriteLine("To continue, press Enter"); Console.ReadKey();

              Console.WriteLine("Download archive");
              DownloadAnArchive(archiveId, client);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      finally
      {
        // Delete SNS topic and SQS queue.
```

```
        snsClient.DeleteTopic(new DeleteTopicRequest() { TopicArn = topicArn });
        sqsClient.DeleteQueue(new DeleteQueueRequest() { QueueUrl = queueUrl });
      }
    }

       static void SetupTopicAndQueue()
    {
      long ticks = DateTime.Now.Ticks;

      // Setup SNS topic.
      snsClient = new
AmazonSimpleNotificationServiceClient(Amazon.RegionEndpoint.USWest2);
      sqsClient = new AmazonSQSClient(Amazon.RegionEndpoint.USWest2);

      topicArn = snsClient.CreateTopic(new CreateTopicRequest { Name =
"GlacierDownload-" + ticks }).TopicArn;
      Console.Write("topicArn: "); Console.WriteLine(topicArn);

      CreateQueueRequest createQueueRequest = new CreateQueueRequest();
      createQueueRequest.QueueName = "GlacierDownload-" + ticks;
      CreateQueueResponse createQueueResponse =
sqsClient.CreateQueue(createQueueRequest);
      queueUrl = createQueueResponse.QueueUrl;
      Console.Write("QueueURL: "); Console.WriteLine(queueUrl);

      GetQueueAttributesRequest getQueueAttributesRequest = new
GetQueueAttributesRequest();
      getQueueAttributesRequest.AttributeNames = new List<string> { "QueueArn" };
      getQueueAttributesRequest.QueueUrl = queueUrl;
      GetQueueAttributesResponse response =
sqsClient.GetQueueAttributes(getQueueAttributesRequest);
      queueArn = response.QueueARN;
      Console.Write("QueueArn: "); Console.WriteLine(queueArn);

      // Setup the Amazon SNS topic to publish to the SQS queue.
      snsClient.Subscribe(new SubscribeRequest()
      {
        Protocol = "sqs",
        Endpoint = queueArn,
        TopicArn = topicArn
      });

      // Add the policy to the queue so SNS can send messages to the queue.
```

```
      var policy = SQS_POLICY.Replace("{TopicArn}", topicArn).Replace("{QuernArn}",
queueArn);

      sqsClient.SetQueueAttributes(new SetQueueAttributesRequest()
      {
          QueueUrl = queueUrl,
          Attributes = new Dictionary<string, string>
          {
              { QueueAttributeName.Policy, policy }
          }
      });
    }

    static void DownloadAnArchive(string archiveId, AmazonGlacierClient client)
    {
      // Initiate job.
      InitiateJobRequest initJobRequest = new InitiateJobRequest()
      {

        VaultName = vaultName,
        JobParameters = new JobParameters()
        {
          Type = "archive-retrieval",
          ArchiveId = archiveId,
          Description = "This job is to download the archive.",
          SNSTopic = topicArn,
        }
      };
      InitiateJobResponse initJobResponse = client.InitiateJob(initJobRequest);
      string jobId = initJobResponse.JobId;

      // Check queue for a message and if job completed successfully, download archive.
      ProcessQueue(jobId, client);
    }

    private static void ProcessQueue(string jobId, AmazonGlacierClient client)
    {
        var receiveMessageRequest = new ReceiveMessageRequest() { QueueUrl = queueUrl,
MaxNumberOfMessages = 1 };
        bool jobDone = false;
        while (!jobDone)
        {
            Console.WriteLine("Poll SQS queue");
```

```
            ReceiveMessageResponse receiveMessageResponse =
sqsClient.ReceiveMessage(receiveMessageRequest);
            if (receiveMessageResponse.Messages.Count == 0)
            {
                Thread.Sleep(10000 * 60);
                continue;
            }
            Console.WriteLine("Got message");
            Message message = receiveMessageResponse.Messages[0];
            Dictionary<string, string> outerLayer =
JsonConvert.DeserializeObject<Dictionary<string, string>>(message.Body);
            Dictionary<string, object> fields =
JsonConvert.DeserializeObject<Dictionary<string, object>>(outerLayer["Message"]);
            string statusCode = fields["StatusCode"] as string;
            if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_SUCCEEDED,
StringComparison.InvariantCultureIgnoreCase))
            {
                long archiveSize = Convert.ToInt64(fields["ArchiveSizeInBytes"]);
                Console.WriteLine("Downloading job output");
                DownloadOutput(jobId, archiveSize, client); // This where we save job
output to the specified file location.
            }
            else if (string.Equals(statusCode, GlacierUtils.JOB_STATUS_FAILED,
StringComparison.InvariantCultureIgnoreCase))
                Console.WriteLine("Job failed... cannot download the archive.");
            jobDone = true;
            sqsClient.DeleteMessage(new DeleteMessageRequest() { QueueUrl = queueUrl,
ReceiptHandle = message.ReceiptHandle });
        }
    }

    private static void DownloadOutput(string jobId, long archiveSize,
AmazonGlacierClient client)
    {
      long partSize = 4 * (long)Math.Pow(2, 20);  // 4 MB.
      using (Stream fileToSave = new FileStream(fileName, FileMode.Create,
FileAccess.Write))
      {

        long currentPosition = 0;
        do
        {
          GetJobOutputRequest getJobOutputRequest = new GetJobOutputRequest()
          {
```

```
            JobId = jobId,
            VaultName = vaultName
          };

          long endPosition = currentPosition + partSize - 1;
          if (endPosition > archiveSize)
            endPosition = archiveSize;

          getJobOutputRequest.SetRange(currentPosition, endPosition);
          GetJobOutputResponse getJobOutputResponse =
  client.GetJobOutput(getJobOutputRequest);

          using (Stream webStream = getJobOutputResponse.Body)
          {
            CopyStream(webStream, fileToSave);
          }
          currentPosition += partSize;
        } while (currentPosition < archiveSize);
      }
    }

    public static void CopyStream(Stream input, Stream output)
    {
      byte[] buffer = new byte[65536];
      int length;
      while ((length = input.Read(buffer, 0, buffer.Length)) > 0)
      {
        output.Write(buffer, 0, length);
      }
    }
  }
}
```

使用 REST API 下載封存

使用 REST API 下載封存

下載封存是兩個步驟程序。

1. 啟動 `archive-retrieval` 類型的任務。如需更多詳細資訊，請參閱 啟動任務 (POST 任務)。
2. 任務完成後，下載封存資料。如需更多詳細資訊，請參閱 「取得任務輸出」 (GET 輸出)。

使用 AWS CLI 下載 Amazon S3 Glacier 中的封存

您可以使用 AWS Command Line Interface (AWS CLI)，在 Amazon S3 Glacier (S3 Glacier) 中下載封存。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 下載封存

> **Note**
>
> 為了下載封存，您必須知道封存 ID。步驟 1-4 將擷取封存 ID。如果您已經知道要下載的封存 ID，請跳至步驟 5。

1. 使用 `initiate-job` 命令開始庫存擷取任務。庫存報告將列出封存 ID。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters="{\"Type\":\"inventory-retrieval\"}"
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. 使用 `describe-job` 命令檢查先前 任務命令的狀態。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

預期的輸出結果：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. 等候任務完成。

   您必須等到任務輸出準備好供您下載。如果您在文件庫上設定通知設定，或者在起始任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

   您可以為文件庫中的特定事件設定通知組態。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 SNS 主題。

4. 完成時，請使用 `get-job-output` 命令將擷取任務下載至檔案 `output.json`。該檔案將包含封存 ID。

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

   這個命令會產生一個包含下列欄位的檔案。

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[
{"ArchiveId":"*** archiveid ***",
"ArchiveDescription":*** archive description (if set) ***,
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}
{"ArchiveId":
...
]}
```

5. 使用 `initiate-job` 命令，從文件庫開始每個封存的擷取程序。您將需要將任務參數指定為 `archive-retrieval`，如下所示。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333
 --job-parameters="{\"Type\":\"archive-retrieval\",\"ArchiveId\":\"*** archiveId
 ***\"}"
```

6. 等候 `archive-retrieval` 任務完成。使用 `describe-job` 命令檢查先前命令的狀態。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

7. 當上述任務完成後，使用 get-job-output 命令下載封存。

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output_file_name
```

刪除 Amazon S3 Glacier 中的封存

您無法使用 Amazon S3 Glacier (S3 Glacier) 管理主控台來刪除封存。若要刪除封存，您必須使用 AWS Command Line Interface (CLI) 或撰寫程式碼直接使用 REST API 或 AWS SDK for Java 和 .NET 包裝函式程式庫進行刪除請求。下列主題說明如何使用 AWS SDK for Java 和 .NET 包裝函式程式庫、REST API 和 AWS CLI。

主題

- 使用 AWS SDK for Java 刪除 Amazon S3 Glacier 中的封存
- 使用 AWS SDK for .NET 刪除 Amazon S3 Glacier 中的封存
- 使用 REST API 刪除 Amazon S3 Glacier 封存
- 使用 AWS Command Line Interface 刪除 Amazon S3 Glacier 中的封存

您可以一次從文件庫刪除一個封存。若要刪除封存，您必須提在刪除請求中提供其封存 ID。您可以透過為包含封存的文件庫下載文件庫庫存來取得封存 ID。如需下載文件庫庫存的詳細資訊，請參閱在 Amazon S3 Glacier 中下載文件庫庫存。

在您刪除封存之後，您可能仍然能夠成功請求起始任務，擷取已刪除的封存，但封存擷取任務將會失敗。

根據以下情況，當您刪除封存時，進行中的封存 ID 擷取可能會也可能不會成功：

- 如果 S3 Glacier 收到刪除封存請求時，封存擷取任務正在積極準備下載資料，則封存擷取操作可能會失敗。
- 如果 S3 Glacier 收到刪除封存請求時，封存擷取任務已成功準備下載封存，則可以下載輸出。

如需封存擷取的詳細資訊，請參閱 在 S3 Glacier 中下載封存。

此為等冪操作。刪除已刪除的封存不會導致錯誤。

刪除封存後，如果立即下載文件庫庫存，則可能會將已刪除的封存納入清單中，因為 S3 Glacier 一天僅準備一次文件庫庫存。

使用 AWS SDK for Java 刪除 Amazon S3 Glacier 中的封存

以下是使用 AWS SDK for Java 低階 API 來刪除封存的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要刪除之封存所存放的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。

2. 您可以透過建立 `DeleteArchiveRequest` 類別的執行個體來提供請求資訊。

   您需要提供封存 ID、文件庫名稱和您的帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 [將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用]。

3. 以參數形式提供請求物件，以便執行 `deleteArchive` 方法。

下列 Java 程式碼片段描述前述步驟。

```
AmazonGlacierClient client;

DeleteArchiveRequest request = new DeleteArchiveRequest()
    .withVaultName("*** provide a vault name ***")
    .withArchiveId("*** provide an archive ID ***");

client.deleteArchive(request);
```

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 [刪除封存 (DELETE archive)]。

範例：使用 AWS SDK for Java 刪除存檔

以下 Java 程式碼範例使用 AWS SDK for Java 來刪除封存。如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例。您需要如所示，使用文件庫名稱和要刪除之封存的封存 ID 更新程式碼。

Example

```
import java.io.IOException;

import com.amazonaws.auth.profile.ProfileCredentialsProvider;
import com.amazonaws.services.glacier.AmazonGlacierClient;
import com.amazonaws.services.glacier.model.DeleteArchiveRequest;

public class ArchiveDelete {

    public static String vaultName = "*** provide vault name ****";
    public static String archiveId = "*** provide archive ID***";
    public static AmazonGlacierClient client;

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

     ProfileCredentialsProvider credentials = new ProfileCredentialsProvider();

        client = new AmazonGlacierClient(credentials);
        client.setEndpoint("https://glacier.us-east-1.amazonaws.com/");

        try {

            // Delete the archive.
            client.deleteArchive(new DeleteArchiveRequest()
                .withVaultName(vaultName)
                .withArchiveId(archiveId));

            System.out.println("Deleted archive successfully.");

        } catch (Exception e) {
            System.err.println("Archive not deleted.");
            System.err.println(e);
        }
    }
}
```

使用 AWS SDK for .NET 刪除 Amazon S3 Glacier 中的封存

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了刪除封存的方法。

主題

- 使用 AWS SDK for .NET 的高階 API 刪除封存
- 使用 AWS SDK for .NET 的低階 API 刪除封存

使用 AWS SDK for .NET 的高階 API 刪除封存

高階 API 的 `ArchiveTransferManager` 類別提供可用來刪除封存的 `DeleteArchive` 方法。

範例：使用 AWS SDK for .NET 的高階 API 刪除封存

以下 C# 程式碼範例使用 AWS SDK for .NET 的高階 API 來刪除封存。如需執行此範例的逐步說明，請參閱 執行程式碼範例。您需要如所示，使用要刪除之封存的封存 ID 更新程式碼。

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Transfer;
using Amazon.Runtime;

namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteHighLevel
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
    {
      try
      {
        var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
        manager.DeleteArchive(vaultName, archiveId);
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
```

```
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }
  }
}
```

使用 AWS SDK for .NET 的低階 API 刪除封存

以下是使用 AWS SDK for .NET 刪除文件庫的步驟。

1. 建立 `AmazonGlacierClient` 類別的執行個體 (用戶端)。

   您需要指定要刪除之封存所存放的 AWS 區域。所有您使用此用戶端執行的操作均會套用到該 AWS 區域。
2. 您可以透過建立 `DeleteArchiveRequest` 類別的執行個體來提供請求資訊。

   您需要提供封存 ID、文件庫名稱和您的帳戶 ID。如果您不提供帳戶 ID，則會使用與您提供來簽署請求之登入資料關聯的帳戶 ID。如需更多詳細資訊，請參閱 使用 AWS 軟體開發套件搭配 Amazon S3 冰川。
3. 以參數形式提供請求物件，以便執行 `DeleteArchive` 方法。

範例：使用 AWS SDK for .NET 的低階 API 刪除封存

下列 C# 範例描述前述步驟。範例使用 AWS SDK for .NET 的低階 API 來刪除封存。

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 刪除封存 (DELETE archive)。

如需執行此範例的逐步說明，請參閱 執行程式碼範例。您需要如所示，使用要刪除之封存的封存 ID 更新程式碼。

Example

```
using System;
using Amazon.Glacier;
using Amazon.Glacier.Model;
using Amazon.Runtime;
```

```
namespace glacier.amazon.com.docsamples
{
  class ArchiveDeleteLowLevel
  {
    static string vaultName = "examplevault";
    static string archiveId = "*** Provide archive ID ***";

    public static void Main(string[] args)
    {
      AmazonGlacierClient client;
      try
      {
        using (client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2))
        {
          Console.WriteLine("Deleting the archive");
          DeleteAnArchive(client);
        }
        Console.WriteLine("Operations successful. To continue, press Enter");
        Console.ReadKey();
      }
      catch (AmazonGlacierException e) { Console.WriteLine(e.Message); }
      catch (AmazonServiceException e) { Console.WriteLine(e.Message); }
      catch (Exception e) { Console.WriteLine(e.Message); }
      Console.WriteLine("To continue, press Enter");
      Console.ReadKey();
    }

    static void DeleteAnArchive(AmazonGlacierClient client)
    {
      DeleteArchiveRequest request = new DeleteArchiveRequest()
      {
        VaultName = vaultName,
        ArchiveId = archiveId
      };
      DeleteArchiveResponse response = client.DeleteArchive(request);
    }
  }
}
```

使用 REST API 刪除 Amazon S3 Glacier 封存

您可以使用刪除封存 API 來刪除封存。

- 如需刪除封存 API 的資訊，請參閱 刪除封存 (DELETE archive)。
- 如需使用 REST API 的詳細資訊，請參閱Amazon S3 Glacier API 參考。

使用 AWS Command Line Interface 刪除 Amazon S3 Glacier 中的封存

您可以使用 AWS Command Line Interface (AWS CLI)，來刪除 Amazon S3 Glacier (S3 Glacier) 中的封存。

主題

(必要條件) 設定 AWS CLI

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：

   安裝 AWS Command Line Interface

   設定 AWS Command Line Interface

2. 在命令提示字元中輸入下列命令，以驗證 AWS CLI 設定。這些命令不會明確提供登入資料，因此會使用預設描述檔的登入資料。

   - 嘗試使用幫助命令。

     ```
     aws help
     ```

   - 若要取得已設定帳戶上 S3 Glacier 文件庫的清單，請使用 `list-vaults` 命令。將 *123456789012* 替換為 AWS 帳戶 ID。

     ```
     aws glacier list-vaults --account-id 123456789012
     ```

   - 若要查看 AWS CLI 目前的設定資料，請使用 `aws configure list` 命令。

     ```
     aws configure list
     ```

範例：使用 AWS CLI 刪除存檔

1. 使用 initiate-job 命令啟動清查擷取任務。

```
aws glacier initiate-job --vault-name awsexamplevault --account-id 111122223333 --
job-parameters="{\"Type\":\"inventory-retrieval\"}"
```

預期的輸出結果：

```
{
    "location": "/111122223333/vaults/awsexamplevault/jobs/*** jobid ***",
    "jobId": "*** jobid ***"
}
```

2. 使用 describe-job 命令檢查先前擷取任務的狀態。

```
aws glacier describe-job --vault-name awsexamplevault --account-id 111122223333 --
job-id *** jobid ***
```

預期的輸出結果：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "*** vault arn ***",
    "Completed": false,
    "JobId": "*** jobid ***",
    "Action": "InventoryRetrieval",
    "CreationDate": "*** job creation date ***",
    "StatusCode": "InProgress"
}
```

3. 等候任務完成。

您必須等到任務輸出準備好供您下載。如果您在文件庫上設定通知設定，或者在起始任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，則 S3 Glacier 會在完成任務後向該主題傳送訊息。

您可以為文件庫中的特定事件設定通知組態。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。無論何時發生特定事件，S3 Glacier 都會傳送訊息到指定的 SNS 主題。

4. 完成時，請使用 `get-job-output` 命令將擷取任務下載至檔案 `output.json`。

```
aws glacier get-job-output --vault-name awsexamplevault --account-id 111122223333
 --job-id *** jobid *** output.json
```

這個命令會產生一個包含下列欄位的檔案。

```
{
"VaultARN":"arn:aws:glacier:region:111122223333:vaults/awsexamplevault",
"InventoryDate":"*** job completion date ***",
"ArchiveList":[
{"ArchiveId":"*** archiveid ***",
"ArchiveDescription":*** archive description (if set) ***,
"CreationDate":"*** archive creation date ***",
"Size":"*** archive size (in bytes) ***",
"SHA256TreeHash":"*** archive hash ***"
}
{"ArchiveId":
...
]}
```

5. 使用 `delete-archive` 命令從文件庫中刪除每個存檔，直到沒有存檔為止。

```
aws glacier delete-archive --vault-name awsexamplevault --account-id 111122223333
 --archive-id *** archiveid ***
```

使用 AWS 軟體開發套件搭配 Amazon S3 冰川

AWS 提供開發套件供您開發適用於 Amazon S3 冰川的應用程式。軟體開發套件程式庫包裝基礎 S3 Glacier API，可簡化程式設計任務。例如，對於傳送到 S3 Glacier 的每個請求，您必須加入簽章以驗證請求。使用 SDK 程式庫時，您只需要在程式碼中提供 AWS 安全登入資料，而程式庫會計算必要的簽章，並將其包含在傳送至 S3 Glacier 的請求中。 AWS SDK 提供可對應至基礎 REST API 的程式庫，並提供物件，讓您可以使用這些物件輕鬆建構要求和處理回應。

主題

- AWS 適用於 Java 和. NET 的開發套件庫
- 搭配 AWS 開發套件使用 S3 冰川
- 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用
- 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用

AWS Command Line Interface (AWS CLI) 是一個統一的工具來管理您的 AWS 服務，包括 S3 冰川。如需有關下載的資訊 AWS CLI，請參閱AWS Command Line Interface。如需有關 S3 Glacier CLI 命令的清單，請參閱 AWS CLI 命令參考。

AWS 適用於 Java 和. NET 的開發套件庫

適用於 Java 和 .NET 的 AWS 開發套件提供高階和低階的包裝函式庫。

您可以使用本開發人員指南 AWS SDK for .NET 中的 AWS SDK for Java 和，找到使用 Amazon S3 Glacier 的範例。

什麼是低階 API？

低階包裝函式程式庫緊密對應 S3 Glacier 所支援的基礎 REST API (Amazon S3 Glacier API 參考)。對於每個 S3 Glacier REST 作業，低階 API 提供對應的方法，請求物件為您提供請求資訊和回應物件，以供您處理 S3 Glacier 回應。低階包裝函式程式庫是基礎 S3 Glacier 作業的最完整的實作。

如需開發套件程式庫的詳細資訊，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用 和 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。

什麼是高階 API？

為了進一步簡化應用程式的開發，這些程式庫為某些操作提供較高階抽象概念。例如：

- 上傳封存 - 使用低階 API 上傳封存，除了要儲存封存的檔案名稱和保存庫名稱外，您需要提供承載的檢查總和 (SHA-256 樹雜湊)。但是，高階 API 會為您計算檢查總和。
- 下載封存或保存庫庫存 - 使用低階 API 下載封存，您首先要啟動工作，等待工作完成，然後取得工作輸出。您需要編寫額外的程式碼來為 S3 Glacier 設定 Amazon Simple Notification Service (Amazon SNS) 主題，以便在工作完成時通知您。您還需要一些輪詢機制來檢查是否已將作業完成訊息發佈到主題。高階 API 提供一種方法，可下載處理所有這些步驟的封存。您只能指定要儲存下載資料的封存 ID 和資料夾路徑。

如需開發套件程式庫的詳細資訊，請參閱 將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用 和 將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用。

何時使用高階和低階 API

一般而言，如果高階 API 提供了執行操作所需的方法，則應使用高階 API ，因為它本身提供簡易性。但是，如果高階 API 不提供此功能，則可以使用低階 API。此外，低階 API 允許在發生故障時對操作進行精細控制，例如重試邏輯。例如，在上傳封存時，高階 API 會利用檔案大小來判斷是在單一操作上傳封存，還是使用分段上傳 API。API 還具有內建的重試邏輯，以防上傳失敗。但是，您的應用程式可能需要對這些決策進行精細控制，在這種情況下，您可以使用低階 API。

搭配 AWS 開發套件使用 S3 冰川

AWS 軟件開發套件（SDK）可用於許多流行的編程語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
AWS SDK for C++	AWS SDK for C++ 程式碼範例
AWS CLI	AWS CLI 程式碼範例
AWS SDK for Go	AWS SDK for Go 程式碼範例
AWS SDK for Java	AWS SDK for Java 程式碼範例
AWS SDK for JavaScript	AWS SDK for JavaScript 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例

SDK 文件	代碼範例
AWS SDK for .NET	AWS SDK for .NET 程式碼範例
AWS SDK for PHP	AWS SDK for PHP 程式碼範例
AWS Tools for PowerShell	PowerShell 程式碼範例的工具
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) 程式碼範例
AWS SDK for Ruby	AWS SDK for Ruby 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

如需 S3 Glacier 專屬範例，請參閱使用 AWS 開發套件的 S3 冰川程式碼範例。

> **可用性範例**
>
> 找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

將 AWS SDK for Java 與 Amazon S3 Glacier 搭配使用

AWS SDK for Java 為 Amazon S3 Glacier (S3 Glacier) 提供高階和低階的 API，如 使用 AWS 軟體開發套件搭配 Amazon S3 冰川 所述。如需下載 AWS SDK for Java 的相關資訊，請參閱適用於 Java 的 Amazon 開發套件。

> **Note**
>
> AWS SDK for Java 為存取 S3 Glacier 提供安全執行緒用戶端。根據最佳實務，您的應用程式應該建立一個用戶端，並在執行緒之間重複使用該用戶端。

主題

- 使用低階 API

- 使用高階 API
- 使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例
- 設定終端節點

使用低階 API

低階 `AmazonGlacierClient` 類別提供對應到 S3 Glacier (Amazon S3 Glacier API 參考) 基礎 REST 操作的所有方法。在呼叫這些方法中的任何一種方法時，您必須建立對應的請求物件，並提供一個回應物件，其中該方法可以將 S3 Glacier 回應傳回到該操作。

例如，`AmazonGlacierClient` 類別提供 `createVault` 方法來建立文件庫。此方法對應到底層建立文件庫 REST 操作 (請參閱 建立文件庫 (PUT 文件庫))。若要使用這個方法，您必須建立接收 S3 Glacier 回應之 `CreateVaultResult` 物件的執行個體，如以下 Java 程式碼片段所示：

```
AmazonGlacierClient client = new AmazonGlacierClient(credentials);
client.setEndpoint("https://glacier.us-west-2.amazonaws.com/");

CreateVaultRequest request = new CreateVaultRequest()
    .withAccountId("-")
    .withVaultName(vaultName);
CreateVaultResult result = client.createVault(createVaultRequest);
```

在指南中的所有低階範例都使用此模式。

> **Note**
>
> 在建立請求時，前置程式碼區段指定 `AccountID`。但是，當使用 AWS SDK for Java，請求中的 `AccountId` 是可選的，因此本指南中的所有低階範例都不設定這個值。`AccountId` 是 AWS 帳戶 識別碼。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 識別碼或選擇性使用「-」，在這種情況下，S3 Glacier 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在其中包含任何連字號。當您使用 AWS SDK for Java，如果不提供帳戶 ID，則該程式庫將帳戶 ID 設定為「-」。

使用高階 API

為了進一步簡化您的應用程式開發，AWS SDK for Java 提供 `ArchiveTransferManager` 類別，它為低階 API 中的某些方法實作較高階抽象概念。它為封存操作提供有用的方法，例如 `upload` 和 `download` 方法。

例如，以下 Java 程式碼片段使用 `upload` 高階方法來上傳封存檔案。

```
String vaultName = "examplevault";
String archiveToUpload = "c:/folder/exampleArchive.zip";

ArchiveTransferManager atm = new ArchiveTransferManager(client, credentials);
String archiveId = atm.upload(vaultName, "Tax 2012 documents", new
 File(archiveToUpload)).getArchiveId();
```

請注意，您執行的任何操作都適用於在建立 `ArchiveTransferManager` 物件時所指定的 AWS 區域。如果您不指定任何 AWS 區域，AWS SDK for Java 將 `us-east-1` 設定為預設 AWS 區域。

在指南中的所有高階範例都使用此模式。

> **Note**
>
> 高階 `ArchiveTransferManager` 類別可以使用 `AmazonGlacierClient` 執行個體或 `AWSCredentials` 執行個體來建構。

使用 Eclipse 執行 Amazon S3 Glacier 的 Java 範例

開始使用 Java 程式碼範例的最簡單方式是安裝最新的 AWS Toolkit for Eclipse。如需安裝或更新到最新工具組的詳細資訊，請前往 http://aws.amazon.com/eclipse。下列任務會引導您建立與測試本節中所提供的 Java 程式碼範例。

建立 Java 程式碼之一般程序的範例

1	建立 AWS 憑證的預設憑證設定檔，如 AWS SDK for Java 主題在適用於 Java 的 Amazon 開發套件中提供 AWS 憑證中所述。
2	在 Eclipse 中建立新 AWS Java 專案。專案是使用AWS SDK for Java所預先設定。

3	將程式碼從您所讀取的區段複製至專案。
4	提供任何必要資料，以更新程式碼。例如，如果上傳檔案，請提供檔案路徑與儲存貯體名稱。
5	執行程式碼。驗證是使用 AWS Management Console來建立物件。如需有關 AWS Management Console 的詳細資訊，請移至 http://aws.amazon.com/console/。

設定終端節點

預設情況下，AWS SDK for Java 使用端點 `https://glacier.us-east-1.amazonaws.com`。您可以明確地設定終端節點，如以下 Java 程式碼片段所示。

以下程式碼片段介紹如何將端點設定為低階 API 中的美國西部 (奧勒岡) 區域 (`us-west-2`)。

Example

```
client = new AmazonGlacierClient(credentials);
client.setEndpoint("glacier.us-west-2.amazonaws.com");
```

以下程式碼片段介紹如何將端點設定為高階 API 中的美國西部 (奧勒岡) 區域。

```
glacierClient = new AmazonGlacierClient(credentials);
sqsClient = new AmazonSQSClient(credentials);
snsClient = new AmazonSNSClient(credentials);

glacierClient.setEndpoint("glacier.us-west-2.amazonaws.com");
sqsClient.setEndpoint("sqs.us-west-2.amazonaws.com");
snsClient.setEndpoint("sns.us-west-2.amazonaws.com");

ArchiveTransferManager atm = new ArchiveTransferManager(glacierClient, sqsClient,
 snsClient);
```

如需支援 AWS 區域和端點的清單，請參閱存取 Amazon S3 Glacier。

將 AWS SDK for .NET 與 Amazon S3 Glacier 搭配使用

AWS SDK for .NET API 可於 `AWSSDK.dll` 使用。如需下載 AWS SDK for .NET 的資訊，請前往範本程式碼程式庫。如 使用 AWS 軟體開發套件搭配 Amazon S3 冰川 所述，AWS SDK for .NET 提供高階和低階的 API。

> **Note**
>
> 低階 API 和高階 API 為存取 S3 Glacier 提供安全執行緒用戶端。根據最佳實務，您的應用程式應該建立一個用戶端，並在執行緒之間重複使用該用戶端。

主題

使用低階 API

低階 `AmazonGlacierClient` 類別提供對應到 Amazon S3 Glacier (S3 Glacier) (Amazon S3 Glacier API 參考) 基礎 REST 操作的所有方法。在呼叫這些方法中的任何一種方法時，您必須建立對應的請求物件，並提供一個回應物件，其中該方法可以將 S3 Glacier 回應傳回到該操作。

例如，`AmazonGlacierClient` 類別提供 `CreateVault` 方法來建立文件庫。此方法對應到底層建立文件庫 REST 操作 (請參閱 建立文件庫 (PUT 文件庫))。若要使用這個方法，您必須建立 `CreateVaultRequest` 和 `CreateVaultResponse` 類別的執行個體，以提供請求資訊並接收 S3 Glacier 回應，如以下的 C# 程式碼片段所示：

```
AmazonGlacierClient client;
client = new AmazonGlacierClient(Amazon.RegionEndpoint.USEast1);

CreateVaultRequest request = new CreateVaultRequest()
{
  AccountId = "-",
  VaultName = "*** Provide vault name ***"
};
```

```
CreateVaultResponse response = client.CreateVault(request);
```

在指南中的所有低階範例都使用此模式。

> **Note**
>
> 在建立請求時，前置程式碼區段指定 AccountId。但是，當使用 AWS SDK for .NET，請求中的 AccountId 是可選的，因此本指南中的所有低階範例都不設定這個值。AccountId 是 AWS 帳戶 識別碼。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 識別碼或選擇性使用「-」，在這種情況下，S3 Glacier 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在其中包含任何連字號。當您使用 AWS SDK for .NET，如果不提供帳戶 ID，則該程式庫將帳戶 ID 設定為「-」。

使用高階 API

為了進一步簡化您的應用程式開發，AWS SDK for .NET 提供 ArchiveTransferManager 類別，它為低階 API 中的某些方法實作較高階抽象概念。它為封存操作提供有用的方法，例如 Upload 和 Download。

例如，以下 C# 程式碼片段使用 Upload 高階方法來上傳封存。

```
string vaultName = "examplevault";
string archiveToUpload = "c:\folder\exampleArchive.zip";

var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USEast1);
string archiveId = manager.Upload(vaultName, "archive description",
 archiveToUpload).ArchiveId;
```

請注意，您執行的任何操作都適用於在建立 ArchiveTransferManager 物件時所指定的 AWS 區域。在指南中的所有高階範例都使用此模式。

> **Note**
>
> 高階 ArchiveTransferManager 類別仍需要低層 AmazonGlacierClient 用戶端，您可以明確地通過該用戶端或 ArchiveTransferManager 建立用戶端。

執行程式碼範例

開始使用 .NET 程式碼範例的最簡單方式是安裝AWS SDK for .NET。如需詳細資訊，請前往適用於 .NET 的 Amazon 開發套件。

下列程序概述的步驟，讓您可以測試本指南中所提供的程式碼範例。

建立 .NET 程式碼範例之一般程序 (使用 Visual Studio)

1	建立 AWS 憑證的憑證描述檔，如適用於 .NET 的 Amazon 開發套件主題中的設定 AWS 憑證所述。
2	使用 AWS 空白專案範本建立新的 Visual Studio 專案。
3	將專案檔 `Program.cs` 中的程式碼取代為您所讀取區段中的程式碼。
4	執行程式碼。驗證是使用 AWS Management Console來建立物件。如需有關 AWS Management Console 的詳細資訊，請移至 http://aws.amazon.com/console/。

設定終端節點

依預設，AWS SDK for .NET 會將端點設定為美國西部 (奧勒岡) 區域 (`https://glacier.us-west-2.amazonaws.com`)。您可以將端點設定為其他 AWS 區域，如以下 C# 程式碼片段所示。

以下程式碼片段介紹如何將端點設定為低階 API 中的美國西部 (奧勒岡) 區域 (`us-west-2`)。

Example

```
AmazonGlacierClient client = new AmazonGlacierClient(Amazon.RegionEndpoint.USWest2);
```

以下程式碼片段介紹如何將端點設定為高階 API 中的美國西部 (奧勒岡) 區域。

```
var manager = new ArchiveTransferManager(Amazon.RegionEndpoint.USWest2);
```

如需支援 AWS 區域和端點的最新清單，請參閱存取 Amazon S3 Glacier。

使用 AWS 開發套件的 S3 冰川程式碼範例

下列程式碼範例顯示如何搭配 AWS 軟體開發套件 (SDK) 使用 S3 Glacier。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境和跨服務範例中查看內容中的動作。

Scenarios (案例) 是向您展示如何呼叫相同服務中的多個函數來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含入門相關資訊和舊版 SDK 的詳細資訊。

開始使用

哈囉，Amazon S3 Glacier

下列程式碼範例示範如何開始使用 Amazon S3 Glacier。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
using Amazon.Glacier;
using Amazon.Glacier.Model;

namespace GlacierActions;

public static class HelloGlacier
{
    static async Task Main()
    {
        var glacierService = new AmazonGlacierClient();
```

```
        Console.WriteLine("Hello Amazon Glacier!");
        Console.WriteLine("Let's list your Glacier vaults:");

        // You can use await and any of the async methods to get a response.
        // Let's get the vaults using a paginator.
        var glacierVaultPaginator = glacierService.Paginators.ListVaults(
            new ListVaultsRequest { AccountId = "-" });

        await foreach (var vault in glacierVaultPaginator.VaultList)
        {
            Console.WriteLine($"{vault.CreationDate}:{vault.VaultName}, ARN:
{vault.VaultARN}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考ListVaults中的。

程式碼範例

- 使用 AWS 開發套件執行 S3 冰川的動作
  - 搭AddTagsToVault配 AWS 開發套件或 CLI 使用
  - 搭CreateVault配 AWS 開發套件或 CLI 使用
  - 搭DeleteArchive配 AWS 開發套件或 CLI 使用
  - 搭DeleteVault配 AWS 開發套件或 CLI 使用
  - 搭DeleteVaultNotifications配 AWS 開發套件或 CLI 使用
  - 搭DescribeJob配 AWS 開發套件或 CLI 使用
  - 搭DescribeVault配 AWS 開發套件或 CLI 使用
  - 搭GetJobOutput配 AWS 開發套件或 CLI 使用
  - 搭GetVaultNotifications配 AWS 開發套件或 CLI 使用
  - 搭InitiateJob配 AWS 開發套件或 CLI 使用
  - 搭ListJobs配 AWS 開發套件或 CLI 使用
  - 搭ListTagsForVault配 AWS 開發套件或 CLI 使用
  - 搭ListVaults配 AWS 開發套件或 CLI 使用
  - 搭SetVaultNotifications配 AWS 開發套件或 CLI 使用
  - 搭UploadArchive配 AWS 開發套件或 CLI 使用

  - 搭UploadMultipartPart配 AWS 開發套件或 CLI 使用
- 使用 AWS 開發套件的 S3 冰川案例
  - 使用開 AWS 發套件將檔案存檔到 Amazon S3 Glacier、取得通知並啟動任務
  - 使用 AWS 開發套件取得 Amazon S3 冰川存檔內容並刪除存檔

使用 AWS 開發套件執行 S3 冰川的動作

下列程式碼範例示範如何使用 AWS 開發套件執行個別 S3 Glacier 動作。這些摘錄會呼叫 S3 Glacier API，是必須在內容中執行之大型程式的程式碼摘錄。每個範例都包含一個連結 GitHub，您可以在其中找到設定和執行程式碼的指示。

下列範例僅包含最常使用的動作。如需完整清單，請參閱 Amazon S3 Glacier API 參考。

範例

- 搭AddTagsToVault配 AWS 開發套件或 CLI 使用
- 搭CreateVault配 AWS 開發套件或 CLI 使用
- 搭DeleteArchive配 AWS 開發套件或 CLI 使用
- 搭DeleteVault配 AWS 開發套件或 CLI 使用
- 搭DeleteVaultNotifications配 AWS 開發套件或 CLI 使用
- 搭DescribeJob配 AWS 開發套件或 CLI 使用
- 搭DescribeVault配 AWS 開發套件或 CLI 使用
- 搭GetJobOutput配 AWS 開發套件或 CLI 使用
- 搭GetVaultNotifications配 AWS 開發套件或 CLI 使用
- 搭InitiateJob配 AWS 開發套件或 CLI 使用
- 搭ListJobs配 AWS 開發套件或 CLI 使用
- 搭ListTagsForVault配 AWS 開發套件或 CLI 使用
- 搭ListVaults配 AWS 開發套件或 CLI 使用
- 搭SetVaultNotifications配 AWS 開發套件或 CLI 使用
- 搭UploadArchive配 AWS 開發套件或 CLI 使用
- 搭UploadMultipartPart配 AWS 開發套件或 CLI 使用

搭AddTagsToVault配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用AddTagsToVault。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Add tags to the items in an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to add tags to.</param>
    /// <param name="key">The name of the object to tag.</param>
    /// <param name="value">The tag value to add.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddTagsToVaultAsync(string vaultName, string key,
string value)
    {
        var request = new AddTagsToVaultRequest
        {
            Tags = new Dictionary<string, string>
                {
                    { key, value },
                },
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.AddTagsToVaultAsync(request);
        return response.HttpStatusCode == HttpStatusCode.NoContent;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考AddTagsToVault中的。

CLI

AWS CLI

以下命令會將兩個標籤加入名為 my-vault 的文件庫：

```
aws glacier add-tags-to-vault --account-id - --vault-name my-vault --tags
 id=1234,date=july2015
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考AddTagsToVault中的。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭CreateVault配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用CreateVault。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// Create an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to create.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
```

```
    public async Task<bool> CreateVaultAsync(string vaultName)
    {
        var request = new CreateVaultRequest
        {
            // Setting the AccountId to "-" means that
            // the account associated with the current
            // account will be used.
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.CreateVaultAsync(request);

        Console.WriteLine($"Created {vaultName} at: {response.Location}");

        return response.HttpStatusCode == HttpStatusCode.Created;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考CreateVault中的。

CLI

AWS CLI

以下命令建立一個名為 my-vault 的文件庫：

```
aws glacier create-vault --vault-name my-vault --account-id -
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考CreateVault中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.CreateVaultRequest;
import software.amazon.awssdk.services.glacier.model.CreateVaultResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;

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

                Usage:    <vaultName>

                Where:
                   vaultName - The name of the vault to create.

                """;

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

        String vaultName = args[0];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createGlacierVault(glacier, vaultName);
        glacier.close();
    }

    public static void createGlacierVault(GlacierClient glacier, String
 vaultName) {
        try {
```

```
            CreateVaultRequest vaultRequest = CreateVaultRequest.builder()
                    .vaultName(vaultName)
                    .build();

            CreateVaultResponse createVaultResult =
 glacier.createVault(vaultRequest);
            System.out.println("The URI of the new vault is " +
 createVaultResult.location());

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考CreateVault中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

建立用戶端。

```
const { GlacierClient } = require("@aws-sdk/client-glacier");
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const glacierClient = new GlacierClient({ region: REGION });
export { glacierClient };
```

建立保存庫。

```
// Load the SDK for JavaScript
import { CreateVaultCommand } from "@aws-sdk/client-glacier";
import { glacierClient } from "./libs/glacierClient.js";

// Set the parameters
const vaultname = "VAULT_NAME"; // VAULT_NAME
const params = { vaultName: vaultname };

const run = async () => {
  try {
    const data = await glacierClient.send(new CreateVaultCommand(params));
    console.log("Success, vault created!");
    return data; // For unit tests.
  } catch (err) {
    console.log("Error");
  }
};
run();
```

- 如需詳細資訊，請參閱《AWS SDK for JavaScript 開發人員指南》。
- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考CreateVault中的。

適用於 JavaScript (v2) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the SDK for JavaScript
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create a new service object
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" });
// Call Glacier to create the vault
glacier.createVault({ vaultName: "YOUR_VAULT_NAME" }, function (err) {
  if (!err) {
    console.log("Created vault!");
```

```
  }
});
```

- 如需詳細資訊，請參閱《AWS SDK for JavaScript 開發人員指南》。
- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考CreateVault中的。

PowerShell

適用的工具 PowerShell

範例 1：為使用者的帳戶建立新儲存庫。由於沒有提供任何值給-AccountId 參數，指令程式會使用預設值「-」來表示目前帳戶。

```
New-GLCVault -VaultName myvault
```

輸出：

```
/01234567812/vaults/myvault
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程CreateVault式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
```

```
        self.glacier_resource = glacier_resource


    def create_vault(self, vault_name):
        """
        Creates a vault.

