aws

開發人員指南

Amazon Glacier

API 版本 2012-06-01

Amazon Glacier: 開發人員指南

Table of Contents

此頁面僅適用於使用 Vaults 和 2012 年原始 REST API 的 Amazon Glacier 服務的現有客戶。

如果您要尋找封存儲存解決方案，建議您在 Amazon Glacier Instant Retrieval、S3 Glacier Flexible Retrieval 和 S3 Glacier Deep Archive 中使用 Amazon Glacier 儲存類別。 Amazon S3 若要進一步了解這些儲存選項，請參閱 Amazon Glacier 儲存類別。

Amazon Glacier （原始獨立保存庫型服務）不再接受新客戶。Amazon Glacier 是一項獨立服務，具有自己的 APIs，可將資料存放在保存庫中，並與 Amazon S3 和 Amazon S3 Glacier 儲存類別不同。您現有的資料將在 Amazon Glacier 中無限期保持安全且可存取。不需要遷移。對於低成本、長期的封存儲存， AWS 建議使用 Amazon S3 Glacier 儲存類別，透過 S3 儲存貯體型 APIs、完整 AWS 區域可用性、降低成本 AWS 和服務整合，提供卓越的客戶體驗。如果您想要增強功能，請考慮使用我們的解決方案指南，將資料從 Amazon S3 Glacier 保存庫傳輸至 Amazon S3 Glacier 儲存類別，以遷移至 Amazon S3 Glacier 儲存類別。 AWS Amazon Glacier Amazon S3

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。

什麼是 Amazon Glacier？

如果您目前正在使用 Amazon Glacier 服務且想要進一步了解，您可以在本指南中找到所需的資訊。Amazon Glacier 是一種安全且耐用的服務，用於使用保存庫進行低成本資料封存和長期備份。如需 Amazon Glacier 服務定價的詳細資訊，請參閱 Amazon Glacier 定價。

主題

您目前是否使用 Amazon Glacier？

> **Note**
>
> 本節是關於 Amazon Glacier 服務。如果您目前使用 Amazon S3 Glacier 儲存類別 (S3 Glacier Instant Retrieval、S3 Glacier Flexible Retrieval 和 S3 Glacier Deep Archive)，請參閱《Amazon S3 使用者指南》中的用於封存物件的儲存類別。

如果您目前使用 Amazon Glacier 服務且想要進一步了解，建議您先閱讀以下章節：

- 什麼是 Amazon Glacier – 本節的其餘部分說明基礎資料模型、其支援的操作，以及您可以用來與服務互動 AWS SDKs。
- 入門：Amazon 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**
>
> Amazon Glacier 會提供主控台。不過，任何封存操作，例如上傳、下載或刪除，都需要您使用 AWS Command Line Interface (AWS CLI) 或寫入程式碼。沒有主控台支援封存操作。例如，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，方法是直接使用 REST API 或使用 AWS SDKs。

若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配 使用 Amazon Glacier AWS Command Line Interface。

除了入門章節之外，您可能會想要進一步了解 Amazon Glacier 操作。下列各節提供使用 REST API 和適用於 Java 和 Microsoft .NET AWS SDKs 來使用 Amazon Glacier 的詳細資訊：

- 搭配 Amazon Glacier 使用 AWS SDKs

  本節提供本指南中各種程式碼範例所使用的 AWS SDKs 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 Glacier 中使用保存庫

  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 Amazon Glacier 主控台之外，您還可以將 AWS SDKs用於各種保存庫操作。本節說明 API，並使用 和 適用於 Java 的 AWS SDK 提供工作範例 適用於 .NET 的 AWS SDK。

- 在 Amazon 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 適用於 Java 的 AWS SDK 與 適用於 .NET 的 AWS SDK的範例。

- Amazon Glacier 的 API 參考

  Amazon Glacier 是一項 RESTful 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 程式庫會包裝此 API，簡化您的程式設計任務。

Amazon Glacier 資料模型

Amazon Glacier 資料模型核心元件包含保存庫和封存。Amazon 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，Amazon Glacier 資料模型包含任務和通知組態資源。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 Amazon Glacier 中，保存庫是用於儲存封存的容器。保存庫類似於 Amazon S3 儲存貯體。建立保存庫時，您可以指定名稱，然後選擇您要建立保存庫 AWS 區域 的 。

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 是擁有保存庫的 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 可以在任何支援的 中建立保存庫 AWS 區域。如需支援的清單 AWS 區域，請參閱 存取 Amazon 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同名保存庫。

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。

Amazon Glacier 支援各種保存庫操作。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。當您請求保存庫清單時，您可以從特定 請求， AWS 區域而產生的清單只會包含在該特定區域中建立的保存庫。

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，是 Amazon Glacier 中儲存的基本單位。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。Amazon Glacier 會指派封存 ID，該 ID 在存放封存 AWS 區域 的 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

Amazon Glacier 任務可以擷取封存，或取得保存庫的庫存。

擷取封存和保存庫庫存（封存清單）是 Amazon Glacier 中的非同步操作，您會先啟動任務，然後在 Amazon Glacier 完成任務後下載任務輸出。

> **Note**
>
> Amazon Glacier 提供冷儲存資料存檔解決方案。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。當您傳送任務請求（啟動任務）時，Amazon Glacier 會傳回任務 ID 以追蹤任務。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
```

對於每個任務，Amazon Glacier 會維護資訊，例如任務類型、描述、建立日期、完成日期和任務狀態。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。Amazon Glacier 傳回的任務清單包含所有進行中和最近完成的任務。

通知組態

由於任務需要一些時間才能執行，Amazon Glacier 支援通知機制，以便在任務完成時通知您。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。

Amazon Glacier 會將通知組態儲存為 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
```

Amazon Glacier 支援設定、取得和刪除通知組態的操作。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。

Amazon Glacier 中支援的操作

若要使用保存庫和封存 （請參閱Amazon Glacier 資料模型），Amazon Glacier 支援一組操作。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。下列各節摘要說明 Amazon Glacier 操作。

保存庫作業

Amazon Glacier 提供建立和刪除保存庫的操作。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。保存庫描述提供資訊，例如建立日期、保存庫中的封存數量、保存庫中所有封存所使用的總位元組大小，以及 Amazon Glacier 產生保存庫庫存的日期。Amazon Glacier 也提供在保存庫上設定、擷取和刪除通知組態的操作。如需詳細資訊，請參閱在 Amazon Glacier 中使用保存庫。

封存操作

Amazon Glacier 為您提供上傳和刪除封存的操作。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。每次上傳封存時，Amazon Glacier 都會產生新的封存 ID。如需詳細資訊，請參閱在 Amazon Glacier 中使用封存。

任務

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

以下是 Amazon Glacier 任務的類型：

- `archive-retrieval`：擷取封存。

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

  如需詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

存取 Amazon Glacier

Amazon Glacier 是一種 RESTful Web 服務，使用 HTTP 和 HTTPS 做為傳輸通訊協定，並使用 JavaScript 物件標記法 (JSON) 做為訊息序列化格式。您的應用程式程式碼可以直接向 Amazon Glacier Web 服務 API 提出請求。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 Glacier 的 API 參考」。