        :param vault_name: The name to give the vault.
        :return: The newly created vault.
        """
        try:
            vault = self.glacier_resource.create_vault(vaultName=vault_name)
            logger.info("Created vault %s.", vault_name)
        except ClientError:
            logger.exception("Couldn't create vault %s.", vault_name)
            raise
        else:
            return vault
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件CreateVault中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteArchive配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteArchive。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 取得封存內容並刪除封存

CLI

AWS CLI

從文件庫刪除封存

下列 delete-archive 範例會從 example_vault 中移除指定的封存。

```
aws glacier delete-archive \
    --account-id 111122223333  \
    --vault-name example_vault \
    --archive-id Sc0u9ZP8yaWkmh-XGlIvAVprtLhaLCGnNwNl5I5x9HqPIkX5mjc0DrId3Ln-
Gi_k2HzmlIDZUz117KSdVMdMXLuFWi9PJUitxWO73edQ43eTlMWkH0pd9zVSAuV_XXZBVhKhyGhJ7w
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteArchive中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DeleteArchiveRequest;
import software.amazon.awssdk.services.glacier.model.GlacierException;

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

                Usage:    <vaultName> <accountId> <archiveId>

                Where:
```

```
                    vaultName - The name of the vault that contains the archive to
 delete.
                    accountId - The account ID value.
                    archiveId - The archive ID value.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        String accountId = args[1];
        String archiveId = args[2];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteGlacierArchive(glacier, vaultName, accountId, archiveId);
        glacier.close();
    }

    public static void deleteGlacierArchive(GlacierClient glacier, String
 vaultName, String accountId,
            String archiveId) {
        try {
            DeleteArchiveRequest delArcRequest = DeleteArchiveRequest.builder()
                    .vaultName(vaultName)
                    .accountId(accountId)
                    .archiveId(archiveId)
                    .build();

            glacier.deleteArchive(delArcRequest);
            System.out.println("The archive was deleted.");

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考DeleteArchive中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def delete_archive(archive):
        """
        Deletes an archive from a vault.

        :param archive: The archive to delete.
        """
        try:
            archive.delete()
            logger.info(
                "Deleted archive %s from vault %s.", archive.id,
 archive.vault_name
            )
        except ClientError:
            logger.exception("Couldn't delete archive %s.", archive.id)
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteArchive中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVault配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVault。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 取得封存內容並刪除封存

CLI

AWS CLI

以下命令刪除一個名為 my-vault 的文件庫：

```
aws glacier delete-vault --vault-name my-vault --account-id -
```

此命令不會產生任何輸出。Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVault中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DeleteVaultRequest;
import software.amazon.awssdk.services.glacier.model.GlacierException;

/**
```

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

        final String usage = """

                Usage:    <vaultName>

                Where:
                   vaultName - The name of the vault to delete.\s
                """;

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

        String vaultName = args[0];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        deleteGlacierVault(glacier, vaultName);
        glacier.close();
    }

    public static void deleteGlacierVault(GlacierClient glacier, String
 vaultName) {
        try {
            DeleteVaultRequest delVaultRequest = DeleteVaultRequest.builder()
                    .vaultName(vaultName)
                    .build();

            glacier.deleteVault(delVaultRequest);
            System.out.println("The vault was deleted!");

        } catch (GlacierException e) {
```

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考DeleteVault中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def delete_vault(vault):
        """
        Deletes a vault.

        :param vault: The vault to delete.
        """
        try:
            vault.delete()
            logger.info("Deleted vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't delete vault %s.", vault.name)
            raise
```

```
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteVault中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DeleteVaultNotifications配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DeleteVaultNotifications。

CLI

AWS CLI

若要移除文件庫的 SNS 通知

下列 delete-vault-notifications 範例會在指定的文件庫移除 Amazon Simple Notification Service (Amazon SNS) 傳送之通知。

```
aws glacier delete-vault-notifications \
    --account-id 111122223333 \
    --vault-name example_vault
```

此命令不會產生輸出。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DeleteVaultNotifications中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
```

```
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def stop_notifications(notification):
        """
        Stops notifications to the configured Amazon SNS topic.

        :param notification: The notification configuration to remove.
        """
        try:
            notification.delete()
            logger.info("Notifications stopped.")
        except ClientError:
            logger.exception("Couldn't stop notifications.")
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DeleteVaultNotifications中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭DescribeJob配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeJob。

CLI

AWS CLI

以下命令會擷取文件庫上名為 my-vault 的庫存擷取工作相關資訊：

```
aws glacier describe-job --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW
```

輸出：

```
{
    "InventoryRetrievalParameters": {
        "Format": "JSON"
    },
    "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-vault",
    "Completed": false,
    "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "Action": "InventoryRetrieval",
    "CreationDate": "2015-07-17T20:23:41.616Z",
    "StatusCode": "InProgress"
}
```

工作 ID 可以在 `aws glacier initiate-job` 和 `aws glacier list-jobs` 的輸出中找到。Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeJob中的。

PowerShell

適用的工具 PowerShell

範例 1：傳回指定工作的詳細資訊。當工作順利完成時，Read-GC JobOutput 指令程式可用來將工作的內容 (封存或詳細目錄清單) 擷取至本機檔案系統。

```
Get-GLCJob -VaultName myvault -JobId "op1x...JSbthM"
```

輸出：

```
Action                      : ArchiveRetrieval
ArchiveId                   : o9O9j...X-TpIhQJw
ArchiveSHA256TreeHash       : 79f3ea754c02f58...dc57bf4395b
ArchiveSizeInBytes          : 38034480
Completed                   : False
```

```
CompletionDate               : 1/1/0001 12:00:00 AM
CreationDate                 : 12/13/2018 11:00:14 AM
InventoryRetrievalParameters :
InventorySizeInBytes         : 0
JobDescription               :
JobId                        : op1x...JSbthM
JobOutputPath                :
OutputLocation               :
RetrievalByteRange           : 0-38034479
SelectParameters             :
SHA256TreeHash               : 79f3ea754c02f58...dc57bf4395b
SNSTopic                     :
StatusCode                   : InProgress
StatusMessage                :
Tier                         : Standard
VaultARN                     : arn:aws:glacier:us-west-2:012345678912:vaults/test
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程DescribeJob式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def get_job_status(job):
        """
```

```
        Gets the status of a job.

        :param job: The job to query.
        :return: The current status of the job.
        """
        try:
            job.load()
            logger.info(
                "Job %s is performing action %s and has status %s.",
                job.id,
                job.action,
                job.status_code,
            )
        except ClientError:
            logger.exception("Couldn't get status for job %s.", job.id)
            raise
        else:
            return job.status_code
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件DescribeJob中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**DescribeVault**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用DescribeVault。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
```

```
    /// Describe an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to describe.</param>
    /// <returns>The Amazon Resource Name (ARN) of the vault.</returns>
    public async Task<string> DescribeVaultAsync(string vaultName)
    {
        var request = new DescribeVaultRequest
        {
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.DescribeVaultAsync(request);

        // Display the information about the vault.
        Console.WriteLine($"{response.VaultName}\tARN: {response.VaultARN}");
        Console.WriteLine($"Created on: {response.CreationDate}\tNumber
of Archives: {response.NumberOfArchives}\tSize (in bytes):
{response.SizeInBytes}");
        if (response.LastInventoryDate != DateTime.MinValue)
        {
            Console.WriteLine($"Last inventory: {response.LastInventoryDate}");
        }

        return response.VaultARN;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考DescribeVault中的。

CLI

AWS CLI

以下命令會擷取名為 `my-vault` 的文件庫資料：

```
aws glacier describe-vault --vault-name my-vault --account-id -
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考DescribeVault中的。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭GetJobOutput配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用GetJobOutput。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 取得封存內容並刪除封存

CLI

AWS CLI

以下命令會將文件庫庫存任務的輸出儲存到目前目錄名為 output.json 的檔案中：

```
aws glacier get-job-output --account-id - --vault-name my-
vault --job-id zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW output.json
```

job-id 可在 aws glacier list-jobs 的輸出中使用。請注意，輸出檔案名稱是位置引數，不以選項名稱作為前綴。Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

輸出：

```
{
    "status": 200,
    "acceptRanges": "bytes",
    "contentType": "application/json"
}
```

output.json:

```
{"VaultARN":"arn:aws:glacier:us-west-2:0123456789012:vaults/
my-vault","InventoryDate":"2015-04-07T00:26:18Z","ArchiveList":
[{"ArchiveId":"kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
```

```
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw","ArchiveDescription":"multipart
 upload
 test","CreationDate":"2015-04-06T22:24:34Z","Size":3145728,"SHA256TreeHash":"9628195fcdb
```

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考GetJobOutput中的。

PowerShell

適用的工具 PowerShell

範例 1：下載排程在指定工作中擷取的封存內容，並將內容儲存到磁碟上的檔案中。下載會為您驗證總和檢查碼 (如果有的話)。如果需要，可以從服務響應歷史記錄中獲取校驗和，如下所示（假設此 cmdlet 是上次運行）：。**$AWSHistory.LastServiceResponse**如果指令程式不是最近執行的，請檢查**$AWSHistory.Commands**集合以取得相關的服務回應。

```
Read-GLCJobOutput -VaultName myvault -JobId "HSWjArc...Zq2XLiW" -FilePath "c:
\temp\blue.bin"
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程GetJobOutput式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource
```

```
    @staticmethod
    def get_job_output(job):
        """
        Gets the output of a job, such as a vault inventory or the contents of an
        archive.

        :param job: The job to get output from.
        :return: The job output, in bytes.
        """
        try:
            response = job.get_output()
            out_bytes = response["body"].read()
            logger.info("Read %s bytes from job %s.", len(out_bytes), job.id)
            if "archiveDescription" in response:
                logger.info(
                    "These bytes are described as '%s'",
response["archiveDescription"]
                )
        except ClientError:
            logger.exception("Couldn't get output for job %s.", job.id)
            raise
        else:
            return out_bytes
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件GetJobOutput中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭GetVaultNotifications配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用GetVaultNotifications。

CLI

AWS CLI

以下命令會取得名為 my-vault 的文件庫通知組態描述：

```
aws glacier get-vault-notifications --account-id - --vault-name my-vault
```

輸出：

```
{
    "vaultNotificationConfig": {
        "Events": [
            "InventoryRetrievalCompleted",
            "ArchiveRetrievalCompleted"
        ],
        "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault"
    }
}
```

如果尚未為文件庫設定任何通知，則會傳回錯誤。Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考GetVaultNotifications中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
```

```
    def get_notification(vault):
        """
        Gets the currently notification configuration for a vault.

        :param vault: The vault to query.
        :return: The notification configuration for the specified vault.
        """
        try:
            notification = vault.Notification()
            logger.info(
                "Vault %s notifies %s on %s events.",
                vault.name,
                notification.sns_topic,
                notification.events,
            )
        except ClientError:
            logger.exception("Couldn't get notification data for %s.",
vault.name)
            raise
        else:
            return notification
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件GetVaultNotifications中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭InitiateJob配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用InitiateJob。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

從資料保險箱擷取歸檔。此範例使用 ArchiveTransferManager 類別。如需 API 詳細資訊，請參閱 ArchiveTransferManager

```
    /// <summary>
    /// Download an archive from an Amazon S3 Glacier vault using the Archive
    /// Transfer Manager.
    /// </summary>
    /// <param name="vaultName">The name of the vault containing the object.</
param>
    /// <param name="archiveId">The Id of the archive to download.</param>
    /// <param name="localFilePath">The local directory where the file will
    /// be stored after download.</param>
    /// <returns>Async Task.</returns>
    public async Task<bool> DownloadArchiveWithArchiveManagerAsync(string
 vaultName, string archiveId, string localFilePath)
    {
        try
        {
            var manager = new ArchiveTransferManager(_glacierService);

            var options = new DownloadOptions
            {
                StreamTransferProgress = Progress!,
            };

            // Download an archive.
            Console.WriteLine("Initiating the archive retrieval job and then
 polling SQS queue for the archive to be available.");
            Console.WriteLine("When the archive is available, downloading will
 begin.");
            await manager.DownloadAsync(vaultName, archiveId, localFilePath,
 options);
```

```
            return true;
        }
        catch (AmazonGlacierException ex)
        {
            Console.WriteLine(ex.Message);
            return false;
        }
    }

    /// <summary>
    /// Event handler to track the progress of the Archive Transfer Manager.
    /// </summary>
    /// <param name="sender">The object that raised the event.</param>
    /// <param name="args">The argument values from the object that raised the
    /// event.</param>
    static void Progress(object sender, StreamTransferProgressArgs args)
    {
        if (args.PercentDone != _currentPercentage)
        {
            _currentPercentage = args.PercentDone;
            Console.WriteLine($"Downloaded {_currentPercentage}%");
        }
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考InitiateJob中的。

CLI

AWS CLI

下列指令會啟動工作以取得 Vault `my-vault` 的詳細目錄：

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
 '{"Type": "inventory-retrieval"}'
```

輸出：

```
{
```

```
    "location": "/0123456789012/vaults/my-vault/jobs/
zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
    "jobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW"
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

下列指令會啟動從 Vault `my-vault` 擷取歸檔的工作：

```
aws glacier initiate-job --account-id - --vault-name my-vault --job-parameters
 file://job-archive-retrieval.json
```

`job-archive-retrieval.json`是本機資料夾中的 JSON 檔案，用來指定工作類型、封存 ID 和一些選用參數：

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
  "Description": "Retrieve archive on 2015-07-17",
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-topic"
}
```

歸檔 ID 可在`aws glacier upload-archive`和的輸出中使用`aws glacier get-job-output`。

輸出：

```
{
    "location": "/011685312445/vaults/mwunderl/jobs/l7IL5-
EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
    "jobId": "l7IL5-EkXy2O5uLYaFdAYOiEY9Ws95fClzIbk-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav"
}
```

如需有關 Job 務參數格式的詳細資訊，請參閱 Amazon Glacier API 參考中的啟動任務。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考InitiateJob中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

擷取儲存庫庫庫存。

```
import software.amazon.awssdk.core.ResponseBytes;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.JobParameters;
import software.amazon.awssdk.services.glacier.model.InitiateJobResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import software.amazon.awssdk.services.glacier.model.InitiateJobRequest;
import software.amazon.awssdk.services.glacier.model.DescribeJobRequest;
import software.amazon.awssdk.services.glacier.model.DescribeJobResponse;
import software.amazon.awssdk.services.glacier.model.GetJobOutputRequest;
import software.amazon.awssdk.services.glacier.model.GetJobOutputResponse;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;

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

        final String usage = """

                Usage:    <vaultName> <accountId> <path>
```

```
                Where:
                   vaultName - The name of the vault.
                   accountId - The account ID value.
                   path - The path where the file is written to.
                """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        String vaultName = args[0];
        String accountId = args[1];
        String path = args[2];
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String jobNum = createJob(glacier, vaultName, accountId);
        checkJob(glacier, jobNum, vaultName, accountId, path);
        glacier.close();
    }

    public static String createJob(GlacierClient glacier, String vaultName,
String accountId) {
        try {
            JobParameters job = JobParameters.builder()
                    .type("inventory-retrieval")
                    .build();

            InitiateJobRequest initJob = InitiateJobRequest.builder()
                    .jobParameters(job)
                    .accountId(accountId)
                    .vaultName(vaultName)
                    .build();

            InitiateJobResponse response = glacier.initiateJob(initJob);
            System.out.println("The job ID is: " + response.jobId());
            System.out.println("The relative URI path of the job is: " +
response.location());
            return response.jobId();

        } catch (GlacierException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);

        }
        return "";
    }

    // Poll S3 Glacier = Polling a Job may take 4-6 hours according to the
    // Documentation.
    public static void checkJob(GlacierClient glacier, String jobId, String name,
String account, String path) {
        try {
            boolean finished = false;
            String jobStatus;
            int yy = 0;

            while (!finished) {
                DescribeJobRequest jobRequest = DescribeJobRequest.builder()
                        .jobId(jobId)
                        .accountId(account)
                        .vaultName(name)
                        .build();

                DescribeJobResponse response = glacier.describeJob(jobRequest);
                jobStatus = response.statusCodeAsString();

                if (jobStatus.compareTo("Succeeded") == 0)
                    finished = true;
                else {
                    System.out.println(yy + " status is: " + jobStatus);
                    Thread.sleep(1000);
                }
                yy++;
            }

            System.out.println("Job has Succeeded");
            GetJobOutputRequest jobOutputRequest = GetJobOutputRequest.builder()
                    .jobId(jobId)
                    .vaultName(name)
                    .accountId(account)
                    .build();

            ResponseBytes<GetJobOutputResponse> objectBytes =
glacier.getJobOutputAsBytes(jobOutputRequest);
```

```
            // Write the data to a local file.
            byte[] data = objectBytes.asByteArray();
            File myFile = new File(path);
            OutputStream os = new FileOutputStream(myFile);
            os.write(data);
            System.out.println("Successfully obtained bytes from a Glacier
 vault");
            os.close();

        } catch (GlacierException | InterruptedException | IOException e) {
            System.out.println(e.getMessage());
            System.exit(1);

        }
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考InitiateJob中的。

PowerShell

適用的工具 PowerShell

範例 1：啟動工作以從使用者擁有的指定儲存庫擷取歸檔。您可以使用 Get-GlcJob 指令程式來檢查工作的狀態。當工作順利完成時，Read-GC JobOutput 指令程式可用來將封存的內容擷取至本機檔案系統。

```
Start-GLCJob -VaultName myvault -JobType "archive-retrieval" -JobDescription
 "archive retrieval" -ArchiveId "o9O9j...TX-TpIhQJw"
```

輸出：

```
JobId            JobOutputPath Location
-----            ------------- --------
op1x...JSbthM                  /012345678912/vaults/test/jobs/
op1xe...I4HqCHkSJSbthM
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程InitiateJob式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

擷取儲存庫庫庫存。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def initiate_inventory_retrieval(vault):
        """
        Initiates an inventory retrieval job. The inventory describes the
 contents
        of the vault. Standard retrievals typically complete within 3–5 hours.
        When the job completes, you can get the inventory by calling
 get_output().

        :param vault: The vault to inventory.
        :return: The inventory retrieval job.
        """
        try:
            job = vault.initiate_inventory_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on vault %s.", vault.name)
            raise
        else:
            return job
```

從資料保險箱擷取歸檔。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def initiate_archive_retrieval(archive):
        """
        Initiates an archive retrieval job. Standard retrievals typically
 complete
        within 3–5 hours. When the job completes, you can get the archive
 contents
        by calling get_output().

        :param archive: The archive to retrieve.
        :return: The archive retrieval job.
        """
        try:
            job = archive.initiate_archive_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on archive %s.", archive.id)
            raise
        else:
            return job
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件InitiateJob中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ListJobs配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ListJobs。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作
- 取得封存內容並刪除封存

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// List Amazon S3 Glacier jobs.
    /// </summary>
    /// <param name="vaultName">The name of the vault to list jobs for.</param>
    /// <returns>A list of Amazon S3 Glacier jobs.</returns>
    public async Task<List<GlacierJobDescription>> ListJobsAsync(string
vaultName)
    {
        var request = new ListJobsRequest
        {
            // Using a hyphen "-" for the Account Id will
            // cause the SDK to use the Account Id associated
            // with the current account.
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.ListJobsAsync(request);

        return response.JobList;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考ListJobs中的。

CLI

AWS CLI

以下命令會列出名為 my-vault 的文件庫正在進行和最近完成之工作：

```
aws glacier list-jobs --account-id - --vault-name my-vault
```

輸出：

```
{
    "JobList": [
        {
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "RetrievalByteRange": "0-3145727",
            "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
            "Completed": false,
            "SHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
            "JobId": "l7IL5-EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
            "ArchiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--
zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
            "JobDescription": "Retrieve archive on 2015-07-17",
            "ArchiveSizeInBytes": 3145728,
            "Action": "ArchiveRetrieval",
            "ArchiveSHA256TreeHash":
 "9628195fcdbcbbe76cdde932d4646fa7de5f219fb39823836d81f0cc0e18aa67",
            "CreationDate": "2015-07-17T21:16:13.840Z",
            "StatusCode": "InProgress"
        },
        {
            "InventoryRetrievalParameters": {
                "Format": "JSON"
            },
```

```
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "Completed": false,
            "JobId": "zbxcm3Z_3z5UkoroF7SuZKrxgGoDc3RloGduS7Eg-
RO47Yc6FxsdGBgf_Q2DK5Ejh18CnTS5XW4_XqlNHS61dsO4CnMW",
            "Action": "InventoryRetrieval",
            "CreationDate": "2015-07-17T20:23:41.616Z",
            "StatusCode": ""InProgress""
        }
    ]
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ListJobs中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
```

```
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件ListJobs中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭**ListTagsForVault**配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ListTagsForVault。

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
    /// <summary>
    /// List tags for an Amazon S3 Glacier vault.
    /// </summary>
    /// <param name="vaultName">The name of the vault to list tags for.</param>
    /// <returns>A dictionary listing the tags attached to each object in the
    /// vault and its tags.</returns>
    public async Task<Dictionary<string, string>> ListTagsForVaultAsync(string
vaultName)
    {
        var request = new ListTagsForVaultRequest
        {
            // Using a hyphen "-" for the Account Id will
            // cause the SDK to use the Account Id associated
            // with the default user.
            AccountId = "-",
            VaultName = vaultName,
        };

        var response = await _glacierService.ListTagsForVaultAsync(request);

        return response.Tags;
    }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考ListTagsForVault中的。

CLI

AWS CLI

下列命令會列出套用到 my-vault 文件庫的標籤：

```
aws glacier list-tags-for-vault --account-id - --vault-name my-vault
```

輸出：

```
{
    "Tags": {
        "date": "july2015",
        "id": "1234"
    }
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ListTagsForVault中的。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭ListVaults配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用ListVaults。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
/// <summary>
/// List the Amazon S3 Glacier vaults associated with the current account.
/// </summary>
/// <returns>A list containing information about each vault.</returns>
public async Task<List<DescribeVaultOutput>> ListVaultsAsync()
{
    var glacierVaultPaginator = _glacierService.Paginators.ListVaults(
        new ListVaultsRequest { AccountId = "-" });
    var vaultList = new List<DescribeVaultOutput>();

    await foreach (var vault in glacierVaultPaginator.VaultList)
    {
        vaultList.Add(vault);
    }

    return vaultList;
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考ListVaults中的。

CLI

AWS CLI

下列命令列出預設帳戶與區域的文件庫。

```
aws glacier list-vaults --account-id -
```

輸出：

```
{
    "VaultList": [
        {
            "SizeInBytes": 3178496,
            "VaultARN": "arn:aws:glacier:us-west-2:0123456789012:vaults/my-
vault",
            "LastInventoryDate": "2015-04-07T00:26:19.028Z",
            "VaultName": "my-vault",
            "NumberOfArchives": 1,
            "CreationDate": "2015-04-06T21:23:45.708Z"
        }
    ]
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考ListVaults中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.model.ListVaultsRequest;
import software.amazon.awssdk.services.glacier.model.ListVaultsResponse;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.DescribeVaultOutput;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.util.List;

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

```
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListVaults {
    public static void main(String[] args) {
        GlacierClient glacier = GlacierClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllVault(glacier);
        glacier.close();
    }

    public static void listAllVault(GlacierClient glacier) {
        boolean listComplete = false;
        String newMarker = null;
        int totalVaults = 0;
        System.out.println("Your Amazon Glacier vaults:");
        try {
            while (!listComplete) {
                ListVaultsResponse response = null;
                if (newMarker != null) {
                    ListVaultsRequest request = ListVaultsRequest.builder()
                            .marker(newMarker)
                            .build();

                    response = glacier.listVaults(request);
                } else {
                    ListVaultsRequest request = ListVaultsRequest.builder()
                            .build();
                    response = glacier.listVaults(request);
                }

                List<DescribeVaultOutput> vaultList = response.vaultList();
                for (DescribeVaultOutput v : vaultList) {
                    totalVaults += 1;
                    System.out.println("* " + v.vaultName());
                }

                // Check for further results.
                newMarker = response.marker();
                if (newMarker == null) {
                    listComplete = true;
```

```
                }
            }

            if (totalVaults == 0) {
                System.out.println("No vaults found.");
            }

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

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考ListVaults中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def list_vaults(self):
        """
        Lists vaults for the current account.
        """
        try:
```

```
            for vault in self.glacier_resource.vaults.all():
                logger.info("Got vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't list vaults.")
            raise
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件ListVaults中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭SetVaultNotifications配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用SetVaultNotifications。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作

CLI

AWS CLI

下列命令可為名為 my-vault 的文件庫設定 SNS 通知：

```
aws glacier set-vault-notifications --account-id - --vault-name my-vault --vault-
notification-config file://notificationconfig.json
```

notificationconfig.json 是目前資料夾中的 JSON 檔案，用來指定 SNS 主題和要發佈的事件：

```
{
  "SNSTopic": "arn:aws:sns:us-west-2:0123456789012:my-vault",
  "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"]
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考SetVaultNotifications中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def set_notifications(self, vault, sns_topic_arn):
        """
        Sets an Amazon Simple Notification Service (Amazon SNS) topic as a target
        for notifications. Amazon S3 Glacier publishes messages to this topic for
        the configured list of events.

        :param vault: The vault to set up to publish notifications.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of the topic that
                              receives notifications.
        :return: Data about the new notification configuration.
        """
        try:
            notification = self.glacier_resource.Notification("-", vault.name)
            notification.set(
                vaultNotificationConfig={
                    "SNSTopic": sns_topic_arn,
                    "Events": [
                        "ArchiveRetrievalCompleted",
                        "InventoryRetrievalCompleted",
                    ],
```

```
                }
            )
            logger.info(
                "Notifications will be sent to %s for events %s from %s.",
                notification.sns_topic,
                notification.events,
                notification.vault_name,
            )
        except ClientError:
            logger.exception(
                "Couldn't set notifications to %s on %s.", sns_topic_arn,
vault.name
            )
            raise
        else:
            return notification
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件SetVaultNotifications中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭UploadArchive配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用UploadArchive。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- 封存檔案、取得通知並啟動工作

.NET

AWS SDK for .NET

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
   /// <summary>
   /// Upload an object to an Amazon S3 Glacier vault.
   /// </summary>
   /// <param name="vaultName">The name of the Amazon S3 Glacier vault to upload
   /// the archive to.</param>
   /// <param name="archiveFilePath">The file path of the archive to upload to
the vault.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<string> UploadArchiveWithArchiveManager(string vaultName,
string archiveFilePath)
   {
       try
       {
           var manager = new ArchiveTransferManager(_glacierService);

           // Upload an archive.
           var response = await manager.UploadAsync(vaultName, "upload archive
test", archiveFilePath);
           return response.ArchiveId;
       }
       catch (AmazonGlacierException ex)
       {
           Console.WriteLine(ex.Message);
           return string.Empty;
       }
   }
```

- 如需 API 詳細資訊，請參閱 AWS SDK for .NET API 參考UploadArchive中的。

CLI

AWS CLI

以下命令會將目前 archive.zip 資料夾中的封存上傳至名為 my-vault 的文件庫：

```
aws glacier upload-archive --account-id - --vault-name my-vault --body
 archive.zip
```

輸出：

```
{
    "archiveId": "kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--
zM_mw6k76ZFGEIWQX-ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw",
    "checksum":
 "969fb39823836d81f0cc028195fcdbcbbe76cdde932d4646fa7de5f21e18aa67",
    "location": "/0123456789012/vaults/my-vault/archives/
kKB7ymWJVpPSwhGP6ycSOAekp9ZYe_--zM_mw6k76ZFGEIWQX-
ybtRDvc2VkPSDtfKmQrj0IRQLSGsNuDp-
AJVlu2ccmDSyDUmZwKbwbpAdGATGDiB3hHO0bjbGehXTcApVud_wyDw"
}
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

若要擷取上傳的封存，請使用 Amazon Glacier 啟動任務命令來啟動擷取任務。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考UploadArchive中的。

Java

適用於 Java 2.x 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glacier.GlacierClient;
import software.amazon.awssdk.services.glacier.model.UploadArchiveRequest;
```

```
import software.amazon.awssdk.services.glacier.model.UploadArchiveResponse;
import software.amazon.awssdk.services.glacier.model.GlacierException;
import java.io.File;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

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

    static final int ONE_MB = 1024 * 1024;

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

                Usage:   <strPath> <vaultName>\s

                Where:
                   strPath - The path to the archive to upload (for example, C:\
\AWS\\test.pdf).
                   vaultName - The name of the vault.
                """;

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

        String strPath = args[0];
        String vaultName = args[1];
        File myFile = new File(strPath);
        Path path = Paths.get(strPath);
        GlacierClient glacier = GlacierClient.builder()
```

```
                .region(Region.US_EAST_1)
                .build();

        String archiveId = uploadContent(glacier, path, vaultName, myFile);
        System.out.println("The ID of the archived item is " + archiveId);
        glacier.close();
    }

    public static String uploadContent(GlacierClient glacier, Path path, String
vaultName, File myFile) {
        // Get an SHA-256 tree hash value.
        String checkVal = computeSHA256(myFile);
        try {
            UploadArchiveRequest uploadRequest = UploadArchiveRequest.builder()
                    .vaultName(vaultName)
                    .checksum(checkVal)
                    .build();

            UploadArchiveResponse res = glacier.uploadArchive(uploadRequest,
path);
            return res.archiveId();

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

    private static String computeSHA256(File inputFile) {
        try {
            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 tree hash = %s\n", toHex(treeHash));
            return toHex(treeHash);

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s",
inputFile, ioe.getMessage());
            System.exit(-1);

        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256:
%s", nsae.getMessage());
            System.exit(-1);
```

```
        }
        return "";
    }

    public static byte[] computeSHA256TreeHash(File inputFile) throws
IOException,
            NoSuchAlgorithmException {

        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }

    /**
     * Computes an SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk, even if it's smaller than 1 MB.
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
            int idx = 0;

            while ((bytesRead = fileStream.read(buff, 0, ONE_MB)) > 0) {
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
            }
```

```
            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.length > 1) {
            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];
            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining.
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes.
                    md.reset();
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();
```

```
                } else { // Take care of the remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }

        return prevLvlHashes[0];
    }

    /**
     * Returns the hexadecimal representation of the input byte array
     */
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);
        for (byte datum : data) {
            String hex = Integer.toHexString(datum & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

- 如需 API 詳細資訊，請參閱 AWS SDK for Java 2.x API 參考UploadArchive中的。

JavaScript

適用於 JavaScript (v3) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

建立用戶端。

```
const { GlacierClient } = require("@aws-sdk/client-glacier");
// Set the AWS Region.
const REGION = "REGION";
//Set the Redshift Service Object
const glacierClient = new GlacierClient({ region: REGION });
export { glacierClient };
```

上傳封存。

```
// Load the SDK for JavaScript
import { UploadArchiveCommand } from "@aws-sdk/client-glacier";
import { glacierClient } from "./libs/glacierClient.js";

// Set the parameters
const vaultname = "VAULT_NAME"; // VAULT_NAME

// Create a new service object and buffer
const buffer = new Buffer.alloc(2.5 * 1024 * 1024); // 2.5MB buffer
const params = { vaultName: vaultname, body: buffer };

const run = async () => {
  try {
    const data = await glacierClient.send(new UploadArchiveCommand(params));
    console.log("Archive ID", data.archiveId);
    return data; // For unit tests.
  } catch (err) {
    console.log("Error uploading archive!", err);
  }
};
run();
```

- 如需詳細資訊，請參閱《AWS SDK for JavaScript 開發人員指南》。
- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考UploadArchive中的。

適用於 JavaScript (v2) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
// Load the SDK for JavaScript
var AWS = require("aws-sdk");
// Set the region
AWS.config.update({ region: "REGION" });

// Create a new service object and buffer
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" });
buffer = Buffer.alloc(2.5 * 1024 * 1024); // 2.5MB buffer

var params = { vaultName: "YOUR_VAULT_NAME", body: buffer };
// Call Glacier to upload the archive.
glacier.uploadArchive(params, function (err, data) {
  if (err) {
    console.log("Error uploading archive!", err);
  } else {
    console.log("Archive ID", data.archiveId);
  }
});
```

- 如需詳細資訊，請參閱《AWS SDK for JavaScript 開發人員指南》。
- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考UploadArchive中的。

PowerShell

適用的工具 PowerShell

範例 1：將單一檔案上載至指定的儲存庫，傳回歸檔 ID 和計算出來的總和檢查碼。

```
Write-GLCArchive -VaultName myvault -FilePath c:\temp\blue.bin
```

輸出：

```
FilePath                    ArchiveId              Checksum
--------                    ---------              --------
C:\temp\blue.bin            o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
```

範例 2：將資料夾階層的內容上載至使用者帳戶中指定的 Vault。指令程式會針對每個上傳的檔案發出檔案名稱、對應的封存 ID 和計算的封存總和檢查碼。

```
Write-GLCArchive -VaultName myvault -FolderPath . -Recurse
```

輸出：

```
FilePath                    ArchiveId              Checksum
--------                    ---------              --------
C:\temp\blue.bin            o9O9jUUs...TTX-TpIhQJw 79f3e...f4395b
C:\temp\green.bin           qXAfOdSG...czo729UHXrw d50a1...9184b9
C:\temp\lum.bin             39aNifP3...q9nb8nZkFIg 28886...5c3e27
C:\temp\red.bin             vp7E6rU_...Ejk_HhjAxKA e05f7...4e34f5
C:\temp\Folder1\file1.txt   _eRINlip...5Sxy7dD2BaA d0d2a...c8a3ba
C:\temp\Folder2\file2.iso   -Ix3jlmu...iXiDh-XfOPA 7469e...3e86f1
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 指令程UploadArchive式參考中的。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

```
class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
```

```
        self.glacier_resource = glacier_resource


    @staticmethod
    def upload_archive(vault, archive_description, archive_file):
        """
        Uploads an archive to a vault.

        :param vault: The vault where the archive is put.
        :param archive_description: A description of the archive.
        :param archive_file: The archive file to put in the vault.
        :return: The uploaded archive.
        """
        try:
            archive = vault.upload_archive(
                archiveDescription=archive_description, body=archive_file
            )
            logger.info(
                "Uploaded %s with ID %s to vault %s.",
                archive_description,
                archive.id,
                vault.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't upload %s to %s.", archive_description, vault.name
            )
            raise
        else:
            return archive
```

- 如需 API 的詳細資訊，請參閱AWS 開發套件UploadArchive中的 Python (博托 3) API 參考。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭UploadMultipartPart配 AWS 開發套件或 CLI 使用

下列程式碼範例會示範如何使用UploadMultipartPart。

CLI

AWS CLI

下列命令會上傳封存的前 1 MiB (1024 x 1024 位元組) 部分：

```
aws glacier upload-multipart-part --body part1 --range 'bytes
 0-1048575/*' --account-id - --vault-name my-vault --upload-
id 19gaRezEXAMPLES6Ry5YYdqthHOC_kGRCT03L9yetr220UmPtBYKk-
OssZtLqyFu7sY1_lR7vgFuJV6NtcV5zpsJ
```

Amazon Glacier 在執行操作時需要帳戶 ID 引數，但您可以使用連字號來指定使用中的帳戶。

內文參數會取得本機檔案系統上分段檔案的路徑。範圍參數採用 HTTP 內容範圍，指示分段在完成的封存中佔用的位元組。上傳 ID 由 `aws glacier initiate-multipart-upload` 命令傳回，也可以透過 `aws glacier list-multipart-uploads` 獲取。

如需有關使用 CLI 分段上傳至 Amazon Glacier 的詳細資訊，請參閱 AWS CLI 使用者指南中的 AWS 使用 Amazon Glacier。

- 如需 API 詳細資訊，請參閱AWS CLI 命令參考UploadMultipartPart中的。

JavaScript

適用於 JavaScript (v2) 的開發套件

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

建立 Buffer 物件 1 MB 區塊的分段上傳。

```
// Create a new service object and some supporting variables
var glacier = new AWS.Glacier({ apiVersion: "2012-06-01" }),
  vaultName = "YOUR_VAULT_NAME",
  buffer = new Buffer(2.5 * 1024 * 1024), // 2.5MB buffer
  partSize = 1024 * 1024, // 1MB chunks,
  numPartsLeft = Math.ceil(buffer.length / partSize),
  startTime = new Date(),
  params = { vaultName: vaultName, partSize: partSize.toString() };
```

```
// Compute the complete SHA-256 tree hash so we can pass it
// to completeMultipartUpload request at the end
var treeHash = glacier.computeChecksums(buffer).treeHash;

// Initiate the multipart upload
console.log("Initiating upload to", vaultName);
// Call Glacier to initiate the upload.
glacier.initiateMultipartUpload(params, function (mpErr, multipart) {
  if (mpErr) {
    console.log("Error!", mpErr.stack);
    return;
  }
  console.log("Got upload ID", multipart.uploadId);

  // Grab each partSize chunk and upload it as a part
  for (var i = 0; i < buffer.length; i += partSize) {
    var end = Math.min(i + partSize, buffer.length),
      partParams = {
        vaultName: vaultName,
        uploadId: multipart.uploadId,
        range: "bytes " + i + "-" + (end - 1) + "/*",
        body: buffer.slice(i, end),
      };

    // Send a single part
    console.log("Uploading part", i, "=", partParams.range);
    glacier.uploadMultipartPart(partParams, function (multiErr, mData) {
      if (multiErr) return;
      console.log("Completed part", this.request.params.range);
      if (--numPartsLeft > 0) return; // complete only when all parts uploaded

      var doneParams = {
        vaultName: vaultName,
        uploadId: multipart.uploadId,
        archiveSize: buffer.length.toString(),
        checksum: treeHash, // the computed tree hash
      };

      console.log("Completing upload...");
      glacier.completeMultipartUpload(doneParams, function (err, data) {
        if (err) {
          console.log("An error occurred while uploading the archive");
          console.log(err);
```

```
        } else {
          var delta = (new Date() - startTime) / 1000;
          console.log("Completed upload in", delta, "seconds");
          console.log("Archive ID:", data.archiveId);
          console.log("Checksum:  ", data.checksum);
        }
      });
    });
  }
});
```

- 如需詳細資訊，請參閱《AWS SDK for JavaScript 開發人員指南》。
- 如需 API 詳細資訊，請參閱 AWS SDK for JavaScript API 參考UploadMultipartPart中的。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS 開發套件的 S3 冰川案例

下列程式碼範例說明如何使用 AWS SDK 在 S3 Glacier 中實作常見案例。這些案例會向您展示如何呼叫 S3 Glacier 中的多個函數來完成特定任務。每個案例都包含一個連結 GitHub，您可以在其中找到如何設定和執行程式碼的指示。

範例

- 使用開 AWS 發套件將檔案存檔到 Amazon S3 Glacier、取得通知並啟動任務
- 使用 AWS 開發套件取得 Amazon S3 冰川存檔內容並刪除存檔

使用開 AWS 發套件將檔案存檔到 Amazon S3 Glacier、取得通知並啟動任務

以下程式碼範例顯示做法：

- 建立 Amazon S3 Glacier 保存庫。
- 設定保存庫，以將通知發布至 Amazon SNS 主題。
- 將封存檔上傳至保存庫。
- 啟動封存擷取任務。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

建立包裝 S3 Glacier 作業的類別。

```
import argparse
import logging
import os
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    def create_vault(self, vault_name):
        """
        Creates a vault.

        :param vault_name: The name to give the vault.
        :return: The newly created vault.
        """
        try:
            vault = self.glacier_resource.create_vault(vaultName=vault_name)
            logger.info("Created vault %s.", vault_name)
        except ClientError:
            logger.exception("Couldn't create vault %s.", vault_name)
```

```
            raise
        else:
            return vault


    def list_vaults(self):
        """
        Lists vaults for the current account.
        """
        try:
            for vault in self.glacier_resource.vaults.all():
                logger.info("Got vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't list vaults.")
            raise


    @staticmethod
    def upload_archive(vault, archive_description, archive_file):
        """
        Uploads an archive to a vault.

        :param vault: The vault where the archive is put.
        :param archive_description: A description of the archive.
        :param archive_file: The archive file to put in the vault.
        :return: The uploaded archive.
        """
        try:
            archive = vault.upload_archive(
                archiveDescription=archive_description, body=archive_file
            )
            logger.info(
                "Uploaded %s with ID %s to vault %s.",
                archive_description,
                archive.id,
                vault.name,
            )
        except ClientError:
            logger.exception(
                "Couldn't upload %s to %s.", archive_description, vault.name
            )
            raise
        else:
            return archive
```

```
    @staticmethod
    def initiate_archive_retrieval(archive):
        """
        Initiates an archive retrieval job. Standard retrievals typically
complete
        within 3–5 hours. When the job completes, you can get the archive
contents
        by calling get_output().

        :param archive: The archive to retrieve.
        :return: The archive retrieval job.
        """
        try:
            job = archive.initiate_archive_retrieval()
            logger.info("Started %s job with ID %s.", job.action, job.id)
        except ClientError:
            logger.exception("Couldn't start job on archive %s.", archive.id)
            raise
        else:
            return job


    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
```

```
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list


    def set_notifications(self, vault, sns_topic_arn):
        """
        Sets an Amazon Simple Notification Service (Amazon SNS) topic as a target
        for notifications. Amazon S3 Glacier publishes messages to this topic for
        the configured list of events.

        :param vault: The vault to set up to publish notifications.
        :param sns_topic_arn: The Amazon Resource Name (ARN) of the topic that
                              receives notifications.
        :return: Data about the new notification configuration.
        """
        try:
            notification = self.glacier_resource.Notification("-", vault.name)
            notification.set(
                vaultNotificationConfig={
                    "SNSTopic": sns_topic_arn,
                    "Events": [
                        "ArchiveRetrievalCompleted",
                        "InventoryRetrievalCompleted",
                    ],
                }
            )
            logger.info(
                "Notifications will be sent to %s for events %s from %s.",
                notification.sns_topic,
                notification.events,
                notification.vault_name,
            )
        except ClientError:
```

```
            logger.exception(
                "Couldn't set notifications to %s on %s.", sns_topic_arn,
 vault.name
            )
            raise
        else:
            return notification
```

呼叫包裝函式類別上的函數，以建立保存庫並上傳檔案，然後將保存庫設定為發布通知與啟動工作以擷取封存。

```
def upload_demo(glacier, vault_name, topic_arn):
    """
    Shows how to:
    * Create a vault.
    * Configure the vault to publish notifications to an Amazon SNS topic.
    * Upload an archive.
    * Start a job to retrieve the archive.

    :param glacier: A Boto3 Amazon S3 Glacier resource.
    :param vault_name: The name of the vault to create.
    :param topic_arn: The ARN of an Amazon SNS topic that receives notification
 of
                      Amazon S3 Glacier events.
    """
    print(f"\nCreating vault {vault_name}.")
    vault = glacier.create_vault(vault_name)
    print("\nList of vaults in your account:")
    glacier.list_vaults()
    print(f"\nUploading glacier_basics.py to {vault.name}.")
    with open("glacier_basics.py", "rb") as upload_file:
        archive = glacier.upload_archive(vault, "glacier_basics.py", upload_file)
    print(
        "\nStarting an archive retrieval request to get the file back from the "
        "vault."
    )
    glacier.initiate_archive_retrieval(archive)
    print("\nListing in progress jobs:")
    glacier.list_jobs(vault, "in_progress")
    print(
        "\nBecause Amazon S3 Glacier is intended for infrequent retrieval, an "
```

```
            "archive request with Standard retrieval typically completes within 3-5 "
            "hours."
        )
        if topic_arn:
            notification = glacier.set_notifications(vault, topic_arn)
            print(
                f"\nVault {vault.name} is configured to notify the "
                f"{notification.sns_topic} topic when {notification.events} "
                f"events occur. You can subscribe to this topic to receive "
                f"a message when the archive retrieval completes.\n"
            )
        else:
            print(
                f"\nVault {vault.name} is not configured to notify an Amazon SNS
topic "
                f"when the archive retrieval completes so wait a few hours."
            )
        print("\nRetrieve your job output by running this script with the --retrieve
flag.")
```

- 如需 API 的詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考資料》中的下列主題。
  - CreateVault
  - InitiateJob
  - ListJobs
  - ListVaults
  - SetVaultNotifications
  - UploadArchive

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 AWS 開發套件取得 Amazon S3 冰川存檔內容並刪除存檔

以下程式碼範例顯示做法：

- 列出 Amazon S3 Glacier 保存庫的工作，並取得工作狀態。

- 取得已完成封存擷取工作的輸出。
- 刪除封存。
- 刪除保存庫。

Python

適用於 Python (Boto3) 的 SDK

> **Note**
>
> 還有更多關於 GitHub。尋找完整範例，並了解如何在AWS 設定和執行程式碼範例儲存庫。

建立包裝 S3 Glacier 作業的類別。

```
import argparse
import logging
import os
import boto3
from botocore.exceptions import ClientError

logger = logging.getLogger(__name__)


class GlacierWrapper:
    """Encapsulates Amazon S3 Glacier API operations."""

    def __init__(self, glacier_resource):
        """
        :param glacier_resource: A Boto3 Amazon S3 Glacier resource.
        """
        self.glacier_resource = glacier_resource


    @staticmethod
    def list_jobs(vault, job_type):
        """
        Lists jobs by type for the specified vault.

        :param vault: The vault to query.
```

```
        :param job_type: The type of job to list.
        :return: The list of jobs of the requested type.
        """
        job_list = []
        try:
            if job_type == "all":
                jobs = vault.jobs.all()
            elif job_type == "in_progress":
                jobs = vault.jobs_in_progress.all()
            elif job_type == "completed":
                jobs = vault.completed_jobs.all()
            elif job_type == "succeeded":
                jobs = vault.succeeded_jobs.all()
            elif job_type == "failed":
                jobs = vault.failed_jobs.all()
            else:
                jobs = []
                logger.warning("%s isn't a type of job I can get.", job_type)
            for job in jobs:
                job_list.append(job)
                logger.info("Got %s %s job %s.", job_type, job.action, job.id)
        except ClientError:
            logger.exception("Couldn't get %s jobs from %s.", job_type,
vault.name)
            raise
        else:
            return job_list


    @staticmethod
    def get_job_output(job):
        """
        Gets the output of a job, such as a vault inventory or the contents of an
        archive.

        :param job: The job to get output from.
        :return: The job output, in bytes.
        """
        try:
            response = job.get_output()
            out_bytes = response["body"].read()
            logger.info("Read %s bytes from job %s.", len(out_bytes), job.id)
            if "archiveDescription" in response:
                logger.info(
```

```
                    "These bytes are described as '%s'",
response["archiveDescription"]
                )
        except ClientError:
            logger.exception("Couldn't get output for job %s.", job.id)
            raise
        else:
            return out_bytes


    @staticmethod
    def delete_archive(archive):
        """
        Deletes an archive from a vault.

        :param archive: The archive to delete.
        """
        try:
            archive.delete()
            logger.info(
                "Deleted archive %s from vault %s.", archive.id,
archive.vault_name
            )
        except ClientError:
            logger.exception("Couldn't delete archive %s.", archive.id)
            raise


    @staticmethod
    def delete_vault(vault):
        """
        Deletes a vault.

        :param vault: The vault to delete.
        """
        try:
            vault.delete()
            logger.info("Deleted vault %s.", vault.name)
        except ClientError:
            logger.exception("Couldn't delete vault %s.", vault.name)
            raise
```