或者，您可以使用包裝 Amazon Glacier REST API 呼叫的 AWS SDKs 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，請參閱搭配 Amazon Glacier 使用 AWS SDKs。

Amazon Glacier 也提供主控台。不過，所有封存和任務操作都需要您直接使用 REST API 或 AWS SDK 包裝函式程式庫來撰寫程式碼和提出請求。若要存取 Amazon Glacier 主控台，請前往 Amazon Glacier 主控台。

區域與終端節點

您可以在特定 中建立保存庫 AWS 區域。您一律會將 Amazon Glacier 請求傳送至 特定的端點 AWS 區域。如需 Amazon Glacier 支援的 AWS 區域 清單，請參閱《 AWS 一般參考》中的 Amazon Glacier 端點和配額。

Amazon Glacier 入門

您可以使用保存庫和封存來開始使用 Amazon Glacier (Amazon 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)。存檔是 Amazon Glacier 中儲存體的基本單位。此入門練習提供說明，讓您探索保存庫和封存的基本 Amazon Glacier 操作。如需有關這些資源的詳細資訊，請參閱 Amazon 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。不過，入門練習會使用 Amazon Glacier 管理主控台來建立和刪除保存庫。對於上傳和下載封存，此入門區段使用 適用於 Java 的 AWS SDK 和 的高階 API 適用於 .NET 的 AWS SDK。高階 API 可在使用 Amazon Glacier 時提供簡化的程式設計體驗。如需搭配 AWS SDKs 使用高階 API 的詳細資訊，請參閱 搭配 Amazon Glacier 使用 AWS SDKs。

> **Important**
>
> Amazon Glacier 會提供主控台。不過，上傳、下載或刪除等任何封存操作都需要您使用 AWS Command Line Interface (CLI) 或寫入程式碼。沒有主控台支援封存操作。例如，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，方法是直接使用 REST API 或使用 AWS SDKs。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配使用 Amazon Glacier 與 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步驟，可讓您進一步了解 Amazon Glacier 的開發人員體驗。

主題

步驟 1：開始使用 Amazon Glacier 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 SDKs，才能開始此練習。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，則必須註冊 ， AWS 帳戶 並在帳戶中建立管理員使用者。

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 AWS

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

如果您已有 AWS 帳戶，請跳至 下載適當的 AWS SDK。如果您沒有 AWS 帳戶，請使用下列程序建立一個。

如果您沒有 AWS 帳戶，請完成下列步驟來建立一個。

註冊 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或簡訊，並在電話鍵盤輸入驗證碼。

當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

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	根據	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。	透過在 AWS Command Line Interface 使用者指南中設定 AWS CLI 以使用 來設定 AWS IAM Identity Center程式設計存取。
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 IAM 使用者以進行緊急存取 的指示。	請依照《IAM 使用者指南》中的管理 IAM 使用者的存取金鑰設定以程式設計方式存取。

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，然後為您的開發平台下載適當的 AWS SDK。

入門練習提供 Java 和 C# 範例。

下載 適用於 Java 的 AWS SDK

若要測試此開發人員指南中的 Java 範例，您需要 適用於 Java 的 AWS SDK。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 https AWS Toolkit for Eclipse ：//http://aws.amazon.com/eclipse/ 下載並安裝 。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 適用於 Java 的 AWS SDK。

下載 適用於 .NET 的 AWS SDK

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

- 如果您使用的是 Visual Studio，則可以同時安裝 適用於 .NET 的 AWS SDK 和 AWS Toolkit for Visual Studio。工具組提供可用於開發的 AWS Explorer for Visual Studio 和專案範本。若要下載 適用於 .NET 的 AWS SDK，請前往 https：//http://aws.amazon.com/sdkfornet。根據預設，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 適用於 .NET 的 AWS SDK。

步驟 2：在 Amazon Glacier 中建立保存庫

文件庫是儲存封存的一種容器。您的第一個步驟是在其中一個支援的 中建立保存庫 AWS 區域。如需 Amazon Glacier AWS 區域 支援的 清單，請參閱《 AWS 一般參考》中的 Amazon Glacier 端點和配額。

您可以編寫程式或使用 Amazon 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 管理主控台 ，並在 https：//https://console.aws.amazon.com/glacier/home 開啟 Amazon Glacier 主控台。
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 從區域選擇器中選取 。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：

- 保存庫名稱在建立保存庫 AWS 區域 的 AWS 帳戶 和 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 Glacier 主控台設定保存庫通知。

7. 如果 AWS 區域 和保存庫名稱正確，請選擇建立保存庫。

您的新保存庫現在已列在 Amazon Glacier 主控台的保存庫頁面上。

步驟 3：將封存上傳至 Amazon 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 Amazon 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，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，方法是直接使用 REST API 或使用 AWS SDKs。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配使用 Amazon Glacier 與 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)。存檔是 Amazon Glacier 中儲存體的基本單位。您可以在單一請求中上傳封存。對於大型封存，Amazon Glacier 提供分段上傳 API 操作，可讓您分段上傳封存。

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 將封存上傳至 Amazon Glacier 中的保存庫 適用於 Java 的 AWS SDK
- 使用 將封存上傳至 Amazon Glacier 中的保存庫 適用於 .NET 的 AWS SDK

使用 將封存上傳至 Amazon Glacier 中的保存庫 適用於 Java 的 AWS SDK

下列 Java 程式碼範例使用 的高階 API， 適用於 Java 的 AWS SDK 將範例封存上傳至保存庫。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 適用於 Java 的 AWS SDK高階 API 的 `ArchiveTransferManager` 類別的 `upload` API 作業。
- 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 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 Glacier 會保留保存庫中所有封存的庫存。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。

SDK for Java 2.x

> **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 將封存上傳至 Amazon Glacier 中的保存庫 適用於 .NET 的 AWS SDK

下列 C# 程式碼範例使用 的高階 API， 適用於 .NET 的 AWS SDK 將範例封存上傳至保存庫。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon 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 作業以上傳封存。對於小型封存，此操作會將封存直接上傳至 Amazon Glacier。對於較大的封存，如果在將資料串流至 Amazon Glacier 時遇到任何錯誤，此操作會使用 Amazon Glacier 中的分段上傳 API 操作將上傳分割為多個部分，以獲得更好的錯誤復原。

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，如所示的保存庫名稱和要上傳的封存檔案的名稱。

> **Note**
>
> Amazon Glacier 會保留保存庫中所有封存的庫存。當您上傳下列範例中的封存時，直到保存庫庫存更新前，封存都不會顯示在管理主控台的保存庫中。此更新通常一天執行一次。