呼叫包裝函式類別的函數，從已完成的工作取得封存內容，然後刪除封存。

```
def retrieve_demo(glacier, vault_name):
    """
    Shows how to:
    * List jobs for a vault and get job status.
    * Get the output of a completed archive retrieval job.
    * Delete an archive.
    * Delete a vault.

    :param glacier: A Boto3 Amazon S3 Glacier resource.
    :param vault_name: The name of the vault to query for jobs.
    """
    vault = glacier.glacier_resource.Vault("-", vault_name)
    try:
        vault.load()
    except ClientError as err:
        if err.response["Error"]["Code"] == "ResourceNotFoundException":
            print(
                f"\nVault {vault_name} doesn't exist. You must first run this
 script "
                f"with the --upload flag to create the vault."
            )
            return
        else:
            raise

    print(f"\nGetting completed jobs for {vault.name}.")
    jobs = glacier.list_jobs(vault, "completed")
    if not jobs:
        print("\nNo completed jobs found. Give it some time and try again
 later.")
        return

    retrieval_job = None
    for job in jobs:
        if job.action == "ArchiveRetrieval" and job.status_code == "Succeeded":
            retrieval_job = job
            break
    if retrieval_job is None:
        print(
            "\nNo ArchiveRetrieval jobs found. Give it some time and try again "
            "later."
        )
```

```
        return

    print(f"\nGetting output from job {retrieval_job.id}.")
    archive_bytes = glacier.get_job_output(retrieval_job)
    archive_str = archive_bytes.decode("utf-8")
    print("\nGot archive data. Printing the first 10 lines.")
    print(os.linesep.join(archive_str.split(os.linesep)[:10]))

    print(f"\nDeleting the archive from {vault.name}.")
    archive = glacier.glacier_resource.Archive(
        "-", vault.name, retrieval_job.archive_id
    )
    glacier.delete_archive(archive)

    print(f"\nDeleting {vault.name}.")
    glacier.delete_vault(vault)
```

- 如需 API 的詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考資料》中的下列主題。
  - DeleteArchive
  - DeleteVault
  - GetJobOutput
  - ListJobs

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱搭配 AWS 開發套件使用 S3 冰川。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon S3 Glacier 中的安全

雲端安全是 AWS 最重視的一環。身為 AWS 的客戶，您將能從資料中心和網路架構中獲益，這些都是專為最重視安全的組織而設計的。

安全是 AWS 與您共同的責任。共同的責任模型 將此描述為雲端 本身 的安全和雲端 內部 的安全：

- 雲端本身的安全 – AWS 負責保護執行 AWS 雲端 內 AWS 服務的基礎設施。AWS 提供的服務，也可讓您安全使用。第三方稽核人員定期檢測及驗證安全的效率也是我們 AWS 合規計劃的一部分。若要了解適用於 Amazon S3 Glacier (S3 Glacier) 的合規計劃，請參閱合規計劃範圍內的 AWS 服務。
- 雲端內部的安全：您的責任取決於所使用的 AWS 服務。您也必須對資料敏感度、組織要求，以及適用法律和法規等其他因素負責。

本文件會協助您了解使用 S3 Glacier 時共同責任模型的適用情形。下列主題說明如何將 S3 Glacier 設定為達到您的安全及合規目標。您也將了解如何使用可以幫助您監控並保護 S3 Glacier 資源的其他 AWS 服務。

主題

Amazon S3 Glacier 的資料保護

Amazon S3 Glacier (S3 Glacier) 為資料封存和長期備份提供高耐用性的雲端儲存。S3 Glacier 的設計可提供 99.999999999 的耐久性，並提供全面的安全性與合規功能，協助您符合嚴格的法規要求。S3 Glacier 會以冗餘方式將資料存放在多個 AWS 可用區域 (AZ) 和每個可用區域內的多個裝置上。為了增加耐用性，S3 Glacier 會在確認成功上傳之前，將資料同步存放到多個 AZ。

如需 AWS 全球雲端基礎架構的詳細資訊，請參閱全球基礎架構。

基於資料保護目的，我們建議您保護 AWS 帳戶 認證，並僅授予個別使用者、群組或角色履行其工作職責所需的權限。

如果您在透過命令列介面或 API 存取時需要經 AWS 過 FIPS 140-2 驗證的加密模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-2 概觀。

主題

資料加密

資料保護是指在傳輸中 (往返 Amazon S3 Glacier) 和靜態 (儲存在 AWS 資料中心時) 時保護資料。您可以使用 Secure Sockets Layer (SSL) 或用戶端加密，保護直接上傳至 S3 Glacier 之傳輸中的資料。

您也可以透過 Amazon S3 存取 S3 Glacier。Amazon S3 對 Amazon S3 儲存貯體支援生命週期設定，其可讓您將物件轉換為 S3 Glacier 儲存類別以供封存使用。透過生命週期政策，使用 SSL 加密在 Amazon S3 與 S3 Glacier 之間傳輸中的資料。

存放在 S3 Glacier 的靜態資料會自動在伺服器端使用 256 位元進階加密標準 (AES-256) 搭配 AWS 所維護的金鑰進行加密。如果您偏好管理自己的金鑰，也可以先使用用戶端加密，再將資料存放在 S3 Glacier 中。如需如何設定 Amazon S3 預設加密的詳細資訊，請參閱《Amazon Simple Storage Service 開發人員指南》中的 Amazon S3 預設加密。

金鑰管理

伺服器端加密可處理靜態資料加密；亦即，Amazon S3 Glacier 會在將資料寫入資料中心時將其加密，以及在您存取該資料時解密資料。只要您有驗證請求並具備存取許可，存取加密資料或未加密資料的方式並無不同。

存放在 S3 Glacier 的靜態資料會自動在伺服器端使用 AES-256 和 AWS所維護的金鑰進行加密。作為額外的保護措施，請使用我們定期輪換的根金鑰來 AWS 加密金鑰本身。

網際網路流量隱私權

您需要經由 AWS 發布的 API 來透過網路存取 Amazon S3 Glacier。用戶端必須支援 Transport Layer Security (TLS) 1.2。建議使用 TLS 1.3 或更新版本。用戶端也必須支援具備完整轉寄密碼 (PFS) 的密碼套件，例如暫時性 Diffie-Hellman (DHE) 或橢圓曲線 Diffie-Hellman Ephemeral (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。此外，您必須使用存取金鑰 ID，以及與 IAM 委託人相關聯的私密存取金鑰來簽署請求，或者您可以使用 AWS Security Token Service (AWS STS) 來產生臨時安全憑證來簽署請求。

VPC 端點

虛擬私有雲端 (VPC) 端點可讓您將 VPC 私下連線至支援的 AWS 服務以及具有 AWS PrivateLink 功能的 VPC 端點服務，而不需要網際網路閘道、NAT 裝置、VPN 連接或 AWS Direct Connect 連線。雖然 S3 Glacier 不直接支援 VPC 端點，但是如果您將 S3 Glacier 作為與 Amazon S3 整合的儲存層進行存取，則可以利用 Amazon Simple Storage Service (Amazon S3) VPC 端點。

如需有關 Amazon S3 生命週期設定以及將物件轉換到 S3 Glacier 儲存類別的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件生命週期管理和轉換物件。如需 VPC 端點的詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 端點。

Amazon S3 Glacier 的 Identity and Access Management

AWS Identity and Access Management (IAM) 可協助管理員安全地控制 AWS 資源存取權。 AWS 服務 IAM 管理員可以控制誰能完成身分驗證 (登入) 和獲得授權 (取得許可)，而得以使用 S3 Glacier 資源。您可以使用 IAM AWS 服務 ，無需額外付費。

主題

- 物件
- 使用身分驗證
- 使用政策管理存取權
- Amazon S3 Glacier 如何與 IAM 搭配運作
- Amazon S3 Glacier 的身分型政策範例
- Amazon S3 Glacier 的資源型政策範例
- Amazon S3 Glacier 身分識別和存取疑難排解
- API 許可參考

物件

您使用 AWS Identity and Access Management (IAM) 的方式會有所不同，視您在 S3 Glacier 中所做的工作而定。

服務使用者：如果使用 S3 Glacier 服務執行工作，管理員會為您提供所需的憑證和許可。隨著您為了執行作業而使用的 S3 Glacier 功能數量變多，您可能會需要額外的許可。了解存取許可的管理方式可

協助您向管理員請求正確的許可。若您無法存取 S3 Glacier 中的某項功能，請參閱Amazon S3 Glacier 身分識別和存取疑難排解。

服務管理員：如果您負責管理公司內的 S3 Glacier 資源，您可能具備 S3 Glacier 的完整存取權限。您的工作是判斷服務使用者應存取的 S3 Glacier 功能及資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步了解貴公司可搭配 S3 Glacier 使用 IAM 的方式，請參閱Amazon S3 Glacier 如何與 IAM 搭配運作。

IAM 管理員如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 S3 Glacier 存取權的詳細資訊。若要檢視您可以在 IAM 中使用的範例 S3 Glacier 身分型政策，請參閱Amazon S3 Glacier 的身分型政策範例。

使用身分驗證

驗證是您 AWS 使用身分認證登入的方式。您必須以 IAM 使用者身分或假設 IAM 角色進行驗證 (登入 AWS)。 AWS 帳戶根使用者

您可以使用透過 AWS 身分識別來源提供的認證，以聯合身分識別身分登入。 AWS IAM Identity Center (IAM 身分中心) 使用者、貴公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料都是聯合身分識別的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使 AWS 用同盟存取時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需有關登入的詳細資訊 AWS，請參閱《AWS 登入 使用指南》 AWS 帳戶中的如何登入您的。

如果您 AWS 以程式設計方式存取，請 AWS 提供軟體開發套件 (SDK) 和命令列介面 (CLI)，以使用您的認證以加密方式簽署要求。如果您不使用 AWS 工具，則必須自行簽署要求。如需使用建議的方法自行簽署請求的詳細資訊，請參閱 IAM 使用者指南中的簽署 AWS API 請求。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重要素驗證 (MFA) 來增加帳戶的安全性。如需更多資訊，請參閱 AWS IAM Identity Center 使用者指南中的多重要素驗證和 IAM 使用者指南中的在 AWS中使用多重要素驗證 (MFA)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取該帳戶中的所有資源 AWS 服務和資源。此身分稱為 AWS 帳戶 root 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱 IAM 使用者指南中的需要根使用者憑證的任務。

聯合身分

最佳作法是要求人類使用者 (包括需要系統管理員存取權的使用者) 使用與身分識別提供者的同盟，才能使用臨時登入資料進行存取 AWS 服務 。

聯合身分識別是來自企業使用者目錄的使用者、Web 身分識別提供者、Identi ty Center 目錄，或使用透過身分識別來源提供的認證進行存取 AWS 服務 的任何使用者。 AWS Directory Service同盟身分存取時 AWS 帳戶，他們會假設角色，而角色則提供臨時認證。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，也可以連線並同步到自己身分識別來源中的一組使用者和群組，以便在所有應用程式 AWS 帳戶 和應用程式中使用。如需 IAM Identity Center 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者是您內部的身分，具 AWS 帳戶 有單一人員或應用程式的特定許可。建議您盡可能依賴暫時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供暫時憑證。如需進一步了解，請參閱IAM 使用者指南中的建立 IAM 使用者 (而非角色) 的時機。

IAM 角色

IAM 角色是您 AWS 帳戶 內部具有特定許可的身分。它類似 IAM 使用者，但不與特定的人員相關聯。您可以切換角色，在中暫時擔任 IAM 角色。 AWS Management Console 您可以透過呼叫 AWS CLI 或 AWS API 作業或使用自訂 URL 來擔任角色。如需使用角色的方法詳細資訊，請參閱 IAM 使用者指南中的使用 IAM 角色。

使用暫時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱 IAM 使用者指南中的為第三方身分提供者建立角色。如果您使用 IAM Identity Center，

則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。

- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權：您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的主體) 存取您帳戶的資源。角色是授予跨帳戶存取權的主要方式。但是，對於某些策略 AWS 服務，您可以將策略直接附加到資源（而不是使用角色作為代理）。若要了解跨帳戶存取角色和以資源為基礎的政策之間的差異，請參閱 IAM 使用者指南中的 IAM 中的跨帳戶資源存取。
- 跨服務訪問 — 有些 AWS 服務 使用其他 AWS 服務功能。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉寄存取工作階段 (FAS) — 當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
  - 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務服務。
  - 服務連結角色 — 服務連結角色是連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。
- 在 Amazon EC2 上執行的應用程式 — 您可以使用 IAM 角色來管理在 EC2 執行個體上執行的應用程式以及發出 AWS CLI 或 AWS API 請求的臨時登入資料。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並提供給其所有應用程式，請建立連接至執行個體的執行個體設定檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得暫時憑證。如需詳細資訊，請參閱 IAM 使用者指南中的利用 IAM 角色來授予許可給 Amazon EC2 執行個體上執行的應用程式。

如需了解是否要使用 IAM 角色或 IAM 使用者，請參閱 IAM 使用者指南中的建立 IAM 角色 (而非使用者) 的時機。

使用政策管理存取權

您可以透 AWS 過建立原則並將其附加至 AWS 身分識別或資源來控制中的存取。原則是一個物件 AWS ，當與身分識別或資源相關聯時，會定義其權限。 AWS 當主參與者 (使用者、root 使用者或角色

工作階段) 提出要求時，評估這些原則。政策中的許可決定是否允許或拒絕請求。大多數原則會 AWS 以 JSON 文件的形式儲存在中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該原則的使用者可以從 AWS Management Console AWS CLI、或 AWS API 取得角色資訊。

身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱 IAM 使用者指南中的建立 IAM 政策。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管理的策略是獨立策略，您可以將其附加到您的 AWS 帳戶. 受管政策包括 AWS 受管政策和客戶管理的策略。如需了解如何在受管政策及內嵌政策間選擇，請參閱 IAM 使用者指南中的在受管政策和內嵌政策間選擇。

資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

資源型政策是位於該服務中的內嵌政策。您無法在以資源為基礎的政策中使用 IAM 的 AWS 受管政策。

存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 和 Amazon VPC 是支援 ACL 的服務範例。 AWS WAF如需進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

其他政策類型

AWS 支援其他較不常見的原則類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。
- 服務控制策略（SCP）— SCP 是 JSON 策略，用於指定中組織或組織單位（OU）的最大權限。AWS Organizations AWS Organizations 是一種用於分組和集中管理您企業擁 AWS 帳戶 有的多個服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 限制成員帳戶中實體的權限，包括每個 AWS 帳戶根使用者帳戶。如需 Organizations 和 SCP 的詳細資訊，請參閱 AWS Organizations 使用者指南中的 SCP 運作方式。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過編寫程式的方式建立角色或聯合使用者的暫時工作階段時，作為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需詳細資訊，請參閱 IAM 使用者指南中的工作階段政策。

多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。要了解如何在涉及多個政策類型時 AWS 確定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

Amazon S3 Glacier 如何與 IAM 搭配運作

在您使用 IAM 管理 S3 Glacier 的存取權之前，請了解可與 S3 Glacier 搭配使用的 IAM 功能有哪些。

您可以搭配 Amazon S3 Glacier 使用的 IAM 功能

IAM 功能	S3 Glacier 支援
身分型政策	是

IAM 功能	S3 Glacier 支援
資源型政策	是
政策動作	是
政策資源	是
政策條件索引鍵 (服務特定)	是
ACL	否
ABAC(政策中的標籤)	否
臨時憑證	是
主體許可	否
服務角色	否
服務連結角色	否

若要深入瞭解 S3 Glacier 和其他 AWS 服務如何搭配大多數 IAM 功能使用，請參閱 IAM 使用者指南中的搭配 IAM 使用的AWS 服務。

適用於 S3 Glacier 的身分型政策

支援身分型政策	是

身分型政策是可以連接到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。若要了解如何建立身分類型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

S3 Glacier 的身分型政策範例

若要檢視 S3 Glacier 身分型政策範例，請參閱Amazon S3 Glacier 的身分型政策範例。

S3 Glacier 內的資源型政策

支援以資源基礎的政策	是

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。主參與者可以包括帳戶、使用者、角色、同盟使用者或。 AWS 服務

如需啟用跨帳戶存取權，您可以指定在其他帳戶內的所有帳戶或 IAM 實體，作為資源型政策的主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當主體和資源位於不同時 AWS 帳戶，受信任帳戶中的 IAM 管理員也必須授與主體實體 (使用者或角色) 權限，才能存取資源。其透過將身分型政策連接到實體來授與許可。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需詳細資訊，請參閱 IAM 使用者指南中的 IAM 中的跨帳戶資源存取。

S3 Glacier 服務支援一種名為保存庫政策，且已附加到保存庫的資源型政策。此政策定義哪些主體可以對保存庫執行動作。

S3 Glacier 保存庫政策會以下列方式管理許可：

- 在帳戶中以單一保存庫政策 (而不是以個別使用者政策) 管理使用者許可。
- 使用 IAM 角色的替代方式是管理跨帳戶許可。

S3 Glacier 內的資源型政策範例

若要檢視 S3 Glacier 資源型政策範例，請參閱Amazon S3 Glacier 的資源型政策範例。

S3 Glacier 的政策動作

支援政策動作	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 Action 元素描述您可以用來允許或拒絕政策中存取的動作。原則動作通常與關聯的 AWS API 作業具有相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

如要查看 S3 Glacier 動作的清單，請參閱《服務授權參考》中的 Amazon S3 Glacier 定義的動作。

S3 Glacier 中的政策動作會在動作之前使用以下字首：

```
glacier
```

如需在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
            "glacier:CreateVault",
            "glacier:DescribeVault",
            "glacier:ListVaults"
         ]
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，若要指定開頭是 Describe 文字的所有動作，請包含以下動作：

```
"Action": "glacier:GetVault*"
```

若要檢視 S3 Glacier 身分型政策範例，請參閱Amazon S3 Glacier 的身分型政策範例。

S3 Glacier 的政策資源

支援政策資源	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。陳述式必須包含 Resource 或 NotResource 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

```
"Resource": "*"
```

如要查看 S3 Glacier 資源類型及其 ARN 的清單，請參閱《服務授權參考》中的 Amazon S3 Glacier 定義的資源。若要了解您可以指定每個資源的 ARN，請參閱 Amazon S3 Glacier 定義的動作。

在 S3 Glacier 中，主要資源是保存庫。S3 Glacier 僅支援保存庫層級的政策。也就是說，在 IAM 政策中，您指定的`Resource`值可以是特定保管庫或特定 AWS 區域中的一組保管庫。S3 Glacier 不支援封存層級許可。

對於所有 S3 Glacier 動作，`Resource` 指定您要授予許可的保存庫。這些資源具有與其相關聯的唯一 Amazon Resource Name (ARN)，如下表所示，而且您可以使用 ARN 中的萬用字元 (*)，以比對開頭為相同字首的保存庫名稱。

S3 Glacier 提供讓您使用 S3 Glacier 資源的一組作業。如需有關可用操作的詳細資訊，請參閱 Amazon S3 Glacier API 參考。

部分 S3 Glacier API 動作支援多個資源。例如，`glacier:AddTagsToVault` 存取 examplevault1 和 examplevault2，因此主體必須具有存取這兩個資源的許可。若要在單一陳述式中指定多項資源，請使用逗號分隔 ARN。

```
        "Resource": [
        ""arn:aws:glacier:us-west-2:123456789012:vaults/examplevault1",",
        ""arn:aws:glacier:us-west-2:123456789012:vaults/examplevault2","
        ]
```

S3 Glacier 的政策條件索引鍵

支援服務特定政策條件金鑰	是

管理員可以使用 AWS JSON 政策來指定誰可以存取哪些內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

`Condition` 元素 (或 `Condition` 區塊) 可讓您指定使陳述式生效的條件。`Condition` 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值，請使用邏輯OR運算來 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定條件金鑰。若要查看所有 AWS 全域條件金鑰，請參閱《IAM 使用者指南》中的AWS 全域條件內容金鑰。

如要查看 S3 Glacier 條件索引鍵的清單，請參閱《服務授權參考》中的適用於 Amazon S3 Glacier 的條件索引鍵。若要了解您可以搭配哪些動作和資源使用條件索引鍵，請參閱 Amazon S3 Glacier 定義的動作。

如需使用 glacier 特定之條件索引鍵的範例，請參閱保存庫鎖定政策。

S3 Glacier 中的 ACL

支援 ACL	否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

ABAC 搭配 S3 Glacier

支援 ABAC (政策中的標籤)	否

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在中 AWS，這些屬性稱為標籤。您可以將標籤附加到 IAM 實體 (使用者或角色) 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

如需根據標籤控制存取，請使用 aws:ResourceTag/*key-name*、aws:RequestTag/*key-name* 或 aws:TagKeys 條件索引鍵，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱 IAM 使用者指南中的什麼是 ABAC?。如要查看含有設定 ABAC 步驟的教學課程，請參閱 IAM 使用者指南中的使用屬性型存取控制 (ABAC)。

將臨時憑證與 S3 Glacier 搭配使用

支援臨時憑證	是

當您使用臨時憑據登錄時，某些 AWS 服務 不起作用。如需其他資訊，包括哪些 AWS 服務 與臨時登入資料搭配AWS 服務 使用，請參閱 IAM 使用者指南中的 IAM。

如果您使用除了使用者名稱和密碼以外的任何方法登入，則您正在 AWS Management Console 使用臨時認證。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取時，該程序會自動建立暫時認證。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱 IAM 使用者指南中的切換至角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而非使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

S3 Glacier 的跨服務主體許可

支援轉寄存取工作階段 (FAS)	否

當您使用 IAM 使用者或角色在中執行動作時 AWS，您會被視為主體。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 會使用主體呼叫的權限 AWS 服務，並結合要求 AWS 服務 向下游服務發出要求。只有當服務收到需要與其 AWS 服務 他資源互動才能完成的請求時，才會發出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。

S3 Glacier 的服務角色

支援服務角色	否

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱 IAM 使用者指南中的建立角色以委派許可給 AWS 服務服務。

> **Warning**
>
> 變更服務角色的許可可能會導致 S3 Glacier 功能故障。只有 S3 Glacier 提供指引時，才能編輯服務角色。

S3 Glacier 的服務連結角色

支援服務連結角色。	否

服務連結角色是連結至. AWS 服務服務可以擔任代表您執行動作的角色。服務連結角色會顯示在您的中， AWS 帳戶 且屬於服務所有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服務連結角色的詳細資訊，請參閱可搭配 IAM 運作的AWS 服務。在表格中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

Amazon S3 Glacier 的身分型政策範例

根據預設，使用者和角色不具備建立或修改 S3 Glacier 資源的許可。他們也無法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 來執行工作。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

若要了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策。

如需 S3 Glacier 所定義之動作和資源類型的詳細資訊，包括每種資源類型的 ARN 格式，請參閱《服務授權參考》中的適用於 Amazon S3 Glacier 的動作、資源和條件索引鍵。

以下是範例政策glacier:CreateVault，使用可識別區域中所有保存庫的 Amazon 資源名稱 (ARNglacier:ListVaults) 授與資源上三個 S3 Glacier 保存庫相關動作 (glacier:DescribeVault和) 的許可。us-west-2 AWS ARN 可唯一識別 AWS 資源。如需有關將 ARN 與 S3 Glacier 搭配使用的詳細資訊，請參閱 S3 Glacier 的政策資源。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
```

```
            "Effect": "Allow",
            "Action": [
            "glacier:CreateVault",
            "glacier:DescribeVault",
            "glacier:ListVaults"
            ],
            "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/*"
         }
      ]
   }
```

政策授與建立、列出並取得 us-west-2 區域中文件庫之相關說明的許可。ARN 結尾處的萬用字元 (*) 表示此陳述式可以符合任何保存庫名稱。

> **Important**
>
> 當您授與使用 glacier:CreateVault 作業來建立保存庫的許可，您必須指定萬用字元 (*)，因為您在建立保存庫之前不會知道保存庫的名稱。

主題

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 S3 Glacier 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管原則並邁向最低權限權限 — 若要開始授與使用者和工作負載的權限，請使用可授與許多常見使用案例權限的AWS 受管理原則。它們在您的 AWS 帳戶. 建議您透過定義特定於您使用案例的 AWS 客戶管理政策，進一步降低使用權限。如需更多資訊，請參閱 IAM 使用者指南中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱 IAM 使用者指南中的 IAM 中的政策和許可。

- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。您也可以使用條件來授與對服務動作的存取權 (如透過特定) 使用這些動作 AWS 服務，例如 AWS CloudFormation。如需詳細資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您編寫安全且實用的政策。如需更多資訊，請參閱 IAM 使用者指南中的 IAM Access Analyzer 政策驗證。
- 需要多因素身份驗證 (MFA) — 如果您的案例需要 IAM 使用者或根使用者 AWS 帳戶，請開啟 MFA 以獲得額外的安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需更多資訊，請參閱 IAM 使用者指南中的設定 MFA 保護的 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱 IAM 使用者指南中的 IAM 安全最佳實務。

使用 S3 Glacier 主控台

若要存取 Amazon S3 Glacier 主控台，您必須擁有最基本的一組許可。這些許可必須允許您列出和檢視 AWS 帳戶中 S3 Glacier 資源的詳細資訊。如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

您不需要為僅對 AWS CLI 或 AWS API 進行呼叫的使用者允許最低主控台權限。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

S3 Glacier 主控台為您提供整合環境，以建立並管理 S3 Glacier 保存庫。至少必須為您建立的 IAM 身分授予 `glacier:ListVaults` 動作的許可，以便檢視 S3 Glacier 主控台，如以下範例所示。

```
{
      "Version": "2012-10-17",
      "Statement": [
         {
            "Action": [
            "glacier:ListVaults"
            ],
            "Effect": "Allow",
            "Resource": "*"
         }
      ]
   }
```

AWS 透過提供由建立和管理的獨立 IAM 政策來解決許多常見使用案例 AWS。受管政策授與常見使用案例中必要的許可，讓您免於查詢需要哪些許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

下列 AWS 受管政策 (您可以附加至帳戶中的使用者) 專屬於 S3 Glacier：

- AmazonGlacierReadOnlyAccess— 授予透過 S3 冰川的唯讀存取權限 AWS Management Console。
- AmazonGlacierFullAccess— 授予對 S3 冰川的完整存取權 AWS Management Console。

您也可以建立專有的自訂 IAM 政策，以允許 S3 Glacier API 動作與資源的許可。您可以將這些自訂政策連接至您為 S3 Glacier 保存庫建立的自訂 IAM 角色。

下一節中討論的 S3 Glacier AWS 受管政策都授與的許可`glacier:ListVaults`。

如需詳細資訊，請參閱《IAM 使用者指南》中的新增許可到使用者。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此原則包含在主控台上或以程式設計方式使用 AWS CLI 或 AWS API 完成此動作的權限。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
```

```
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

客戶受管政策範例

您可以在本節中找到使用者政策範例，此政策會授予各種 S3 Glacier 動作的許可。當您使用 S3 冰川 REST API、Amazon 開發套件、或 S3 冰川管理主控台 (如果適用) 時，這些政策會運作。 AWS CLI

> **Note**
>
> 所有範例皆使用美國西部 (奧勒岡) 區域 (us-west-2) 及虛構帳戶 ID。

範例

- 範例 1：允許使用者從保存庫下載封存
- 範例 2：允許使用者建立保存庫和設定通知
- 範例 3：允許使用者上傳封存至特定保存庫
- 範例 4：允許使用者授與特定保存庫的完整許可

範例 1：允許使用者從保存庫下載封存

若要下載封存，首先啟動工作以擷取封存。擷取作業完成後，您可以下載資料。以下範例政策授與 `glacier:InitiateJob` 動作的許可來啟動工作 (可讓使用者擷取封存，或從保存庫擷取保存庫的庫存)，以及 `glacier:GetJobOutput` 動作的許可，以下載已擷取資料。該政策也會授與執行 `glacier:DescribeJob` 動作的許可，好讓使用者可以取得工作狀態。如需詳細資訊，請參閱 啟動任務 (POST 任務)。

該政策在名為 examplevault 的保存庫上授與這些許可。您可以從 Amazon S3 Glacier 主控台取得保存庫 ARN，或以程式設計方式呼叫 描述文件庫 (GET 文件庫) 或 「列出文件庫」(GET 文件庫) API 動作。

```
{
               "Version":"2012-10-17",
               "Statement": [
                  {
                     "Effect": "Allow",
                     "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/
examplevault",
                     "Action":["glacier:InitiateJob",
                              "glacier:GetJobOutput",
                              "glacier:DescribeJob"]
                  }
               ]
               }
```

範例 2：允許使用者建立保存庫和設定通知

以下範例政策授予在 Resource 元素中指定的 us-west-2 區域中建立保存庫和設定通知的許可。如需有關使用通知的詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。該策略還授予列出 AWS 區域中儲存庫並取得特定儲存庫描述的權限。

> **Important**
>
> 當您授與使用 glacier:CreateVault 作業來建立保存庫的許可，您必須在 Resource 值中指定萬用字元 (*)，因為您在建立保存庫之前不會知道保存庫的名稱。

```
{
               "Version":"2012-10-17",
               "Statement": [
                  {
                     "Effect": "Allow",
                     "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/*",
                     "Action":["glacier:CreateVault",
                              "glacier:SetVaultNotifications",
                              "glacier:GetVaultNotifications",
                              "glacier:DeleteVaultNotifications",
                              "glacier:DescribeVault",
```

```
                                "glacier:ListVaults"]
                   }
                ]
                }
```

範例 3：允許使用者上傳封存至特定保存庫

以下範例政策授予將封存上傳到 us-west-2 區域中特定保存庫的許可。這些許可允許使用者使用 上傳封存 (POST 封存) API 操作將封存一次全部上傳，或使用 啟動分段上傳 (POST 分段 - 上傳) API 操作分段上傳。

```
{
                "Version":"2012-10-17",
                "Statement": [
                   {
                      "Effect": "Allow",
                      "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/
examplevault",
                      "Action":["glacier:UploadArchive",
                               "glacier:InitiateMultipartUpload",
                               "glacier:UploadMultipartPart",
                               "glacier:ListParts",
                               "glacier:ListMultipartUploads",
                               "glacier:CompleteMultipartUpload"]
                   }
                ]
                }
```

範例 4：允許使用者授與特定保存庫的完整許可

以下範例政策授予名為 `examplevault` 保存庫上所有 S3 Glacier 動作的許可。

```
{
             "Version":"2012-10-17",
             "Statement": [
                {
                   "Effect": "Allow",
                   "Resource": "arn:aws:glacier:us-west-2:123456789012:vaults/
examplevault",
                   "Action":["glacier:*"]
                }
             ]
```

```
        }
```

Amazon S3 Glacier 的資源型政策範例

S3 Glacier 保存庫可讓一個保存庫存取政策和一個保存庫鎖定政策與其建立關聯。Amazon S3 Glacier 保存庫存取政策是資源型政策，可用來管理保存庫的許可。保存庫鎖定政策是可以鎖定的保存庫存取政策。鎖定保存庫鎖定政策後，便無法變更政策。您可以使用保存庫鎖定政策強制執行合規性控制。

主題

- 保存庫存取政策
- 保存庫鎖定政策

保存庫存取政策

Amazon S3 Glacier 保存庫存取政策是資源型的政策，可用來管理保存庫的許可。

您可以為每個保存庫建立一個保存庫存取政策，以管理許可。您隨時可以修改保存庫存取政策中的許可。S3 Glacier 也支援每個保存庫的保存庫鎖定政策，在您鎖定之後便無法更改。如需使用保存庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

範例

- 範例 1：授予特定 Amazon S3 Glacier 動作的跨帳戶許可
- 範例 2：授予 MFA 刪除操作的跨帳戶許可

範例 1：授予特定 Amazon S3 Glacier 動作的跨帳戶許可

以下範例政策為名為 `examplevault` 之保存庫上的一組 S3 Glacier 作業，將跨帳戶許可授予給兩個 AWS 帳戶 。

> **Note**
>
> 擁有保存庫的帳戶會被收取與保存庫關聯的所有費用。外部帳戶允許的所有請求、資料傳輸和擷取費用均計入擁有保存庫的帳戶。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"cross-account-upload",
         "Principal": {
            "AWS": [
               "arn:aws:iam::123456789012:root",
               "arn:aws:iam::444455556666:root"
            ]
         },
         "Effect":"Allow",
         "Action": [
            "glacier:UploadArchive",
            "glacier:InitiateMultipartUpload",
            "glacier:AbortMultipartUpload",
            "glacier:CompleteMultipartUpload"
         ],
         "Resource": [
            "arn:aws:glacier:us-west-2:999999999999:vaults/examplevault"

         ]
      }
   ]
}
```

範例 2：授予 MFA 刪除操作的跨帳戶許可

您可以使用多重要素驗證 (MFA) 保護 S3 Glacier 資源。為提供額外等級的安全性，MFA 要求使用者透過提供有效的 MFA 代碼來證明實際擁有 MFA 裝置。如需有關設定 MFA 存取的詳細資訊，請參閱《IAM 使用者指南》中的設定 MFA 保護的 API 存取。

範例原則會授 AWS 帳戶 與具有臨時登入資料的權限，以便從名為 examplevault 的儲存庫中刪除歸檔，前提是要求已透過 MFA 裝置進行驗證。政策使用 `aws:MultiFactorAuthPresent` 條件金鑰來指定此額外的需求。如需詳細資訊，請參閱《IAM 使用者指南》中的適用於條件的可用索引鍵。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "add-mfa-delete-requirement",
```

```
                        "Principal": {
                           "AWS": [
                              "arn:aws:iam::123456789012:root"
                           ]
                        },
                        "Effect": "Allow",
                        "Action": [
                           "glacier:Delete*"
                        ],
                        "Resource": [
                           "arn:aws:glacier:us-west-2:999999999999:vaults/
examplevault"
                        ],
                        "Condition": {
                           "Bool": {
                              "aws:MultiFactorAuthPresent": true
                           }
                        }
                     }
                  ]
               }
```

保存庫鎖定政策

Amazon S3 Glacier (S3 Glacier) 保存庫可以有一個資源型保存庫存取政策，和一個連接到該保存庫的保存庫鎖定政策。保存庫鎖定政策是您可以鎖定的保存庫存取政策。使用保存庫鎖定政策可協助您強制執行法規和合規要求。Amazon S3 Glacier 提供一組 API 作業，用於管理保存庫鎖定政策，請參閱使用 S3 Glacier API 鎖定文件庫。

做為保存庫鎖定政策的範例，假設您的保存庫必須保留封存一年才能將其刪除。為了實作此要求，您可以建立保存庫鎖定政策，拒絕使用者刪除封存的權限，直到封存已存在一年。您可以在鎖定之前測試此政策。鎖定政策後，政策將不可改變。如需鎖定程序的詳細資訊，請參閱保存庫鎖定政策。如果您要管理可以變更的其他使用者許可，則可以使用保存庫存取政策 (請參閱 保存庫存取政策)。

您可以使用 S3 Glacier API、Amazon 開發套件或 S3 Glacier 主控台來建立和管理文件庫鎖定政策。AWS CLI如需有關保存庫資源型政策所允許的 S3 Glacier 動作清單，請參閱 API 許可參考。

範例

- 範例 1：用於不超過 365 天的封存之拒絕刪除權限
- 範例 2：根據標籤拒絕刪除權限

範例 1：用於不超過 365 天的封存之拒絕刪除權限

假設有一個法規要求，封存保留長達一年，才能將它們刪除。您可以透過實作以下保存庫鎖定政策來強制執行該要求。文件庫存取政策拒絕 glacier:DeleteArchive 動作 examplevault 如果要刪除的檔案小於 1 年。該政策使用特定於 S3 Glacier 的條件索引鍵 ArchiveAgeInDays，來強制執行一年的保留要求。

```
{
               "Version":"2012-10-17",
               "Statement":[
                  {
                     "Sid": "deny-based-on-archive-age",
                     "Principal": "*",
                     "Effect": "Deny",
                     "Action": "glacier:DeleteArchive",
                     "Resource": [
                        "arn:aws:glacier:us-west-2:123456789012:vaults/examplevault"
                     ],
                     "Condition": {
                        "NumericLessThan" : {
                              "glacier:ArchiveAgeInDays" : "365"
                              }
                           }
                     }
                     ]
                  }
```

範例 2：根據標籤拒絕刪除權限

假設您有一個以時間為基礎的保留規則，即一個封存可以在不到一年的時間內被刪除。同時，假設您需要對封存進行合法保留，以防止在法律調查期間無限期地刪除或修改。在這種情況下，法律保留優先於保存庫鎖定政策中指定的以時間為基礎的保留規則。

為了實施這兩個規則，以下範例政策有兩個陳述式：

- 第一個陳述式拒絕給每個人的刪除許可、鎖定文件庫。此鎖是使用 LegalHold 標籤執行的。
- 第二個陳述式授予封存小於 365 天的刪除權限。但是，即使存檔少於 365 天，在第一個陳述式中的條件符合時，就沒有人可以將封存刪除。