Example— 使用 的高階 API 上傳封存 適用於 .NET 的 AWS SDK

```
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 Amazon 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 Amazon Glacier 中的保存庫 中上傳的範例封存。

> **Important**
>
> Amazon Glacier 會提供主控台。不過，上傳、下載或刪除等任何封存操作都需要您使用 AWS Command Line Interface (CLI) 或寫入程式碼。沒有主控台支援封存操作。例如，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，方法是直接使用 REST API 或使用 AWS SDKs。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配使用 Amazon Glacier 與 AWS Command Line Interface。

一般而言，從 Amazon Glacier 擷取您的資料需要兩個步驟：

1. 啟動擷取任務。
2. 工作完成後，下載資料的位元組。

若要從 Amazon Glacier 擷取封存，您必須先啟動任務。在工作完成後下載資料。如需封存擷取的詳細資訊，請參閱 擷取 Amazon Glacier Archives。

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 Amazon Glacier 常見問答集。如需資料擷取費用的資訊，請參閱 Amazon 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 從 Amazon Glacier 中的保存庫下載封存 適用於 Java 的 AWS SDK
- 使用 從 Amazon Glacier 中的保存庫下載封存 適用於 .NET 的 AWS SDK

使用 從 Amazon Glacier 中的保存庫下載封存 適用於 Java 的 AWS SDK

下列 Java 程式碼範例使用 的高階 API 適用於 Java 的 AWS SDK 來下載您在上一個步驟中上傳的封存。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 Amazon Glacier 中建立保存庫 中建立保存庫的位置。
- 此範例使用 適用於 Java 的 AWS SDK高階 API 的 `ArchiveTransferManager` 類別的 `download` API 作業。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。如果您依照 中的指示建立 AWS Identity and Access Management (IAM) 管理員使用者步驟 1：開始使用 Amazon Glacier 之前，則您的使用者具有建立和使用 Amazon SNS 主題和 Amazon SQS 佇列的必要 IAM 許可。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon Glacier 的 Java 範例。您需要按照在 步驟 3：將封存上傳至 Amazon 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 下載封存 適用於 Java 的 AWS SDK

```
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 從 Amazon Glacier 中的保存庫下載封存 適用於 .NET 的 AWS SDK

下列 C# 程式碼範例使用 的高階 API 適用於 .NET 的 AWS SDK 來下載您先前在 中上傳的封存使用 將封存上傳至 Amazon Glacier 中的保存庫 適用於 .NET 的 AWS SDK。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon 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 Amazon Glacier 中建立保存庫 中建立保存庫的位置。
- 本範例使用 `ArchiveTransferManager` 類別的 `Download` API 作業以下載封存。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。如果您依照 中的指示建立 AWS Identity and Access

Management (IAM) 管理員使用者步驟 1：開始使用 Amazon Glacier 之前，則您的使用者具有建立和使用 Amazon SNS 主題和 Amazon SQS 佇列的必要 IAM 許可。

- 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 Amazon 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 的高階 API 下載封存 適用於 .NET 的 AWS SDK

```
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 Amazon 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 Amazon Glacier 中的保存庫 中上傳的範例封存。

> **Important**
>
> 您無法使用 Amazon Glacier 主控台刪除封存。任何封存操作，例如上傳、下載或刪除，都需要您使用 AWS Command Line Interface (CLI) 或寫入程式碼。若要上傳資料，例如相片、影片和其他文件，您必須直接使用 REST API AWS CLI 或使用 AWS SDKs，使用 或 撰寫程式碼來提出請求。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配使用 Amazon Glacier 與 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 從 Amazon Glacier 中的保存庫刪除封存 適用於 Java 的 AWS SDK
- 使用 從 Amazon Glacier 中的保存庫刪除封存 適用於 .NET 的 AWS SDK

- 使用 在 Amazon Glacier 中刪除封存 AWS CLI

相關章節

- 步驟 3：將封存上傳至 Amazon Glacier 中的保存庫
- 在 Amazon Glacier 中刪除封存

使用 從 Amazon Glacier 中的保存庫刪除封存 適用於 Java 的 AWS SDK

下列程式碼範例使用 適用於 Java 的 AWS SDK 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保存庫名稱和封存 ID。
- `deleteArchive` API 操作會將請求傳送至 Amazon 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 Glacier 的 Java 範例。您需要按照在 步驟 3：將封存上傳至 Amazon 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 刪除封存 適用於 Java 的 AWS SDK

```
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 從 Amazon Glacier 中的保存庫刪除封存 適用於 .NET 的 AWS SDK

下列 C# 程式碼範例使用 的高階 API 適用於 .NET 的 AWS SDK 來刪除您在上一個步驟中上傳的封存。在程式碼範例中，請注意下列事項：

- 此範例會為指定的 Amazon 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 作業作為 適用於 .NET 的 AWS SDK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 Amazon 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 的高階 API 刪除封存 適用於 .NET 的 AWS SDK

```
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 在 Amazon Glacier 中刪除封存 AWS CLI

您可以使用 AWS Command Line Interface () 刪除 Amazon Glacier 中的封存AWS CLI。

主題

- （先決條件）設定 AWS CLI
- 範例：使用 刪除封存 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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

- 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