```
{
"Version":"2012-10-17",
"Statement":[
   {
      "Sid": "lock-vault",
      "Principal": "*",
      "Effect": "Deny",
      "Action": [
         "glacier:DeleteArchive"
      ],
      "Resource": [
         "arn:aws:glacier:us-west-2:123456789012:vaults/examplevault"
      ],
      "Condition": {
         "StringLike": {
            "glacier:ResourceTag/LegalHold": [
            "true",
            ""
            ]
         }
      }
      },
      {
      "Sid": "you-can-delete-archive-less-than-1-year-old",
      "Principal": {
            "AWS": "arn:aws:iam::123456789012:root"
         },
      "Effect": "Allow",
      "Action": [
         "glacier:DeleteArchive"
      ],
      "Resource": [
         "arn:aws:glacier:us-west-2:123456789012:vaults/examplevault"
      ],
      "Condition": {
         "NumericLessThan": {
            "glacier:ArchiveAgeInDays": "365"
         }
      }
      }
   ]
}
```

Amazon S3 Glacier 身分識別和存取疑難排解

請使用以下資訊來協助您診斷和修正使用 S3 Glacier 和 IAM 時發生的常見問題。

主題

我未獲授權，不得在 S3 Glacier 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 glacier:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 glacier:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 glacier:*GetWidget* 動作存取 *my-example-widget* 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我沒有授權執行 iam：PassRole

如果您收到錯誤，告知您無權執行 iam:PassRole 動作，則必須更新您的政策，以允許您將角色傳遞至 S3 Glacier。

有些 AWS 服務 允許您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 marymajor 的 IAM 使用者嘗試使用主控台在 S3 Glacier 中執行動作時，發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想允許我以外的人訪問我 AWS 帳戶 的 S3 Glacier 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 S3 Glacier 是否支援這些功能，請參閱Amazon S3 Glacier 如何與 IAM 搭配運作。
- 若要了解如何提供對您所擁有資源 AWS 帳戶 的存取權，請參閱 IAM 使用者指南中您擁有的另一 AWS 帳戶 個 IAM 使用者提供存取權限。
- 若要了解如何將資源存取權提供給第三方 AWS 帳戶，請參閱 IAM 使用者指南中的提供第三方 AWS 帳戶 擁有的存取權。
- 如需了解如何透過聯合身分提供存取權，請參閱 IAM 使用者指南中的將存取權提供給在外部進行身分驗證的使用者 (聯合身分)。
- 若要了解跨帳戶存取使用角色和以資源為基礎的政策之間的差異，請參閱 IAM 使用者指南中的 IAM 中的跨帳戶資源存取。

API 許可參考

當您在設定 Amazon S3 Glacier 如何與 IAM 搭配運作 並編寫可連接到 IAM 身分 (身分型政策) 或資源 (資源型政策) 的許可政策時，可以使用以下資料表作為參考。此單包括每個 S3 Glacier API 作業、您可以授與執行動作權限的對應動作，以及您可以授與權限的 AWS 資源。

您在政策的 `Action` 元素中指定動作，然後在政策的 `Resource` 元素中指定資源值。您也可以使用 IAM 政策語言 `Condition` 元素，來指定政策生效時間。

若要指定動作，請使用後接 API 操作名稱的 `glacier:` 字首 (例如，`glacier:CreateVault`)。對於大多數 S3 Glacier 動作，`Resource` 是您要授予許可的保存庫。您透過使用保存庫 ARN，將保存庫指定為 `Resource` 值。欲表示條件，您可以使用預先定義的條件金鑰。如需詳細資訊，請參閱 S3 Glacier 內的資源型政策。

下表列出可與以身分為基礎的政策和以資源為基礎的政策搭配使用的動作。

> **Note**
>
> 有些動作只能與以身分為基礎的政策搭配使用。這些動作在第一個欄 API 作業名稱後以星號 (*) 標示。

S3 Glacier API 和動作所需的許可

中止分段上傳 (DELETE uploadID)

所需許可 (API 動作)： `glacier:AbortMultipartUpload`

資源：`arn:aws:glacier:region:account-id:vaults/vault-name`、`arn:aws:glacier:region:account-id:vaults/example*`、`arn:aws:glacier:region:account-id:vaults/*`

S3 Glacier 條件索引鍵：

「中止文件庫鎖定」(DELETE 鎖定政策)

所需許可 (API 動作)： `glacier:AbortVaultLock`

資源：

S3 Glacier 條件索引鍵：

新增標籤至文件庫 (POST 標籤新增)

所需許可 (API 動作)：`glacier:AddTagsToVault`

資源：`arn:aws:glacier:region:account-id:vaults/vault-name`、`arn:aws:glacier:region:account-id:vaults/example*`、`arn:aws:glacier:region:account-id:vaults/*`

S3 Glacier 條件索引鍵：`glacier:ResourceTag/TagKey`

完成分段上傳 (POST uploadID)

所需許可 (API 動作)：`glacier:CompleteMultipartUpload`

資源：`arn:aws:glacier:region:account-id:vaults/vault-name`、`arn:aws:glacier:region:account-id:vaults/example*`、`arn:aws:glacier:region:account-id:vaults/*`

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

完成文件庫鎖定 (POST lockId)

所需許可 (API 動作)：glacier:CompleteVaultLock

資源：

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

建立文件庫 (PUT 文件庫) *

所需許可 (API 動作)：glacier:CreateVault

資源：

S3 Glacier 條件索引鍵：

刪除封存 (DELETE archive)

所需許可 (API 動作)：glacier:DeleteArchive

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵： glacier:ArchiveAgeInDays、glacier:ResourceTag/*TagKey*

刪除文件庫 (DELETE 文件庫)

所需許可 (API 動作)：glacier:DeleteVault

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

刪除文件庫存取政策 (DELETE 存取政策)

所需許可 (API 動作)：glacier:DeleteVaultAccessPolicy

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

刪除文件庫通知 (DELETE 通知的組態)

所需許可 (API 動作)：glacier:DeleteVaultNotifications

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

描述任務 (GET JobID)

所需許可 (API 動作)：glacier:DescribeJob

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

描述文件庫 (GET 文件庫)

所需許可 (API 動作)：glacier:DescribeVault

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

取得資料擷取政策 (GET 政策) *

所需許可 (API 動作)：glacier:GetDataRetrievalPolicy

資源：arn:aws:glacier:*region*:*account-id*:policies/retrieval-limit-policy

S3 Glacier 條件索引鍵：

「取得任務輸出」 (GET 輸出)

所需許可 (API 動作)：glacier:GetJobOutput

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

取得文件庫存取政策 (GET 存取政策)

所需許可 (API 動作)：glacier:GetVaultAccessPolicy

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

取得文件庫鎖定 (GET 鎖定政策)

所需許可 (API 動作)：glacier:GetVaultLock

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

取得文件庫通知 (GET 通知的組態)

所需許可 (API 動作)：glacier:GetVaultNotifications

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

啟動任務 (POST 任務)

所需許可 (API 動作)：glacier:InitiateJob

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵： glacier:ArchiveAgeInDays、glacier:ResourceTag/*TagKey*

啟動分段上傳 (POST 分段 - 上傳)

所需許可 (API 動作)：glacier:InitiateMultipartUpload

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

啟動文件庫鎖定 (POST 鎖定政策)

所需許可 (API 動作)：glacier:InitiateVaultLock

資源：

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

列出工作 (GET 工作)

所需許可 (API 動作)：glacier:ListJobs

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

列出分段上傳 (GET 分段 - 上傳)

所需許可 (API 動作)：glacier:ListMultipartUploads

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

清單部分 (GET uploadID)

所需許可 (API 動作)：glacier:ListParts

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

列出文件庫的標籤 (GET 標籤)

所需許可 (API 動作)：glacier:ListTagsForVault

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：

「列出文件庫」(GET 文件庫)

所需許可 (API 動作)：glacier:ListVaults

資源：

S3 Glacier 條件索引鍵：

從文件庫移除標籤 (POST tags remove)

所需許可 (API 動作)：glacier:RemoveTagsFromVault

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

設定資料擷取政策 (PUT 政策) *

所需許可 (API 動作)：glacier:SetDataRetrievalPolicy

資源：arn:aws:glacier:*region*:*account-id*:policies/retrieval-limit-policy

S3 Glacier 條件索引鍵：

設定文件庫存取政策 (PUT 存取政策)

所需許可 (API 動作)：glacier:SetVaultAccessPolicy

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

設定文件庫通知組態 (PUT 通知的組態)

所需許可 (API 動作)：glacier:SetVaultNotifications

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

上傳封存 (POST 封存)

所需許可 (API 動作)：glacier:UploadArchive

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

分段上傳 (PUT uploadID)

所需許可 (API 動作)：glacier:UploadMultipartPart

資源：arn:aws:glacier:*region*:*account-id*:vaults/vault-name、arn:aws:glacier:*region*:*account-id*:vaults/example*、arn:aws:glacier:*region*:*account-id*:vaults/*

S3 Glacier 條件索引鍵：glacier:ResourceTag/*TagKey*

在 Amazon S3 Glacier 中記錄和監控

監控是維護 Amazon S3 Glacier (S3 Glacier) 及 AWS 解決方案的可靠性、可用性和效能所不可或缺。您應該從 AWS 解決方案各個部分收集監控資料，以便在發生失敗時，可更輕鬆地識別和偵錯該失敗的來源。AWS 提供下列工具，讓您監控 S3 Glacier 資源及回應潛在的事件：

Amazon CloudWatch 警示

透過 Amazon S3 使用 S3 Glacier 時，您可以使用 Amazon CloudWatch 警示，在指定的一段時間內監看單一指標。如果指標超過指定的閾值，會傳送一則通知至 Amazon SNS 主題或 AWS Auto Scaling 政策。CloudWatch 警示不會因為處於特定狀態而叫用動作。必須是狀態已變更並維持了所指定的時間長度，才會呼叫動作。如需詳細資訊，請參閱透過 Amazon CloudWatch 監控指標。

AWS CloudTrail 日誌

CloudTrail 會提供由使用者、角色或 AWS 服務在 S3 Glacier 中採取之動作的記錄。CloudTrail 將 S3 Glacier 的所有 API 呼叫擷取為事件，包括來自 S3 Glacier 主控台以及來自對 S3 Glacier API 發出的程式碼呼叫。如需更多詳細資訊，請參閱 使用 AWS CloudTrail 記錄 Amazon S3 Glacier API 呼叫。

AWS Trusted Advisor

為成千上萬 AWS 客戶提供服務的過程中，學習到的最佳實務，都體現在 Trusted Advisor 中。Trusted Advisor 可檢查您的 AWS 環境，並在有可能節省成本、提升系統可用性與效能或填補安全漏洞時向您提出建議。所有 AWS 客戶都能存取五項 Trusted Advisor 檢查。商業或企業支援方案客戶，可以檢視所有 Trusted Advisor 檢查。

如需詳細資訊，請參閱《AWS Support 使用者指南》中的 AWS Trusted Advisor。

Amazon S3 Glacier 的合規驗證

由第三方稽核人員評估 Amazon S3 Glacier (S3 Glacier) 安全與合規性是多個 AWS 合規計劃中的一環，包括：

- 系統和組織控制 (SOC)
- 支付卡產業資料安全標準 (PCI DSS)
- 聯邦風險與授權管理計劃 (FedRAMP)
- 美國健康保險流通與責任法案 (HIPAA)

AWS 會經常提供更新的 AWS 服務清單，旨在列出符合 AWS 服務範圍合規計劃特定合規計劃範圍的服務。

您可以使用 AWS Artifact 下載第三方稽核報告。如需詳細資訊，請參閱《AWS Artifact 使用者指南》中的下載 AWS 成品中的報告。

如需 AWS 合規計劃的詳細資訊，請參閱 AWS 合規計劃。

使用 S3 Glacier 時的合規責任會取決於資料的敏感度、組織的合規目標，以及適用法律和法規。若您使用的 S3 Glacier 必須遵循特定標準 (如 HIPAA、PCI 或 FedRAMP)，AWS 會提供資源予以協助：

- S3 Glacier 文件庫鎖定 可讓您使用文件庫鎖定政策，輕鬆部署並強制執行各個 S3 Glacier 文件庫的合規性控制。您可以在文件庫鎖定政策中指定控制功能，例如「寫入一次、讀取多次」(WORM)，並鎖定該政策以防未來進行編輯。在鎖定政策之後，再也無法變更它。文件庫鎖定政策可協助您符合法規框架，例如 SEC17a-4 和 HIPAA。
- 安全與合規快速入門指南會討論架構考量和步驟，說明如何在 AWS 上部署以安全及合規為重心的基準環境。
- HIPAA 安全及合規架構白皮書概述公司如何使用 AWS 來協助他們符合 HIPAA 要求。
- AWS Well-Architected Tool (AWS WA Tool) 是雲端中可提供一致程序的服務，您可以使用 AWS 最佳實務來檢閱和衡量架構。AWS WA Tool 提供的建議可讓工作負載更可靠、安全、有效率且經濟實惠。
- AWS 合規資源提供數種可能適用於您產業和位置的不同手冊和指南。
- AWS Config 可協助您評定資源組態與內部實務、業界準則和法規的合規狀態。
- AWS Security Hub 可供您全方位檢視 AWS 中的安全狀態，並協助您檢查是否符合安全產業標準和最佳實務。

Amazon S3 Glacier 中的恢復能力

AWS 全球基礎設施是以區域與可用區域為中心建置的。AWS 區域提供多個分開且隔離的實際可用區域，並以具備低延遲、高輸送量和高度備援特性的聯網相互連結。這些可用區域可讓您有效設計和操作應用程式與資料庫，它們的可用性、容錯能力和擴展性都比單一或多個資料中心的傳統基礎設施還高。S3 Glacier 會以備援方式，將資料存放在跨越至少三個可用區域的多個裝置中。為了增加耐用性，S3 Glacier 會在確認成功上傳之前，將資料同步存放到多個 AZ。

如需 AWS 區域與可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

Amazon S3 Glacier 的基礎設施安全性

Amazon S3 Glacier (S3 Glacier) 受管服務受到 Amazon Web Services：安全程序概觀所述的 AWS 全球網路安全程序保護。

您需要經由 AWS 發布的 API 來透過網路存取 S3 Glacier。用戶端必須支援 Transport Layer Security (TLS) 1.2。建議使用 TLS 1.3 或更新版本。用戶端還必須支援具備完全正向加密 (PFS) 功能的密碼套件，例如臨時 Diffie-Hellman (DHE) 或橢圓曲線臨時 Diffie-Hellman (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。此外，必須使用存取金鑰 ID，以及與 IAM 委託人相關聯的私密存取金鑰來簽署請求，或者您可以使用 AWS Security Token Service (AWS STS) 來產生臨時安全憑證來簽署請求。

VPC 端點

虛擬私有雲端 (VPC) 端點可讓您將 VPC 私下連線至支援的 AWS 服務以及具有 AWS PrivateLink 功能的 VPC 端點服務，而不需要網際網路閘道、NAT 裝置、VPN 連接或 AWS Direct Connect 連線。雖然 S3 Glacier 不直接支援 VPC 端點，但是如果您將 S3 Glacier 作為與 Amazon S3 整合的儲存層進行存取，則可以利用 Amazon S3 VPC 端點。

如需有關 Amazon S3 生命週期設定以及將物件轉換到 S3 Glacier 儲存類別的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的物件生命週期管理和轉換物件。如需 VPC 端點的詳細資訊，請參閱《Amazon VPC 使用者指南》中的 VPC 端點。

S3 Glacier 資料擷取政策

使用 Amazon S3 Glacier 資料擷取政策，您可以輕鬆設定資料擷取配額，並管理每個配額 AWS 帳戶中的資料擷取活動 AWS 區域。如需有關 S3 Glacier 資料擷取費用的詳細資訊，請參閱 S3 Glacier 定價。

> **Important**
>
> 資料擷取政策僅適用於標準擷取，並管理直接向 S3 Glacier 提出的擷取請求。
> 如需 S3 Glacier 儲存類別的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的封存物件的儲存類別和轉換物件。

主題

選擇 S3 Glacier 資料擷取政策

您可以從三種類型的 S3 Glacier 資料擷取政策中選擇：無擷取限制、僅限免費方案和最高擷取率。

「無擷取限制」是用於擷取的預設資料擷取政策。如果使用「無擷取限制」政策，則不會設定擷取配額，並接受所有有效的資料擷取請求。

使用僅限免費方案政策，您可以將擷取保留在每日 AWS 免費方案額度內，而不會產生任何資料擷取費用。如果您想要擷取的資料多於 AWS 免費方案限額中的資料，可以使用「最大擷取速率」政策來設定 bytes-per-hour 擷取率配額。「最大擷取速率」政策可確保帳戶中所有擷取工作的尖峰擷取速率 AWS 區域 不會超過您設定的 bytes-per-hour 配額。

使用「僅限免費方案」和「最高擷取率」政策，將不會接受超出指定的擷取配額的資料擷取請求。如果您使用的是「僅限免費方案」政策，S3 Glacier 將同步拒絕超出 AWS 免費方案限額的擷取請求。如果您使用「最大擷取速率」政策，S3 Glacier 會拒絕導致進行中任務尖峰擷取速率超出政策設定的bytes-per-hour 配額的擷取請求。這些政策協助您簡化資料擷取成本管理。

以下是有關資料擷取政策的一些有用資料：

- 資料擷取政策設定不會變更從 S3 Glacier 使用標準擷取所擷取資料所需的 3 到 5 小時期間。
- 設置新的資料擷取政策不會影響之前接受的已在進行中的擷取任務。
- 如果由於資料擷取政策而拒絕擷取工作請求，則您無需支付該工作或請求的費用。
- 您可以為每個策略設置一個數據檢索策略 AWS 區域，該策略將管理您帳戶 AWS 區域 下的所有數據檢索活動。資料擷取政策是特定特定的， AWS 區域 因為資料擷取成本各有不同 AWS 區域。如需詳細資訊，請參閱 Amazon S3 Glacier 定價。

僅限免費方案政策

您可以將資料擷取政策設定為「僅免費方案」，以確保擷取始終保持在 AWS 免費方案額度範圍內，這樣就不會產生資料擷取費用。如果擷取請求遭到拒絕，您將會收到錯誤訊息，指出目前的資料擷取政策拒絕了請求。

您將每個區域的資料擷取政策設為「僅限免費方案」。您無法在設定政策後，於一天內擷取到比 AWS 區域按比例分配的每日 AWS 免費擷取限額的更多資料。也不會產生資料擷取費用。

在一個月內產生資料擷取費用後，您也可以切換到僅限免費方案政策。此僅限免費方案政策在該情況下將對新的擷取請求生效，但不會影響過去的請求。您需要支付之前產生的費用。

最高擷取率政策

您可以將資料擷取政策設定為「最大擷取速率」，透過指定具有最大值的資料擷取配額來控制尖峰擷取速 bytes-per-hour 率。當您將資料擷取原則設定為「最大擷取速率」時，如果新的擷取要求會導致進行中工作的尖峰擷取速率超過原則所指定的 bytes-per-hour 配額，則會拒絕新的擷取要求。如果擷取工作請求遭到拒絕，您會收到錯誤訊息，指出目前的資料擷取政策拒絕了請求。

將您的資料擷取政策設定為「最大擷取速率」政策可能會影響您一天中可以使用的 AWS 免費方案額度。例如，假設您將最高擷取率設定為每小時 1 MB。這低於 AWS 免費方案政策費率。為了確保您充分利用每日 AWS 免費方案限額，您可以先將政策設定為「僅限免費方案」，然後再視需要切換至「最大擷取率」政策。如需更多有關如何計算擷取限額的資訊，請參閱 Amazon S3 Glacier 常見問答集。

無擷取限制政策

如果您的資料擷取政策設定為「無擷取限制」，則所有有效的資料擷取請求都將被接受，並且您的資料擷取成本將根據使用情況而變化。

使用 S3 Glacier 主控台設定資料擷取政策

使用 Amazon S3 Glacier 主控台建立資料擷取政策

1. 登入 AWS Management Console 並開啟 S3 冰川主控台，網址為 https://console.aws.amazon.com/glacier/home。
2. 在 「選取地區」下方， AWS 區域 從下拉式選單中選擇。您可以為每個項目設定資料擷取原則 AWS 區域。
3. 在左側的導覽窗格中，選擇資料擷取設定。
4. 選擇編輯。即會顯示編輯資料擷取政策頁面。
5. 在資料擷取政策下選擇政策。

   您可以選取三種資料擷取政策之一：無擷取限制、僅限免費方案或指定最高擷取率。

   - 如果您選擇無擷取限制，則會接受所有有效的擷取請求。
   - 如果您選擇僅限免費方案，則不接受超過 AWS 免費方案的資料擷取要求。
   - 如果您選擇指定最高擷取率，則資料擷取請求會造成進行中工作的尖峰擷取率超過您指定的最高擷取率時，就會拒絕這些請求。您必須在最高擷取率下的 GB/小時方塊中指定每小時 GB 的值。當您在 GB/小時中輸入一個值時，主控台將為您計算估計費用。
6. 選擇儲存變更。

使用 Amazon S3 Glacier API 設定資料擷取政策

您可以使用 Amazon S3 Glacier REST API 或使用 AWS 開發套件，以檢視和設定資料擷取政策。

使用 Amazon S3 Glacier REST API 設定資料擷取政策

您可以使用 Amazon S3 Glacier REST API，檢視和設定資料擷取政策。您可以使用 取得資料擷取政策 (GET 政策) 操作查看現有的資料擷取政策。您可以使用 設定資料擷取政策 (PUT 政策) 作業設定資料擷取政策。

您可以在使用 PUT 政策作業時，透過將 JSON `Strategy` 欄位值設定為 `BytesPerHour`、`FreeTier` 或 `None`，來選取資料擷取政策類型。`BytesPerHour` 等於在主控台中選擇指定最高擷取率、`FreeTier` 以選擇僅限免費方案，`None` 以選擇無擷取限制。

當您使用 啟動任務 (POST 任務) 作業以啟動資料擷取工作時，該工作將超出資料擷取政策中設定的最高擷取率，則 `Initiate Job` 作業將停止並擲出異常。

使用 AWS SDK 設定資料擷取原則

AWS 提供開發套件供您開發適用於 Amazon S3 冰川的應用程式。這些開發套件提供對應到底層 REST API 的程式庫，並提供物件，可讓您輕鬆建構請求和處理回應。如需更多詳細資訊，請參閱 使用 AWS 軟體開發套件搭配 Amazon S3 冰川。

標記 Amazon S3 Glacier 資源

標籤是您指派給 AWS 資源的標籤。每個標籤皆包含由您定義的索引鍵和值。您可以將定義的標籤指派給 Amazon S3 Glacier (S3 Glacier) 文件庫資源。標籤是個簡單的工具，但功能強大，可管理 AWS 資源並整理資料，包括帳單資料。

主題

標記基本概念

您將使用 S3 Glacier 主控台、AWS Command Line Interface (AWS CLI) 或 S3 Glacier API 完成以下任務：

- 新增標籤到文件庫
- 列出文件庫的標籤
- 從文件庫移除標籤

有關如何新增、列出和移除標籤的詳細資訊，請參閱 標記 S3 Glacier 文件庫。

您可以使用標籤來分類您的文件庫。例如，您可以依用途、擁有者或環境來分類文件庫。由於您定義了每個標籤的金鑰和值，您可以建立一組自訂的類別，以符合您的特定需求。例如，您可以定義一組標籤，可協助您依照文件庫的擁有者和用途來追蹤文件庫。以下是一些標籤的範例：

- 擁有者：名稱
- 用途：影片封存
- 環境：生產

標籤限制

基本標籤限制，如下所示：

- 資源的標籤 (文件庫) 上限為 50。
- 標籤金鑰與值皆區分大小寫。

標籤金鑰限制，如下所示：

- 在文件庫的一組標籤中，每個標籤金鑰必須是唯一的。如果您新增具有已使用金鑰的標籤，則新的標籤會覆寫現有金鑰值對。
- 標籤索引鍵開頭不能為 `aws:`，因為此字首保留供 AWS 使用。AWS 會代表您建立開頭為此字首的標籤，但您無法加以編輯或刪除。
- 標籤金鑰的長度必須為 1 到 128 個 Unicode 字元。
- 標籤索引鍵必須包含下列字元：Unicode 字母、數字、空格以及下列特殊字元：`_ . / = + - @`。

標籤值限制，如下所示：

- 標籤值的長度必須為 0 到 255 個 Unicode 字元。
- 標籤值可以空白。否則，它們必須包含下列字元：Unicode 字母、數字、空格以及下列任何特殊字元：`_ . / = + - @`。

使用標記追蹤成本

您可以使用標籤來分類並追蹤 AWS 成本。當您將標籤套用至任何 AWS 資源 (包括文件庫) 時，AWS 成本分配報告就會包含依標籤彙總的使用情況和成本。您可以套用代表業務類別 (例如成本中心、應用程式名稱和擁有者) 的標籤，將多種服務的成本分類整理。如需詳細資訊，請參閱《AWS Billing 使用者指南》中的將成本分配標籤用於自訂帳單報告。

使用標籤管理存取控制

可以將標籤用作存取政策陳述式中的條件。例如，您可以設定合法的保留標籤，並將其做為條件包含在資料保留政策中，該條件聲明「如果合法保留標籤值設定為 `True`，則將拒絕任何人的封存刪除。」 您可以部署資料保留政策，並在正常情況下將合法保留標籤設定為 `False`。如果您的資料必須保留以協助調查，您可以透過將標籤值設定為 `True`，並稍後以類似方式刪除保留，從而輕鬆開啟法律保留。如需詳細資訊，請參閱《IAM 使用者指南》中的使用標籤控制存取權限。

相關章節

- 標記 S3 Glacier 文件庫

使用 AWS CloudTrail 記錄 Amazon S3 Glacier API 呼叫

Amazon S3 Glacier (S3 Glacier) 已與 AWS CloudTrail 整合，這項服務可提供由使用者、角色或 S3 Glacier 中 AWS 服務所採取之動作的記錄。CloudTrail 將 S3 Glacier 的所有 API 呼叫擷取為事件，包括來自 S3 Glacier 主控台以及來自對 S3 Glacier API 發出的程式碼呼叫。若您建立追蹤，便可將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 S3 Glacier 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的 Event history (事件歷史記錄) 檢視最新事件。您可以利用 CloudTrail 所收集的資訊來判斷向 S3 Glacier 發出的請求，以及發出請求的 IP 地址、人員、時間和其他詳細資訊。

若要進一步了解 CloudTrail，請參閱《AWS CloudTrail 使用者指南》。

CloudTrail 中的 Amazon S3 Glacier 資訊

當您建立帳戶時，系統即會在 AWS 帳戶 中啟用 CloudTrail。S3 Glacier 中發生活動時，系統便會將該活動記錄於 CloudTrail 事件，並將其他 AWS 服務事件記錄至事件歷史中。您可以檢視、搜尋和下載 AWS 帳戶 的最新事件。如需詳細資訊，請參閱使用 CloudTrail 事件歷史記錄檢視事件。

若要持續記錄 AWS 帳戶 中正在進行的事件 (包括 S3 Glacier 的事件)，請建立追蹤。追蹤能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台建立線索時，線索會套用到所有 AWS 區域。權杖會記錄來自 AWS 分割區中所有 AWS 區域的事件，然後將記錄檔案交付到您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中所收集的事件資料。如需詳細資訊，請參閱：

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 從多個區域接收 CloudTrail 日誌檔案，以及從多個帳戶接收 CloudTrail 日誌檔案

CloudTrail 會記錄所有 S3 Glacier 動作，並記錄在 Amazon S3 Glacier API 參考 中。例如，對 建立文件庫 (PUT 文件庫)、刪除文件庫 (DELETE 文件庫) 以及 「列出文件庫」(GET 文件庫) 動作發出的呼叫會在 CloudTrail 日誌檔案中產生項目。

每一筆事件或日誌項目都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 是否使用根使用者或憑證提出該請求。

- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

了解 Amazon S3 Glacier 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌項目。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔案並非依公有 API 呼叫追蹤記錄的堆疊排序，因此不會以任何特定順序出現。

以下範例示範的是展示 建立文件庫 (PUT 文件庫)、刪除文件庫 (DELETE 文件庫)、「列出文件庫」(GET 文件庫) 及 描述文件庫 (GET 文件庫) 動作的 CloudTrail 日誌項目。

```
{
    "Records": [
        {
            "awsRegion": "us-east-1",
            "eventID": "52f8c821-002e-4549-857f-8193a15246fa",
            "eventName": "CreateVault",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "HJiLgvfXCY88QJAC6rRoexS9ThvI21Q1Nqukfly02hcUPPo",
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": {
                "location": "/999999999999/vaults/myVaultName"
            },
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
```

```
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "cdd33060-4758-416a-b7b9-dafd3afcec90",
            "eventName": "DeleteVault",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "GGdw-VfhVfLCFwAM6iVUvMQ6-fMwSqSO9FmRd0eRSa_Fc7c",
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "355750b4-e8b0-46be-9676-e786b1442470",
            "eventName": "ListVaults",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "yPTs22ghTsWprFivb-2u30FAaDALIZP17t4jM_xL9QJQyVA",
            "requestParameters": {
                "accountId": "-"
```

```
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        },
        {
            "awsRegion": "us-east-1",
            "eventID": "569e830e-b075-4444-a826-aa8b0acad6c7",
            "eventName": "DescribeVault",
            "eventSource": "glacier.amazonaws.com",
            "eventTime": "2014-12-10T19:05:15Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.02",
            "recipientAccountId": "999999999999",
            "requestID": "QRt1ZdFLGn0TCm784HmKafBmcB2lVaV81UU3fsOR3PtoIiM",
            "requestParameters": {
                "accountId": "-",
                "vaultName": "myVaultName"
            },
            "responseElements": null,
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/1.9.6 Mac_OS_X/10.9.5 Java_HotSpot(TM)_64-
Bit_Server_VM/25.25-b02/1.8.0_25",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "999999999999",
                "arn": "arn:aws:iam::999999999999:user/myUserName",
                "principalId": "A1B2C3D4E5F6G7EXAMPLE",
                "type": "IAMUser",
                "userName": "myUserName"
            }
        }
    ]
}
```

Amazon S3 Glacier API 參考

Amazon S3 Glacier 支援一組操作，尤其是一組 RESTful API 呼叫。可讓您與服務互動。

您可以使用任何可以傳送 HTTP 請求的程式設計庫，將 REST 請求傳送給 S3 Glacier。傳送 REST 請求時，S3 Glacier 需要您簽署請求來驗證每一個請求。此外，在上傳封存時，您還必須計算檢查總和的承載，並將其包含在您的請求。如需更多詳細資訊，請參閱 簽署請求。

如果發生錯誤，您需要了解 S3 Glacier 在錯誤回應中傳送的內容，以便處理錯誤。除了記錄 REST 操作之外，本節還提供了所有這些資訊，以便您可以直接進行 REST API 呼叫。

您可以直接使用 REST API 呼叫，或者使用提供包裝函式庫的 Amazon 開發套件。這些程式庫對您傳送的每個請求進行簽署，並計算請求中承載的檢查總和。因此，使用 Amazon 開發套件可簡化編碼任務。此開發人員指南提供使用 AWS SDK for Java 和 .NET 的基本 S3 Glacier 操作的工作範例。如需詳細資訊，請參閱 使用 AWS 軟體開發套件搭配 Amazon S3 冰川。

主題

常見請求標題

Amazon S3 Glacier (S3 Glacier) REST 請求包含的標頭包含有關請求的基本資訊。下表描述了所有 S3 Glacier REST 請求可以使用的標頭。

標頭名稱	描述	必要
Authorization	簽署請求所需的標頭。S3 Glacier 需要簽章版本 4。如需更多詳細資訊，請參閱 簽署請求。 類型：字串	是
Content-Length	請求本文的長度 (不含標頭)。 類型：字串 條件：僅適用於 上傳封存 (POST 封存) API。	有條件
Date	用來建立 Authorization 標頭中包含的簽章的日期。如果要使用 Date 標頭簽署時，必須指定要使用 ISO 8601 基本格式。在這種情況下，不需要 x-amz-date 標頭。請注意，當 x-amz-date 存在時，一律覆寫 Date 標頭的值。 如果日期標頭不會用來簽署，則它可以是 RFC 2616 (3.3 節) 指定的完整日期格式之一。例如，以下日期/時間 Wed, 10 Feb 2017 12:00:00 GMT 是與 S3 Glacier 搭配使用的有效日期/時間標頭。 如果您使用 Date 標頭進行簽署，則必須採用 ISO 8601 基本的 YYYYMMDD'T'HHMMSS'Z' 格式。 類型：字串 條件：如果已指定 Date，但不符合 ISO 8601 基本格式，則必須也包含 x-amz-date 標頭。如果 Date 是以 ISO 8601 基本格式指定的，那麼這對於簽署請求已足夠，並且您不需要 x-amz-date 標頭。如需詳細資訊，請參閱 Amazon Web Services 詞彙表中的 Signature 第 4 版中的處理日期。	有條件
Host	此標頭指定您傳送請求的服務終端節點。值格式必須是「glacier.*region*.amazonaws.com 」，	是

標頭名稱	描述	必要
	其中 *region* 會替換為 AWS 區域目的地，例如 us-west-2 。 類型：字串	
x-amz-content-sha256	使用 上傳封存 (POST 封存) 或 分段上傳 (PUT uploadID) 上傳的整個承載的計算 SHA256 檢查總和。這個標頭與 x-amz-sha256-tree-hash 標頭不同，但是對於某些小型承載，這些值是相同的。當 x-amz-content-sha256 是必要的，必須指定 x-amz-content-sha256 和 x-amz-sha256-tree-hash 。 類型：字串 條件：串流 API 的必要項目、上傳封存 (POST 封存) 和 分段上傳 (PUT uploadID)。	有條件
x-amz-date	用來在授權標頭中建立簽章的日期。格式必須是 ISO 8601 基本的 YYYYMMDD'T'HHMMSS'Z' 格式。例如，下列日期/時間 20170210T120000Z 是與 S3 Glacier 搭配使用的有效 x-amz-date 。 類型：字串 條件：x-amz-date 對所有請求都是選用的，都可以用於覆寫用於簽署請求的日期。如果 Date 標頭是以 ISO 8601 基本格式指定的，則不需要 x-amz-date 。當 x-amz-date 存在時，一律覆寫 Date 標頭的值。如需詳細資訊，請參閱 Amazon Web Services 詞彙表中的 Signature 第 4 版中的處理日期。	有條件
x-amz-glacier-version	要使用的 S3 Glacier API 版本。目前版本是 2012-06-01 。 類型：字串	是

標頭名稱	描述	必要
x-amz-sha256-tree-hash	上傳封存 (上傳封存 (POST 封存)) 或封存部分 (分段上傳 (PUT uploadID)) 的計算 SHA256 樹雜湊檢查總和。如需計算此檢查總和的詳細資訊，請參閱 運算檢查總和。 類型：字串 預設：無 條件：上傳封存 (POST 封存) 和 分段上傳 (PUT uploadID) 是必要的。	有條件

常見回應標頭

下表說明常用於大部分 API 回應的回應標頭。

名稱	描述
Content-Length	回應內文的長度，以位元組為單位。 類型：字串
Date	Amazon S3 Glacier (S3 Glacier) 回應的日期和時間，例如，Wed, 10 Feb 2017 12:00:00 GMT。日期格式必須是 RFC 2616 第 3.3 節中規定的完整日期格式之一 。請注意，傳回的 Date 可能從其他日期稍微偏移，因此，從上傳封存 (POST 封存) 請求傳回的日期，可能不符合文件庫的庫存清單中為封存顯示的日期。 類型：字串
x-amzn-RequestId	S3 Glacier 所建立之唯一識別您請求的值。當 S3 Glacier 發生問題時，AWS 可能會使用此值來解決問題。建議您記錄這些值。 類型：字串

名稱	描述
x-amz-sha256-tree-hash	封存或庫存內文的 SHA256 樹雜湊檢查總和。如需計算此檢查總和的詳細資訊，請參閱 運算檢查總和。 類型：字串

簽署請求

S3 Glacier 會要求您簽署請求，對您發送的每個請求進行身分驗證。若要簽署請求，請使用加密雜湊函數來計算數位簽章。加密雜湊是一個函數, 其根據輸入傳回一個唯一的雜湊值。此雜湊函數的輸入包含請求和私密存取金鑰的文字。雜湊函數會傳回一個雜湊值，您將此值包含在請求中做為簽章。該簽章是請求 `Authorization` 標頭中的一部分。

收到請求後，S3 Glacier 會使用您原先用以簽署請求的相同雜湊函數與輸入，重新計算簽章。如果產生的簽章符合請求中的簽章，則 S3 Glacier 會處理請求。否則，請求會遭到拒絕。

S3 Glacier 支援使用 AWS Signature 第 4 版進行身分驗證。計算簽章的程序可以分成三個任務：

- 任務 1：建立正式請求

  將 HTTP 請求重新編排為正式格式。使用標準表單是必要的，因為 S3 Glacier 在重新計算簽章以與所傳送的簽章進行比較時，會使用相同的標準表單。
- 任務 2：建立登入字串

  建立一個字串，您會使用此字串做為密碼編譯雜湊函數的其中一個輸入值。此字串，稱為登入字串，是雜湊演算法的名稱、請求日期、登入資料範圍字串和前一個任務的正式請求的串連。憑證範圍字串本身是日期、AWS 區域和服務資訊的串連。
- 任務 3：建立簽章

  使用接受兩個輸入字串的密碼編譯雜湊函數來建立請求的簽章：您的 登入字串和衍生金鑰。藉由從您的私密存取金鑰開始來計算此衍生金鑰和使用登入資料範圍 字串來建立一系列雜湊型訊息身分驗證代碼 (HMAC)。請注意，在此簽署步驟中所使用的雜湊函數不是用於上傳資料的 S3 Glacier API 中所使用的樹雜湊演算法。

主題

- 簽章計算範例

- 計算串流操作的簽章

簽章計算範例

以下範例會逐步解說為 建立文件庫 (PUT 文件庫) 建立簽章的詳細資訊 。此範例可用作檢查簽名簽章計算方法的參考。如需詳細資訊，請參閱《IAM 使用者指南》中的簽署 AWS API 請求。

該範例假設如下：

- 請求的時間戳記為 `Fri, 25 May 2012 00:24:53 GMT`。
- 端點是美國東部 (維吉尼亞北部) 區域 (`us-east-1`)。

一般請求語法 (包括 JSON 內文) 是：

```
PUT /-/vaults/examplevault HTTP/1.1
Host: glacier.us-east-1.amazonaws.com
Date: Fri, 25 May 2012 00:24:53 GMT
Authorization: SignatureToBeCalculated
x-amz-glacier-version: 2012-06-01
```

對於任務1：建立正式請求計算的請求的正式形式是：

```
PUT
/-/vaults/examplevault

host:glacier.us-east-1.amazonaws.com
x-amz-date:20120525T002453Z
x-amz-glacier-version:2012-06-01

host;x-amz-date;x-amz-glacier-version
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

正式請求的最後一行是請求內文的雜湊值。另外，請注意正式請求中的空的第三行。這是因為此 API 沒有查詢參數。

要為任務 2：建立要簽章的字串 簽章的字串是：

```
AWS4-HMAC-SHA256
```

```
20120525T002453Z
20120525/us-east-1/glacier/aws4_request
5f1da1a2d0feb614dd03d71e87928b8e449ac87614479332aced3a701f916743
```

登入字串的第一行是演算法，第二行是時間戳記，第三行是登入資料範圍，最後一行是來自任務 1：建立正式請求的正式請求的雜湊。在憑證範圍內使用的服務名稱是 glacier。

對於任務 3：建立簽章，衍生金鑰可以呈現為：

```
derived key = HMAC(HMAC(HMAC(HMAC("AWS4" + YourSecretAccessKey,"20120525"),"us-
east-1"),"glacier"),"aws4_request")
```

如果使用私密存取金鑰 wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY，則計算簽章是：

```
3ce5b2f2fffac9262b4da9256f8d086b4aaf42eba5f111c21681a65a127b7c2a
```

最後步驟是建立 Authorization 標頭。對於示範存取金鑰 AKIAIOSFODNN7EXAMPLE，標頭 (為了可讀性而新增了換行) 是：

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20120525/us-east-1/
glacier/aws4_request,
SignedHeaders=host;x-amz-date;x-amz-glacier-version,
Signature=3ce5b2f2fffac9262b4da9256f8d086b4aaf42eba5f111c21681a65a127b7c2a
```

計算串流操作的簽章

上傳封存 (POST 封存) 和 分段上傳 (PUT uploadID) 是串流操作，需要您在簽章和發送請求時包含附加標頭 x-amz-content-sha256。串流操作的簽章步驟與其他操作的簽章步驟完全相同，但增加了串流標頭。

串流標頭 x-amz-content-sha256 的計算是依據要上傳的整個內容 (承載) 的 SHA256 雜湊。請注意，這個計算不同於 SHA256 樹狀雜湊 (運算檢查總和)。除了微不足道的情況外，承載資料的 SHA 256 雜湊值將與承載資料的 SHA256 樹狀雜湊不同。

如果承載資料指定為位元組陣列，則可以使用以下 Java 程式碼片段來運算 SHA256 雜湊。

```
public static byte[] computePayloadSHA256Hash2(byte[] payload) throws
 NoSuchAlgorithmException, IOException {
    BufferedInputStream bis =
       new BufferedInputStream(new ByteArrayInputStream(payload));
    MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
    byte[] buffer = new byte[4096];
    int bytesRead = -1;
    while ( (bytesRead = bis.read(buffer, 0, buffer.length)) != -1 ) {
        messageDigest.update(buffer, 0, bytesRead);
    }
    return messageDigest.digest();
}
```

同樣地，在 C# ，您可以計算承載資料的 SHA256 雜湊，如以下的程式碼片段所示。

```
public static byte[] CalculateSHA256Hash(byte[] payload)
{
    SHA256 sha256 = System.Security.Cryptography.SHA256.Create();
    byte[] hash = sha256.ComputeHash(payload);

    return hash;
}
```

串流 API 的簽章計算範例

以下範例會逐步解說為 上傳封存 (POST 封存) 建立簽章的詳細資訊，這是 S3 Glacier 中兩個串流 API 的其中之一。該範例假設如下：

- 請求的時間戳記為 `Mon, 07 May 2012 00:00:00 GMT`。
- 端點是美國東部 (維吉尼亞北部) 區域 (us-east-1)。
- 內容承載是「歡迎使用 S3 Glacier」的字串。

下面的範例顯示一般請求語法 (包括 JSON 內文)。請注意，包含 `x-amz-content-sha256` 標頭。在這個簡化的範例中，`x-amz-sha256-tree-hash` 和 `x-amz-content-sha256` 是相同的值。但是，對於大於 1 MB 的封存上傳，情況並非如此。

```
POST /-/vaults/examplevault HTTP/1.1
```

```
Host: glacier.us-east-1.amazonaws.com
Date: Mon, 07 May 2012 00:00:00 GMT
x-amz-archive-description: my archive
x-amz-sha256-tree-hash: SHA256 tree hash
x-amz-content-sha256: SHA256 payload hash
Authorization: SignatureToBeCalculated
x-amz-glacier-version: 2012-06-01
```

對於任務 1：建立正式請求 計算的請求的正式形式如下所示。請注意，串流標頭 x-amz-content-sha256 包含其值。這表示您必須先讀取承載和計算 SHA256 雜湊，然後再計算簽章。

```
POST
/-/vaults/examplevault

host:glacier.us-east-1.amazonaws.com
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
x-amz-date:20120507T000000Z
x-amz-glacier-version:2012-06-01

host;x-amz-content-sha256;x-amz-date;x-amz-glacier-version
726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
```

剩餘的簽章計算遵循 簽章計算範例 中概述的步驟。使用私密存取金鑰Authorization 和存取金鑰 wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY 的 AKIAIOSFODNN7EXAMPLE 標頭如下所示 (為了可讀性而增加了換行)：

```
Authorization=AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20120507/us-east-1/glacier/aws4_request,
SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-amz-glacier-version,
Signature=b092397439375d59119072764a1e9a144677c43d9906fd98a5742c57a2855de6
```

運算檢查總和

當上傳封存時，您必須同時包含 x-amz-sha256-tree-hash 和 x-amz-content-sha256 標頭。x-amz-sha256-tree-hash 標題是您的請求內文中承載的檢查總和。此主題說明如何計算 x-amz-sha256-tree-hash 標頭。x-amz-content-sha256 標頭是整個承載的雜湊，並且是授權所需。如需詳細資訊，請參閱 串流 API 的簽章計算範例。

請求的承載內容可以是：

- 整個封存：在單一請求中使用上傳封存 API 上傳封存時，您在請求內文傳送整個封存。在這種情況下，您必須包含整個封存的檢查總和。
- 封存部分：使用分段上傳 API 以部分形式上傳封存時，您僅在請求內文傳送部分封存。在這種情況下，會包含封存部分的檢查總和。而且在上傳所有的部分後，您會完成分段上傳請求，其必須包含整個封存的檢查總和。

承載的檢查總和是 SHA-256 樹雜湊。其稱為樹雜湊的原因是，在運算檢查總和的過程中，您會計算 SHA-256 樹雜湊值。根的雜湊值是整個封存的檢查總和。

> **Note**
>
> 本節說明運算 SHA-256 樹雜湊的方式。不過，您可以使用任何會產生相同結果的程序。

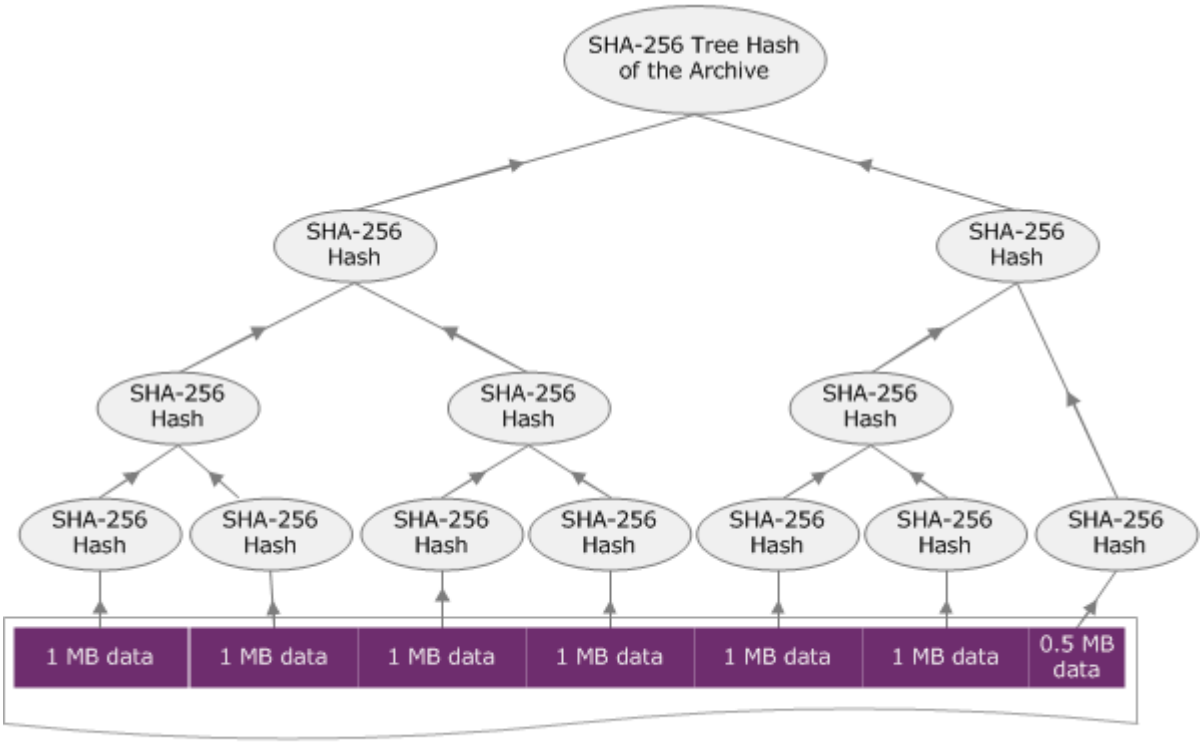
您運算 SHA-256 樹雜湊，如下所示：

1. 對於每個 1 MB 區塊的承載資料，運算 SHA-256 雜湊。最後區塊的資料可能小於 1 MB。例如，如果上傳 3.2 MB 的封存，您為前三個 1 MB 的資料區塊的每一個運算 SHA-256 雜湊值，然後運算剩餘 0.2 MB 資料的 SHA-256 雜湊。這些雜湊值形成樹狀結構的分葉節點。
2. 建置下一個層級的樹狀結構。
   a. 串連兩個連續的子節點雜湊值，並且運算已串連雜湊值的 SHA-256 雜湊。這個串連及 SHA-256 雜湊的產生會產生兩個子節點的父節點。
   b. 如果只剩一個子節點，就會將該雜湊值提升到樹狀結構的下一個層級。
3. 重複步驟 2，直到產生的樹狀結構有根。樹狀結構的根提供整個封存的雜湊，而適當子樹狀結構則提供分段上傳部分的雜湊。

主題

樹雜湊範例 1：在單一請求上傳封存

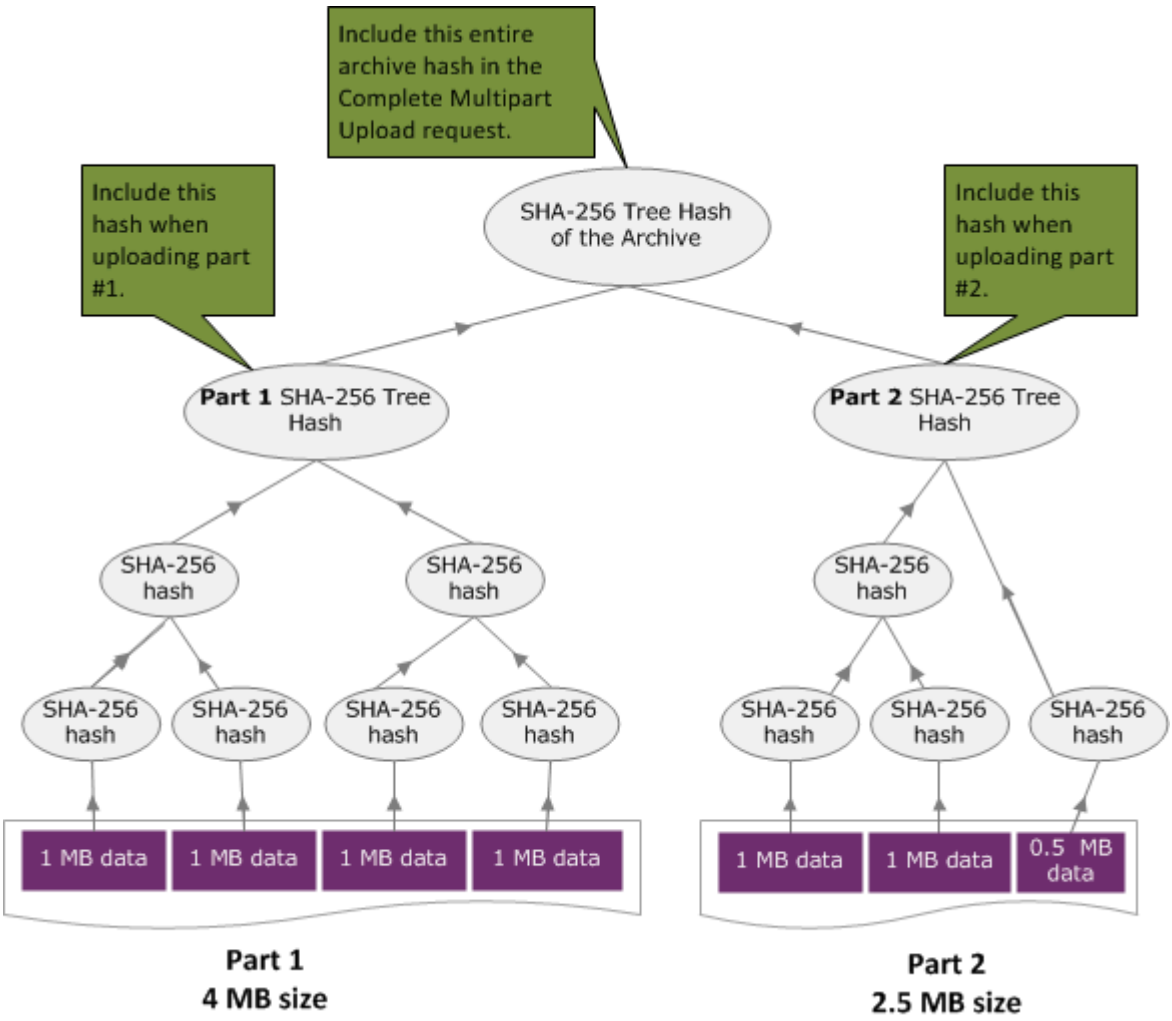
在單一請求中使用上傳封存 API 上傳封存時 (請參閱 上傳封存 (POST 封存)) 請求承載包含整個封存。因此，您必須在 x-amz-sha256-tree-hash 請求標頭中包含整個封存的樹雜湊。假設您想要上傳 6.5 MB 的封存。下圖說明建立封存的 SHA-256 雜湊的程序。您讀取封存並運算每 1 MB 區塊的 SHA-256 雜湊。您也運算剩餘 0.5 MB 資料的雜湊，然後依所述的樹狀結構建置程序。



6.5 MB Archive

樹雜湊範例 2：使用分段上傳來上傳封存

在使用分段上傳來上傳封存時，其運算樹雜湊的程序和在單一請求上傳封存的程序是相同的。唯一的差別是，在分段上傳只上傳每個請求的一部分封存 (使用 分段上傳 (PUT uploadID) API)，因此，您只提供 x-amz-sha256-tree-hash 請求標頭部分的檢查總和。不過，在上傳所有部分後，您必須隨著 完成分段上傳 (POST uploadID) 請求標頭中整個封存的樹雜湊，傳送「完成分段上傳」(請參閱 x-amz-sha256-tree-hash) 請求。



運算檔案的樹雜湊

這裡所顯示的演算法僅示範之目定而選定。您可以視您實作情況的需要最佳化程式碼。如果您使用 Amazon 開發套件，針對 Amazon S3 Glacier (S3 Glacier) 進行程式設計，系統便會為您完成樹雜湊計算，您只需提供檔案參考即可。

Example 1: Java 範例

以下範例說明如何計算使用 Java 的檔案的 SHA256 樹雜湊。您可以執行這個範例，做法是提供檔案位置做為引數，或可直接從您的程式碼使用 `TreeHashExample.computeSHA256TreeHash` 方法。

```
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
```

```
public class TreeHashExample {

static final int ONE_MB = 1024 * 1024;

    /**
     * Compute the Hex representation of the SHA-256 tree hash for the specified
     * File
     *
     * @param args
     *            args[0]: a file to compute a SHA-256 tree hash for
     */
    public static void main(String[] args) {

        if (args.length < 1) {
            System.err.println("Missing required filename argument");
            System.exit(-1);
        }

        File inputFile = new File(args[0]);
        try {

            byte[] treeHash = computeSHA256TreeHash(inputFile);
            System.out.printf("SHA-256 Tree Hash = %s\n", toHex(treeHash));

        } catch (IOException ioe) {
            System.err.format("Exception when reading from file %s: %s", inputFile,
                    ioe.getMessage());
            System.exit(-1);

        } catch (NoSuchAlgorithmException nsae) {
            System.err.format("Cannot locate MessageDigest algorithm for SHA-256: %s",
                    nsae.getMessage());
            System.exit(-1);
        }
    }

    /**
     * Computes the SHA-256 tree hash for the given file
     *
     * @param inputFile
     *            a File to compute the SHA-256 tree hash for
     * @return a byte[] containing the SHA-256 tree hash
     * @throws IOException
```

```
     *             Thrown if there's an issue reading the input file
     * @throws NoSuchAlgorithmException
     */
    public static byte[] computeSHA256TreeHash(File inputFile) throws IOException,
            NoSuchAlgorithmException {

        byte[][] chunkSHA256Hashes = getChunkSHA256Hashes(inputFile);
        return computeSHA256TreeHash(chunkSHA256Hashes);
    }

    /**
     * Computes a SHA256 checksum for each 1 MB chunk of the input file. This
     * includes the checksum for the last chunk even if it is smaller than 1 MB.
     *
     * @param file
     *            A file to compute checksums on
     * @return a byte[][] containing the checksums of each 1 MB chunk
     * @throws IOException
     *             Thrown if there's an IOException when reading the file
     * @throws NoSuchAlgorithmException
     *             Thrown if SHA-256 MessageDigest can't be found
     */
    public static byte[][] getChunkSHA256Hashes(File file) throws IOException,
            NoSuchAlgorithmException {

        MessageDigest md = MessageDigest.getInstance("SHA-256");

        long numChunks = file.length() / ONE_MB;
        if (file.length() % ONE_MB > 0) {
            numChunks++;
        }

        if (numChunks == 0) {
            return new byte[][] { md.digest() };
        }

        byte[][] chunkSHA256Hashes = new byte[(int) numChunks][];
        FileInputStream fileStream = null;

        try {
            fileStream = new FileInputStream(file);
            byte[] buff = new byte[ONE_MB];

            int bytesRead;
```

```
            int idx = 0;
            int offset = 0;

            while ((bytesRead = fileStream.read(buff, offset, ONE_MB)) > 0) {
                md.reset();
                md.update(buff, 0, bytesRead);
                chunkSHA256Hashes[idx++] = md.digest();
                offset += bytesRead;
            }

            return chunkSHA256Hashes;

        } finally {
            if (fileStream != null) {
                try {
                    fileStream.close();
                } catch (IOException ioe) {
                    System.err.printf("Exception while closing %s.\n %s",
file.getName(),
                            ioe.getMessage());
                }
            }
        }
    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1 MB chunk
     * checksums.
     *
     * This method uses a pair of arrays to iteratively compute the tree hash
     * level by level. Each iteration takes two adjacent elements from the
     * previous level source array, computes the SHA-256 hash on their
     * concatenated value and places the result in the next level's destination
     * array. At the end of an iteration, the destination array becomes the
     * source array for the next level.
     *
     * @param chunkSHA256Hashes
     *            An array of SHA-256 checksums
     * @return A byte[] containing the SHA-256 tree hash for the input chunks
     * @throws NoSuchAlgorithmException
     *             Thrown if SHA-256 MessageDigest can't be found
     */
    public static byte[] computeSHA256TreeHash(byte[][] chunkSHA256Hashes)
            throws NoSuchAlgorithmException {
```

```
        MessageDigest md = MessageDigest.getInstance("SHA-256");

        byte[][] prevLvlHashes = chunkSHA256Hashes;

        while (prevLvlHashes.length > 1) {

            int len = prevLvlHashes.length / 2;
            if (prevLvlHashes.length % 2 != 0) {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];

            int j = 0;
            for (int i = 0; i < prevLvlHashes.length; i = i + 2, j++) {

                // If there are at least two elements remaining
                if (prevLvlHashes.length - i > 1) {

                    // Calculate a digest of the concatenated nodes
                    md.reset();
                    md.update(prevLvlHashes[i]);
                    md.update(prevLvlHashes[i + 1]);
                    currLvlHashes[j] = md.digest();

                } else { // Take care of remaining odd chunk
                    currLvlHashes[j] = prevLvlHashes[i];
                }
            }

            prevLvlHashes = currLvlHashes;
        }

        return prevLvlHashes[0];
    }

    /**
     * Returns the hexadecimal representation of the input byte array
     *
     * @param data
     *            a byte[] to convert to Hex characters
     * @return A String containing Hex characters
     */
```

```
    public static String toHex(byte[] data) {
        StringBuilder sb = new StringBuilder(data.length * 2);

        for (int i = 0; i < data.length; i++) {
            String hex = Integer.toHexString(data[i] & 0xFF);

            if (hex.length() == 1) {
                // Append leading zero.
                sb.append("0");
            }
            sb.append(hex);
        }
        return sb.toString().toLowerCase();
    }
}
```

Example 2: C# .NET 範例

以下範例說明如何計算使檔案的 SHA256 樹雜湊。您可以執行此範例，做法是提供檔案位置做為引數。

```
using System;
using System.IO;

using System.Security.Cryptography;

namespace ExampleTreeHash
{
    class Program
    {
        static int ONE_MB = 1024 * 1024;

        /**
        * Compute the Hex representation of the SHA-256 tree hash for the
        * specified file
        *
        * @param args
        *            args[0]: a file to compute a SHA-256 tree hash for
        */
        public static void Main(string[] args)
        {
            if (args.Length < 1)
            {
```

```
                Console.WriteLine("Missing required filename argument");
                Environment.Exit(-1);
            }
            FileStream inputFile = File.Open(args[0], FileMode.Open, FileAccess.Read);
            try
            {
                byte[] treeHash = ComputeSHA256TreeHash(inputFile);
                Console.WriteLine("SHA-256 Tree Hash = {0}",
BitConverter.ToString(treeHash).Replace("-", "").ToLower());
                Console.ReadLine();
                Environment.Exit(-1);
            }
            catch (IOException ioe)
            {
                Console.WriteLine("Exception when reading from file {0}: {1}",
                    inputFile, ioe.Message);
                Console.ReadLine();
                Environment.Exit(-1);
            }
            catch (Exception e)
            {
                Console.WriteLine("Cannot locate MessageDigest algorithm for SHA-256:
{0}",
                    e.Message);
                Console.WriteLine(e.GetType());
                Console.ReadLine();
                Environment.Exit(-1);
            }
            Console.ReadLine();
        }


        /**
         * Computes the SHA-256 tree hash for the given file
         *
         * @param inputFile
         *            A file to compute the SHA-256 tree hash for
         * @return a byte[] containing the SHA-256 tree hash
         */
        public static byte[] ComputeSHA256TreeHash(FileStream inputFile)
        {
            byte[][] chunkSHA256Hashes = GetChunkSHA256Hashes(inputFile);
            return ComputeSHA256TreeHash(chunkSHA256Hashes);
        }
```

```
/**
 * Computes a SHA256 checksum for each 1 MB chunk of the input file. This
 * includes the checksum for the last chunk even if it is smaller than 1 MB.
 *
 * @param file
 *            A file to compute checksums on
 * @return a byte[][] containing the checksums of each 1MB chunk
 */
public static byte[][] GetChunkSHA256Hashes(FileStream file)
{
    long numChunks = file.Length / ONE_MB;
    if (file.Length % ONE_MB > 0)
    {
        numChunks++;
    }

    if (numChunks == 0)
    {
        return new byte[][] { CalculateSHA256Hash(null, 0) };
    }
    byte[][] chunkSHA256Hashes = new byte[(int)numChunks][];

    try
    {
        byte[] buff = new byte[ONE_MB];

        int bytesRead;
        int idx = 0;

        while ((bytesRead = file.Read(buff, 0, ONE_MB)) > 0)
        {
            chunkSHA256Hashes[idx++] = CalculateSHA256Hash(buff, bytesRead);
        }
        return chunkSHA256Hashes;
    }
    finally
    {
        if (file != null)
        {
            try
            {
                file.Close();
```

```
                }
                catch (IOException ioe)
                {
                    throw ioe;
                }
            }
        }

    }

    /**
     * Computes the SHA-256 tree hash for the passed array of 1MB chunk
     * checksums.
     *
     * This method uses a pair of arrays to iteratively compute the tree hash
     * level by level. Each iteration takes two adjacent elements from the
     * previous level source array, computes the SHA-256 hash on their
     * concatenated value and places the result in the next level's destination
     * array. At the end of an iteration, the destination array becomes the
     * source array for the next level.
     *
     * @param chunkSHA256Hashes
     *            An array of SHA-256 checksums
     * @return A byte[] containing the SHA-256 tree hash for the input chunks
     */
    public static byte[] ComputeSHA256TreeHash(byte[][] chunkSHA256Hashes)
    {
        byte[][] prevLvlHashes = chunkSHA256Hashes;
        while (prevLvlHashes.GetLength(0) > 1)
        {

            int len = prevLvlHashes.GetLength(0) / 2;
            if (prevLvlHashes.GetLength(0) % 2 != 0)
            {
                len++;
            }

            byte[][] currLvlHashes = new byte[len][];

            int j = 0;
            for (int i = 0; i < prevLvlHashes.GetLength(0); i = i + 2, j++)
            {

                // If there are at least two elements remaining
```

```
                    if (prevLvlHashes.GetLength(0) - i > 1)
                    {

                        // Calculate a digest of the concatenated nodes
                        byte[] firstPart = prevLvlHashes[i];
                        byte[] secondPart = prevLvlHashes[i + 1];
                        byte[] concatenation = new byte[firstPart.Length +
 secondPart.Length];
                        System.Buffer.BlockCopy(firstPart, 0, concatenation, 0,
 firstPart.Length);
                        System.Buffer.BlockCopy(secondPart, 0, concatenation,
 firstPart.Length, secondPart.Length);

                        currLvlHashes[j] = CalculateSHA256Hash(concatenation,
 concatenation.Length);

                    }
                    else
                    { // Take care of remaining odd chunk
                        currLvlHashes[j] = prevLvlHashes[i];
                    }
                }

                prevLvlHashes = currLvlHashes;
            }

            return prevLvlHashes[0];
        }

        public static byte[] CalculateSHA256Hash(byte[] inputBytes, int count)
        {
            SHA256 sha256 = System.Security.Cryptography.SHA256.Create();
            byte[] hash = sha256.ComputeHash(inputBytes, 0, count);
            return hash;
        }
    }
}
```

下載資料時接收檢查總和

當您使用啟動任務 API 擷取封存時 (請參閱 啟動任務 (POST 任務))，您可以選擇性指定擷取封存的範圍。同樣地，當您使用取得任務輸出 API 下載資料時 (請參閱 「取得任務輸出」 (GET 輸出))，您可以選擇指定各種下載資料。當您在擷取和下載封存的資料時，務必要了解這些範圍的兩個特性。擷取範圍

對封存而言需要符合 MB。擷取範圍和下載範圍都必須符合樹雜湊，才能在您下載資料時接收檢查總和值。這兩種類型的範圍合規遵循的定義如下：

- 百萬位元組對齊-範圍 [StartByte, EndBytes] 是百萬位元組 (1024*1024) 對齊時StartBytes可以被 1 MB 整除，或者等於指定的歸檔結尾 (封存位元組EndBytes大小減 1)。如果有指定，啟動任務 API 中使用的範圍需要符合 MB。
- 樹哈希對齊-當且僅當建立在範圍上的樹哈希的根等同於整個存檔的樹哈希中的節點時，range [StartBytes, EndBytes] 才是樹哈希相對於存檔對齊的樹哈希值。擷取範圍和下載範圍都必須符合樹雜湊，才能接收您下載資料的檢查總和值。如需範圍範例及其與封存樹雜湊之關係的詳細資訊，請參閱 樹雜湊範例：擷取符合樹雜湊的封存範圍。

  請注意，符合樹雜湊之範圍，也符合 MB。然而，符合 MB 範圍並不一定需符合樹雜湊。

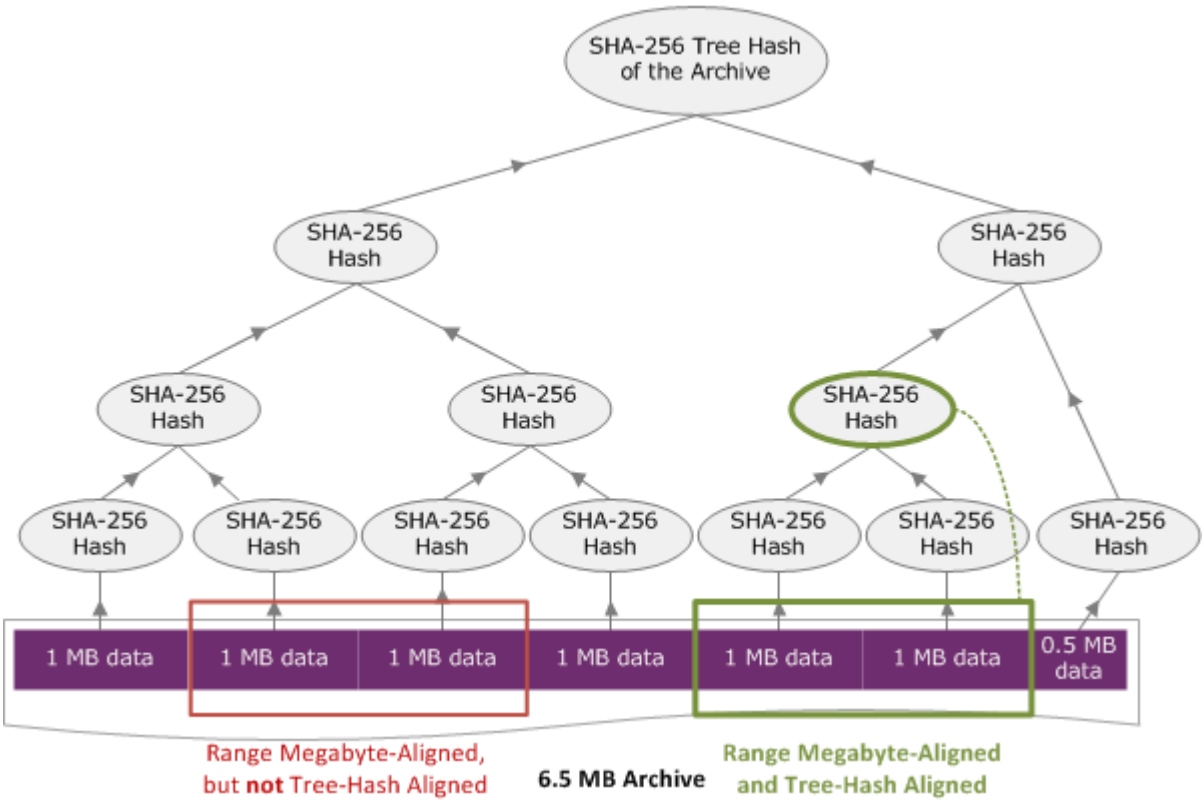
以下案例描述您何時在下載封存資料接收檢查總和值：

- 如果您未在啟動任務請求中指定擷取的範圍，而且在取得任務請求中下載整個封存。
- 如果您未在啟動任務請求中指定擷取的範圍，卻在取得任務請求中指定符合樹雜湊的範圍。
- 如果您在啟動任務請求中指定符合樹雜湊的範圍，而且在取得任務請求中下載整個範圍。
- 如果您在啟動任務請求中指定符合樹雜湊的範圍，並且在取得任務請求中指定符合樹雜湊的範圍。

如果您在啟動任務請求中指定非符合樹雜湊的擷取範圍，您仍然可以取得封存資料，但在取得任務請求中下載資料時並未有任何檢查總和值傳回。

樹雜湊範例：擷取符合樹雜湊的封存範圍

假設您的保存庫中有 6.5 MB 的封存，而您想要擷取 2 MB 的封存。您如何在啟動任務請求中指定 2 MB 範圍，決定您是否會在下載資料時接收資料檢查總和值。下圖說明您可以下載的兩個用於 6.5 MB 封存的 2 MB 範圍。兩個範圍都是符合 MB，但只有一個符合樹雜湊。



符合樹雜湊範圍規格

本節提供構成符合樹雜湊範圍的確切規格。下載封存的一部分，以及當指定擷取資料的範圍，並從擷取之資料下載的範圍時，符合樹雜湊範圍至關重要。如果這兩種範圍都符合樹雜湊，您將在下載資料時接收檢查總和資料。

範圍 [A、B] 是與封存相關的符合樹雜湊，如果且只有在新的樹雜湊建置在 [A、B] 上，且該範圍的樹雜湊根目錄等同於整個封存的樹雜湊節點。您可以查看相關內容，如 樹雜湊範例：擷取符合樹雜湊的封存範圍 圖中所示。在本節中，我們提供了適用於符合樹雜湊的規格。

考量 [P、Q] 為 N 百萬位元組 (MB) 的封存範圍查詢，而 P 和 Q 則是一 MB 的倍數。請注意，實際包含範圍是 [P MB、Q MB – 1 個位元組]，但為簡單化，我們會將它顯示為 [P、Q]。在這些考量下，然後

- 如果 P 是奇數，僅有一個可能的符合樹雜湊範圍 - 就是[P、P + 1 MB]。
- 如果 P 是偶數，而 k 是最大數，其中 P 可以編寫為 2k * X，則最多有 k 個符合樹雜湊範圍以 P 開頭。X 是大於 0 的整數。符合樹雜湊的範圍落在下列類別：
  - 對於每個 i，其中 (0 <= i <= k) 而且 $P + 2^i < N$，然後 [P、$Q + 2^i$] 是符合樹雜湊的範圍。
  - P = 0 是特殊情況，其中 A = 2[lgN]*0

錯誤回應

如果發生錯誤，此 API 會傳回以下其中一個例外狀況：

代碼	描述	HTTP 狀態碼	類型
AccessDeniedException	如果嘗試存取 AWS Identity and Access Management (IAM) 政策所不允許的資源，或者在請求 URI 中使用了錯誤的 AWS 帳戶 ID，則傳回。如需更多詳細資訊，請參閱 Amazon S3 Glacier 的 Identity and Access Management。	403 Forbidden	用戶端
BadRequest	如果請求無法處理，則傳回。	400 Bad Request	用戶端
ExpiredTokenException	如果請求中使用的安全權杖已過期，則傳回。	403 Forbidden	用戶端
InsufficientCapacityException	如果沒有足夠的能力處理急件請求，則傳回。此錯誤僅適用於急件擷取，而不適用於標準或大量擷取。	503 Service Unavailable	伺服器
InvalidParameterValueException	如果未正確指定請求的參數，則傳回。	400 Bad Request	用戶端
InvalidSignatureException	如果請求簽章無效，則傳回。	403 Forbidden	用戶端
LimitExceededException	如果請求導致超出以下其中一個限額：文件庫限制、標籤限制或佈建的容量限制，則傳回。	400 Bad Request	用戶端
MissingAuthenticationTokenException	如果沒有該請求的驗證資料，則傳回。	400 Bad Request	用戶端

代碼	描述	HTTP 狀態碼	類型
MissingParameterValueException	如果請求中缺少必要的標題或參數，則傳回。	400 Bad Request	用戶端
PolicyEnforcedException	如果擷取作業超過目前資料政策的擷取速率限制，則傳回。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。	400 Bad Request	用戶端
ResourceNotFoundException	如果指定的資源不存在，例如文件庫、上傳 ID 或任務 ID，則傳回。	404 Not Found	用戶端
RequestTimeoutException	如果上傳封存並在接收上傳時 Amazon S3 Glacier (S3 Glacier) 逾時，則傳回。	408 Request Timeout	用戶端
SerializationException	如果請求內文無效，則傳回。如果包括 JSON 承載，請檢查格式完好。	400 Bad Request	用戶端
ServiceUnavailableException	如果服務無法完成請求，則傳回。	500 Internal Server Error	伺服器
ThrottlingException	如果您需要將請求速率降至 S3 Glacier，則傳回。	400 Bad Request	用戶端
UnrecognizedClientException	如果存取金鑰 ID 或安全權杖無效，則傳回。	400 Bad Request	用戶端

各種 S3 Glacier API 會傳回相同的例外狀況，但具有不同的例外狀況訊息，可協助您對遇到的特定錯誤進行故障診斷。

S3 Glacier 在回應內文中傳回錯誤資訊。以下範例顯示一些錯誤回應。

範例 1：描述不存在的任務 ID 的任務請求

假設您傳送不存在任務的 描述任務 (GET JobID) 請求。也就是說，您指定了不存在的任務 ID。

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVEXAMPLEbadJobID HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

S3 Glacier 會在回應中傳回以下錯誤回應。

```
HTTP/1.1 404 Not Found
x-amzn-RequestId: AAABaZ9N92Iiyv4N7sru3ABEpSQkuFtmH3NP6aAC51ixfjg
Content-Type: application/json
Content-Length: 185
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "ResourceNotFoundException",
  "message": "The job ID was not found: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVEXAMPLEbadJobID",
  "type": "Client"
  }
```

其中：

Code

其中一個一般例外狀況。

類型：字串

Message

特定於傳回錯誤的 API 的錯誤狀況的一般描述。

類型：字串

類型

錯誤來源。該欄位可以是以下其中一個值：`Client`、`Server` 或 `Unknown`。

類型：字串

在先前的回應中，請注意下列事項：

- 對於錯誤回應，S3 Glacier 傳回 4xx 和 5xx 的狀態碼值。在此範例中，該狀態碼為 404 Not Found。
- 此 Content-Type 標頭值 application/json 表示內文中的 JSON
- 在內文的 JSON 提供錯誤資訊。

在之前的請求，假設您指定不存在的文件庫，而不是錯誤的任務 ID。該回應傳回不同的訊息。

```
HTTP/1.1 404 Not Found
x-amzn-RequestId: AAABBeC9Zw0rp_5D0L8VfB3FA_WlTupqTKAUehMcPhdgni0
Content-Type: application/json
Content-Length: 154
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "ResourceNotFoundException",
  "message": "Vault not found for ARN: arn:aws:glacier:us-west-2:012345678901:vaults/
examplevault",
  "type": "Client"
}
```

範例 2：列出請求參數具有無效值的任務請求

在這個範例中，您傳送 列出工作 (GET 工作) 請求以擷取具有特定 statuscode 的文件庫任務，而且您提供錯誤的 statuscode 值 finished，而不是可接受的值 InProgress、Succeeded 或 Failed。

```
GET /-/vaults/examplevault/jobs?statuscode=finished HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

S3 Glacier 以適當的訊息傳回 InvalidParameterValueException。

```
HTTP/1.1 400 Bad Request
x-amzn-RequestId: AAABaZ9N92Iiyv4N7sru3ABEpSQkuFtmH3NP6aAC51ixfjg
Content-Type: application/json
Content-Length: 141
Date: Wed, 10 Feb 2017 12:00:00 GMT
{
  "code": "InvalidParameterValueException",
  "message": "The job status code is not valid: finished",
  "type: "Client"
}
```

文件庫操作

以下是 S3 Glacier 中可用的文件庫操作。

主題

- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 新增標籤至文件庫 (POST 標籤新增)
- 建立文件庫 (PUT 文件庫)
- 完成文件庫鎖定 (POST lockId)
- 刪除文件庫 (DELETE 文件庫)
- 刪除文件庫存取政策 (DELETE 存取政策)
- 刪除文件庫通知 (DELETE 通知的組態)
- 描述文件庫 (GET 文件庫)
- 取得文件庫存取政策 (GET 存取政策)
- 取得文件庫鎖定 (GET 鎖定政策)
- 取得文件庫通知 (GET 通知的組態)
- 啟動文件庫鎖定 (POST 鎖定政策)
- 列出文件庫的標籤 (GET 標籤)
- 「列出文件庫」(GET 文件庫)
- 從文件庫移除標籤 (POST tags remove)
- 設定文件庫存取政策 (PUT 存取政策)
- 設定文件庫通知組態 (PUT 通知的組態)

「中止文件庫鎖定」(DELETE 鎖定政策)

描述

如果文件庫鎖定不是 Locked 狀態，此操作會停止文件庫鎖定程序。如果在請求此操作時，文件庫鎖定處於 Locked 狀態，則該操作將傳回 AccessDeniedException 錯誤。停止文件庫鎖定程序，會將文件庫鎖定政策從指定的文件庫中移除。

透過呼叫 InProgress，文件庫鎖定進入 啟動文件庫鎖定 (POST 鎖定政策) 狀態。透過呼叫 Locked，文件庫鎖定進入 完成文件庫鎖定 (POST lockId) 狀態。您可以透過呼叫 取得文件庫鎖定 (GET 鎖定政策) 來取得文件庫鎖定的狀態。如需文件庫鎖定程序的詳細資訊，請參閱S3 Glacier 文件庫鎖定。如需文件庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

此為等冪操作。如果文件庫鎖定處於 InProgress 狀態或者沒有與文件庫關聯的政策，則可以多次成功呼叫此操作。

請求

若要刪除文件庫鎖定政策，請將 HTTP DELETE 請求傳送到文件庫 lock-policy 子資源的 URI。

語法

```
DELETE /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

如果成功刪除該政策，S3 Glacier 將傳回 HTTP 204 No Content 回應。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何停止文件庫鎖定程序。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 **examplevault** 的文件庫的 lock-policy 子資源。

```
DELETE /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

```
x-amz-glacier-version: 2012-06-01
```

回應範例

如果政策成功刪除，則 S3 Glacier 會如以下範例所示傳回 HTTP 204 No Content 回應。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 完成文件庫鎖定 (POST lockId)
- 取得文件庫鎖定 (GET 鎖定政策)
- 啟動文件庫鎖定 (POST 鎖定政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

新增標籤至文件庫 (POST 標籤新增)

該操作將指定的標籤新增到文件庫。每個標籤皆包含鍵與值。每個文件庫最多可擁有 50 個標籤。如果您的請求會導致超出文件庫的標籤限制，則該操作會擲出 LimitExceededException 錯誤。

如果標籤已存在於指定金鑰下的文件庫，則現有的索引鍵值將會被覆寫。如需標籤的詳細資訊，請參閱標記 Amazon S3 Glacier 資源。

請求語法

若要將標籤新增到文件庫，請將 HTTP POST 請求傳送到標籤 URI，如以下語法範例所示。

```
POST /AccountId/vaults/vaultName/tags?operation=add HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
   "Tags":
      {
         "string": "string",
         "string": "string"
      }
}
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

名稱	描述	必要
operation=add	單一查詢字串參數 operation 的值為 add，以將其與 從文件庫移除標籤 (POST tags remove) 做區分。	是

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文包含以下 JSON 欄位。

Tags (標籤)

此標籤新增到文件庫。每個標籤皆包含鍵與值。此數值可以是空字串。

類型：字串到字串對應

長度限制：最小長度為 1。長度上限為 10。

必要：是

回應

如果操作請求成功，則服務會傳回 HTTP 204 No Content 回應。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例傳送帶有標籤的 HTTP POST 以新增至文件庫。

```
POST /-/vaults/examplevault/tags?operation=add HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

```
Content-Length: length
x-amz-glacier-version: 2012-06-01

{
  "Tags":
    {
       "examplekey1": "examplevalue1",
       "examplekey2": "examplevalue2"
    }
}
```

回應範例

如果請求成功，則 S3 Glacier 會如以下範例所示傳回 HTTP 204 No Content。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 列出文件庫的標籤 (GET 標籤)
- 從文件庫移除標籤 (POST tags remove)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

建立文件庫 (PUT 文件庫)

描述

此操作將使用指定的名稱建立新的文件庫。 文件庫的名稱在 AWS 帳戶 的 AWS 區域中必須是唯一的。您可以為每個帳戶建立最多 1,000 個文件庫。如需建立更多文件庫的詳細資訊，請前往 Amazon S3 Glacier 產品詳細資訊頁面。

命名文件庫時必須使用以下準則。

- 名稱長度可介於 1 到 255 個字元之間。
- 有效字元為 a-z、A-Z、0-9、'_' (底線)、'-' (連字號) 和 '.' (句號)。

這個操作是等冪的，您可以多次傳送相同的請求，並且在 Amazon S3 Glacier (S3 Glacier) 第一次建立指定的文件庫之後，不會再有任何影響。

請求

語法

若要建立文件庫，請將 HTTP PUT 請求傳送到要建立的文件庫 URI 。

```
PUT /AccountId/vaults/VaultName HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作的請求內文必須為空 (0 位元組)。

回應

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
Location	建立的文件庫的相對 URI 路徑。 類型：字串

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例傳送 HTTP PUT 請求來建立名為 examplevault 的文件庫。

```
PUT /-/vaults/examplevault HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Content-Length: 0
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

S3 Glacier 建立文件庫，並在 Location 標頭中傳回文件庫的相對 URI 路徑。無論帳戶 ID 或連字號 ('Location') 是否在請求中指定，帳戶 ID 一律會顯示在 - 標頭中。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Location: /111122223333/vaults/examplevault
```

相關章節

- 「列出文件庫」(GET 文件庫)
- 刪除文件庫 (DELETE 文件庫)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

完成文件庫鎖定 (POST lockId)

描述

此操作透過將文件庫鎖定從 InProgress 狀態轉移為 Locked 狀態來完成文件庫鎖定程序，這會導致文件庫鎖定政策無法變更。透過呼叫 InProgress，文件庫鎖定進入 啟動文件庫鎖定 (POST 鎖定政策) 狀態。您可以透過呼叫 取得文件庫鎖定 (GET 鎖定政策) 來獲得文件庫鎖定的狀態。如需文件庫鎖定程序的詳細資訊，請參閱S3 Glacier 文件庫鎖定。

此為等冪操作。如果文件庫鎖定處於 Locked 狀態並且提供的鎖定 ID 與最初用於鎖定文件庫的鎖定 ID 符合，則該請求一律成功。

如果文件庫鎖定處於 Locked 狀態時在請求中傳遞了無效的鎖定 ID，則該操作將傳回 AccessDeniedException 錯誤。如果文件庫鎖定處於 InProgress 狀態時在請求中傳遞了無效的鎖定 ID，則該操作將擲出 InvalidParameter 錯誤。

請求

要完成文件庫鎖定程序，請使用有效的鎖定 ID 將HTTP POST 請求傳送到文件庫的 lock-policy 子資源的 URI。

語法

```
POST /AccountId/vaults/vaultName/lock-policy/lockId HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

此 lockId 值是從 啟動文件庫鎖定 (POST 鎖定政策) 請求取得的鎖定 ID 。

請求參數

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

如果操作請求成功，則服務會傳回 HTTP 204 No Content 回應。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例使用鎖定 ID 傳送 HTTP POST 請求來完成文件庫鎖定程序。

```
POST /-/vaults/examplevault/lock-policy/AE863rKkWZU53SLW5be4DUcW HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示傳回 HTTP 204 No Content 回應。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 取得文件庫鎖定 (GET 鎖定政策)
- 啟動文件庫鎖定 (POST 鎖定政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

刪除文件庫 (DELETE 文件庫)

描述

這個操作會刪除文件庫。只有在文件庫中沒有按照上次庫存進行封存，並且自上次清點以來沒有再寫入文件庫時，Amazon S3 Glacier (S3 Glacier) 才會刪除文件庫。如果無法滿足這些條件，則文件庫刪除將失敗 (即無法刪除文件庫)，S3 Glacier 將傳回錯誤。

您可以使用提供文件庫資訊的 描述文件庫 (GET 文件庫) 操作，包括在文件庫中的封存數量，但是，該資訊是根據 S3 Glacier 上次產生的文件庫庫存。

此為等冪操作。

> **Note**
>
> 當您刪除文件庫，附加到文件庫的文件庫存取政策也會被刪除。如需文件庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

請求

若要刪除文件庫，請將 DELETE 請求傳送至文件庫資源 URI。

語法

```
DELETE /AccountId/vaults/VaultName HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例刪除名為「examplevault」的文件庫。此範例請求是對於要刪除的資源 (文件庫) 的 URI 之 DELETE 請求。

```
DELETE /-/vaults/examplevault HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 建立文件庫 (PUT 文件庫)
- 「列出文件庫」(GET 文件庫)
- 啟動任務 (POST 任務)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

刪除文件庫存取政策 (DELETE 存取政策)

描述

這個操作會刪除與指定文件庫關聯的存取政策。此操作最終是一致的，也就是說，Amazon S3 Glacier (S3 Glacier) 需要花費一些時間才能完全移除存取政策，並且在傳送刪除請求後，仍可能在短時間內看到政策的作用。

此為等冪操作。即使沒有與文件庫關聯的政策，也可以多次叫用刪除。如需文件庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

請求

若要刪除目前文件庫存取政策，請將 HTTP DELETE 請求傳送到文件庫 access-policy 子資源的 URI。

語法

```
DELETE /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

在回應中，如果成功刪除政策，則 S3 Glacier 會傳回 `204 No Content`。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何刪除文件庫存取政策。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 **examplevault** 的文件庫的 `access-policy` 子資源。

```
DELETE /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
x-amz-glacier-version: 2012-06-01
```

回應範例

如果在回應中成功刪除政策，則 S3 Glacier 會如以下範例所示傳回 `204 No Content`。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 取得文件庫存取政策 (GET 存取政策)
- 設定文件庫存取政策 (PUT 存取政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

刪除文件庫通知 (DELETE 通知的組態)

描述

此操作將刪除為 設定文件庫通知組態 (PUT 通知的組態) 設定的通知組態。此操作最終是一致的，也就是說，Amazon S3 Glacier (S3 Glacier) 需要花費一些時間才能完成停用通知，並且在傳送刪除請求後，可能在短時間內仍會收到一些通知。

請求

要刪除文件庫的通知組態，請向文件庫的 DELETE 子資源發送 `notification-configuration` 請求。

語法

```
DELETE /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何刪除文件庫的通知組態。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 notification-configuration 的文件庫的 examplevault 子資源。

```
DELETE /111122223333/vaults/examplevault/notification-configuration HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 取得文件庫通知 (GET 通知的組態)
- 設定文件庫通知組態 (PUT 通知的組態)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

描述文件庫 (GET 文件庫)

描述

這個操作會傳回有關文件庫的資訊，包括 Amazon Resource Name (ARN)、文件庫建立日期、文件庫內含的封存數量，以及文件庫中所有封存的大小總計。封存的數量，以及截至所產生最後文件庫庫存 Amazon S3 Glacier (S3 Glacier) 的大小總計 (請參閱 在 Amazon S3 Glacier 中使用文件庫)。S3

Glacier 幾乎每天產生文件庫庫存。這表示如果您新增或移除文件庫的封存，然後立即傳送描述文件庫請求、回應可能不會反映變更。

請求

若要取得有關文件庫的資訊，請傳送 GET 請求至特定文件庫資源的 URI。

語法

```
GET /AccountId/vaults/VaultName HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

```
Content-Type: application/json
Content-Length: Length

{
  "CreationDate" : String,
  "LastInventoryDate" : String,
  "NumberOfArchives" : Number,
  "SizeInBytes" : Number,
  "VaultARN" : String,
  "VaultName" : String
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

CreationDate

建立文件庫時的 UTC 日期。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

LastInventoryDate

當 S3 Glacier 完成最後一個文件庫庫存時的 UTC 日期。如需有關啟動文件庫之庫存的詳細資訊，請參閱 啟動任務 (POST 任務)。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

NumberOfArchives

根據最後一個文件庫的庫存的文件庫封存數量。如果文件庫尚未執行庫存，此欄位將傳回 Null，例如，您只建立文件庫。

類型：數字

SizeInBytes

文件庫中的封存大小總計以位元組為單位，包括截至最後一個庫存日期的各封存成本。如果文件庫尚未執行庫存，此欄位將傳回 null，例如，您只建立文件庫。

類型：數字

VaultARN

文件庫的 Amazon Resource Name (ARN)。

類型：字串

VaultName

在建立時指定的文件庫名稱。文件庫名稱也包含在文件庫的 ARN 中。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例示範如何取得有關稱為 `examplevault` 之文件庫的資訊。

```
GET /-/vaults/examplevault HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 260

{
  "CreationDate" : "2012-02-20T17:01:45.198Z",
  "LastInventoryDate" : "2012-03-20T17:03:43.221Z",
  "NumberOfArchives" : 192,
  "SizeInBytes" : 78088912,
```

```
  "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault",
  "VaultName" : "examplevault"
}
```

相關章節

- 建立文件庫 (PUT 文件庫)
- 「列出文件庫」(GET 文件庫)
- 刪除文件庫 (DELETE 文件庫)
- 啟動任務 (POST 任務)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

取得文件庫存取政策 (GET 存取政策)

描述

此操作會擷取文件庫上設定的 `access-policy` 子資源，如需設定此子資源的詳細資訊，請參閱 設定文件庫存取政策 (PUT 存取政策)。如果文件庫上沒有設定存取政策，則該操作會傳回 `404 Not found` 錯誤。如需文件庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

請求

若要傳回目前文件庫存取政策，請將 HTTP `GET` 請求傳送到文件庫的 `access-policy` 子資源的 URI。

語法

```
GET /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

在回應中，Amazon S3 Glacier (S3 Glacier) 會在回應內文中以 JSON 格式傳回文件庫存取政策。

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length

{
  "Policy": "string"
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

政策

文件庫存取政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何取得文件庫存取政策。

範例請求

在這個範例中，將 GET 請求傳送到文件庫的 access-policy 子資源的 URI。

```
GET /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

如果請求成功，S3 Glacier 會在回應的內文中傳回作為 JSON 字串的文件庫存取政策。傳回的 JSON 字串使用 "\" 做為逸出字元，如 設定文件庫存取政策 (PUT 存取政策) 範例中所示。不過，以下範例顯示傳回的 JSON 字串，無需逸出字元即可讀取。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: length

{
  "Policy": "
    {
      "Version": "2012-10-17",
```

```
      "Statement": [
        {
          "Sid": "allow-time-based-deletes",
          "Principal": {
            "AWS": "999999999999"
          },
          "Effect": "Allow",
          "Action": "glacier:Delete*",
          "Resource": [
            "arn:aws:glacier:us-west-2:999999999999:vaults/examplevault"
          ],
          "Condition": {
            "DateGreaterThan": {
              "aws:CurrentTime": "2018-12-31T00:00:00Z"
            }
          }
        }
      ]
    }
  "
}
```

相關章節

- 刪除文件庫存取政策 (DELETE 存取政策)
- 設定文件庫存取政策 (PUT 存取政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

取得文件庫鎖定 (GET 鎖定政策)

描述

此操作從指定的保存的 `lock-policy` 子資源集中擷取以下屬性：

- 文件庫上設定的文件庫鎖定政策。
- 文件庫鎖定的狀態，它是 InProgess 或 Locked。
- 當鎖定 ID 過期時。鎖定 ID 用於完成文件庫鎖定程序。
- 當文件庫鎖定啟動並進入 InProgress 狀態時。

透過呼叫 InProgress，文件庫鎖定進入 啟動文件庫鎖定 (POST 鎖定政策) 狀態。透過呼叫 Locked，文件庫鎖定進入 完成文件庫鎖定 (POST lockId) 狀態。您可呼叫 「中止文件庫鎖定」 (DELETE 鎖定政策)，停止文件庫鎖定程序。如需文件庫鎖定程序的詳細資訊，請參閱S3 Glacier 文件庫鎖定。

如果在文件庫沒有設定文件庫鎖定政策，則該操作將傳回 404 Not found 錯誤。如需文件庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

請求

若要傳回目前文件庫鎖定政策和其他屬性，請將 HTTP GET 請求傳送到文件庫的 lock-policy 子資源的 URI，如以下語法範例所示。

語法

```
GET /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

在回應中，Amazon S3 Glacier (S3 Glacier) 會在回應內文中以 JSON 格式傳回文件庫存取政策。

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length

{
  "Policy": "string",
  "State": "string",
  "ExpirationDate": "string",
  "CreationDate":"string"
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

政策

文件庫鎖定政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

州

文件庫鎖定的狀態。

類型：字串

有效值：InProgress|Locked

ExpirationDate

鎖定 ID 過期的 UTC 日期和時間。如果文件庫鎖定處於 null 狀態，則此值可以為 Locked。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