```
aws configure list
```

範例：使用 刪除封存 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組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在任務完成後傳送訊息至主題。

   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 Glacier 中設定保存庫通知。每當發生特定事件時，Amazon Glacier 就會傳送訊息至指定的 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：刪除 Amazon Glacier 中的保存庫

文件庫是儲存封存的一種容器。若要刪除 Amazon Glacier 保存庫，您必須先從 Amazon Glacier 計算的最後一個庫存中刪除保存庫中的所有現有封存。

您可以編寫程式或使用 Amazon 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 在 Amazon Glacier 中刪除保存庫。

> **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。Amazon 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 管理主控台 ，並在 https：//https://console.aws.amazon.com/glacier/home 開啟 Amazon Glacier 主控台。
2. 從選取區域功能表中，選擇您要刪除之保存庫 AWS 區域 的 。

   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，請參閱在 Amazon Glacier 中刪除封存。

   > **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。您可以使用 REST API、 適用於 Java 的 AWS SDK、 適用於 .NET 的 AWS SDK 或 撰寫程式碼來提出刪除封存請求 AWS CLI。如需刪除封存的資訊，請參閱 步驟 5：從 Amazon 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？

現在您已嘗試了入門練習，您可以探索下列各節，以進一步了解 Amazon Glacier。

- 在 Amazon Glacier 中使用保存庫
- 在 Amazon Glacier 中使用封存

在 Amazon Glacier 中使用保存庫

文件庫是儲存封存的一種容器。建立保存庫時，您可以指定保存庫名稱，以及要在 AWS 區域 其中建立保存庫的 。如需 Amazon Glacier AWS 區域 支援的 清單，請參閱《 AWS 一般參考》中的 Amazon Glacier 端點和配額。

您可以在保存庫中存放無限數量的封存。

> **Important**
>
> Amazon Glacier 會提供主控台。不過，任何封存操作，例如上傳、下載或刪除，都需要您使用 AWS Command Line Interface (AWS CLI) 或寫入程式碼。沒有主控台支援封存操作。例如，若要上傳資料，例如相片、影片和其他文件，您必須直接使用 REST API AWS CLI 或使用 AWS SDKs，使用 或寫入程式碼來提出請求。
>
> 若要安裝 AWS CLI，請參閱 AWS Command Line Interface。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。如需使用 AWS CLI 將封存上傳至 Amazon Glacier 的範例，請參閱搭配使用 Amazon Glacier 與 AWS Command Line Interface。

主題

Amazon Glacier 中的保存庫操作

Amazon Glacier 支援各種保存庫操作。保存庫作業專用於特定 AWS 區域。換句話說，當您建立保存庫時，會在特定 AWS 區域中建立保存庫。當您列出保存庫時，Amazon Glacier 會從 AWS 區域 您在請求中指定的 傳回保存庫清單。

建立和刪除文件庫

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

只有在保存庫中截至 Amazon Glacier 計算的最後一個庫存沒有封存，以及自上次庫存以來沒有寫入保存庫時，您才能刪除保存庫。

> **Note**
>
> Amazon Glacier 會每 24 小時定期為每個保存庫準備好庫存。由於庫存可能不會反映最新資訊，Amazon Glacier 會檢查自上次保存庫庫存以來是否有任何寫入操作，以確保保存庫確實是空的。

如需詳細資訊，請參閱在 Amazon Glacier 中建立保存庫及在 Amazon Glacier 中刪除保存庫。

擷取保存庫中繼資料

您可以擷取保存庫資訊，例如保存庫建立日期、保存庫中的封存數和保存庫中所有封存的總大小。Amazon Glacier 為您提供 API 呼叫，以擷取 AWS 區域 您帳戶中特定保存庫或所有保存庫的資訊。如需詳細資訊，請參閱在 Amazon Glacier 中擷取保存庫中繼資料。

下載保存庫庫存

保存庫庫存是指保存庫中的封存清單。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。Amazon Glacier 每天更新一次保存庫庫存，從第一個封存上傳到保存庫的那一天開始。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，Amazon Glacier 會準備您的庫存以供下載。工作完成後，您即可下載庫存資料。

若由於工作的非同步本質，您可以使用 Amazon Simple Notification Service (Amazon SNS) 通知，以在工作完成時通知您。您可以為每個個別工作請求指定一個 Amazon SNS 主題，或者將保存庫設定為在特定保存庫事件發生時傳送通知。

Amazon Glacier 會每 24 小時定期為每個保存庫準備好庫存。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。

當您啟動保存庫庫存的任務時，Amazon Glacier 會傳回其產生的最後一個庫存，這是point-in-time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。不過，假設您在用戶端維護資料庫，其中包含與您上傳至 Amazon Glacier 的封存相關聯的中繼資料。然後，您可能會發現保存庫庫存的好處，可以在資料庫中使用實際的保存庫庫存來調節資訊。

如需有關擷取庫存的詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

設定保存庫通知

從 Amazon Glacier 擷取任何內容，例如從保存庫或保存庫庫存封存，都是兩個步驟。請先啟動工作。工作完成後，即可下載輸出。若要了解任務何時完成，您可以使用 Amazon Glacier 通知。Amazon Glacier 會將通知訊息傳送至您提供的 Amazon Simple Notification Service (Amazon SNS) 主題。

您可以設定保存庫通知，以及識別保存庫事件和事件發生時要通知的 Amazon SNS 主題。每當保存庫事件發生時，Amazon Glacier 都會傳送通知至指定的 Amazon SNS 主題。如需詳細資訊，請參閱在 Amazon Glacier 中設定保存庫通知。

在 Amazon Glacier 中建立保存庫

建立保存庫可將保存庫新增到您帳戶中的一組保存庫中。每個 AWS 區域最多可 AWS 帳戶 建立 1，000 個保存庫。如需 Amazon Glacier (Amazon Glacier) 支援 AWS 的區域清單，請參閱《 AWS 一般參考》中的區域和端點。

建立保存庫時，必須提供保存庫名稱。以下是保存庫命名要求：

- 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)。

保存庫名稱在帳戶和建立保存庫的 AWS 區域中必須是唯一的。也就是說，帳戶可以在不同的 AWS 區域中建立同名的保存庫，但不能在相同的 AWS 區域中建立。

主題

使用 在 Amazon Glacier 中建立保存庫 適用於 Java 的 AWS SDK

低階 API 提供所有保存庫操作的方法，包括建立和刪除保存庫、取得保存庫描述，以及取得在特定 中建立的保存庫清單 AWS 區域。以下是使用 適用於 Java 的 AWS SDK建立保存庫的步驟。

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

   您需要指定要在 AWS 區域 其中建立保存庫的 。所有您使用此用戶端執行的作業均會套用到該 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 Glacier (Amazon Glacier) 要求您提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。
3. 以參數形式提供請求物件，以便執行 `createVault` 方法。

   Amazon Glacier 傳回的回應可在 `CreateVaultResult` 物件中使用。

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。`Location` 列印的 是保存庫的相對 URI，其中包含您的帳戶 ID AWS 區域、 和保存庫名稱。

```
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 建立保存庫 (PUT 保存庫)。

範例：使用 建立保存庫 適用於 Java 的 AWS SDK

下列 Java 程式碼範例會在 us-west-2區域中建立保存庫 （ 如需詳細資訊 AWS 區域，請參閱 存取 Amazon Glacier)。此外，程式碼範例會擷取保存庫資訊、列出相同 中的所有保存庫 AWS 區域，然後刪除建立的保存庫。

如需如何執行下列範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon 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 在 Amazon Glacier 中建立保存庫 適用於 .NET 的 AWS SDK

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

主題

- 使用 的高階 API 建立保存庫 適用於 .NET 的 AWS SDK
- 使用 的低階 API 建立保存庫 適用於 .NET 的 AWS SDK

使用 的高階 API 建立保存庫 適用於 .NET 的 AWS SDK

高階 API 的 `ArchiveTransferManager` 類別提供可用於在 AWS 區域中建立保存庫的 `CreateVault` 方法。

範例：使用 的高階 API 進行保存庫操作 適用於 .NET 的 AWS SDK

以下 C# 程式碼範例建立和刪除美國西部 (奧勒岡) 區域中的保存庫。如需您可以在 AWS 區域 其中建立保存庫的清單，請參閱 存取 Amazon 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 的低階 API 建立保存庫 適用於 .NET 的 AWS SDK

低階 API 為所有保存庫操作提供方法，包括建立和刪除保存庫、取得保存庫描述，以及取得在特定 中建立的保存庫清單 AWS 區域。以下是使用 適用於 .NET 的 AWS SDK建立保存庫的步驟。

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

您需要指定要在 AWS 區域 其中建立保存庫的 。所有您使用此用戶端執行的作業均會套用到該 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 Glacier (Amazon Glacier) 要求您提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。