CreationDate

文件庫鎖定進入 InProgress 狀態的 UTC 日期和時間。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何取得文件庫鎖定政策。

範例請求

在這個範例中，將 GET 請求傳送到文件庫的 lock-policy 子資源的 URI。

```
GET /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

如果請求成功，S3 Glacier 會在回應的內文中傳回作為 JSON 字串的文件庫存取政策。傳回的 JSON 字串使用 "\" 做為逸出字元，如 啟動文件庫鎖定 (POST 鎖定政策) 範例請求所示。不過，以下範例顯示傳回的 JSON 字串，無需逸出字元即可讀取。

```
HTTP/1.1 200 OK
```

```
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: length

{
  "Policy": "
    {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Sid": "Define-vault-lock",
          "Principal": {
            "AWS": "arn:aws:iam::999999999999:root"
          },
          "Effect": "Deny",
          "Action": "glacier:DeleteArchive",
          "Resource": [
            "arn:aws:glacier:us-west-2:999999999999:vaults/examplevault"
          ],
          "Condition": {
            "NumericLessThanEquals": {
              "glacier:ArchiveAgeInDays": "365"
            }
          }
        }
      ]
    }
  ",
  "State": "InProgress",
  "ExpirationDate": "exampledate",
  "CreationDate": "exampledate"
}
```

相關章節

- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 完成文件庫鎖定 (POST lockId)

- 啟動文件庫鎖定 (POST 鎖定政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

取得文件庫通知 (GET 通知的組態)

描述

此操作擷取在文件庫 (請參閱 `notification-configuration` 上設定的 設定文件庫通知組態 (PUT 通知的組態) 子資源。如果未設定文件庫的通知組態，則操作將傳回 `404 Not Found` 錯誤。如需文件庫通知的詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。

請求

要擷取通知組態資訊，請向文件庫的 GET 子資源的 URI 發送 `notification-configuration` 請求。

語法

```
GET /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length
{
  "Events": [
    String,
    ...
  ],
  "SNSTopic": String
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

事件

Amazon S3 Glacier (S3 Glacier) 將向其指定的 Amazon SNS 主題傳送通知的一或多個事件清單。如需有關可將文件庫設定為發佈通知的文件庫事件的詳細資訊，請參閱 設定文件庫通知組態 (PUT 通知的組態)。

類型：陣列

SNSTopic

Amazon Simple Notification Service (Amazon SNS) 主題 Amazon Resource Name (ARN)。如需詳細資訊，請參閱《Amazon Simple Notification Service 入門指南》中的 Amazon SNS 入門。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何擷取文件庫的通知組態。

範例請求

在這個範例中，將 GET 請求傳送到文件庫的 `notification-configuration` 子資源。

```
GET /-/vaults/examplevault/notification-configuration HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

成功的回應以 JSON 格式顯示回應內文中的稽核記錄組態文件。此範例中的此設定示範如何將兩個事件 (`ArchiveRetrievalCompleted` 和 `InventoryRetrievalCompleted`) 的通知傳送到 Amazon SNS 主題 `arn:aws:sns:us-west-2:012345678901:mytopic`。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 150

{
  "Events": [
    "ArchiveRetrievalCompleted",
    "InventoryRetrievalCompleted"
```

```
  ],
  "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic"
}
```

相關章節

- 刪除文件庫通知 (DELETE 通知的組態)
- 設定文件庫通知組態 (PUT 通知的組態)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

啟動文件庫鎖定 (POST 鎖定政策)

描述

這個操作透過執行以下動作來啟動文件庫鎖定程序：

- 在指定的文件庫上安裝文件庫鎖定政策。
- 將文件庫鎖定的鎖定狀態設定為 `InProgress`。
- 傳回用於完成文件庫鎖定程序的鎖定 ID。

您可以為每個文件庫設定文件庫鎖定政策，並且該政策的大小最多可達 20 KB。如需文件庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

文件庫鎖定進入 `InProgress` 狀態後，您必須在 24 小時內完成文件庫鎖定程序。24 小時視窗結束後，鎖定 ID 到期，文件庫自動結束 `InProgress` 狀態，並且從文件庫中刪除文件庫鎖定政策。透過將文件庫鎖定狀態設定為 完成文件庫鎖定 (POST lockId)，呼叫 `Locked` 以完成文件庫鎖定程序。

> **Note**
>
> 文件庫鎖定處於 `Locked` 狀態後，您無法為文件庫啟動新的文件庫鎖定。

您可呼叫「中止文件庫鎖定」(DELETE 鎖定政策)，停止文件庫鎖定程序。您可以透過呼叫 取得文件庫鎖定 (GET 鎖定政策) 來取得文件庫鎖定的狀態。如需文件庫鎖定程序的詳細資訊，請參閱S3 Glacier 文件庫鎖定。

如果在文件庫鎖定處於 InProgress 狀態時呼叫此操作，該操作將傳回 AccessDeniedException 錯誤。當文件庫鎖定處於 InProgress 狀態時，必須先呼叫 「中止文件庫鎖定」(DELETE 鎖定政策)，才能啟動新的文件庫鎖定政策。

請求

若要啟動文件庫鎖定政策，請將 HTTP POST 請求傳送到文件庫的 lock-policy 子資源的 URI，如以下語法範例所示。

語法

```
POST /AccountId/vaults/vaultName/lock-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
  "Policy": "string"
}
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文包含以下 JSON 欄位。

政策

文件庫鎖定政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

必要：是

回應

如果接受此政策，則 Amazon S3 Glacier (S3 Glacier) 會傳回 `HTTP 201 Created` 回應。

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-lock-id: lockId
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
`x-amz-lock-id`	鎖定 ID 用於完成文件庫鎖定程序。 類型：字串

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例是發送 HTTP PUT 請求至文件庫的lock-policy 子資源的 URI。Policy JSON 字串使用 "\" 做為逸出字元。

```
PUT /-/vaults/examplevault/lock-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01

{"Policy":"{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-vault-
lock\",\"Effect\":\"Deny\",\"Principal\":{\"AWS\":\"arn:aws:iam::999999999999:root
\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-
west-2:999999999999:vaults/examplevault\",\"Condition\":{\"NumericLessThanEquals\":
{\"glacier:ArchiveAgeinDays\":\"365\"}}}]}"}
```

回應範例

如果請求成功，則 S3 Glacier 會如以下範例所示傳回 HTTP 201 Created 回應。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
x-amz-lock-id: AE863rKkWZU53SLW5be4DUcW
```

相關章節

- 「中止文件庫鎖定」(DELETE 鎖定政策)
- 完成文件庫鎖定 (POST lockId)
- 取得文件庫鎖定 (GET 鎖定政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

列出文件庫的標籤 (GET 標籤)

此操作列出了附加到文件庫的所有標籤。如果沒有標籤，該操作傳回空的對應。如需標籤的詳細資訊，請參閱標記 Amazon S3 Glacier 資源。

請求語法

要列出文件庫的標籤，請將 HTTP GET 請求傳送到標籤 URI，如以下語法範例所示。

```
GET /AccountId/vaults/vaultName/tags HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

如果操作成功，則服務傳回 HTTP 200 OK 回應。

回應語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length
{
   "Tags":
      {
         "string" : "string",
         "string" : "string"
      }
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

Tags (標籤)

此標籤附加到文件庫。每個標籤皆包含鍵與值。

類型：字串到字串對應

必要：是

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例：列出文件庫的標籤

以下範例列出文件庫的標籤。

範例請求

在這個範例中，傳送 GET 請求以從指定的文件庫中擷取標籤清單。

```
GET /-/vaults/examplevault/tags HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示，傳回帶有文件庫標籤清單的 HTTP 200 OK。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: length

{
   "Tags",
      {
         "examplekey1": "examplevalue1",
         "examplekey2": "examplevalue2"
      }
}
```

相關章節

- 新增標籤至文件庫 (POST 標籤新增)

- 從文件庫移除標籤 (POST tags remove)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

「列出文件庫」(GET 文件庫)

描述

此操作列出了呼叫使用者帳戶擁有的所有文件庫。回應中傳回的清單是 ASCII 依文件庫名稱排序。

在預設情況下，這個操作會傳回每個請求的最多 10 個項目。如果要列出更多的文件庫，則回應內文中的 `marker` 欄位包含文件庫 Amazon Resource Name (ARN)，以便在該列表中繼續使用新的「列出文件庫」請求，否則 `marker` 欄位是 `null`。在下一個「列出文件庫」請求中，將 `marker` 參數設為 Amazon S3 Glacier (S3 Glacier) 在上一個「列出文件庫」請求之回應中傳回的值。您可以透過在請求中指定 `limit` 參數來限制回應中傳回的文件庫數量。

請求

若要取得文件庫清單，請向GET文件庫 資源傳送 請求。

語法

```
GET /AccountId/vaults HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

這個操作會傳回以下請求參數。

名稱	描述	必要
limit	所要傳回的文件庫數量上限。預設限制為 10。傳回的文件庫數量可能少於指定的限制，但傳回的文件庫數量永遠不會超過限制。 類型：字串 限制：最小整數值為 1。最大整數值為 10。	否
marker	用於分頁的字串。marker 指定文件庫 ARN，之後應該開始文件庫清單。(由 marker 指定的文件庫不包括在傳回的清單中。) 從之前的「列出文件庫」回應中取得 marker 值。只有在您繼續對之前的「列出文件庫」請求中開始的結果進行分頁時，才需要包含 marker。指定標記的空白值 (「」) 會傳回從第一個文件庫開始的文件庫清單。 類型：字串 限制條件：無	否

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length
```

```
{
  "Marker": String
  "VaultList": [
   {
    "CreationDate": String,
    "LastInventoryDate": String,
    "NumberOfArchives": Number,
    "SizeInBytes": Number,
    "VaultARN": String,
    "VaultName": String
   },
   ...
  ]
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

CreationDate

建立文件庫的日期，以國際標準時間 (UTC) 為準。

類型：字串 ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

LastInventoryDate

上次文件庫庫存的日期，以國際標準時間 (UTC) 為準。如果文件庫尚未執行庫存，此欄位可以為 null，例如，您剛剛建立文件庫。如需有關啟動文件庫之庫存的詳細資訊，請參閱 啟動任務 (POST 任務)。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

Marker

vaultARN 代表繼續分頁結果。您在另一個「列出文件庫」請求中使用 marker 來取得清單中的更多文件庫。如果沒有更多的文件庫，則此值為 null。

類型：字串

NumberOfArchives

截至上次庫存日期的文件庫中的封存數量。

類型：數字

SizeInBytes

文件庫中的所有封存大小總計以位元組為單位，包括截至最後一個庫存日期的各封存成本。

類型：數字

VaultARN

文件庫的 Amazon Resource Name (ARN)。

類型：字串

VaultList

物件陣列，每個物件提供文件庫的說明。

類型：陣列

VaultName

此文件庫名稱。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例：列出所有文件庫

下列範例列出文件庫。由於在請求中未指定 `marker` 和d `limit` 參數，因此傳回最多 10 個文件庫。

範例請求

```
GET /-/vaults HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
```

```
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

Marker 是 null 表明沒有更多的文件庫列出。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 497

{
  "Marker": null,
  "VaultList": [
   {
    "CreationDate": "2012-03-16T22:22:47.214Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 2,
    "SizeInBytes": 12334,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault1",
    "VaultName": "examplevault1"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault2",
    "VaultName": "examplevault2"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-25T12:14:31.121Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault3",
    "VaultName": "examplevault3"
   }
  ]
}
```

範例：部分文件庫清單

以下範例傳回從 marker 指定的文件庫開始的兩個文件庫。

範例請求

```
GET /-/vaults?limit=2&marker=arn:aws:glacier:us-west-2:012345678901:vaults/
examplevault1 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

清單中傳回兩個文件庫。Marker 包含文件庫 ARN 以在另一個「列出文件庫」請求中繼續分頁。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: 497

{
  "Marker": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault3",
  "VaultList": [
   {
    "CreationDate": "2012-03-16T22:22:47.214Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 2,
    "SizeInBytes": 12334,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault1",
    "VaultName": "examplevault1"
   },
   {
    "CreationDate": "2012-03-19T22:06:51.218Z",
    "LastInventoryDate": "2012-03-21T22:06:51.218Z",
    "NumberOfArchives": 0,
    "SizeInBytes": 0,
    "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault2",
    "VaultName": "examplevault2"
```

```
    }
  ]
}
```

相關章節

- 建立文件庫 (PUT 文件庫)
- 刪除文件庫 (DELETE 文件庫)
- 啟動任務 (POST 任務)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

從文件庫移除標籤 (POST tags remove)

此操作會從附加到文件庫的一組標籤移除一或多個標籤。如需標籤的詳細資訊，請參閱標記 Amazon S3 Glacier 資源。

此為等冪操作。操作將會成功，即使沒有標籤附加到文件庫。

請求語法

若要從文件庫移除標籤，請如以下語法範例所示，將 HTTP POST 請求傳送到標籤 URI。

```
POST /AccountId/vaults/vaultName/tags?operation=remove HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
{
   "TagKeys": [
      "string",
      "string"
   ]
```

```
}
```

> **Note**
>
> 此 `AccountId` 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

名稱	描述	必要
`operation=remove`	單一查詢字串參數 `operation` 的值為 `remove`，以將其與 新增標籤至文件庫 (POST 標籤新增) 做區分。	是

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文包含以下 JSON 欄位。

TagKeys

標籤金鑰的清單。每個對應的標籤會從文件庫移除。

類型：字串的陣列

長度限制條件：清單中最少 1 個項目。清單中最多 10 個項目。

必要：是

回應

如果動作成功，則服務會傳回具空的 HTTP 內文的 HTTP `204 No Content` 回應。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例會傳送 HTTP POST 請求以移除指定的標籤。

```
POST /-/vaults/examplevault/tags?operation=remove HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01

{
   "TagsKeys": [
      "examplekey1",
      "examplekey2"
   ]
}
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示傳回 HTTP 204 No Content。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 新增標籤至文件庫 (POST 標籤新增)
- 列出文件庫的標籤 (GET 標籤)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

設定文件庫存取政策 (PUT 存取政策)

描述

這個操作將設定文件庫的存取政策，並會覆寫現有的政策。要設定文件庫存取政策，傳送 PUT 請求到文件庫的 `access-policy` 子資源。您可以為每個文件庫設定存取政策，並且該政策的大小最多可達 20 KB。如需文件庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

請求

語法

若要設定文件庫存取政策，請將 HTTP PUT 請求傳送到文件庫的 `access-policy` 子資源的 URI，如以下語法範例所示。

```
PUT /AccountId/vaults/vaultName/access-policy HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

```
{
  "Policy": "string"
}
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文包含以下 JSON 欄位。

政策

文件庫存取政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

必要：是

回應

如果在回應中接受政策被，則 S3 Glacier 將傳回 `204 No Content`。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
```

```
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例是發送 HTTP PUT 請求至文件庫的access-policy 子資源的 URI。Policy JSON 字串使用 "\" 做為逸出字元。

```
PUT /-/vaults/examplevault/access-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01

{"Policy":"{\"Version\":\"2012-10-17\",\"Statement\":[{\"Sid\":\"Define-owner-access-
rights\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::999999999999:root
\"},\"Action\":\"glacier:DeleteArchive\",\"Resource\":\"arn:aws:glacier:us-
west-2:999999999999:vaults/examplevault\"}]}"}
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示傳回 HTTP 204 No Content。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
```

```
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 刪除文件庫存取政策 (DELETE 存取政策)
- 取得文件庫存取政策 (GET 存取政策)

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

設定文件庫通知組態 (PUT 通知的組態)

描述

擷取封存和文件庫庫存是 Amazon S3 Glacier (S3 Glacier) 中的非同步操作，您必須先啟動任務並等待任務完成，然後才能下載任務輸出。您可以設定文件庫，以在這些任務完成時，將訊息發佈到 Amazon Simple Notification Service (Amazon SNS) 主題。可以使用此操作在文件庫上設定通知組態。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中設定文件庫通知。

要設定文件庫通知，傳送 PUT 請求到文件庫的 `notification-configuration` 子資源。通知組態特定於文件庫；因此，也被稱為文件庫子資源。該請求應包含提供 Amazon Simple Notification Service (Amazon SNS) 主題的 JSON 文件，以及希望 S3 Glacier 向該主題傳送通知的事件。

可以設定文件庫以發佈以下文件庫事件的通知：

- **`ArchiveRetrievalCompleted`**：當為封存擷取啟動的任務完成時，會發生此事件 (啟動任務 (POST 任務))。完成的工作的狀態可以是 `Succeeded` 或 `Failed`。傳送至 SNS 主題的通知與從 描述任務 (GET JobID) 傳回的輸出相同。
- **`InventoryRetrievalCompleted`**：當為庫存擷取啟動的任務完成時，會發生此事件 (啟動任務 (POST 任務))。完成的工作的狀態可以是 `Succeeded` 或 `Failed`。傳送至 SNS 主題的通知與從 描述任務 (GET JobID) 傳回的輸出相同。

Amazon SNS 主題必須授予對該文件庫的許可，才能將通知發布到該主題。

請求

要在文件庫上設定通知組態，請將 PUT 請求傳送到文件庫的 `notification-configuration` 子資源的 URI。您可以在請求內文中指定組態。該設定包含 Amazon SNS 主題名稱和一連串觸發每個主題通知的事件。

語法

```
PUT /AccountId/vaults/VaultName/notification-configuration HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01

{
   "SNSTopic": String,
   "Events":[String, ...]
}
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文中的 JSON 包含以下欄位。

事件

您希望 S3 Glacier 傳送通知之一或多個事件的陣列。

有效值：`ArchiveRetrievalCompleted | InventoryRetrievalCompleted`

必要：是

類型：陣列

SNSTopic

Amazon SNS 主題 ARN。如需詳細資訊，請前往《Amazon Simple Notification Service 入門指南》中的 Amazon SNS 入門。

必要：是

類型：字串

回應

如果在回應中接受通知設定，則 Amazon S3 Glacier (S3 Glacier) 會傳回 `204 No Content`。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何設定文件庫通知。

範例請求

以下請求設定 examplevault 通知設定，以便將兩個事件 (ArchiveRetrievalCompleted 和 InventoryRetrievalCompleted) 的通知傳送到 Amazon SNS 主題 arn:aws:sns:us-west-2:012345678901:mytopic。

```
PUT /-/vaults/examplevault/notification-policy HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
   "Events": ["ArchiveRetrievalCompleted", "InventoryRetrievalCompleted"],
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic"
}
```

回應範例

成功的回應會傳回 204 No Content。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 取得文件庫通知 (GET 通知的組態)
- 刪除文件庫通知 (DELETE 通知的組態)
- Amazon S3 Glacier 的 Identity and Access Management

另請參閱

如需在語言特定的 Amazon 開發套件之一中使用此 API 的詳細資訊，請參閱下列說明：

- AWS Command Line Interface

封存操作

以下是可在 S3 Glacier 中使用的封存操作。

主題

- 刪除封存 (DELETE archive)
- 上傳封存 (POST 封存)

刪除封存 (DELETE archive)

描述

此操作會從文件庫刪除封存。您可以一次從文件庫刪除一個封存。若要刪除封存，您必須提在刪除請求中提供其封存 ID。您可以透過為包含封存的文件庫下載文件庫庫存來取得封存 ID。如需下載文件庫庫存的詳細資訊，請參閱在 Amazon S3 Glacier 中下載文件庫庫存。

在您刪除封存之後，您可能仍然能夠成功請求起始任務，擷取已刪除的封存，但封存擷取任務將會失敗。

根據以下情況，當您刪除封存時，進行中的封存 ID 擷取可能會也可能不會成功：

- 如果封存擷取任務在 Amazon S3 Glacier (S3 Glacier) 接收刪除封存請求時，正在積極準備下載資料，則封存擷取操作可能會失敗。
- 如果在 S3 Glacier 接收刪除封存請求時，封存擷取任務已成功準備下載封存，則可以下載輸出。

如需封存擷取的詳細資訊，請參閱 在 S3 Glacier 中下載封存。

此為等冪操作。嘗試刪除已刪除的封存不會導致錯誤。

請求

為刪除封存，您傳送 DELETE 請求到封存資源 URI。

語法

```
DELETE /AccountId/vaults/VaultName/archives/ArchiveID HTTP/1.1
Host: glacier.Region.amazonaws.com
x-amz-Date: Date
```

```
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何從稱為 examplevault 之文件庫刪除封存。

範例請求

要刪除之封存的 ID 被指定為 archives 的子資源。

```
DELETE /-/vaults/examplevault/archives/NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

如果請求成功，S3 Glacier 會回應 204 No Content 以表示封存已刪除。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 上傳封存 (POST 封存)
- Amazon S3 Glacier 的 Identity and Access Management

上傳封存 (POST 封存)

描述

此操作將封存新增到文件庫。如需成功上傳，您的資料要長期保留。Amazon S3 Glacier (S3 Glacier) 會在回應中，傳回此回應之 x-amz-archive-id 標頭中的封存 ID。您應該儲存傳回的封存 ID，以便稍後可以存取封存。

您必須提供正在上傳的資料的 SHA256 樹狀雜湊。如需有關運算 SHA256 樹雜湊的資訊，請參閱 運算檢查總和。

> **Note**
>
> 使用 API 時，只有「上傳封存」(POST 封存) 動作才需要 SHA256 樹雜湊。使用 AWS CLI 時不需要此雜湊。

當上傳封存時，您可以選擇指定多達 1,024 可列印 ASCII 字元的封存說明。當您擷取封存或取得文件庫庫存時，S3 Glacier 會傳回封存說明。S3 Glacier 不以任何方式解釋說明。封存說明不需要是唯一的。您不能使用說明來擷取或排序封存清單。

除了選填的封存說明外，S3 Glacier 不支援封存的任何額外中繼資料。封存 ID 是一個不透明的字元序列，您無法從中推斷出封存的任何含義。因此，您可以在用戶端維護封存的中繼資料。如需更多詳細資訊，請參閱 在 Amazon S3 Glacier 中使用封存。

封存是不可變的。在您上傳封存後，您不能編輯封存或其說明。

請求

要上傳封存，請使用 HTTP POST 方法，並將請求範圍限定在要儲存封存的文件庫的 `archives` 子資源。請求必須包括封存承載大小、檢查總和(SHA256 樹狀雜湊)，並且可以選擇包含封存的說明。

語法

```
POST /AccountId/vaults/VaultName/archives
Host: glacier.Region.amazonaws.com
x-amz-glacier-version: 2012-06-01
Date: Date
Authorization: SignatureValue
x-amz-archive-description: Description
x-amz-sha256-tree-hash: SHA256 tree hash
x-amz-content-sha256: SHA256 linear hash
Content-Length: Length

<Request body.>
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此實作操作不使用請求參數。

請求標頭

除了所有操作通用的請求標頭之外，此操作還會使用下列請求標頭。如需常見請求標頭的資訊，請參閱常見請求標題。

名稱	描述	必要
`Content-Length`	物件的大小 (位元組)。如需詳細資訊，請參閱 http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.13。 類型：數字 預設：無 限制條件：無	是
`x-amz-archive-description`	正在上傳的封存的可選說明。它可以是純語言描述或您選擇指派的某個識別符。說明在封存中不必是唯一的。當您擷取文件庫庫存 (請參閱 啟動任務 (POST 任務)) 時，它將為其傳回的每個封存的說明包括在內。 類型：字串 預設：無	否

名稱	描述	必要
	限制：說明必須小於或等於 1,024 字元。允許的字元是沒有控制代碼的 7 位元 ASCII，尤其 ASCII 值是 32-126 十進制或 0x20-0x7E 十六進制。	
x-amz-content-sha256	承載的 SHA256-256 檢查總和 (線性雜湊)。這與您在 x-amz-sha256-tree-hash 標頭中指定的值不同。 類型：字串 預設：無 限制條件：無	是
x-amz-sha256-tree-hash	承載的使用者計算的檢查總和，SHA256-256 樹狀雜湊。如需有關運算 SHA256 樹狀雜湊的資訊，請參閱 運算檢查總和。如果 S3 Glacier 計算承載的不同檢查總和，其會拒絕該請求。 類型：字串 預設：無 限制條件：無	是

請求主體

請求內文包含要上傳的資料。

回應

S3 Glacier 會在回應中長期存放封存，並傳回封存 ID 的 URI 路徑。

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier
```

```
Location: Location
x-amz-archive-id: ArchiveId
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
Location	新加入的封存資源的相對 URI 路徑。 類型：字串
x-amz-archive-id	此封存的 ID。此值也包含在 Location 標頭中。 類型：字串
x-amz-sha256-tree-hash	由 S3 Glacier 計算的封存檢查總和。 類型：字串

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例顯示上傳封存的請求。

```
POST /-/vaults/examplevault/archives HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
```

```
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
x-amz-content-sha256: 7f2fe580edb35154041fa3d4b41dd6d3adaef0c85d2ff6309f1d4b520eeecda3
Content-Length: 2097152
x-amz-glacier-version: 2012-06-01
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
amz-glacier-
version,Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace

<Request body (2097152 bytes).>
```

回應範例

下面的成功回應有一個 Location 標頭，您可以從中取得S3 Glacier 指派給封存的 ID。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
Location: /111122223333/vaults/examplevault/archives/
NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
x-amz-archive-id: NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

相關章節

- 在 Amazon S3 Glacier 中使用封存
- 上傳分段中的大型封存 (分段上傳)
- 刪除封存 (DELETE archive)
- Amazon S3 Glacier 的 Identity and Access Management

分段上傳操作

以下是可用於 S3 Glacier 的分段上傳操作。

主題

中止分段上傳 (DELETE uploadID)

描述

此分段上傳操作命令會停止由上傳 ID 識別的分段上傳。

中止分段上傳請求成功後，您不能使用上傳 ID 上傳更多部分或執行任何其他操作。停止已完成的分段上傳失敗。但是，停止已經停止的上傳動作將會在短時間內成功。

此為等冪操作。

如需有關分段上傳的資訊，請參閱上傳分段中的大型封存 (分段上傳)。

請求

若要停止分段上傳，請將 HTTP DELETE 請求傳送至文件庫 `multipart-uploads` 子資源的 URI，並將特定的分段上傳 ID 識別為 URI 的一部分。

語法

```
DELETE /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

在下列範例中，將 DELETE 請求傳送到分段上傳 ID 資源的 URI。

```
DELETE /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE  HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
```

```
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 列出分段上傳 (GET 分段 - 上傳)
- 清單部分 (GET uploadID)
- 上傳分段中的大型封存 (分段上傳)
- Amazon S3 Glacier 的 Identity and Access Management

完成分段上傳 (POST uploadID)

描述

您呼叫此分段上傳操作以通知 Amazon S3 Glacier (S3 Glacier) 所有封存部分均已上傳，而且 S3 Glacier 現在可以將上傳的部分組合封存。

如需有關分段上傳的資訊，請參閱上傳分段中的大型封存 (分段上傳)。

在將封存整合並儲存到文件庫後，S3 Glacier 會傳回新建封存資源的封存 ID。上傳封存後，應該儲存傳回的封存 ID 以便稍後擷取封存。

在請求中，必須包括已上傳的整個封存的計算 SHA256 樹雜湊。如需有關運算 SHA256 樹雜湊的資訊，請參閱 運算檢查總和。在伺服器端，S3 Glacier 也會建構組合封存的 SHA256 樹雜湊。如果值相符，S3 Glacier 會將封存儲存到文件庫，否則其會傳回錯誤，而且操作會失敗。此 清單部分 (GET

uploadID) 操作會傳回為特定的分段上傳所上傳的組件清單。其中包含每個上傳部分的檢查總和資訊，可用於偵測錯誤的檢查總和問題。

此外，S3 Glacier 還會檢查任何遺漏的內容範圍。上傳部分時，可指定範圍值，以識別每個部分所適合在該封存中的最終整合位置。在為任何遺漏的內容範圍組合最終封存 S3 Glacier 檢查時，如果有任何遺漏的內容範圍，S3 Glacier 會傳回錯誤，而完整分段上傳操作將失敗。

完整分段上傳是等冪操作。在您首次成功完整分段上傳後，如果在短時間內再次呼叫該操作，該操作將成功並傳回相同的封存 ID。這在遇到網路問題或收到 500 伺服器錯誤時非常有用，在這種情況下，您可以重複完整分段上傳請求，並在不建立重複封存的情況下獲得相同的封存 ID。但是請注意，在分段上傳完成後，您無法呼叫清單組件操作，並且分段上傳不會出現在清單分段上傳回應中，即使冪等完整也是如此。

請求

若要完成分段上傳，請將 HTTP POST 請求傳送到 S3 Glacier 為回應初始分段上傳請求而建立之上傳 ID 的 URI。這與上傳組件時使用的 URI 相同。除了常見的必要標頭外，還必須包括整個封存的 SHA256 樹雜湊結果和封存的總大小 (以位元組為單位)。

語法

```
POST /AccountId/vaults/VaultName/multipart-uploads/uploadID
Host: glacier.Region.amazonaws.com
Date: date
Authorization: SignatureValue
x-amz-sha256-tree-hash: SHA256 tree hash of the archive
x-amz-archive-size: ArchiveSize in bytes
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

除了所有操作通用的請求標頭之外，此操作還會使用下列請求標頭。如需常見請求標頭的資訊，請參閱常見請求標題。

名稱	描述	必要
x-amz-archive-size	整個封存的總大小 (以位元組為單位)。這個值應該是您上傳的個別組件的所有大小的總和。 類型：字串 預設：無 限制條件：無	是
x-amz-sha256-tree-hash	整個封存的 SHA256 樹雜湊。它是個別組件的 SHA256 樹雜湊的樹雜湊。如果在請求中指定的值與 S3 Glacier 計算的最終組合封存的 SHA256 樹雜湊值不相符，則 S3 Glacier 會傳回錯誤，請求會失敗。 類型：字串 預設：無 限制條件：無	是

要求元素

此操作不使用請求元素。

回應

Amazon S3 Glacier (S3 Glacier) 建立整個封存的 SHA256 樹雜湊。如果該值與您在請求中指定的整個封存 SHA256 樹雜湊相符，則 S3 Glacier 會將封存新增到文件庫中。在回應中，會傳回具有新增的封存資源的 URL 路徑的 HTTP Location 標頭。如果您在請求中傳送的封存大小或 SHA256 不相符，則 S3 Glacier 將傳回錯誤，而且上傳仍處於未完成狀態。稍後可以使用正確的值重試完整分段上傳操作，此時您可以成功建立封存。如果分段上傳未完成，則最終 S3 Glacier 將回收上傳 ID。

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
x-amz-archive-id: ArchiveId
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
Location	新建立的封存的相對 URI 路徑。此 URL 包含 S3 Glacier 所產生的封存 ID。 類型：字串
x-amz-archive-id	此封存的 ID。此值也包含在 Location 標頭中。 類型：字串

回應欄位

此操作不會傳回任何回應內文。

範例

範例請求

在這個範例中，HTTP POST 請求被傳送到由初始分段上傳請求傳回的 URI。該請求同時指定整個封存的 SHA256 樹雜湊和總封存大小。

```
POST /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
```

```
z-amz-Date: 20170210T120000Z
x-amz-sha256-tree-hash:1ffc0f54dd5fdd66b62da70d25edacd0
x-amz-archive-size:8388608
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

以下範例回應示範 S3 Glacier 已成功從您上傳的部分中建立封存。該回應包含具有完整路徑的封存 ID。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/archives/
NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
x-amz-archive-id: NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
```

您現在可以將 HTTP 請求傳送到新增的資源/封存的 URI。例如，您可以傳送 GET 請求以擷取封存。

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 中止分段上傳 (DELETE uploadID)
- 列出分段上傳 (GET 分段 - 上傳)
- 清單部分 (GET uploadID)
- 上傳分段中的大型封存 (分段上傳)
- 刪除封存 (DELETE archive)
- Amazon S3 Glacier 的 Identity and Access Management

啟動分段上傳 (POST 分段 - 上傳)

描述

這個操作會啟動分段上傳 (請參閱上傳分段中的大型封存 (分段上傳))。Amazon S3 Glacier (S3 Glacier) 會建立分段上傳資源和在回應中傳回其 ID。在後續分段上傳操作中使用此上傳 ID。

當您啟動分段上傳時，可以指定部分大小 (以位元組為單位)。部分大小必須是 1 MiB (1024 KiB) 乘以 2 的次方，例如 1048576 (1 MiB)、2097152 (2 MiB)、4194304 (4 MiB)、8388608 (8 MiB) 等。最小允許部分大小為 1 MiB，最大為 4 GiB。

使用此上傳 ID上傳的每個部分（除最後一個外）都必須具有相同的大小。最後一個可以是相同的大小或較小。例如，假設您想要上傳 16.2 MiB 的檔案。如果您以 4 MiB 的部分大小啟動分段上傳，則將上傳四個部分的 4 MiB 和一個部分的 0.2 MiB。

> **Note**
>
> 當您啟動分段上傳時，不需要知道封存的大小，因為 S3 Glacier 不需要指定整體封存大小。

完成分段上傳後，S3 Glacier 會移除由該 ID 引用的分段上傳資源。如果取消分段上傳，或者如果在 24 小時內沒有活動，則 S3 Glacier 也將移除分段上傳資源。該 ID 可能在24 小時後後仍然可用，但應用程式不應預期這種行為。

請求

若要啟動分段上傳，您可以將 HTTP POST 請求傳送到要儲存封存的文件庫的 `multipart-uploads` 子資源的 URI。請求必須包括部分大小，並且可以選擇包含封存的說明。

語法

```
POST /AccountId/vaults/VaultName/multipart-uploads
Host: glacier.us-west-2.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
x-amz-archive-description: ArchiveDescription
x-amz-part-size: PartSize
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

除了所有操作通用的請求標頭之外，此操作還會使用下列請求標頭。如需常見請求標頭的資訊，請參閱常見請求標題。

名稱	描述	必要
`x-amz-part-size`	除了最後一個外，每個部分的大小 (以位元組為單位)。最後一個部分可以小於此部分大小。 類型：字串 預設：無 限制：部分大小必須是 1 MiB (1024 KiB) 乘以 2 的次方，例如 1048576 (1 MiB)、2097152 (2 MiB)、4194304 (4 MiB)、8388608 (8 MiB) 等。最小允許部分大小為 1 MiB，最大為 4 GiB (4096 MiB)。	是
`x-amz-archive-description`	封存描述您正在上傳的部分。它可以是純語言描述或您選擇指派的某個唯一識別符。當您擷取文件庫庫存 (請參閱 啟動任務 (POST 任務)) 時，庫存將為其傳回的每個封存的描述包括在內。封存描述中的前方空格會遭到移除。 類型：字串	否

名稱	描述	必要
	預設：無 限制：描述必須小於或等於 1,024 位元組。允許的字元是沒有控制代碼的 7 位元 ASCII，尤其 ASCII 值是 32-126 十進制或 0x20-0x7E 十六進制。	

請求主體

此操作沒有請求內文。

回應

在回應中，S3 Glacier 會建立由 ID 識別的分段上傳資源，並傳回分段上傳 ID 的相對 URI 路徑。

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
x-amz-multipart-upload-id: multiPartUploadId
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
Location	已建立分段上傳 ID S3 Glacier 的相對 URI 路徑。您使用此 URI 路徑來限制您的請求以上傳部分，並完成分段上傳。 類型：字串
x-amz-multipart-upload-id	分段上傳的 ID。此值也包含在 Location 標頭中。

名稱	描述
	類型：字串

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例透過名為 POST 的文件庫的 multipart-uploads 子資源的 URI 傳送 HTTP examplevault 請求來啟動分段上傳。該請求包括標頭以指定 4 MiB (4194304 位元組) 的部分大小和選填的封存描述。

```
POST /-/vaults/examplevault/multipart-uploads
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-archive-description: MyArchive-101
x-amz-part-size: 4194304
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

S3 Glacier 建立分段上傳資源，並將其新增到文件庫的 multipart-uploads 子資源。Location 回應標頭包括分段上傳 ID 的相對 URI 路徑。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE
```

```
x-amz-multipart-upload-id:
 OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE
```

如需上傳單個部分的詳細資訊，請參閱 分段上傳 (PUT uploadID)。

相關章節

- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 中止分段上傳 (DELETE uploadID)
- 列出分段上傳 (GET 分段 - 上傳)
- 清單部分 (GET uploadID)
- 刪除封存 (DELETE archive)
- 上傳分段中的大型封存 (分段上傳)
- Amazon S3 Glacier 的 Identity and Access Management

清單部分 (GET uploadID)

描述

此分段上傳操作列出已上傳到由上傳 ID 識別的特定分段上傳中的封存部分。如需有關分段上傳的資訊，請參閱上傳分段中的大型封存 (分段上傳)。

在完成分段上傳之前，您可以在正在進行的分段上傳期間隨時提出此請求。S3 Glacier 會傳回部分清單，這些清單會按您在每個部分上傳中指定的範圍排序。如果在完成分段上傳後傳送清單部分請求，Amazon S3 Glacier (S3 Glacier) 會傳回錯誤。

此清單部分操作支援分頁。您應該經常檢查回應內文 `Marker` 欄位中註明可繼續列表的標記，如果沒有其他項目，`marker` 欄位為 `null`。如果 `marker` 不是 Null，若要擷取下一組部分，請將另一個清單部分請求與 `marker` 請求參數設定為 S3 Glacier 為回應您之前的清單部分請求傳回的標記值。

您可以透過在請求中指定 `limit` 參數來限制回應中傳回的部分數量。

請求

語法

要列出正在進行的分段上傳的部分，請將 GET 請求傳送到分段上傳 ID 資源的 URI。當您啟動分段上傳時，將傳回分段上傳 ID (啟動分段上傳 (POST 分段 - 上傳))。您可以選擇指定 marker 和 limit 參數。

```
GET /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

名稱	描述	必要
limit	所要傳回的部分數量上限。預設限制為 50。傳回的部分數量可能少於指定的限制，但傳回的部分數量永遠不會超過限制。 類型：字串 限制：最小整數值為 1。最大整數值為 50。	否
marker	用於分頁的不透明字串。marker 指定部分清單應開始的部分。從之前的清單部分回應的回應中獲取 marker 值。如果您要繼續對之前的清單部分請求中開始的結果進行分頁，則只需包含 marker。 類型：字串	否

名稱	描述	必要
	限制條件：無	

請求標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
    "ArchiveDescription" : String,
    "CreationDate" : String,
    "Marker": String,
    "MultipartUploadId" : String,
    "PartSizeInBytes" : Number,
    "Parts" :
    [ {
      "RangeInBytes" : String,
      "SHA256TreeHash" : String
      },
      ...
     ],
    "VaultARN" : String
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

ArchiveDescription

在啟動分段上傳請求中指定的封存說明。如果在啟動分段上傳操作中未指定封存說明，則此欄位為 null。

類型：字串

CreationDate

啟動分段上傳的 UTC 時間。

類型：字串 ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

Marker

一個不透明字串，表示繼續結果分頁之處。您可以在新的清單部分請求中使用 marker 來取得清單中的更多任務。如果沒有更多的部分列出，則此值為 null。

類型：字串

MultipartUploadId

與部分關聯的上傳的 ID。

類型：字串

PartSizeInBytes

部分大小 (以位元組為單位)。這與您在啟動分段上傳請求中指定的值相同。

類型：數字

部分

分段上傳的部分大小清單。陣列中的每個物件都包含 RangeBytes 和 sha256-tree-hash 名稱/值對。

類型：陣列

RangeInBytes

部分的位元組範圍，包含範圍的上限值。

類型：字串

SHA256TreeHash

S3 Glacier 為部分所計算的 SHA256 樹雜湊值。此欄位永遠不為 null。

類型：字串

VaultARN

啟動分段上傳的文件庫的 Amazon Resource Name (ARN)。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例：列出分段上傳的部分

以下範例列出了上傳的所有部分。此範例將 HTTP GET 請求傳送到正在進行的分段上傳的特定分段上傳 ID 的 URI，最多可傳回 1,000 個部分。

範例請求

```
GET /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

S3 Glacier 會在回應中，傳回與指定的分段上傳 ID 關聯之上傳部分的清單。在這個範例中，只有兩個部分。傳回的 Marker 欄位是 null，表示沒有更多部分的分段上傳。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
```

```
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 412

{
    "ArchiveDescription" : "archive description",
    "CreationDate" : "2012-03-20T17:03:43.221Z",
    "Marker": null,
    "MultipartUploadId" :
 "OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE",
    "PartSizeInBytes" : 4194304,
    "Parts" :
    [ {
      "RangeInBytes" : "0-4194303",
      "SHA256TreeHash" : "01d34dabf7be316472c93b1ef80721f5d4"
      },
      {
      "RangeInBytes" : "4194304-8388607",
      "SHA256TreeHash" : "0195875365afda349fc21c84c099987164"
      }],
    "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/demo1-vault"
}
```

範例：列出分段上傳的部分 (指定標記和限制請求參數)

以下範例示範如何使用分頁來取得有限數量的結果。此範例將 HTTP GET 請求傳送到正在進行的分段上傳的特定分段上傳 ID 的 URI，以傳回一部分。起始 marker 參數指定在哪些部分上啟動部分清單。您可以從對部分清單的上一個請求的回應中獲取 marker 值。此外，在這個範例中，limit 參數設為 1 和傳回一個部分。請注意，Marker 欄位不是 null，表示至少還有一個要獲取的部分。

範例請求

```
GET /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE?marker=1001&limit=1 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

S3 Glacier 會在回應中，傳回與指定之進行中分段上傳 ID 關聯的上傳部分清單。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: text/json
Content-Length: 412

{
    "ArchiveDescription" : "archive description 1",
    "CreationDate" : "2012-03-20T17:03:43.221Z",
    "Marker": "MfgsKHVjbQ6EldVl72bn3_n5h2TaGZQUO-Qb3B9j3TITf7WajQ",
    "MultipartUploadId" :
 "OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE",
    "PartSizeInBytes" : 4194304,
    "Parts" :
    [ {
      "RangeInBytes" : "4194304-8388607",
      "SHA256TreeHash" : "01d34dabf7be316472c93b1ef80721f5d4"
      }],
    "VaultARN" : "arn:aws:glacier:us-west-2:012345678901:vaults/demo1-vault"
}
```

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 中止分段上傳 (DELETE uploadID)
- 列出分段上傳 (GET 分段 - 上傳)
- 上傳分段中的大型封存 (分段上傳)
- Amazon S3 Glacier 的 Identity and Access Management

列出分段上傳 (GET 分段 - 上傳)

描述

這個分段上傳操作列出了用於指定的文件庫的進行中分段上傳。進行中的分段上傳是已由 啟動分段上傳 (POST 分段 - 上傳) 請求啟動，但尚未完成或停止的分段上傳。「列出分段上傳」回應中傳回的清單中，並沒有保證順序。

「列出分段上傳」操作支援分頁。在預設情況下，在回應中這項操作會傳回最多 50 個分段上傳。您應該經常檢查回應內文 `marker` 欄位中註明可繼續列表的標記，如果沒有其他項目，`marker` 欄位為 `null`。

如果 `marker` 不是 Null，若要擷取下一組分段上傳，可將另一個「列出分段上傳」請求與 `marker` 請求參數，設定為 Amazon S3 Glacier (S3 Glacier) 為回應您之前「列出分段上傳」請求所傳回的標記值。

請注意這個操作和 清單部分 (GET uploadID) 操作之間的不同。「列出分段上傳」操作列出了文件庫的所有分段上傳。「列出分段」操作會傳回上傳 ID 識別之特定分段上傳部分。

如需有關分段上傳的資訊，請參閱上傳分段中的大型封存 (分段上傳)。

請求

語法

若要列出段上傳，可將 GET 請求傳送到文件庫的 `multipart-uploads` 子資源的 URI。您可以選擇指定 `marker` 和 `limit` 參數。

```
GET /AccountId/vaults/VaultName/multipart-uploads HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

名稱	描述	必要
limit	指定回應內文中傳回的上傳數量上限。如果未指定，「列出上傳」操作會傳回最多 50 個上傳。 類型：字串 限制：最小整數值為 1。最大整數值為 50。	否
marker	用於分頁的不透明字串。marker 指定上傳清單應開始的上傳部分。從之前的「列出上傳」回應中取得 marker 值。如果您要繼續對之前的「列出上傳」請求中開始的結果進行分頁，則只需包含 marker。 類型：字串 限制條件：無	否

請求標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
```

```
  "Marker": String,
  "UploadsList" : [
    {
      "ArchiveDescription": String,
      "CreationDate": String,
      "MultipartUploadId": String,
      "PartSizeInBytes": Number,
      "VaultARN": String
    },
   ...
  ]
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

ArchiveDescription

在啟動分段上傳請求中指定的封存說明。如果在啟動分段上傳操作中未指定封存說明，則此欄位為 null。

類型：字串

CreationDate

啟動分段上傳的 UTC 時間。

類型：字串 ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

Marker

一個不透明字串，表示繼續結果分頁之處。您在新的「列出分段上傳」請求中使用 marker 來取得清單中的更多上傳。如果沒有更多的上傳，則此值為 null。

類型：字串

PartSizeInBytes

啟動分段上傳 (POST 分段 - 上傳) 請求中指定的分段大小。這是在上傳的所有部分的大小，除了最後一個分段外，其可能小於此大小。

類型：數字

MultipartUploadId

分段上傳的 ID。

類型：字串

UploadsList

關於分段上傳物件的中繼資料清單。清單中的每一個項目包含一組用於對應上傳的名稱值的配對，包括 ArchiveDescription CreationDate、MultipartUploadId、PartSizeInBytes 和 VaultARN。

類型：陣列

VaultARN

文件庫的 Amazon Resource Name (ARN)，其中包含存檔。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例：列出所有分段上傳

以下範例列出文件庫所有進行中的分段上傳。範例對指定之文件庫的 GET 子資源的 URI 顯示 HTTP multipart-uploads 請求。由於在請求中未指定 marker 和 limit 參數，因此傳回最多 1,000 個進行中分段上傳。

範例請求

```
GET /-/vaults/examplevault/multipart-uploads HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

在回應中，S3 Glacier 針對指定的文件庫傳回所有進行中分段上傳的清單。marker 欄位為 null，其表示沒有其他上傳可列出。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 1054

{
  "Marker": null,
  "UploadsList": [
    {
      "ArchiveDescription": "archive 1",
      "CreationDate": "2012-03-19T23:20:59.130Z",
      "MultipartUploadId":
 "xsQdFIRsfJr20CW2AbZBKpRZAFTZSJIMtL2hYf8mvp8dM0m4RUzlaqoEye6g3h3ecqB_zqwB7zLDMeSWhwo65re4C4Ev"
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "ArchiveDescription": "archive 2",
      "CreationDate": "2012-04-01T15:00:00.000Z",
      "MultipartUploadId": "nPyGOnyFcx67qqX7E-0tSGiRi88hHMOwOxR-
_jNyM6RjVMFfV29lFqZ3rNsSaWBugg6OP92pRtufeHdQH7ClIpSF6uJc",
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "ArchiveDescription": "archive 3",
      "CreationDate": "2012-03-20T17:03:43.221Z",
      "MultipartUploadId": "qt-RBst_7yO8gVIonIBsAxr2t-db0pE4s8MNeGjKjGdNpuU-
cdSAcqG62guwV9r5jh5mLyFPzFEitTpNE7iQfHiu1XoV",
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ]
}
```

範例：部分列出分段上傳

以下範例示範如何使用分頁來取得有限數量的結果。範例對指定之文件庫的 GET 子資源的 URI 顯示 HTTP multipart-uploads 請求。在這個範例中，limit 參數設為 1，其表示只傳回清單中一個上傳，而 marker 參數指出傳回清單開始進行的分段上傳 ID。

範例請求

```
GET /-/vaults/examplevault/multipart-uploads?
limit=1&marker=xsQdFIRsfJr20CW2AbZBKpRZAFTZSJIMtL2hYf8mvp8dM0m4RUzlaqoEye6g3h3ecqB_zqwB7zLDMeSW
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

在回應中，Amazon S3 Glacier (S3 Glacier) 會針對指定的文件庫傳回一個不超過兩個進行中分段上傳的清單，從指定的標記開始，並傳回兩個結果。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 470

{
  "Marker": "qt-RBst_7yO8gVIonIBsAxr2t-db0pE4s8MNeGjKjGdNpuU-
cdSAcqG62guwV9r5jh5mLyFPzFEitTpNE7iQfHiu1XoV",
  "UploadsList" : [
    {
      "ArchiveDescription": "archive 2",
      "CreationDate": "2012-04-01T15:00:00.000Z",
      "MultipartUploadId": "nPyGOnyFcx67qqX7E-0tSGiRi88hHMOwOxR-
_jNyM6RjVMFfV29lFqZ3rNsSaWBugg6OP92pRtufeHdQH7ClIpSF6uJc",
      "PartSizeInBytes": 4194304,
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ]
```

```
}
```

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 中止分段上傳 (DELETE uploadID)
- 清單部分 (GET uploadID)
- 上傳分段中的大型封存 (分段上傳)
- Amazon S3 Glacier 的 Identity and Access Management

分段上傳 (PUT uploadID)

描述

這個分段上傳操作會上傳封存的一部分。您可以任何順序上傳封存部分，因為在您的分段上傳請求中您指定將在部分中上傳之整合封存的位元組範圍。您也可以平行上傳這些部分。您可以上傳多達 10,000 個部分的分段上傳。

如需有關分段上傳的資訊，請參閱上傳分段中的大型封存 (分段上傳)。

如果以下情況為 true 時，Amazon S3 Glacier (S3 Glacier) 會拒絕上傳部分請求：

- SHA256 樹雜湊不相符：為確保部分資料不會在傳輸時毀損，您運算該部分的 SHA256 樹雜湊並將其包含在請求中。在收到該部分資料時，S3 Glacier 也會運算 SHA256 樹雜湊。如果兩個雜湊值不相符，操作就會失敗。如需有關運算 SHA256 樹雜湊的資訊，請參閱 運算檢查總和。
- SHA256 線性雜湊不相符：由於授權需要，您運算整個上傳承載的 SHA256 線性雜湊，並將其包含在請求中。如需有關運算 SHA256 線性雜湊的資訊，請參閱運算檢查總和。
- 部分大小不相符：每個部分 (最後部分除外) 的大小，必須符合在對應的 啟動分段上傳 (POST 分段 - 上傳) 請求中所指定的大小。最後分段的大小必須與指定的大小相同或小於指定的大小。

> **Note**
>
> 如果您所上傳分段的大小小於在起始分段上傳請求中所指定的分段大小，而且該分段不是最後一個分段，則上傳分段請求將會成功。不過，後續的「完成分段上傳」請求將會失敗。

- 範圍不符合：請求中的位元組範圍值，不符合對應的起始請求中所指定的部分大小。例如，如果您指定 4194304 位元組 (4 MB) 的分段大小，則 0 到 4194303 位元組 (4 MB -1) 和 4194304 (4 MB) 到 8388607 (8 MB -1) 是有效的分段範圍。不過，如果您設定範圍值為 2 MB 到 6 MB，範圍就不符合分段大小且上傳將會失敗。

此為等冪操作。如果您多次上傳相同的分段，最新請求中所包含的資料會覆寫之前上傳的資料。

請求

您傳送此 HTTP PUT 請求至由您起始分段上傳請求所傳回的上傳 ID 的 URI。S3 Glacier 使用上傳 ID，來建立分段上傳與特定部分上傳之間的關聯。請求必須包含分段資料的 SHA256 樹雜湊 (x-amz-SHA256-tree-hash 標頭)、整個承載的 SHA256 線性雜湊 (x-amz-content-sha256 標頭)、位元組範圍 (Content-Range 標頭)，以及以位元組表示的分段長度 (Content-Length 標頭)。

語法

```
PUT /AccountId/vaults/VaultName/multipart-uploads/uploadID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Range: ContentRange
Content-Length: PayloadSize
Content-Type: application/octet-stream
x-amz-sha256-tree-hash: Checksum of the part
x-amz-content-sha256: Checksum of the entire payload
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

除了所有操作通用的請求標頭之外，此操作還會使用下列請求標頭。如需常見請求標頭的資訊，請參閱常見請求標題。

名稱	描述	必要
Content-Length	識別以位元組表示的分段長度。 類型：字串 預設：無 限制條件：無	否
Content-Range	識別在此分段將上傳之整合封存的位元組範圍。S3 Glacier 使用此資訊以正確的順序組合封存。此標頭的格式遵循 RFC 2616。範例標頭為 Content-Range:bytes 0-4194303/* 。 類型：字串 預設：無 限制條件：範圍不得大於您在起始分段上傳時所指定的分段大小。	是
x-amz-content-sha256	所上傳承載的 SHA256 檢查總和 (線性雜湊)。這與您在 x-amz-sha256-tree-hash 標頭中指定的值不同。 類型：字串 預設：無	是

名稱	描述	必要
	限制條件：無	
x-amz-sha256-tree-hash	指定將上傳之資料的 SHA256 樹雜湊。如需有關運算 SHA256 樹雜湊的資訊，請參閱 運算檢查總和。 類型：字串 預設：無 限制條件：無	是

請求主體

請求內文包含要上傳的資料。

回應

成功上傳部分時，S3 Glacier 傳回 204 No Content 的回應。

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
x-amz-sha256-tree-hash	S3 Glacier 為上傳的部分所運算的 SHA256 樹雜湊。 類型：字串

回應內文

此操作不會傳回任何回應內文。

範例

以下請求上傳 4 MB 的分段。請求會設定位元組範圍以將此成為封存的第一個分段。

範例請求

範例傳送 HTTP PUT 請求以上傳 4 MB 的分段。請求是傳送至由起始分段上傳請求所傳回之上傳 ID 的 URI。Content-Range 標頭將分段識別為封存的第一個 4 MB 資料分段。

```
PUT /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range:bytes 0-4194303/*
x-amz-sha256-tree-hash:c06f7cd4baacb087002a99a5f48bf953
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
Content-Length: 4194304
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
amz-glacier-
version,Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace
```

若要上傳下一個分段，程序是相同的；不過，您必須計算要上傳分段的新 SHA256 樹雜湊，也可以指定新的位元組範圍，以指出分段將進入最終組件的哪裡。以下請求會上傳使用相同上傳 ID 的另一個分段。請求指定之前請求之後的下一個 4 MB 封存和 4 MB 的分段大小。

```
PUT /-/vaults/examplevault/multipart-uploads/
OW2fM5iVylEpFEMM9_HpKowRapC3vn5sSL39_396UW9zLFUWVrnRHaPjUJddQ5OxSHVXjYtrN47NBZ-
khxOjyEXAMPLE HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range:bytes 4194304-8388607/*
Content-Length: 4194304
x-amz-sha256-tree-hash:f10e02544d651e2c3ce90a4307427493
x-amz-content-sha256:726e392cb4d09924dbad1cc0ba3b00c3643d03d14cb4b823e2f041cff612a628
x-amz-glacier-version: 2012-06-01
Authorization: Authorization=AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20120525/
us-west-2/glacier/aws4_request, SignedHeaders=host;x-amz-content-sha256;x-amz-date;x-
```

```
amz-glacier-version,
 Signature=16b9a9e220a37e32f2e7be196b4ebb87120ca7974038210199ac5982e792cace
```

可以任何順序上傳部分；S3 Glacier 對於每個部分會使用範圍規格，以決定組合部分的順序。

回應範例

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
x-amz-sha256-tree-hash: c06f7cd4baacb087002a99a5f48bf953
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

相關章節

- 啟動分段上傳 (POST 分段 - 上傳)
- 分段上傳 (PUT uploadID)
- 完成分段上傳 (POST uploadID)
- 中止分段上傳 (DELETE uploadID)
- 列出分段上傳 (GET 分段 - 上傳)
- 清單部分 (GET uploadID)
- 上傳分段中的大型封存 (分段上傳)
- Amazon S3 Glacier 的 Identity and Access Management

任務操作

以下是 S3 Glacier 中可用的任務操作。

主題

- 描述任務 (GET JobID)
- 「取得任務輸出」 (GET 輸出)
- 啟動任務 (POST 任務)
- 列出工作 (GET 工作)

描述任務 (GET JobID)

描述

此操作傳回您之前啟動任務的相關資訊，包括任務啟動日期、啟動任務的使用者、任務狀態碼/訊息，以及在 Amazon S3 Glacier (S3 Glacier) 完成任務後要通知的 Amazon Simple Notification Service (Amazon SNS) 主題。如需有關啟動任務的詳細資訊，請參閱 啟動任務 (POST 任務)。

> **Note**
>
> 此操作可讓您檢查任務的狀態。但是，我們強烈建議您設定 Amazon SNS 主題，並在啟動任務請求中指定該主題，以便在完成任務後，S3 Glacier 可以通知該主題。

S3 Glacier 完成任務後，任務 ID 至少在 24 小時內不會過期。

請求

語法

若要取得有關任務時的資訊，可以使用 HTTP GET 方法，並將請求範圍限定於特定任務。請注意，相對 URI 路徑與啟動任務時 S3 Glacier 傳回給您的路徑是相同的。

```
GET /AccountID/vaults/VaultName/jobs/JobID HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: date
Authorization: signatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

> **Note**
>
> 在請求中，如果您省略 JobID，該回應將傳回在指定文件庫上的所有作用中任務的清單。如需有關列出任務的詳細資訊，請參閱 列出工作 (GET 工作)。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: Length

{
    "Action": "string",
    "ArchiveId": "string",
    "ArchiveSHA256TreeHash": "string",
    "ArchiveSizeInBytes": number,
    "Completed": boolean,
    "CompletionDate": "string",
    "CreationDate": "string",
    "InventoryRetrievalParameters": {
        "EndDate": "string",
        "Format": "string",
        "Limit": "string",
        "Marker": "string",
        "StartDate": "string"
    },
```

```
    "InventorySizeInBytes": number,
    "JobDescription": "string",
    "JobId": "string",
    "JobOutputPath": "string",
    "OutputLocation": {
        "S3": {
            "AccessControlList": [
                {
                    "Grantee": {
                        "DisplayName": "string",
                        "EmailAddress": "string",
                        "ID": "string",
                        "Type": "string",
                        "URI": "string"
                    },
                    "Permission": "string"
                }
            ],
            "BucketName": "string",
            "CannedACL": "string",
            "Encryption": {
                "EncryptionType": "string",
                "KMSContext": "string",
                "KMSKeyId": "string"
            },
            "Prefix": "string",
            "StorageClass": "string",
            "Tagging": {
                "string": "string"
            },
            "UserMetadata": {
                "string": "string"
            }
        }
    },
    "RetrievalByteRange": "string",
    "SelectParameters": {
        "Expression": "string",
        "ExpressionType": "string",
        "InputSerialization": {
            "csv": {
                "Comments": "string",
                "FieldDelimiter": "string",
                "FileHeaderInfo": "string",
```

```
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "RecordDelimiter": "string"
            }
        },
        "OutputSerialization": {
            "csv": {
                "FieldDelimiter": "string",
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "QuoteFields": "string",
                "RecordDelimiter": "string"
            }
        }
    },
    "SHA256TreeHash": "string",
    "SNSTopic": "string",
    "StatusCode": "string",
    "StatusMessage": "string",
    "Tier": "string",
    "VaultARN": "string"
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

Action

工作類型。它是 ArchiveRetrieval、InventoryRetrieval 或 Select。

類型：字串

封存

針對選擇或封存擷取任務請求的封存 ID。否則，此欄位為 null。

類型：字串

ArchiveSHA256TreeHash

用於封存任務的整個封存的 SHA256 樹狀雜湊。對於庫存擷取作業，此欄位為 null。

類型：字串

ArchiveSizeInBytes

對於 ArchiveRetrieval 任務，這是請求下載的封存的大小，以位元組為單位。對於 InventoryRetrieval 任務，值為 null。

類型：數字

已完成

工作狀態。當封存或庫存擷取任務完成後，您可以使用 「取得任務輸出」 (GET 輸出) 獲取任務的輸出。

類型：布林值

CompletionDate

任務請求完成的國際標準時間 (UTC) 的時間。當任務正在進行時，該值為空。

類型：字串

CreationDate

建立任務所需的 UTC 時間。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

InventoryRetrievalParameters

用於各種庫存擷取的輸入參數。

類型：InventoryRetrievalJobInput物件

InventorySizeInBytes

對於 InventoryRetrieval 任務，這是請求下載的庫存的大小，以位元組為單位。對於 ArchiveRetrieval 或 Select 任務，值為 null。

類型：數字

JobDescription

當您啟動的任務所提供的任務說明。

類型：字串

JobId

在 S3 Glacier 中識別任務的 ID。

類型：字串

JobOutputPath

包含任務輸出位置。

類型：字串

OutputLocation

一個物件，其中包含有關儲存選取任務結果和錯誤的位置的資訊。

類型：OutputLocation物件

RetrievalByteRange

用於封存擷取任務的擷取位元組範圍，格式為「*StartByteValue-EndByteValue*。」 如果封存擷取中沒有指定範圍, 則擷取整個封存，也就 StartByteValue 等於 0，而 EndByteValue 等於封存的大小減去 1。對於庫存擷取或選擇作業，此欄位為 null。

類型：字串

SelectParameters

一個物件，其中包含有關用於選擇的參數的資訊。

類型：SelectParameters物件

SHA256TreeHash

所請求的封存範圍的 SHA256 樹狀雜湊值。如果對封存的 啟動任務 (POST 任務) 請求指定了樹狀雜湊值的範圍，則此欄位會傳回一個值。如需關於封存範圍擷取的樹狀雜湊值的詳細資訊，請參閱下載資料時接收檢查總和。

對於擷取整個封存的特定案例，此值與 ArchiveSHA256TreeHash 值相同。

此欄位在以下情況下為 null：

- 封存擷取任務所指定的範圍不符合樹狀雜湊。
- 指定與整個封存和任務狀態相等的範圍的封存任務是 InProgress。
- 庫存任務
- 選取任務。

類型：字串

SNSTopic

接收通知的 Amazon SNS 主題。

類型：字串

StatusCode

表示任務狀態的代碼。

有效值：`InProgress` | `Succeeded` | `Failed`

類型：字串

StatusMessage

描述任務狀態的友善訊息。

類型：字串

層

用於選擇或封存擷取的資料存方案。

有效值：`Bulk` | `Expedited` | `Standard`

類型：字串

VaultARN

該任務是子資源的文件庫 Amazon Resource Name (ARN)。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例顯示對擷取封存的任務的請求。

範例請求：取得任務描述

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID HTTP/1.1
```

```
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

回應內文包含描述指定任務的 JSON。請注意，對於庫存擷取和封存擷取任務，JSON 欄位是相同的。但是，當欄位不適用於任務類型時，其值為 null。以下是封存擷取作業的範例回應。注意下列事項：

- Action欄位值為 ArchiveRetrieval。
- ArchiveSizeInBytes 欄位會顯示在封存擷取作業中請求的封存大小。
- ArchiveSHA256TreeHash 欄位顯示整個封存的 SHA256 樹狀雜湊。
- RetrievalByteRange 欄位顯示在啟動任務請求中請求的範圍。在這個範例中，請求整個封存。
- SHA256TreeHash 欄位顯示在啟動任務請求中請求的範圍的 SHA256 樹狀雜湊。在這個範例中，它與 ArchiveSHA256TreeHash 欄位相同，這表示請求整個封存。
- 此 InventorySizeInBytes 欄位值為 null，因為它不適用於封存擷取任務。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 419
{
  "Action": "ArchiveRetrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "ArchiveSizeInBytes": 16777216,
  "ArchiveSHA256TreeHash":
 "beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60",
  "Completed": false,
  "CompletionDate": null,
  "CreationDate": "2012-05-15T17:21:39.339Z",
  "InventorySizeInBytes": null,
  "JobDescription": "My ArchiveRetrieval Job",
  "JobId": "HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
```

```
  "RetrievalByteRange": "0-16777215",
  "SHA256TreeHash": "beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60",
  "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
  "StatusCode": "InProgress",
  "StatusMessage": "Operation in progress.",
  "Tier": "Bulk",
  "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

以下是庫存擷取任務的範例回應。注意下列事項：

- Action欄位值為 InventoryRetrieval。
- ArchiveSizeInBytes、ArchiveSHA256TreeHash 和 RetrievalByteRange 欄位值為 null，因為這些欄位不適用於庫存擷取任務。
- InventorySizeInBytes 欄位值是 null ，因為該任務仍在進行中，而且要下載的庫存尚未完全準備好。如果任務在描述任務請求之前完成，則此欄位將給出輸出的大小。

```
{
   "Action": "InventoryRetrieval",
   "ArchiveId": null,
   "ArchiveSizeInBytes": null,
   "ArchiveSHA256TreeHash": null,
   "Completed": false,
   "CompletionDate": null,
   "CreationDate": "2012-05-15T23:18:13.224Z",
   "InventorySizeInBytes": null,
   "JobDescription": "Inventory Description",
   "JobId": "HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID",
   "RetrievalByteRange": null,
   "SHA256TreeHash": null,
   "SNSTopic": "arn:aws:sns:us-west-2:012345678901:mytopic",
   "StatusCode": "InProgress",
   "StatusMessage": "Operation in progress.",
   "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
}
```

以下是已完成的清查擷取任務的範例回應，其中包含用來持續對文件庫清查擷取進行分頁的標記。

```
{
```

```
    "Action": "InventoryRetrieval",
    "ArchiveId": null,
    "ArchiveSHA256TreeHash": null,
    "ArchiveSizeInBytes": null,
    "Completed": true,
    "CompletionDate": "2013-12-05T21:51:13.591Z",
    "CreationDate": "2013-12-05T21:51:12.281Z",
    "InventorySizeInBytes": 777062,
    "JobDescription": null,
    "JobId": "sCC2RZNBF2nildYD_roe0J9bHRdPQUbDRkmTdg-mXi2u3lc49uW6TcEhDF2D9pB2phx-
BN30JaBru7PMyOlfXHdStzu8",
    "NextInventoryRetrievalMarker": null,
    "RetrievalByteRange": null,
    "SHA256TreeHash": null,
    "SNSTopic": null,
    "StatusCode": "Succeeded",
    "StatusMessage": "Succeeded",
    "Tier": "Bulk",
    "VaultARN": "arn:aws:glacier-devo:us-west-2:836579025725:vaults/inventory-
icecube-2",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-11-12T13:43:12Z",
        "EndDate": "2013-11-20T08:12:45Z",
        "Limit": "120000",
        "Format": "JSON",
        "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    },
}
```

相關章節

- 「取得任務輸出」 (GET 輸出)
- Amazon S3 Glacier 的 Identity and Access Management

「取得任務輸出」 (GET 輸出)

描述

此操作可下載使用 啟動任務 (POST 任務) 啟動的任務的輸出。根據您在啟動任務時所指定的任務類型，輸出將是存檔或文件庫清查的內容。

您可以透過指定位元組範圍來下載所有任務輸出或下載部分輸出。對於封存和庫存擷取任務，應根據取得任務輸出回應中的標題中傳回的大小來驗證下載的大小。

對於封存擷取任務，您還應該驗證大小是否符合預期。如果您下載部分輸出，則預期的大小取決於您指定的位元組範圍。例如，如果您指定 `bytes=0-1048575` 範圍，則應驗證下載大小為 1,048,576 位元組。如果您下載整個封存，則預期大小是您上傳至 Amazon S3 Glacier (S3 Glacier) 時的封存大小。預期大小也會從取得任務輸出回應的標題中傳回。

對於封存擷取任務，根據指定的位元組範圍而定，S3 Glacier 會傳回資料部分的檢查總和。為確保下載的部分是正確的資料，請計算用戶端上的檢查總和，驗證值是否符合，並驗證大小是否符合您所期望的。

S3 Glacier 完成任務後，任務 ID 至少在 24 小時內不會過期。也就是說，您可以在 S3 Glacier 完成任務後的 24 小時內下載任務輸出。

請求

語法

若要擷取任務輸出，請將 HTTP GET 請求傳送到特定任務的 `output` 的 URI。

```
GET /AccountId/vaults/VaultName/jobs/JobID/output HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Range: ByteRangeToRetrieve
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

除了所有操作通用的請求標頭之外，此操作還會使用下列請求標頭。如需常見請求標頭的資訊，請參閱常見請求標題。

名稱	描述	必要
Range	從輸出中擷取的位元組範圍。例如，如果想要下載第一個 1,048,576 位元組，請將範圍指定為 `bytes=0-1048575` 。如需詳細資訊，請前往範圍標頭欄位定義。範圍與啟動任務請求中指定的任何範圍有關。在預設情況下，這個操作下載整個輸出。 如果任務輸出很大，則您可以使用 Range 請求標頭來擷取輸出的一部分。這可讓您可以以較小的位元組區塊下載整個輸出。例如，假設您有 1 GB 的任務輸出要下載，並且您決定一次下載 128 MB 的資料區塊，總共有八個「取得任務輸出」請求。您將使用以下程序下載任務輸出： 1. 透過使用 Range 標頭指定適當的位元組範圍，下載 128 MB 的輸出。驗證是否收到了所有 128 MB 的資料。 2. 與資料一起，回應將包括承載的檢查總和。您計算用戶端上承載的檢查總和，並將其與回應中收到的檢查總和進行比較，以確保接收到所有預期資料。 3. 對所有八個 128 MB 輸出資料區塊重複步驟 1 和 2，每次指定適當的位元組範圍。 4. 下載任務輸出的所有部分後，您有 8 個檢查總和值的清單。計算這些值的樹狀雜湊以尋找整個輸出的檢查總和。使用 描述任務 (GET JobID) 操作，取得為您提供輸出的任務的任務資訊。此回應包含存放在 S3 Glacier 中整個封存的檢查總和。您可以將此值與計算的檢查總和進行比較，以確保下載的整個存檔封存內容沒有錯誤。 類型：字串 預設：無	否

名稱	描述	必要
	限制條件：無	

請求主體

此操作沒有請求內文。

回應

語法

對於傳回所有任務資料的擷取請求，任務輸出回應將傳回 200 OK 回應程式碼。當請求部分內容時，例如，如果您在請求中指定了 Range 標頭，，則會傳回回應的程式碼 206 Partial Content。

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: ContentType
Content-Length: Length
x-amz-sha256-tree-hash: ChecksumComputedByAmazonGlacier

[Body containing job output.]
```

回應標頭

標頭	描述
Content-Range	S3 Glacier 傳回的位元組範圍。如果只下載部分輸出，則回應將提供 S3 Glacier 傳回的位元組範圍。 例如，bytes 0-1048575/8388608 從 8 MB中傳回第一個 1 MB。 如需有關 Content-Range 標頭的詳細資訊，請參閱內容 - 範圍標頭欄位定義。 類型：字串
Content-Type	內容類型取決於任務輸出是存檔還是文件庫清查。

標頭	描述
	• 對於封存資料，內容類型為 application/octet-stream 。 • 對於文件庫清查，如果您在啟動任務時請求了 CSV 格式，則內容類型為 text/csv。否則，在預設情況下，文件庫清查做為 JSON 傳回，並且內容類型為 application/json 。 類型：字串
x-amz-sha256-tree-hash	在回應中資料的檢查總和。只有在擷取封存擷取任務的輸出時才傳回此標頭。此外，當啟動任務請求中請求的擷取資料範圍與樹狀雜湊符合，並且在「取得任務輸出」中下載的範圍也與樹狀雜湊符合時，會出現此標頭。如需有關樹狀雜湊符合範圍的詳細資訊，請參閱 下載資料時接收檢查總和。 例如，如果在啟動任務請求中指定了要擷取的樹狀雜湊符合範圍 (包括整個封存)，則在下列情況下，您將收到下載資料的檢查總和： • 您可以取得擷取資料的整個範圍。 • 您請求位元組範圍的擷取資料，其大小為百萬位元組 (1024 KB) 乘以 2 的冪，並且以請求範圍的大小的倍數開始和結束。例如，如果您有 3.1 MB 的擷取資料，並且您指定了一個從 1 MB 開始到 2 MB 結束的範圍，則 x-amz-sha256-tree-hash 將做為回應標頭傳回。 • 您請求傳回擷取資料的範圍，該範圍到達傳回資料的結尾，而該範圍的開始是擷取範圍大小的倍數，四捨五入到下一個為 2 的冪位，但不小於 1 百萬位元組 (1024 KB)。例如，如果您有 3.1 MB 的擷取資料，並且您指定了一個從 2 MB 開始，並以 3.1 MB 結束 (資料的結尾) 的範圍，則 x-amz-sha256-tree-hash 將做為回應標頭傳回。 類型：字串

回應內文

S3 Glacier 在回應內文中傳回任務輸出。根據任務類型，輸出可以是存檔內容或文件庫清查。在文件庫清查情況下，根據預設，會以下列 JSON 內文傳回清查清單。

```
{
 "VaultARN": String,
 "InventoryDate": String,
 "ArchiveList": [
      {"ArchiveId": String,
       "ArchiveDescription": String,
       "CreationDate": String,
       "Size": Number,
       "SHA256TreeHash": String
      },
      ...
    ]
}
```

如果您在啟動文件庫清查任務時請求以逗號分隔值 (CSV) 輸出格式，則在內文中以 CSV 格式傳回文件庫清查。CSV 格式有五個欄「ArchiveId」、」 ArchiveDescription」、」 CreationDate」、「Size」和「SHA256TreeHash」，其定義與相應的 JSON 欄位相同。

> **Note**
>
> 在傳回的 CSV 格式中，欄位可能會以整個欄位用雙引號括起來。包含逗號或雙引號的欄位一律以雙引號括起來。例如，`my archive description,1` 做為 `"my archive description,1"` 傳回。在傳回的雙引號所括起的欄位中的雙引號字元，是藉由在其前面加上反斜線字元逸出。例如，`my archive description,1"2` 做為 `"my archive description,1\"2"` 傳回，`my archive description,1\"2` 做為 `"my archive description,1\\"2"` 傳回。反斜線字元不會逸出。

JSON 回應內文包含以下 JSON 欄位。

ArchiveDescription

封存的說明。

類型：字串

封存

封存的 ID。

類型：字串

ArchiveList

封存中繼資料的陣列。陣列中的每個物件表示文件庫中包含的一個存檔的中繼資料。

類型：陣列

CreationDate

建立封存的 UTC 日期和時間。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

InventoryDate

在文件庫變更後完成的文件庫的最後清查的 UTC 日期和時間。儘管 S3 Glacier 每天準備一次文件庫庫存，但只有在自上次清點以來已對文件庫進行封存新增或刪除時，才更新清點日期。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

SHA256TreeHash

封存的樹狀雜湊。

類型：字串

大小

封存的大小 (位元組)。

類型：數字

VaultARN

請求封存擷取的 Amazon Resource Name (ARN) 資源。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例顯示對擷取封存的任務的請求。

範例 1：下載輸出

本範例擷取 S3 Glacier 為回應啟動封存擷取任務請求而準備的資料。

範例請求

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/output
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

以下是封存擷取作業的範例回應。請注意，Content-Type 標頭是 application/octet-stream，並且 x-amz-sha256-tree-hash 標頭包含在回應中，這表示傳回所有任務資料。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
x-amz-sha256-tree-hash:
 beb0fe31a1c7ca8c6c04d574ea906e3f97b31fdca7571defb5b44dca89b5af60
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/octet-stream
Content-Length: 1048576

[Archive data.]
```

以下是庫存擷取任務的範例回應。請注意，Content-Type 標頭是 application/json。另請注意，回應不包含 x-amz-sha256-tree-hash 標頭。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
```

```
Content-Type: application/json
Content-Length: 906

{
 "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault",
 "InventoryDate": "2011-12-12T14:19:01Z",
 "ArchiveList": [
   {
     "ArchiveId": "DMTmICA2n5Tdqq5BV2z7og-
A20xnpAPKt3UXwWxdWsn_D6auTUrW6kwy5Qyj9xd1MCE1mBYvMQ63LWaT8yTMzMaCxB_9VBWrW4Jw4zsvg5kehAPDVKcppU
oA",
     "ArchiveDescription": "my archive1",
     "CreationDate": "2012-05-15T17:19:46.700Z",
     "Size": 2140123,
     "SHA256TreeHash":
 "6b9d4cf8697bd3af6aa1b590a0b27b337da5b18988dbcc619a3e608a554a1e62"
   },
   {
     "ArchiveId": "2lHzwhKhgF2JHyvCS-
ZRuF08IQLuyB4265Hs3AXj9MoAIhz7tbXAvcFeHusgU_hViO1WeCBe0N5lsYYHRyZ7rrmRkNRuYrXUs_sjl2K8ume_7mKO_
uHE1oHqaW9d37pabXrSA",
     "ArchiveDescription": "my archive2",
     "CreationDate": "2012-05-15T17:21:39.339Z",
     "Size": 2140123,
     "SHA256TreeHash":
 "7f2fe580edb35154041fa3d4b41dd6d3adaef0c85d2ff6309f1d4b520eeecda3"
   }
  ]
}
```

範例 2：僅下載部分輸出

本範例僅擷取 S3 Glacier 為回應啟動封存擷取任務請求而準備的封存部分。請求使用可選的 Range 標頭只擷取前 1,024 位元組。

範例請求

```
GET /-/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/output
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Range: bytes=0-1023
```

```
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

以下成功回應顯示 `206 Partial Content` 回應。在這種情況下，回應也包含 `Content-Range` 標頭，該標頭指定了 S3 Glacier 傳回的位元組範圍。

```
HTTP/1.1 206 Partial Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Range: bytes 0-1023/8388608
Content-Type: application/octet-stream
Content-Length: 1024

[Archive data.]
```

相關章節

- 描述任務 (GET JobID)
- 啟動任務 (POST 任務)
- Amazon S3 Glacier 的 Identity and Access Management

啟動任務 (POST 任務)

此操作會啟動下列類型的 Amazon S3 Glacier (S3 Glacier) 任務：

- `archive-retrieval`：擷取封存
- `inventory-retrieval`：清點文件庫

主題

- 初始化封存或文件庫庫存擷取任務
- 請求
- 回應

- 範例
- 相關章節

初始化封存或文件庫庫存擷取任務

擷取封存或文件庫庫存是非同步操作，需要您啟動任務。一旦啟動，就無法取消任務。擷取是兩個步驟：

1. 使用 啟動任務 (POST 任務) 操作啟動擷取任務。

> **Important**
>
> 資料擷取政策可能導致啟動擷取作業請求失敗，並出現 `PolicyEnforcedException`。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。如需 `PolicyEnforcedException` 例外狀況的詳細資訊，請參閱 錯誤回應。

2. 任務完成後，使用 「取得任務輸出」 (GET 輸出) 操作下載位元組。

以非同步方式執行擷取請求。當您開始擷取任務時，S3 Glacier 會建立任務並在回應中傳回任務 ID。當 S3 Glacier 完成任務時，您可以取得任務輸出 (封存或庫存資料)。如需有關取得任務輸出的詳細資訊，請參閱 「取得任務輸出」 (GET 輸出) 操作。

必須完成任務，才能取得其輸出。若要判斷任務何時完成，您有下列選項：

- 使用 Amazon SNS 通知：您可以指定在任務完成後，S3 Glacier 可以發佈通知的 Amazon SNS 主題。您可以為每個任務請求指定一個 SNS 主題。通知僅在 S3 Glacier 完成任務後傳送。除了指定每個任務請求的 SNS 主題外，還可以為文件庫設定文件庫通知，以便為所有擷取傳送任務通知。如需更多詳細資訊，請參閱 設定文件庫通知組態 (PUT 通知的組態)。
- 取得任務詳細資訊：任務進行中時，您可以提出 描述任務 (GET JobID) 請求以取得任務狀態資訊。但是，使用 Amazon SNS 通知來判斷任務何時完成會更有效率。

> **Note**
>
> 您透過通知取得的資訊，與您呼叫 描述任務 (GET JobID) 所取得的資訊相同。

如果對於特定事件，您將通知設定新增到文件庫中，並且在啟動任務請求中指定 SNS 主題，則 S3 Glacier 會同時傳送這兩個通知。如需更多詳細資訊，請參閱 設定文件庫通知組態 (PUT 通知的組態)。

此文件庫庫存

從您第一次將封存上傳到文件庫的那一天起，S3 Glacier 大約每天更新一次文件庫庫存。如果從上次清查以來，沒有新增或刪除文件庫的存檔，則清查日期不會更新。當您為文件庫庫存啟動任務時，S3 Glacier 會傳回其產生的最後一個庫存，這是一個時間點快照，而不是即時資料。

S3 Glacier 為文件庫建立第一個庫存後，通常需要半天到一天的時間才可以擷取庫存。

您可能沒有發現為每個存檔上傳擷取文件庫清查的好處。但是，假設您在用戶端上維護資料庫，該用戶端將上傳到 S3 Glacier 的封存相關中繼資料建立關聯。然後，您可能會發現文件庫清查的好處，可以視需要在資料庫中使用實際的文件庫清查來調節資訊。如需有關庫存任務輸出中傳回的資料欄位的詳細資訊，請參閱 回應內文。

庫存擷取範圍

您可以透過篩選在文件庫建立日期或設定限制來限制擷取到的文件庫庫存項目的數量。

透過封存建立日期進行篩選

您可以透過在啟動任務請求中指定這些參數的值，來擷取在 StartDate 和 EndDate 之間建立之封存的庫存項目。在 StartDate 之時或之後，以及 EndDate 之前所建立的封存被傳回。如果您只提供沒有 StartDate 的 EndDate ，則擷取 StartDate 或之後建立的所有封存的庫存。如果您只提供 EndDate 而沒有 StartDate，則擷取 EndDate 之時或之後所建立的所有封存的庫存。

限制每個擷取的庫存項目

您可以透過在啟動任務請求中設定 Limit 參數，來限制傳回的庫存項目數量。庫存任務輸出包含達到指定 Limit 的庫存項目。如果有更多的庫存項目可用，則結果會進行分頁。在任務完成後，您可以使用 描述任務 (GET JobID) 操作來取得您在後續的啟動任務請求中使用的標記。標記指示要擷取下一組庫存項目的起點。您可以透過使用之前描述任務輸出中的標記重複建立啟動任務請求來瀏覽整個庫存。這樣做一直到您從描述任務取得傳回空標記，表示沒有更多的庫存項目可用。

您可以將 Limit 參數與日期範圍參數一起使用。

遠端封存擷取

您可以為整個封存或封存範圍啟動封存擷取。在遠端封存擷取的情況下，指定要傳回的位元組範圍或整個封存。指定的範圍必須符合百萬位元組 (MB)。換言之，範圍起始值必須可被 1 MB 整除，並且範圍結束值加上 1 必須可整除 1 MB 或等於封存的結束。如果遠端封存擷取不符合 MB，則此操作將傳回

400 回應。此外，為了確保使用取得任務輸 (「取得任務輸出」 (GET 輸出)) 下載的資料取得檢查總和值，該範圍必須必須符合樹狀雜湊。如需有關樹狀雜湊符合範圍的詳細資訊，請參閱 下載資料時接收檢查總和。

急件、標準和大量方案

當啟動封存擷取任務時，可以在請求內文的 Tier 欄位中指定下列選項之一：

- **Expedited**：當您偶爾需要緊急要求還原封存時，急件讓您快速存取資料。對於幾乎最大型的封存 (250 MB 以上)，使用急件方案所存取的資料，通常會在 1-5 分鐘內即可使用。
- **Standard**：讓您能以標準方案在幾小時內存取任何封存。使用標準方案存取的資料通常 3 -5 小時內可用。若未指定方案選項，這會是任務請求的預設選項。
- **Bulk**：大量是 S3 Glacier 的最低成本方案，無須太多費用即可於一天內擷取大量資料 (甚至達到數 PB)。使用大量方案存取的資料通常 5 -12 小時內可用。

如需有關急件和大量擷取的詳細資訊，請參閱 使用 AWS 主控台擷取 S3 Glacier 封存。

請求

要啟動作業，您可以使用 HTTP POST 方法的請求，並將請求範圍擴大到文件庫的 jobs 子資源。您可以在請求的 JSON 文件中指定任務請求的詳細資訊。任務類型是由 Type 欄位指定的。或者，您可以指定 SNSTopic 欄位，以指示在任務完成後 S3 Glacier 可以發佈通知的 Amazon SNS 主題。

> **Note**
>
> 若要將通知發佈到 Amazon SNS，如果主題尚不存在，您必須自行建立該主題。S3 Glacier 不會為您建立主題。該主題必須有權接收來自 S3 Glacier 文件庫的發布。S3 Glacier 不驗證文件庫是否具有發布到該主題的許可。如果未設定適當的權限，即使任務完成後，您也可能不會收到通知。

語法

以下是啟動任務的請求語法。

```
POST /AccountId/vaults/VaultName/jobs HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
```

```
x-amz-glacier-version: 2012-06-01

{
   "jobParameters": {
      "ArchiveId": "string",
      "Description": "string",
      "Format": "string",
      "InventoryRetrievalParameters": {
         "EndDate": "string",
         "Limit": "string",
         "Marker": "string",
         "StartDate": "string"
      },
      "OutputLocation": {
         "S3": {
            "AccessControlList": [
               {
                  "Grantee": {
                     "DisplayName": "string",
                     "EmailAddress": "string",
                     "ID": "string",
                     "Type": "string",
                     "URI": "string"
                  },
                  "Permission": "string"
               }
            ],
            "BucketName": "string",
            "CannedACL": "string",
            "Encryption": {
               "EncryptionType": "string",
               "KMSContext": "string",
               "KMSKeyId": "string"
            },
            "Prefix": "string",
            "StorageClass": "string",
            "Tagging": {
               "string" : "string"
            },
            "UserMetadata": {
               "string" : "string"
            }
         }
      },
```

```
      "RetrievalByteRange": "string",
      "SelectParameters": {
         "Expression": "string",
         "ExpressionType": "string",
         "InputSerialization": {
            "csv": {
               "Comments": "string",
               "FieldDelimiter": "string",
               "FileHeaderInfo": "string",
               "QuoteCharacter": "string",
               "QuoteEscapeCharacter": "string",
               "RecordDelimiter": "string"
            }
         },
         "OutputSerialization": {
            "csv": {
               "FieldDelimiter": "string",
               "QuoteCharacter": "string",
               "QuoteEscapeCharacter": "string",
               "QuoteFields": "string",
               "RecordDelimiter": "string"
            }
         }
      },
      "SNSTopic": "string",
      "Tier": "string",
      "Type": "string"
   }
}
```

> **Note**
>
> 此 `AccountId` 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求主體

該請求接受請求內文中的 JSON 格式的以下資料。

jobParameters

提供指定任務資訊的選項。

類型：jobParameters物件

必要：是

回應

S3 Glacier 建立此任務。在回應中，它會傳回任務的 URI。

語法

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: location
x-amz-job-id: jobId
x-amz-job-output-path: jobOutputPath
```

回應標頭

標頭	描述
Location	任務的相對 URI 路徑。您可以使用此 URI 路徑來尋找任務狀態。如需更多詳細資訊，請參閱 描述任務 (GET JobID)。 類型：字串 預設：無
x-amz-job-id	任務的 ID。此值也包含在 Location 標頭中。 類型：字串 預設：無
x-amz-job-output-path	儲存選取結果的位置的路徑。

標頭	描述
	類型：字串 預設：無

回應內文

此操作不會傳回任何回應內文。

錯誤

除了所有 Amazon S3 Glacier 操作常見的可能錯誤以外，此操作還包括下列一或多個錯誤。如需 Amazon S3 Glacier 錯誤和錯誤代碼清單的相關資訊，請參閱錯誤回應。

代碼	描述	HTTP 狀態碼	類型
`InsufficientCapacityException`	如果沒有足夠的能力處理此急件請求，則退回。此錯誤僅適用於急件擷取，而不適用於標準或大量擷取。	`503 ServiceUnavailable`	伺服器

範例

範例請求：啟動封存擷取任務

```
POST /-/vaults/examplevault/jobs HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Type": "archive-retrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "Description": "My archive description",
```

```
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-ArchiveRetrieval-topic-
Example",
  "Tier" : "Bulk"
}
```

以下是一個請求內文的範例，該請求指定各種封存擷取使用 RetrievalByteRange 欄位。

```
{
  "Type": "archive-retrieval",
  "ArchiveId": "NkbByEejwEggmBz2fTHgJrg0XBoDfjP4q6iu87-
TjhqG6eGoOY9Z8i1_AUyUsuhPAdTqLHy8pTl5nfCFJmDl2yEZONi5L26Omw12vcs01MNGntHEQL8MBfGlqrEXAMPLEArchi
  "Description": "My archive description",
  "RetrievalByteRange": "2097152-4194303",
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-ArchiveRetrieval-topic-
Example",
  "Tier" : "Bulk"
}
```