3. 以參數形式提供請求物件，以便執行 `CreateVault` 方法。

   Amazon Glacier 傳回的回應可在 `CreateVaultResponse` 物件中使用。

範例：使用 的低階 API 的保存庫操作 適用於 .NET 的 AWS SDK

下列 C# 範例描述前述步驟。此範例在美國西部 (奧勒岡) 區域建立保存庫。此外，程式碼範例會擷取保存庫資訊，列出相同 中的所有保存庫 AWS 區域，然後刪除建立的保存庫。`Location` 列印的 是保存庫的相對 URI，其中包含您的帳戶 ID AWS 區域、 和保存庫名稱。

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 建立保存庫 (PUT 保存庫)。

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);
    }
  }
}
```

使用 REST API 在 Amazon Glacier 中建立保存庫

若要使用 REST API 建立保存庫，請參閱 建立保存庫 (PUT 保存庫)。

使用 Amazon Glacier 主控台建立保存庫

若要使用 Amazon Glacier (Amazon Glacier) 主控台建立保存庫，請參閱入門教學步驟 2：在 Amazon Glacier 中建立保存庫中的 。

使用 在 Amazon Glacier 中建立保存庫 AWS Command Line Interface

請依照下列步驟，使用 () 在 AWS Command Line Interface Amazon Glacier (Amazon Glacier) 中建立保存庫AWS CLI。

主題

- （先決條件）設定 AWS CLI
- 範例：使用 建立保存庫 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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

```
aws configure list
```

範例：使用 建立保存庫 AWS CLI

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 Glacier 中擷取保存庫中繼資料

您可以擷取保存庫資訊，例如保存庫建立日期、保存庫中的封存數和保存庫中所有封存的總大小。Amazon Glacier (Amazon Glacier) 為您提供 API 呼叫，以擷取您帳戶中特定 AWS 區域中特定保存庫或所有保存庫的資訊。

如果您擷取保存庫清單，Amazon Glacier 會傳回依保存庫名稱的 ASCII 值排序的清單。該清單最多可包含 1,000 個保存庫。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 回應中傳回的保存庫數量。如果回應中的保存庫比所傳回的更多，則會對結果進行分頁。您需要傳送其他請求，以擷取下一組保存庫。

主題

- 使用 在 Amazon Glacier 中擷取保存庫中繼資料 適用於 Java 的 AWS SDK
- 使用 在 Amazon Glacier 中擷取保存庫中繼資料 適用於 .NET 的 AWS SDK
- 使用 REST API 擷取保存庫中繼資料

- 使用 在 Amazon Glacier 中擷取保存庫中繼資料 AWS Command Line Interface

使用 在 Amazon Glacier 中擷取保存庫中繼資料 適用於 Java 的 AWS SDK

主題

擷取保存庫的保存庫中繼資料

您可以擷取特定保存庫或特定 AWS 區域中所有保存庫的中繼資料。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，為特定保存庫擷取保存庫中繼資料。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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 Glacier (Amazon Glacier) 要求您提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。
3. 以參數形式提供請求物件，以便執行 `describeVault` 方法。

   Amazon Glacier 傳回的保存庫中繼資料資訊可在 `DescribeVaultResult` 物件中使用。

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 描述保存庫 (GET 保存庫)。

擷取區域中所有保存庫的保存庫中繼資料

您也可以使用 `listVaults`方法來擷取特定 AWS 區域中所有保存庫的中繼資料。

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保存庫數量為 5 個。程式碼片段接著會進行一系列`listVaults`呼叫，從 AWS 區域擷取整個保存庫清單。

```
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);
```

在上述程式碼區段中，如果您未在請求中指定 `Limit`值，Amazon Glacier 會傳回最多 10 個保存庫，如 Amazon Glacier API 所設定。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，清單中為每個保存庫傳回的資訊與透過呼叫特定保存庫的 `describeVault` 方法所獲得的資訊相同。

> **Note**
>
> `listVaults` 方法呼叫底層 REST API (請參閱 「列出保存庫」(GET 保存庫))。

範例：使用適用於 Java 的 Amazon 開發套件擷取保存庫中繼資料

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

使用 在 Amazon Glacier 中擷取保存庫中繼資料 適用於 .NET 的 AWS SDK

主題

擷取保存庫的保存庫中繼資料

您可以擷取特定保存庫或特定 AWS 區域中所有保存庫的中繼資料。以下是使用 適用於 .NET 的 AWS SDK的低階 API 為特定保存庫擷取保存庫中繼資料的步驟。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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 Glacier (Amazon Glacier) 要求您提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。

3. 以參數形式提供請求物件，以便執行 DescribeVault 方法。

   Amazon Glacier 傳回的保存庫中繼資料資訊可在 DescribeVaultResult 物件中使用。

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) 區域中現有保存庫的中繼資料資訊。

```
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 描述保存庫 (GET 保存庫)。

擷取區域中所有保存庫的保存庫中繼資料

您也可以使用 ListVaults方法來擷取特定 AWS 區域中所有保存庫的中繼資料。

以下 C# 程式碼片段擷取美國西部 (奧勒岡) 區域中的保存庫清單。請求會限制回應中傳回的保存庫數量為 5 個。程式碼片段接著會進行一系列ListVaults呼叫，從 AWS 區域擷取整個保存庫清單。

```
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);
```

在上述程式碼區段中，如果您未在請求中指定 `Limit`值，Amazon Glacier 會傳回最多 10 個保存庫，如 Amazon Glacier API 所設定。

請注意，清單中為每個保存庫傳回的資訊與透過呼叫特定保存庫的 `DescribeVault` 方法所獲得的資訊相同。

> **Note**
>
> `ListVaults` 方法呼叫底層 REST API (請參閱 「列出保存庫」(GET 保存庫))。

範例：使用 的低階 API 擷取保存庫中繼資料 適用於 .NET 的 AWS SDK

如需運作中程式碼範例，請參閱「範例：使用 的低階 API 的保存庫操作 適用於 .NET 的 AWS SDK」。C# 程式碼範例建立保存庫和擷取保存庫中繼資料。

使用 REST API 擷取保存庫中繼資料

若要使用 REST API 列出保存庫的更多資訊，請參閱 「列出保存庫」(GET 保存庫)。若要描述一個保存庫的詳細資訊，請參閱 描述保存庫 (GET 保存庫)。

使用 在 Amazon Glacier 中擷取保存庫中繼資料 AWS Command Line Interface

此範例說明如何使用 AWS Command Line Interface () 擷取 Amazon Glacier (Amazon Glacier) 中的保存庫資訊和中繼資料AWS CLI。

主題

（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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

     ```
     aws configure list
     ```

範例：使用 擷取保存庫中繼資料 AWS CLI

- 使用 `describe-vault` 命令，在帳戶 *111122223333* 下描述一個名為 *awsexamplevault* 的保存庫。

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

在 Amazon Glacier 中下載保存庫庫存

將第一個封存上傳至保存庫後，Amazon Glacier (Amazon Glacier) 會自動建立保存庫庫存，然後大約每天更新一次。Amazon Glacier 建立第一個庫存後，通常需要半天到一天的時間才能擷取該庫存。您可以使用下列兩個步驟從 Amazon Glacier 擷取保存庫庫存：