回應範例

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

範例請求：啟動庫存擷取任務

以下請求將啟動一個庫存擷取任務，以便從 examplevault 文件庫中取得封存清單。在請求內文中的 Format 設為 CSV 表示庫存以 CSV 格式傳回。

```
POST /-/vaults/examplevault/jobs HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Content-Type: application/x-www-form-urlencoded
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

```
{
  "Type": "inventory-retrieval",
  "Description": "My inventory job",
  "Format": "CSV",
  "SNSTopic": "arn:aws:sns:us-west-2:111111111111:Glacier-InventoryRetrieval-topic-
Example"
}
```

回應範例

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
```

範例請求：通過使用一組限制的日期篩選和後續請求來擷取庫存項目的下一頁，啟動庫存擷取任務。

以下請求透過使用日期篩選和設定限制來啟動文件庫庫存擷取任務。

```
{
    "ArchiveId": null,
    "Description": null,
    "Format": "CSV",
    "RetrievalByteRange": null,
    "SNSTopic": null,
    "Type": "inventory-retrieval",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-12-04T21:25:42Z",
        "EndDate": "2013-12-05T21:25:42Z",
        "Limit" : "10000"
    },
}
```

以下的請求範例，後續請求使用來自 描述任務 (GET JobID) 所取得的標記來擷取庫存項目的下一頁。

```
{
    "ArchiveId": null,
```

```
    "Description": null,
    "Format": "CSV",
    "RetrievalByteRange": null,
    "SNSTopic": null,
    "Type": "inventory-retrieval",
    "InventoryRetrievalParameters": {
        "StartDate": "2013-12-04T21:25:42Z",
        "EndDate": "2013-12-05T21:25:42Z",
        "Limit": "10000",
        "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    },
}
```

回應範例

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Location: /111122223333/vaults/examplevault/jobs/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-id: HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID
x-amz-job-output-path: test/HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID/
```

相關章節

- 描述任務 (GET JobID)
- 「取得任務輸出」 (GET 輸出)
- Amazon S3 Glacier 的 Identity and Access Management

列出工作 (GET 工作)

描述

此操作列出文件庫的任務，包括正在進行的任務和最近完成的任務。

> **Note**
>
> Amazon S3 Glacier (S3 Glacier) 會在刪除之前，保留最近完成的任務一段時間；但是，最終會刪除完成的任務。可以擷取完成工作的輸出。在完成工作後，工作將保留一段時間，可使您可以在錯過工作完成通知的情況下取得工作輸出，否則您第一次嘗試下載工作時會失敗。例如，假設您啟動封存擷取工作以下載封存。工作完成後，您開始下載封存，但發生網路錯誤。在此案例中，您可以在工作存在時重試並下載封存。

List Jobs 操作支援分頁。您應該隨時檢查回應 Marker 欄位。如果沒有更多的工作要列出，Marker 欄位設定為 null。如果要列出更多工作，Marker 欄位將設定為非空值，您可以使用該值來繼續清單的分頁。若要傳回從特定工作開始的工作清單，請將 marker 請求參數設定為從之前的 Marker 請求取得的工作的 List Jobs 值。

您可以透過在請求中指定 limit 參數來設定回應中傳回的工作數量的最大限制。預設限制為 50。傳回的工作數可能少於限制，但傳回的工作數永遠不會超過限制。

此外，還可以透過指定選用的 statuscode 參數或 completed 參數或兩者來篩選傳回的工作清單。使用 statuscode 參數，可以指定僅返回與 InProgress、Succeeded 或 Failed 狀態符合的工作。使用 completed 參數，您可以指定只傳回已完成的 (true) 的工作或未完成的 (false) 的工作。

請求

語法

要傳回所有類型的任務清單, 請將 GET 請求傳送到文件庫的 jobs 子資源的 URI。

```
GET /AccountId/vaults/VaultName/jobs HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是擁有該文件庫之帳戶的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您使用帳號 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

名稱	描述	必要
completed	要傳回的工作狀態。您可指定為 true 或 false。 類型：布林值 限制條件：無	否
limit	所要傳回的工作數量上限。預設限制為 50。傳回的工作數可能少於指定的限制，但傳回的工作數永遠不會超過限制。 類型：字串 限制：最小整數值為 1。最大整數值為 50。	否
marker	用於分頁的不透明字串，用於指定工作清單應開始的工作。您可以從之前的 marker 回應中取得 List Jobs 值。如果您要繼續分析在之前的 marker 請求中啟動的結果，則只需包含 List Jobs 。 類型：字串 限制條件：無	否
statuscode	傳回的工作狀態類型。 類型：字串 限制：InProgress 、Succeeded 或 Failed 的其中一個值。	否

請求標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Location: Location
Content-Type: application/json
Content-Length: Length

{
    "JobList": [
        {
            "Action": "string",
            "ArchiveId": "string",
            "ArchiveSHA256TreeHash": "string",
            "ArchiveSizeInBytes": number,
            "Completed": boolean,
            "CompletionDate": "string",
            "CreationDate": "string",
            "InventoryRetrievalParameters": {
                "EndDate": "string",
                "Format": "string",
                "Limit": "string",
                "Marker": "string",
                "StartDate": "string"
            },
            "InventorySizeInBytes": number,
            "JobDescription": "string",
            "JobId": "string",
            "JobOutputPath": "string",
            "OutputLocation": {
                "S3": {
                    "AccessControlList": [
                        {
                            "Grantee": {
                                "DisplayName": "string",
                                "EmailAddress": "string",
                                "ID": "string",
```

```
                        "Type": "string",
                        "URI": "string"
                    },
                    "Permission": "string"
                }
            ],
            "BucketName": "string",
            "CannedACL": "string",
            "Encryption": {
                "EncryptionType": "string",
                "KMSContext": "string",
                "KMSKeyId": "string"
            },
            "Prefix": "string",
            "StorageClass": "string",
            "Tagging": {
                "string": "string"
            },
            "UserMetadata": {
                "string": "string"
            }
        }
    },
    "RetrievalByteRange": "string",
    "SelectParameters": {
        "Expression": "string",
        "ExpressionType": "string",
        "InputSerialization": {
            "csv": {
                "Comments": "string",
                "FieldDelimiter": "string",
                "FileHeaderInfo": "string",
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "RecordDelimiter": "string"
            }
        },
        "OutputSerialization": {
            "csv": {
                "FieldDelimiter": "string",
                "QuoteCharacter": "string",
                "QuoteEscapeCharacter": "string",
                "QuoteFields": "string",
                "RecordDelimiter": "string"
```

```
                    }
                }
            },
            "SHA256TreeHash": "string",
            "SNSTopic": "string",
            "StatusCode": "string",
            "StatusMessage": "string",
            "Tier": "string",
            "VaultARN": "string"
        }
    ],
    "Marker": "string"
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

JobList

工作物件的清單。每個工作物件包含描述工作的中繼資料。

類型：GlacierJobDescription 物件陣列

Marker

一個不透明字串，表示繼續結果分頁之處。您可以在新的 marker 請求中使用 List Jobs 值來取得清單中的更多工作。如果沒有更多的工作列出，則此值為 null。

類型：字串

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何傳回有關文件庫任務的資訊。第一個範例傳回兩個工作清單，第二個範例傳回部分工作。

範例：傳回所有工作

範例請求

以下 GET 請求傳回文件庫的任務。

```
GET /-/vaults/examplevault/jobs  HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

以下回應包括存檔擷取任務和清查擷取任務，其中包含用來持續對文件庫清查擷取進行分頁的標記。回應還會顯示 Marker 欄位設定為 null，這表示沒有更多的工作列出。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 1444

{
  "JobList": [
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "BDfaUQul0dVzYwAMr8YSa_6_8abbhZq-
i1oT69g8ByClfJyBgAGBkWl2QbF5os851P7Y7KdZDOHWJIn4rh1ZHaOYD3MgFhK_g0oDPesW34uHQoVGwoIqubf6BgUEfQm
      "ArchiveSizeInBytes": 1048576,
      "ArchiveSHA256TreeHash":
 "25499381569ab2f85e1fd0eb93c5406a178ab77c5933056eb5d6e7d4adda609b",
      "Completed": true,
      "CompletionDate": "2012-05-01T00:00:09.304Z",
      "CreationDate": "2012-05-01T00:00:06.663Z",
      "InventorySizeInBytes": null,
      "JobDescription": null,
      "JobId": "hDe9t9DTHXqFw8sBGpLQQOmIM0-
JrGtu1O_YFKLnzQ64548qJc667BRWTwBLZC76Ygy1jHYruqXkdcAhRsh0hYv4eVRU",
      "RetrievalByteRange": "0-1048575",
```

```
      "SHA256TreeHash":
 "25499381569ab2f85e1fd0eb93c5406a178ab77c5933056eb5d6e7d4adda609b",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "Tier": "Bulk",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "Action": "InventoryRetrieval",
      "ArchiveId": null,
      "ArchiveSizeInBytes": null,
      "ArchiveSHA256TreeHash": null,
      "Completed": true,
      "CompletionDate": "2013-05-11T00:25:18.831Z",
      "CreationDate": "2013-05-11T00:25:14.981Z",
      "InventorySizeInBytes": 1988,
      "JobDescription": null,
      "JobId":
 "2cvVOnBL36btzyP3pobwIceiaJebM1bx9vZOOUtmNAr0KaVZ4WkWgVjiPldJ73VU7imlm0pnZriBVBebnqaAcirZq_C5"
      "RetrievalByteRange": null,
      "SHA256TreeHash": null,
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
      "InventoryRetrievalParameters": {
          "StartDate": "2013-11-12T13:43:12Z",
          "EndDate": "2013-11-20T08:12:45Z",
          "Limit": "120000",
          "Format": "JSON",
          "Marker":
 "vyS0t2jHQe5qbcDggIeD50chS1SXwYMrkVKo0KHiTUjEYxBGCqRLKaiySzdN7QXGVVV5XZpNVG67pCZ_uykQXFMLaxOSu
    }
  ],
  "Marker": null
}
```

範例：回傳工作的局部清單

範例請求

以下 GET 請求傳回由 marker 參數指定的工作。將 limit 參數設定為 2 以指定最多傳回兩個工作。

```
GET /-/vaults/examplevault/jobs?marker=HkF9p6o7yjhFx-
K3CGl6fuSm6VzW9T7esGQfco8nUXVYwS0jlb5gq1JZ55yHgt5vP54ZShjoQzQVVh7vEXAMPLEjobID&limit=2
 HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

以下回應顯示傳回的兩個工作，Marker 欄位設定為非空值，可用來繼續工作清單的分頁。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 1744

{
  "JobList": [
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "58-3KpZfcMPUznvMZNPaKyJx9wODCsWTnqcjtx2CjKZ6b-
XgxEuA8yvZOYTPQfd7gWR4GRm2XR08gcnWbLV4VPV_kDWtZJKi0TFhKKVPzwrZnA4-
FXuIBfViYUIVveeiBE51FO4bvg",
      "ArchiveSizeInBytes": 8388608,
      "ArchiveSHA256TreeHash":
 "106086b256ddf0fedf3d9e72f461d5983a2566247ebe7e1949246bc61359b4f4",
      "Completed": true,
      "CompletionDate": "2012-05-01T00:25:20.043Z",
      "CreationDate": "2012-05-01T00:25:16.344Z",
      "InventorySizeInBytes": null,
      "JobDescription": "aaabbbccc",
      "JobId": "s4MvaNHIh6mOa1f8iY4ioG2921SDPihXxh3Kv0FBX-
JbNPctpRvE4c2_BifuhdGLqEhGBNGeB6Ub-JMunR9JoVa8y1hQ",
      "RetrievalByteRange": "0-8388607",
      "SHA256TreeHash":
 "106086b256ddf0fedf3d9e72f461d5983a2566247ebe7e1949246bc61359b4f4",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
```

```
      "Tier": "Bulk",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    },
    {
      "Action": "ArchiveRetrieval",
      "ArchiveId": "2NVGpf83U6qB9M2u-
Ihh61yoFLRDEoh7YLZWKBn80A2i1xG8uieBwGjAr4RkzOHA0E07ZjtI267R03Z-6Hxd8pyGQkBdciCSH1-
Lw63Kx9qKpZbPCdU0uTW_WAdwF6lR6w8iSyKdvw",
      "ArchiveSizeInBytes": 1048576,
      "ArchiveSHA256TreeHash":
 "3d2ae052b2978727e0c51c0a5e32961c6a56650d1f2e4ceccab6472a5ed4a0",
      "Completed": true,
      "CompletionDate": "2012-05-01T16:59:48.444Z",
      "CreationDate": "2012-05-01T16:59:42.977Z",
      "InventorySizeInBytes": null,
      "JobDescription": "aaabbbccc",
      "JobId":
 "CQ_tf6fOR4jrJCL61Mfk6VM03oY8lmnWK93KK4gLig1UPAbZiN3UV4G_5nq4AfmJHQ_dOMLOX5k8ItFv0wCPN0oaz5dG"
      "RetrievalByteRange": "0-1048575",
      "SHA256TreeHash":
 "3d2ae052b2978727e0c51c0a5e32961c6a56650d1f2e4ceccab6472a5ed4a0",
      "SNSTopic": null,
      "StatusCode": "Succeeded",
      "StatusMessage": "Succeeded",
      "Tier": "Standard",
      "VaultARN": "arn:aws:glacier:us-west-2:012345678901:vaults/examplevault"
    }
  ],
  "Marker":
 "CQ_tf6fOR4jrJCL61Mfk6VM03oY8lmnWK93KK4gLig1UPAbZiN3UV4G_5nq4AfmJHQ_dOMLOX5k8ItFv0wCPN0oaz5dG"
}
```


相關章節

- 描述任務 (GET JobID)
- Amazon S3 Glacier 的 Identity and Access Management

在任務操作中使用的資料類型

以下是用於 S3 Glacier 中任務操作的資料類型。

主題

CSVInput

包含有關逗號分隔值 (CSV) 檔案的資訊。

目錄

評論

單一字元用於表示當該字元出現在該資料列的開頭時應應該忽略該資料列。

類型：字串

必要：否

FieldDelimiter

單一字元用於在記錄中分隔個別欄位 此字元必須是 32-126 範圍內的 \n、\r 或 ASCII 字元。預設為逗號 (,)。

類型：字串

預設：,

必要：否

FileHeaderInfo

一個值，用於描述如何處理輸入的第一行。

類型：字串

有效值：`Use` | `Ignore` | `None`

必要：否

QuoteCharacter

用作逸出字元的單一字元，其中欄位分隔符號是該值的一部分。

類型：字串

必要：否

QuoteEscapeCharacter

單一字元，用於在已逸出的值內逸出引號字元。

類型：字串

必要：否

RecordDelimiter

單一字元用於分隔個別紀錄。

類型：字串

必要：否

詳細資訊

- 啟動任務 (POST 任務)

CSVOutput

包含有關儲存任務結果的逗號分隔值 (CSV) 格式的資訊。

目錄

FieldDelimiter

單一字元用於在記錄中分隔個別欄位

類型：字串

必要：否

QuoteCharacter

用作逸出字元的單一字元，其中欄位分隔符號是該值的一部分。

類型：字串

必要：否

QuoteEscapeCharacter

單一字元，用於在已逸出的值內逸出引號字元。

類型：字串

必要：否

QuoteFields

一個值，指示是否應將所有輸出欄位包含在引號內。

有效值：ALWAYS | ASNEEDED

類型：字串

必要：否

RecordDelimiter

單一字元用於分隔個別紀錄。

類型：字串

必要：否

詳細資訊

- 啟動任務 (POST 任務)

加密

包含有關用於在 Amazon S3 中儲存任務結果的加密資訊。

目錄

加密

將任務結果儲存在 Amazon S3 中時使用的伺服器端加密演算法。預設值是不加密。

類型：字串

有效值：`aws:kms | AES256`

必要：否

KMSContext

選用。如果加密類型是 `aws:kms`，，則可以使用此值來指定任務結果的加密內容。

類型：字串

必要：否

KMSKeyId

用於加密物件的 AWS Key Management Service (AWS KMS) 金鑰 ID。

類型：字串

必要：否

詳細資訊

- 啟動任務 (POST 任務)

GlacierJobDescription

包含 Amazon S3 Glacier (S3 Glacier) 任務的說明。

目錄

Action

工作類型。它是 ArchiveRetrieval、InventoryRetrieval 或 Select。

類型：字串

封存

針對選擇或封存擷取任務請求的封存 ID。否則，此欄位為 null。

類型：字串

ArchiveSHA256TreeHash

用於封存擷取的整個封存的 SHA256 樹狀雜湊。對於庫存擷取作業，此欄位為 null。

類型：字串

ArchiveSizeInBytes

對於 ArchiveRetrieval 任務，這是請求下載的封存的大小，以位元組為單位。對於 InventoryRetrieval 任務，值為 null。

類型：數字

已完成

如果任務完成後，則為 true 否則為 false。

類型：布林值

CompletionDate

任務完成的日期。

任務請求完成的國際標準時間 (UTC) 的時間。當任務正在進行時，該值為空。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

CreationDate

任務開始的國際標準時間 (UTC) 日期。

類型：ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

InventoryRetrievalParameters

用於各種庫存擷取的輸入參數。

類型：InventoryRetrievalJobInput物件

InventorySizeInBytes

對於 `InventoryRetrieval` 任務，這是請求下載的庫存的大小，以位元組為單位。對於 `ArchiveRetrieval` 或 `Select` 任務，值為 `null`。

類型：數字

JobDescription

當您啟動的任務所提供的任務說明。

類型：字串

JobId

在 S3 Glacier 中識別任務的 ID。

類型：字串

JobOutputPath

包含任務輸出位置。

類型：字串

OutputLocation

一個物件，其中包含有關儲存選取任務結果和錯誤的位置的資訊。

類型：OutputLocation物件

RetrievalByteRange

用於封存擷取任務的擷取位元組範圍，格式為「*StartByteValue*-*EndByteValue*。」 如果封存擷取中沒有指定範圍, 則擷取整個封存，StartByteValue 等於 0，而 EndByteValue 等於封存的大小減去 1。對於庫存擷取作業，此欄位為 `null`。

類型：字串

SelectParameters

一個物件，其中包含有關用於選擇的參數的資訊。

類型：SelectParameters物件

SHA256TreeHash

所請求的封存範圍的 SHA256 樹狀雜湊值。如果對封存的 啟動任務 (POST 任務) 請求指定了樹狀雜湊值的範圍，則此欄位會傳回一個值。如需關於封存範圍擷取的樹狀雜湊值的詳細資訊，請參閱下載資料時接收檢查總和。

對於擷取整個封存的特定案例，此值與 `ArchiveSHA256TreeHash` 值相同。

此欄位在以下情況下為 `null`：

- 封存擷取任務所指定的範圍不符合樹狀雜湊。
- 指定與整個封存和任務狀態相等的範圍的封存任務是 `InProgress`。
- 庫存任務
- 選取任務。

類型：字串

SNSTopic

如果在任務啟動 (啟動任務 (POST 任務)) 中設定了通知，則 Amazon Resource Name (ARN) 代表傳送任務完成或失敗的通知的 Amazon SNS 主題。

類型：字串

StatusCode

表示任務狀態的代碼。

有效值：`InProgress` | `Succeeded` | `Failed`

類型：字串

StatusMessage

任務狀態訊息。

類型：字串

層

用於選擇或封存擷取的資料存方案。

有效值：Expedited | Standard | Bulk

類型：字串

VaultARN

該任務是子資源的文件庫 ARN。

類型：字串

詳細資訊

- 啟動任務 (POST 任務)

授予

包含關於授予的資訊。

目錄

承授者

承授者

類型：承授者物件

必要：否

許可

給予承授者的許可。

類型：字串

有效值：FULL_CONTROL | WRITE | WRITE_ACP | READ | READ_ACP

必要：否

詳細資訊

- 啟動任務 (POST 任務)

承授者

包含關於承授者的資訊。

目錄

DisplayName

承授者的螢幕名稱。

類型：字串

必要：否

EmailAddress

承授者的電子郵件地址。

類型：字串

必要：否

ID

承授者的正式使用者 ID。

類型：字串

必要：否

類型

承授者的類型。

類型：字串

有效值：`AmazonCustomerByEmail` | `CanonicalUser` | `Group`

必要：否

URI

承授者群組的 URI。

類型：字串

必要：否

詳細資訊

- 啟動任務 (POST 任務)

InputSerialization

描述封存如何序列化。

目錄

CSV

描述 CSV 編碼物件的序列化的物件。

類型：CSVInput物件

必要：否

詳細資訊

- 啟動任務 (POST 任務)

InventoryRetrievalJobInput

提供用於指定各種庫存擷取任務的選項。

目錄

EndDate

文件庫庫存擷取範圍的結束日期，以 UTC 表示，其中包含在此日期之前建立的封存。

有效值：ISO 8601 日期格式 (YYYY-MM-DDThh:mm:ssTZD) 的字串表示法，例如，2013-03-20T17:03:43Z。

類型：字串 ISO 8601 日期格式 (YYYY-MM-DDThh:mm:ssTZD) 的字串表示法 (以秒為單位)，例如，2013-03-20T17:03:43Z。

必要：否

Format (格式)

文件庫庫存清單的輸出格式，該清單是在啟動任務以擷取文件庫庫存時由 啟動任務 (POST 任務) 請求所設定的。

有效值：CSV | JSON

必要：否

類型：字串

限制

每個文件庫庫存擷取請求可以傳回的庫存項目的最大上限數。

有效值：大於或等於 1 的整數值。

類型：字串

必要：否

Marker

一個不透明的字串，表示其繼續分頁的文件庫庫存擷取結果。您可以在新的 Initiate Job 請求中使用此標記來取得額外庫存項目。如果沒有其他庫存項目，這個值為空。

類型：字串

必要：否

StartDate

文件庫庫存擷取的日期範圍的開始，以 UTC 表示，其中包含在此日期或之後建立的封存。

有效值：ISO 8601 日期格式 (YYYY-MM-DDThh:mm:ssTZD) 的字串表示法，例如，2013-03-20T17:03:43Z。

類型：字串 ISO 8601 日期格式 (YYYY-MM-DDThh:mm:ssTZD) 的字串表示法 (以秒為單位)，例如，2013-03-20T17:03:43Z。

必要：否

詳細資訊

- 啟動任務 (POST 任務)

jobParameters

提供選項來定義工作。

目錄

封存

您想要封存的 ID。如果 Type 欄位設定為 `select` 或 `archive-retrieval`，則此欄位是必要的。如果為庫存擷取工作請求指定此欄位，則會發生錯誤。

有效值：必須是從之前向 Amazon S3 Glacier (S3 Glacier) 請求取得的有效封存 ID。

類型：字串

必要 ：是，當 Type 設定為 `select` 或 `archive-retrieval`。

描述

工作的可選說明。

有效值 ：描述必須小於或等於 1,024 位元組。允許的字元是沒有控制代碼的 7 位元 ASCII，尤其 ASCII 值是 32-126 十進制或 0x20-0x7E 十六進制。

類型：字串

必要：否

Format (格式)

(選用) 輸出格式，用於啟動任務以擷取文件庫清查。如果您啟動庫存工作並未指定 `Format` 欄位，則 JSON 是預設的格式。

有效值：`CSV` | `JSON`

類型：字串

必要：否

InventoryRetrievalParameters

用於各種庫存擷取的輸入參數。

類型：InventoryRetrievalJobInput物件

必要：否

OutputLocation

一個物件，其中包含有關儲存選取工作結果的位置的資訊。

類型：OutputLocation物件

必要：是，對於 select 工作而言。

RetrievalByteRange

要為 archive-retrieval 擷取的位元組範圍，格式為「*StartByteValue-EndByteValue*」。如果不指定此欄位，則擷取整個封存。如果指定此欄位，則位元組範圍必須符合 MB (1024* 1024)。MB - 符合表示 StartByteValue 必須被 1 MB 整除，並且 EndByteValue 加上 1 必須可被 1 MB 整除或者是指定為封存位元組大小值減去 1 的封存的結尾。如果 RetrievalByteRange 不符合 MB，則此操作將傳回 400 回應。

如果為 inventory-retrieval 或 select 工作請求指定此欄位，則會發生錯誤。

類型：字串

必要：否

SelectParameters

一個物件，其中包含有關用於選擇的參數的資訊。

類型：SelectParameters物件

必要：否

SNSTopic

Amazon SNS 主題的 Amazon Resource Name (ARN)，任務完成時，S3 Glacier 會傳送通知，並且輸出可供您下載。定的主題將通知發佈到其訂閱伺服器。

SNS 主題必須存在。如果沒有，S3 Glacier 不會為您建立輸出。此外，SNS 主題必須具有政策，可讓已建立工作的帳戶將訊息發佈到該主題。如需有關 SNS 主題名稱的詳細資訊，請參閱 Amazon Simple Notification Service API 參考 中的 CreateTopic。

類型：字串

必要：否

層

用於選擇或封存擷取工作的方案。Standard 是使用的預設值。

有效值：`Expedited | Standard | Bulk`

類型：字串

必要：否

類型

工作類型。您可以啟動 任務，對存檔執行 select 查詢、擷取存檔或取得文件庫的清查。

有效值：`select | archive-retrieval | inventory-retrieval`

類型：字串

必要：是

詳細資訊

- 啟動任務 (POST 任務)

OutputLocation

包含關於儲存任務結果和錯誤的位置的資訊。

目錄

S3

一個物件描述了 Amazon S3 位置以接收還原請求的結果。

Type (類型)：S3Location

必要：是

詳細資訊

- 啟動任務 (POST 任務)

OutputSerialization

描述輸出如何序列化。

目錄

CSV

物件，描述逗號分隔值 (CSV) 編碼的查詢結果的序列化。

類型：CSVOutput物件

必要：否

詳細資訊

- 啟動任務 (POST 任務)

S3Location

包含關於在 Amazon S3 中儲存任務結果之位置的資訊。

目錄

AccessControlList

控制對儲存結果的存取之授權清單。

類型：授予 物件陣列

必要：否

BucketName

任務結果儲存所在之 Amazon S3 儲存貯體的名稱。該儲存貯體必須與包含輸入封存物件的文件庫位於同一 AWS 區域中。

類型：字串

必要：是

CannedACL

要套用至任務結果的固定存取控制清單 (ACL)

類型：字串

有效值：private | public-read | public-read-write | aws-exec-read | authenticated-read | bucket-owner-read | bucket-owner-full-control

必要：否

加密

包含有關用於儲存任務的加密資訊的物件結果將顯示在 Amazon S3 中。

類型：加密物件

必要：否

字首

此字首要追加在該請求結果的前面。字首的最大長度為 512 位元組。

類型：字串

必要：是

StorageClass

用於儲存任務結果的任務類別。

類型：字串

有效值：STANDARD | REDUCED_REDUNDANCY | STANDARD_IA

必要：否

標記

套用到任務結果的標籤集。

類型：字串到字串對應

必要：否

UserMetadata

要隨任務結果儲存於 Amazon S3 之中繼資料的對應。

類型：字串到字串對應

必要：否

詳細資訊

- 啟動任務 (POST 任務)

SelectParameters

包含有關用於選擇的參數的資訊。

目錄

運算式

用於選擇物件的運算式。運算式不得超過 128,000 個字元的配額。

類型：字串

必要：是

ExpressionType

提供的運算式的類型，例如 SQL。

有效值：SQL

類型：字串

必要：是

InputSerialization

描述在該選擇中的物件的序列化格式。

類型：InputSerialization物件

必要：否

OutputSerialization

描述如何序列化選擇任務的結果。

必要：否

類型：OutputSerialization物件

詳細資訊

- 啟動任務 (POST 任務)

資料擷取操作

以下是 S3 Glacier 中可用之與資料擷取相關的操作。

主題

- 取得資料擷取政策 (GET 政策)
- 列出佈建容量 (GET 佈建的容量)
- 購買佈建容量 (POST 佈建的容量)
- 設定資料擷取政策 (PUT 政策)

取得資料擷取政策 (GET 政策)

描述

此操作傳回在 GET 請求中指定的 AWS 帳戶 和 AWS 區域的目前資料擷取政策。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。

請求

要傳回目前的資料擷取政策、傳送 HTTP GET 請求到資料擷取政策 URI 如下語法範例。

語法

```
GET /AccountId/policies/data-retrieval HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用

與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
Content-Type: application/json
Content-Length: length
{
  "Policy":
    {
      "Rules":[
         {
            "BytesPerHour": Number,
            "Strategy": String
         }
       ]
    }
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

BytesPerHour

最多可在一小時內擷取的位元組數。

只有當策略欄位的值為 BytesPerHour 時，此欄位才會出現。

類型：數字

規則

政策規則。雖然這是清單類型，但目前只會有一個規則，其中包含策略欄位和可選的 BytesPerHour 欄位。

類型：陣列

策略

資料擷取政策的類型。

類型：字串

有效值：BytesPerHour|FreeTier|None。BytesPerHour 相當於在主控台中選擇最大擷取率。FreeTier 相當於在主控台中選擇僅限免費方案。None 相當於在主控台中選擇無擷取政策。如需在主控台中選擇資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例示範如何取得資料擷取政策。

範例請求

在這個範例中，將 GET 請求傳送到政策位置的 URI。

```
GET /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

成功的回應以 JSON 格式顯示回應內文中的資料擷取政策。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:00:00 GMT
Content-Type: application/json
Content-Length: 85

{
  "Policy":
    {
      "Rules":[
         {
           "BytesPerHour":10737418240,
           "Strategy":"BytesPerHour"
          }
       ]
    }
}
```

相關章節

- 設定資料擷取政策 (PUT 政策)
- 啟動任務 (POST 任務)

列出佈建容量 (GET 佈建的容量)

此操作會列出指定 AWS 帳戶 的佈建容量單位。如需佈建之容量的詳細資訊，請參閱「封存擷取選項」。

佈建容量單位會持續一個月，從購買的日期和時間開始，即為開始日期。單位會在過期日期當天過期，即開始日期到最接近的秒數的正好一個月。

如果開始日期是在某個月的第 31 天，過期日期則是下個月的最後一天。例如，如果開始日期是 8 月 31 日，過期日期則是 9 月 30 日。如果開始日期是 1 月 31 日，過期日期則是 2 月 28 日。您可以查看回應範例 中的此功能。

請求語法

若要列出帳戶的佈建擷取容量，請將 HTTP GET 請求傳送到佈建容量 URI，如以下語法範例所示。

```
GET /AccountId/provisioned-capacity HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

如果操作成功，則服務傳回 HTTP 200 OK 回應。

回應語法

```
HTTP/1.1 200 OK
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

```
Content-Type: application/json
Content-Length: Length
{
   "ProvisionedCapacityList":
      {
         "CapacityId" : "string",
         "StartDate" : "string"
         "ExpirationDate" : "string"
      }
}
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

回應內文包含以下 JSON 欄位。

CapacityId

此 ID 是用來識別佈建容量單位。

類型：字串

StartDate

購買佈建容量單位的日期，以國際標準時間 (UTC) 為準。

類型：字串 ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

ExpirationDate

佈建容量單位到期的日期，以國際標準時間 (UTC) 為準。

類型：字串 ISO 8601 日期格式的字串表示法，例如，2013-03-20T17:03:43.221Z。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

以下範例列出帳戶的佈建容量單位。

範例請求

在這個範例中，傳送 GET 請求以擷取指定帳戶的佈建容量單位的清單。

```
GET /123456789012/priority-capacity HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會傳回 HTTP 200 OK，其中包含該帳戶的佈建容量單位清單，如以下範例所示。

第一個列出的佈建容量單位是開始日期為 2017 年 1 月 31 日且過期日期為 2017 年 2 月 28 日的單位範例。如先前所述，如果開始日期是在某個月的第 31 天，過期日期則是下個月的最後一天。

```
HTTP/1.1 200 OK
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
Content-Type: application/json
Content-Length: length

{
   "ProvisionedCapacityList",
      {
         "CapacityId": "zSaq7NzHFQDANTfQkDen4V7z",
         "StartDate": "2017-01-31T14:26:33.031Z",
         "ExpirationDate": "2017-02-28T14:26:33.000Z",
      },
      {
         "CapacityId": "yXaq7NzHFQNADTfQkDen4V7z",
         "StartDate": "2016-12-13T20:11:51.095Z"",
         "ExpirationDate": "2017-01-13T20:11:51.000Z" ",
      },
      ...
}
```

相關章節

- 購買佈建容量 (POST 佈建的容量)

購買佈建容量 (POST 佈建的容量)

這個操作為 AWS 帳戶 購買一個佈建容量單位。

佈建容量單位會持續一個月，從購買的日期和時間開始，即為開始日期。單位會在過期日期當天過期，即開始日期到最接近的秒數的正好一個月。

如果開始日期是在某個月的第 31 天，過期日期則是下個月的最後一天。例如，如果開始日期是 8 月 31 日，過期日期則是 9 月 30 日。如果開始日期是 1 月 31 日，過期日期則是 2 月 28 日。

佈建容量有助於確保快速擷取在需要時有可用的擷取容量。容量的每個單位都確保每五分鐘至少可以執行三個快速擷取，並提供最多 150 MB/s 的擷取傳輸量。如需佈建之容量的詳細資訊，請參閱「封存擷取選項」。

> **Note**
>
> 每個 AWS 帳戶 都有兩個佈建容量單位的限制。

請求

若要購買 AWS 帳戶 的佈建容量單位，請向佈建容量 URI 傳送 HTTP POST 請求。

語法

```
POST /AccountId/provisioned-capacity HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用

與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

此操作沒有請求內文。

回應

如果操作請求成功，則服務會傳回 HTTP 201 Created 回應。

語法

```
HTTP/1.1 201 Created
x-amzn-RequestId: x-amzn-RequestId
Date: Date
x-amz-capacity-id: CapacityId
```

回應標頭

成功的回應除了所有操作通用的回應標頭之外，還包含下列回應標頭。如需常見回應標頭的詳細資訊，請參閱常見回應標頭。

名稱	描述
x-amz-capacity-id	此 ID 是用來識別佈建容量單位。 類型：字串

回應內文

此操作不會傳回任何回應內文。

錯誤

除了所有 Amazon S3 Glacier 操作常見的可能錯誤以外，此操作還包括下列一或多個錯誤。如需 Amazon S3 Glacier 錯誤和錯誤代碼清單的相關資訊，請參閱錯誤回應。

代碼	描述	HTTP 狀態碼	類型
LimitExceededException	如果給定的請求超過帳戶的佈建容量單位的限制，則傳回。	400 Bad Request	用戶端

範例

以下範例購買帳戶的佈建容量。

範例請求

以下範例傳送 HTTP POST 請求以購買佈建容量單位。

```
POST /123456789012/provisioned-capacity HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2
Content-Length: length
x-amz-glacier-version: 2012-06-01
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示傳回 HTTP 201 Created 回應。

```
HTTP/1.1 201 Created
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
x-amz-capacity-id: zSaq7NzHFQDANTfQkDen4V7z
```

相關章節

- 列出佈建容量 (GET 佈建的容量)

設定資料擷取政策 (PUT 政策)

描述

此操作會在 PUT 請求中指定的 AWS 區域設定然後制定資料擷取政策。您能夠為 AWS 帳戶 在每個 AWS 區域設定一個政策。政策是在成功的 PUT 操作幾分鐘內制定的。

設定政策操作不會影響擷取任務，其在制定政策之前便已在進行中。如需有關資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。

請求

語法

若要設定資料擷取政策，請如以下語法範例所示傳送 HTTP PUT 請求到資料擷取政策 URI。

```
PUT /AccountId/policies/data-retrieval HTTP/1.1
Host: glacier.Region.amazonaws.com
Date: Date
Authorization: SignatureValue
Content-Length: Length
x-amz-glacier-version: 2012-06-01

{
  "Policy":
    {
      "Rules":[
         {
             "Strategy": String,
             "BytesPerHour": Number
         }
       ]
    }
}
```

> **Note**
>
> 此 AccountId 值是 AWS 帳戶 ID。此值必須符合與用於簽署請求之憑證關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性使用 '-' (連字號)，在這種情況下，Amazon S3 會使用

與用於簽署請求之憑證關聯的 AWS 帳戶 ID。如果您要指定帳戶 ID，請勿在 ID 中包含任何連字號 ('-')。

請求參數

此操作不使用請求參數。

請求標頭

此操作僅使用所有操作常見的請求標頭。如需常見請求標頭的資訊，請參閱 常見請求標題。

請求主體

請求內文包含以下 JSON 欄位。

BytesPerHour

最多可在一小時內擷取的位元組數。

此欄位只有在策略欄位的值為 BytesPerHour 時才需要。若策略欄位未設定為 BytesPerHour 而您設定了此欄位，您的 PUT 操作會被拒絕。

類型：數字

必要：是，如果策略欄位設定為 BytesPerHour。否則為否。

有效值：最低整數值 1。最大整數值為 2^63 - 包含 1。

規則

政策規則。雖然這是清單類型，但目前必須僅有一個規則，其中包含策略欄位和可選的 BytesPerHour 欄位。

類型：陣列

必要：是

策略

要設定的資料擷取政策的類型。

類型：字串

必要：是

有效值：BytesPerHour|FreeTier|None。BytesPerHour 相當於在主控台中選擇最大擷取率。FreeTier 相當於在主控台中選擇僅限免費方案。None 相當於在主控台中選擇無擷取政策。如需在主控台中選擇資料擷取政策的詳細資訊，請參閱 S3 Glacier 資料擷取政策。

回應

語法

```
HTTP/1.1 204 No Content
x-amzn-RequestId: x-amzn-RequestId
Date: Date
```

回應標頭

此操作僅使用大部分回應常見的回應標頭。如需常見回應標頭的資訊，請參閱 常見回應標頭。

回應內文

此操作不會傳回任何回應內文。

錯誤

如需 Amazon S3 Glacier 例外和錯誤訊息的詳細資訊，請參閱錯誤回應。

範例

範例請求

以下範例傳送策略欄位設定為 BytesPerHour 的 HTTP PUT 請求。

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Policy":
```

```
    {
      "Rules":[
         {
             "Strategy":"BytesPerHour",
             "BytesPerHour":10737418240
          }
       ]
    }
}
```

以下範例傳送策略欄位設定為 FreeTier 的 HTTP PUT 請求。

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Policy":
    {
      "Rules":[
         {
             "Strategy":"FreeTier"
          }
       ]
    }
}
```

以下範例傳送策略欄位設定為 None 的 HTTP PUT 請求。

```
PUT /-/policies/data-retrieval HTTP/1.1
Host: glacier.us-west-2.amazonaws.com
x-amz-Date: 20170210T120000Z
x-amz-glacier-version: 2012-06-01
Authorization: AWS4-HMAC-SHA256 Credential=AKIAIOSFODNN7EXAMPLE/20141123/
us-west-2/glacier/aws4_request,SignedHeaders=host;x-amz-date;x-amz-glacier-
version,Signature=9257c16da6b25a715ce900a5b45b03da0447acf430195dcb540091b12966f2a2

{
  "Policy":
```

```
    {
      "Rules":[
         {
             "Strategy":"None"
          }
       ]
    }
}
```

回應範例

如果請求成功，則 Amazon S3 Glacier (S3 Glacier) 會如以下範例所示，設定此政策並傳回 HTTP 204 No Content。

```
HTTP/1.1 204 No Content
x-amzn-RequestId: AAABZpJrTyioDC_HsOmHae8EZp_uBSJr6cnGOLKp_XJCl-Q
Date: Wed, 10 Feb 2017 12:02:00 GMT
```

相關章節

- 取得資料擷取政策 (GET 政策)
- 啟動任務 (POST 任務)

文件歷史記錄

- 目前產品版本：2012-06-01

下表說明 2018 年 7 月 5 日後《Amazon S3 Glacier 開發人員指南》每個版本的重要變更。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

變更	描述	日期
改善透過 S3 批次操作發出標準還原請求的開始時間	透過 S3 批次操作發出之還原請求的標準擷取，現在可在幾分鐘內開始。如需詳細資訊，請參閱封存擷取選項。	2023 年 8 月 9 日
對於 S3 Glacier Flexible Retrieval 和 S3 Glacier Deep Archive，Amazon S3 支援更高的還原請求速率	對於 S3 Glacier Flexible Retrieval 和 S3 Glacier Deep Archive 儲存體類別，Amazon S3 支援速率高達每個 AWS 帳戶 每秒 1,000 筆交易的還原請求。	2022 年 11 月 15 日
Amazon Glacier 名稱變更	Amazon Glacier 現在是 Amazon S3 Glacier，以更好地反映 Glacier 與 Amazon S3 的整合。	2018 年 11 月 20 日
現在可以透過 RSS 獲得更新	您現在可以訂閱更新 RSS 訊息，以接收《Amazon S3 Glacier 開發人員指南》的更新通知。	2018 年 7 月 5 日

舊版更新

下表說明 2018 年 7 月 5 日前《Amazon S3 Glacier 開發人員指南》每個版本的重要變更。

變更	描述	版本日期
快速和大量資料擷取	S3 Glacier 現在除了標準擷取以外，也支援快速和大量資料擷取。如需詳細資訊，請參閱 封存擷取選項。	2016 年 11 月 21 日
文件庫鎖定	S3 Glacier 現在支援文件庫鎖定，可讓您使用文件庫鎖定政策，在個別 S3 Glacier 文件庫上輕鬆部署並強制執行合規性控制。如需詳細資訊，請參閱 S3 Glacier 文件庫鎖定 及 保存庫鎖定政策。	2015 年 7 月 8 日
文件庫標籤	S3 Glacier 現在允許您標記 S3 Glacier 文件庫，以簡化資源和成本管理。標籤是您可以定義和關聯到文件庫，使用標籤可為 AWS 成本報告等操作新增篩選功能。如需詳細資訊，請參閱 標記 Amazon S3 Glacier 資源 及 標記 S3 Glacier 文件庫。	2015 年 6 月 22 日
文件庫存取政策	S3 Glacier 現在支援使用文件庫存取政策，來管理對個別 S3 Glacier 文件庫的存取。現在，您可以直接在文件庫上定義存取政策，從而更容易向組織內部的使用者和業務部門以及外部業務夥伴授予文件庫存取權限。如需詳細資訊，請參閱 保存庫存取政策。	2015 年 4 月 27 日
資料擷取政策和稽核記錄	S3 Glacier 現在支援資料擷取政策和稽核記錄。資料擷取政策可讓您輕鬆地設定資料擷取限制並簡化資料擷取成本管理。您只需在 AWS Management Console 按幾下或使用 S3 Glacier API，即可定義自己的資料擷取限制。如需詳細資訊，請參閱 S3 Glacier 資料擷取政策。 此外，S3 Glacier 現在支援使用 AWS CloudTrail 進行稽核記錄，該記錄將為帳戶記錄 S3 Glacier API 呼叫，並將日誌檔傳送到指定的 Amazon S3 儲存貯體。如需詳細資訊，請參閱 使用 AWS CloudTrail 記錄 Amazon S3 Glacier API 呼叫。	2014 年 12 月 11 日
更新 Java 範例	更新本指南中使用 AWS SDK for Java 的 Java 程式碼範例。	2014 年 6 月 27 日

變更	描述	版本日期
限制文件庫庫存擷取	您現在可以透過篩選在封存建立日期或設定限制來限制擷取到的文件庫庫存項目的數量。如需有關限制庫存擷取的詳細資訊，請參閱 庫存擷取範圍 主題中的 啟動任務 (POST 任務)。	2013 年 31 月 12 日
移除已過期的 URL	從程式碼範例中已移除指向舊的安全憑證頁面 URL 。	2013 年 7 月 26 日
支援範圍擷取	S3 Glacier 現在支援擷取封存的特定範圍。您可以啟動請求 S3 Glacier 的任務，以便為後續的下載準備整個封存或部分封存。當封存非常龐大，您會發現啟動多個連續作業來準備封存的成本效益較高 如需詳細資訊，請參閱 在 S3 Glacier 中下載封存。	2012 年 11 月 13 日
新的指南	這是《Amazon S3 Glacier 開發人員指南》的第一個版本。	2012 年 8 月 20 日

AWS 詞彙表

如需最新的 AWS 術語，請參閱《AWS 詞彙表 參考》中的 AWS 詞彙表。