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 Amazon 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。以非同步方式執行工作請求。當您起始檢索任務時，Amazon Glacier 會建立任務，並在回應中傳回任務 ID。當 Amazon Glacier 完成工作時，您可以取得工作輸出、封存位元組或保存庫庫存資料。

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 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 可以在任務完成後發佈通知。您可以使用以下方法指定此 Amazon SNS 主題：
  - 為每個工作指定 Amazon SNS 主題。

    啟動工作時，需選擇指定 Amazon SNS 主題。
  - 在文件庫上設定通知組態。

您可以為保存庫中的特定事件設定通知設定 (請參閱 在 Amazon Glacier 中設定保存庫通知)。只要發生特定事件，Amazon Glacier 就會傳送訊息至指定的 SNS 主題。

如果您在保存庫上設定了通知組態，且在啟動任務時也指定了 Amazon SNS 主題，Amazon Glacier 會將任務完成訊息傳送至這兩個主題。

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。

- 明確請求任務資訊—Amazon 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

Amazon Glacier 每天至少更新一次保存庫庫存，從您第一次將封存上傳至保存庫的那一天開始。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。當您啟動保存庫庫存的任務時，Amazon Glacier 會傳回其產生的最後一個庫存，這是point-in-time快照，而不是即時資料。請注意，Amazon Glacier 為保存庫建立第一個庫存後，通常需要半天到最多一天的時間，才能擷取該庫存。

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，假設您在用戶端維護資料庫，關聯您上傳至 Amazon Glacier 之封存的相關中繼資料。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 在 Amazon Glacier 中下載保存庫庫存 適用於 Java 的 AWS SDK

以下是使用 適用於 Java 的 AWS SDK低階 API 來擷取保存庫庫存的步驟。高階的 API 不支援擷取保存庫庫存。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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` 物件中的工作資訊來執行 `InitiateJobRequest`。

   > **Note**
   >
   > 請注意，如果保存庫庫存尚未完成，便會傳回錯誤。Amazon Glacier (Amazon Glacier) 每 24 小時定期為每個保存庫準備清查。

   Amazon Glacier 傳回任務 ID 以回應。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 工作完成。

您必須等到任務輸出準備好供您下載。如果您已在保存庫上設定通知組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在完成任務後傳送訊息至主題。

您也可以呼叫 `describeJob`方法來輪詢 Amazon 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。下一節中提供的程式碼範例使用 Amazon SNS for Amazon Glacier 來發佈訊息。

4. 透過執行 `getJobOutput` 方法下載工作輸出 (保存庫庫存資料)。

   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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。

   Amazon Glacier 傳回的輸出可在 `GetJobOutputResult` 物件中使用。

```
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 開發套件擷取保存庫庫存

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。

  Amazon Glacier 會在完成任務後傳送通知至此主題。
- 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，指定已建立的 Amazon SNS 主題，以便 Amazon Glacier 在完成任務後可以將通知發佈至主題。

- 檢查 Amazon SQS 佇列中是否有包含工作 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 在 Amazon Glacier 中下載保存庫庫存 適用於 .NET 的 AWS SDK

以下是使用 適用於 .NET 的 AWS SDK低階 API 來擷取保存庫庫存的步驟。高階的 API 不支援擷取保存庫庫存。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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` 物件中提供工作資訊。Amazon Glacier (Amazon Glacier) 會傳回任務 ID 以回應。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 工作完成。

   您必須等到任務輸出準備好供您下載。如果您已在識別 Amazon Simple Notification Service (Amazon SNS) 主題的保存庫上設定通知組態，或在啟動任務時指定 Amazon SNS 主題，Amazon Glacier 會在任務完成後傳送訊息至該主題。下一節中提供的程式碼範例使用 Amazon SNS for Amazon Glacier 來發佈訊息。

   您也可以呼叫 `DescribeJob`方法來輪詢 Amazon 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` 方法下載工作輸出 (保存庫庫存資料)。

   透過建立 `GetJobOutputRequest` 類別的執行個體來提供帳戶 ID、保存庫名稱和工作 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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。

   Amazon Glacier 傳回的輸出可在 `GetJobOutputResponse` 物件中使用。

```
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 的低階 API 擷取保存庫庫存 適用於 .NET 的 AWS SDK

以下 C# 程式碼範例將擷取指定保存庫的保存庫庫存。

範例會執行下列任務：

- 設定 Amazon SNS 主題。

  Amazon Glacier 會在完成任務後傳送通知至此主題。
- 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，範例會指定 Amazon SNS 主題，以便 Amazon Glacier 在完成任務後傳送訊息。
- 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\",&TCX5-2025-waiver;" +
        "    \"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。如需詳細資訊，請參閱「取得任務輸出」 (GET 輸出)。

使用 在 Amazon Glacier 中下載保存庫庫存 AWS Command Line Interface

請依照下列步驟，使用 AWS Command Line Interface () 在 Amazon Glacier (Amazon Glacier) 中下載保存庫庫存AWS CLI。

主題

（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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

     ```
     aws configure list
     ```

範例：使用 下載保存庫庫存 AWS CLI

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 工作完成。

您必須等到任務輸出準備好供您下載。在 Amazon Glacier 完成任務後至少 24 小時內，任務 ID 不會過期。如果您已在保存庫上設定通知組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在完成任務後傳送訊息至主題。

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 Glacier 中設定保存庫通知。Amazon Glacier 會在發生特定事件時傳送訊息至指定的 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 Glacier 中設定保存庫通知

從 Amazon Glacier 擷取任何內容，例如從保存庫或保存庫庫存封存，都是兩個步驟。

1. 啟動擷取任務。
2. 工作完成後，下載工作輸出。

可以在保存庫中設定通知設定，以便在工作完成後，將訊息傳送到 Amazon Simple Notification Service (Amazon SNS) 主題。

主題

- 在 Amazon Glacier 中設定保存庫通知：一般概念

- 使用 在 Amazon Glacier 中設定保存庫通知 適用於 Java 的 AWS SDK
- 使用 在 Amazon Glacier 中設定保存庫通知 適用於 .NET 的 AWS SDK
- 使用 REST API 在 Amazon Glacier 中設定保存庫通知
- 使用 Amazon Glacier 主控台設定保存庫通知
- 使用 設定保存庫通知 AWS Command Line Interface

在 Amazon Glacier 中設定保存庫通知：一般概念

Amazon Glacier 擷取任務請求會以非同步方式執行。您必須等到 Amazon Glacier 完成任務，才能取得其輸出。您可以定期輪詢 Amazon Glacier 以判斷任務狀態，但這不是最佳方法。Amazon Glacier 也支援通知。工作完成後，該工作就可將訊息張貼到 Amazon Simple Notification Service (Amazon SNS) 主題。使用此功能需要您在保存庫上設定通知設定。在組態中，您可以識別一或多個事件，以及您希望 Amazon Glacier 在事件發生時傳送訊息的 Amazon SNS 主題。 Amazon Glacier

Amazon Glacier 會定義與任務完成 (`ArchiveRetrievalCompleted`、`InventoryRetrievalCompleted`) 特別相關的事件，您可以將這些事件新增至保存庫的通知組態。特定任務完成後，Amazon Glacier 會將通知訊息發佈至 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，您只能為保存庫設定一個 Amazon SNS 主題。

> **Note**
>
> 將通知組態新增至保存庫會導致 Amazon Glacier 每次發生通知組態中指定的事件時傳送通知。您還可以選擇在每個工作啟動請求中指定 Amazon SNS 主題。如果您在保存庫上新增通知組態，並在啟動任務請求中指定 Amazon SNS 主題，Amazon Glacier 會傳送這兩個通知。

Amazon Glacier 傳送的任務完成訊息包含諸如任務類型 (`InventoryRetrieval`、`ArchiveRetrieval`)、任務完成狀態、SNS 主題名稱、任務狀態碼和保

存庫 ARN 等資訊。以下是 Amazon Glacier 在完成InventoryRetrieval任務後傳送至 SNS 主題的範例通知。

```
{
 "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 主題必須允許保存庫發布通知。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 主題和保存庫是由不同的 擁有 AWS 帳戶，則您必須設定 Amazon SNS 主題以接受保存庫中的發佈。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 在 Amazon Glacier 中設定保存庫通知 適用於 Java 的 AWS SDK

以下是使用 適用於 Java 的 AWS SDK低階 API 設定保存庫通知的步驟。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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。

您需要提供保存庫名稱、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，請參閱設定保存庫通知組態 (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`事件發生或`InventoryRetrievalCompleted`事件發生時，組態會要求 Amazon Glacier (Amazon Glacier) 傳送通知至指定的 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 保存庫作業。

範例：使用 在保存庫上設定通知組態 適用於 Java 的 AWS SDK

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 適用於 Java 的 AWS SDK 搭配 Amazon Glacier 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 在 Amazon Glacier 中設定保存庫通知 適用於 .NET 的 AWS SDK

以下是使用 適用於 .NET 的 AWS SDK低階 API 設定保存庫通知的步驟。

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

   您需要指定保存庫所在的 AWS 區域。您使用此用戶端執行的所有操作都會套用到該 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。

   您需要提供保存庫名稱、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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 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，請參閱設定保存庫通知組態 (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. 在設定有關保存庫的通知組態之後，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 在保存庫上設定通知組態 適用於 .NET 的 AWS SDK

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) 區域設定有關保存庫 (「examplevault」) 的通知設定、擷取設定，然後將其刪除。當ArchiveRetrievalCompleted事件發生或InventoryRetrievalCompleted事件發生時，組態會要求 Amazon Glacier (Amazon Glacier) 傳送通知至指定的 Amazon SNS 主題。

> **Note**
>
> 如需基礎 REST API 的資訊，請參閱 保存庫作業。

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，並提供現有的保存庫名稱和 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 Amazon Glacier 中設定保存庫通知

若要使用 REST API 設定文件庫通知，請參閱 設定保存庫通知組態 (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 (刪除保存庫通知 (DELETE 通知的組態))。

使用 Amazon Glacier 主控台設定保存庫通知

本節說明如何使用 Amazon Glacier 主控台設定保存庫通知。設定通知時，您可以指定工作完成事件，其會將通知傳送至 Amazon Simple Notification Service (Amazon SNS) 主題。除了設定保存庫通知外，您也可以指定在啟動工作時發布通知的主題。如果將保存件庫設為通知特定事件，並且在工作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. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在左側導覽窗格中，選擇保存庫。
3. 在保存庫清單中，選擇保存庫。
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作業：

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

Amazon SNS 選項	Action
	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 中必須是唯一的 AWS 區域。 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：

   • 若要僅在封存擷取工作完成時傳送通知，請選取封存擷取工作完成。

- 若要僅在保存庫庫存工作完成時傳送通知，請選取保存庫庫存擷取工作完成。

使用 設定保存庫通知 AWS Command Line Interface

本節說明如何使用 AWS Command Line Interface設定保存庫通知。設定通知時，您要指定工作完成事件，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設定保存庫通知。

主題

（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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

     ```
     aws configure list
     ```

範例：使用 設定保存庫通知 AWS CLI

1. 使用 `set-vault-notifications` 命令，設定保存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"]
}
```

   如需針對 Amazon Glacier 使用 Amazon SNS 主題的詳細資訊，請參閱在 Amazon Glacier 中設定保存庫通知：一般概念 Amazon Glacier

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

在 Amazon Glacier 中刪除保存庫

Amazon Glacier (Amazon Glacier) 只有在保存庫中截至其計算的最後一個庫存沒有封存，且自上次清查以來沒有寫入保存庫時，才會刪除保存庫。如需刪除封存的資訊，請參閱 在 Amazon Glacier 中刪除封存。如需下載文件庫清查的詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

> **Note**
>
> Amazon Glacier 會每 24 小時定期為每個保存庫準備好庫存。由於庫存可能不會反映最新資訊，Amazon Glacier 會檢查自上次保存庫庫存以來是否有任何寫入操作，以確保保存庫確實是空的。

> **Note**
>
> 如需自動刪除保存庫封存，請參閱 Amazon S3 Glacier 中的自動刪除保存庫封存。

主題

- 使用 在 Amazon Glacier 中刪除保存庫 適用於 Java 的 AWS SDK
- 使用 在 Amazon Glacier 中刪除保存庫 適用於 .NET 的 AWS SDK
- 使用 REST API 在 Amazon Glacier 中刪除保存庫
- 使用 Amazon Glacier 主控台刪除空保存庫
- 使用 在 Amazon Glacier 中刪除保存庫 AWS Command Line Interface

使用 在 Amazon Glacier 中刪除保存庫 適用於 Java 的 AWS SDK

以下是使用 適用於 Java 的 AWS SDK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)。

   您需要指定從中刪除保存庫 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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。

   您需要提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。

3. 以參數形式提供請求物件，以便執行 `deleteVault` 方法。

   Amazon Glacier (Amazon Glacier) 只會在保存庫為空時刪除保存庫。如需詳細資訊，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 刪除保存庫 適用於 Java 的 AWS SDK

如需運作中程式碼範例，請參閱「範例：使用 建立保存庫 適用於 Java 的 AWS SDK」。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 在 Amazon Glacier 中刪除保存庫 適用於 .NET 的 AWS SDK

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

主題

- 使用 的高階 API 刪除保存庫 適用於 .NET 的 AWS SDK
- 使用 的低階 API 刪除保存庫 適用於 .NET 的 AWS SDK

使用 的高階 API 刪除保存庫 適用於 .NET 的 AWS SDK

高階 API 的 `ArchiveTransferManager` 類別提供可用來刪除保存庫的 `DeleteVault` 方法。

範例：使用 的高階 API 刪除保存庫 適用於 .NET 的 AWS SDK

如需運作中程式碼範例，請參閱「範例：使用 的高階 API 進行保存庫操作 適用於 .NET 的 AWS SDK」。C# 程式碼範例顯示基本保存庫作業，包括建立和刪除保存庫。

使用 的低階 API 刪除保存庫 適用於 .NET 的 AWS SDK

以下是使用 適用於 .NET 的 AWS SDK刪除保存庫的步驟。

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

   您需要指定從中刪除保存庫 AWS 的區域。您使用此用戶端執行的所有操作都適用於該 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。

   您需要提供保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。

3. 以參數形式提供請求物件，以便執行 DeleteVault 方法。

   Amazon Glacier (Amazon Glacier) 只會在保存庫為空時刪除保存庫。如需詳細資訊，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。程式碼片段會擷取預設 AWS 區域中保存庫的中繼資料資訊。

```
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 的低階 API 刪除保存庫 適用於 .NET 的 AWS SDK

如需運作中程式碼範例，請參閱「範例：使用 的低階 API 的保存庫操作 適用於 .NET 的 AWS SDK」。C# 程式碼範例顯示基本保存庫作業，包括建立和刪除保存庫。

使用 REST API 在 Amazon Glacier 中刪除保存庫

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

使用 Amazon Glacier 主控台刪除空保存庫

> **Note**
>
> 若要刪除保存庫，您必須先刪除該保存庫內所有現有的封存。您可以使用 REST API、、或 AWS Command Line Interface () 撰寫程式碼來提出刪除封存請求 適用於 Java 的 AWS SDK， 適用於 .NET 的 AWS SDK藉此執行此操作AWS CLI。如需刪除封存的資訊，請參閱 步驟 5：從 Amazon Glacier 中的保存庫刪除封存。

保存庫清空後，您就可以使用下列步驟將其刪除。

使用 Amazon Glacier 主控台刪除空保存庫

1. 登入 AWS 管理主控台 ，並在 Amazon Glacier 主控台開啟 Amazon Glacier 主控台。
2. 在選取區域下，選擇保存庫所在的 AWS 區域 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫清單中，選取要刪除之保存庫名稱旁的選項，然後選擇頁面頂端的刪除。
5. 在刪除保存庫對話方塊中，透過選擇刪除來確認您要刪除保存庫。

   > **Important**
   >
   > 刪除保存庫無法復原。

6. 若要確認您是否已刪除保存庫，請開啟保存庫清單，然後輸入您已刪除的保存庫名稱。如果找不到保存庫，就表示已成功刪除。

使用 在 Amazon Glacier 中刪除保存庫 AWS Command Line Interface

您可以使用 () AWS Command Line Interface 刪除 Amazon Glacier (Amazon Glacier) 中的空和非空保存庫AWS CLI。

主題

- （先決條件）設定 AWS CLI
- 範例：使用 刪除空保存庫 AWS CLI
- 範例：使用 刪除非空保存庫 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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

- 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

```
aws configure list
```

範例：使用 刪除空保存庫 AWS CLI

- 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 刪除非空保存庫 AWS CLI

Amazon Glacier 只有在保存庫中截至其計算的最後一個庫存沒有封存，且自上次清查以來沒有寫入保存庫時，才會刪除保存庫。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 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組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在任務完成後傳送訊息至主題。

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 Glacier 中設定保存庫通知。Amazon Glacier 會在發生特定事件時傳送訊息至指定的 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**
>
> 如果您的封存 ID 以連字號或其他特殊字元開頭，您必須以引號括住封存 ID，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
```

標記您的 Amazon Glacier 保存庫

您可以將自己的中繼資料以標籤形式指派給 Amazon Glacier 保存庫。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 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 Glacier 主控台標記保存庫
- 使用 標記保存庫 AWS CLI
- 使用 Amazon Glacier API 標記保存庫
- 相關章節

使用 Amazon Glacier 主控台標記保存庫

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

查看文件庫的標籤

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從區域選取器中選取 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫清單中，選擇保存庫。
5. 選擇保存庫屬性索引標籤。捲動至標籤區段以檢視與保存庫相關聯的標籤。

若要新增標籤到文件庫

您最多可以將 50 個標籤與保存庫建立關聯。與保存庫相關聯的標籤，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 Glacier 資源。

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從區域選取器中選取 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫清單中，選擇要新增標籤的保存庫名稱。
5. 選擇保存庫屬性索引標籤。
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. 選擇儲存變更。

編輯標籤

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從區域選取器中選取 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫清單中，選擇保存庫名稱。
5. 選擇保存庫屬性索引標籤，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. 選擇儲存變更。

若要從文件庫中移除標籤

1. 登入 AWS 管理主控台 ，並在 https：//https://console.aws.amazon.com/glacier/home 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從區域選取器中選取 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫清單中，選擇要從中移除標籤的保存庫名稱。
5. 選擇保存庫屬性索引標籤。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 標記保存庫 AWS CLI

請依照下列步驟，使用 AWS Command Line Interface () 新增、列出或移除標籤AWS CLI。

每個標籤皆包含鍵與值。每個保存庫最多可擁有 50 個標籤。

1. 若要將標籤新增至保存庫，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
```

若要取得此保存庫作業的更多資訊，請參閱將標籤新增至保存庫。

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

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

若要取得此保存庫作業的更多資訊，請參閱列出保存庫的標籤。

3. 若要從連接至保存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
```

若要取得此保存庫作業的更多資訊，請參閱將標籤從保存庫中移除。

使用 Amazon Glacier API 標記保存庫

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

新增標籤至保存庫 (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。

從保存庫移除標籤 (POST tags remove)

從指定的文件庫中移除標籤。

相關章節

- 標記 Amazon Glacier 資源

Amazon Glacier 保存庫鎖定

下列主題說明如何在 Amazon Glacier 中鎖定保存庫，以及如何使用保存庫鎖定政策。

主題

- 文件庫鎖定概觀
- 使用 Amazon Glacier API 鎖定保存庫
- 使用 鎖定保存庫 AWS Command Line Interface
- 使用 Amazon Glacier 主控台鎖定保存庫

文件庫鎖定概觀

Amazon Glacier Vault Lock 可協助您使用保存庫鎖定政策，輕鬆部署和強制執行個別 Amazon Glacier 保存庫的合規控制。您可以在保存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**
>
> 鎖定保存庫鎖定政策後，將無法再變更或刪除該政策。

Amazon Glacier 會強制執行保存庫鎖定政策中設定的控制項，以協助達成您的合規目標。例如，您可以使用保存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，在保存庫鎖定政策中部署各種合規控制。如需保存庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

保存庫鎖定政策不同於保存庫存取政策。兩個政策都可管理文件庫的存取控制。不過，保存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。您可以使用保存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**
>
> 我們建議您先建立保存庫、完成保存庫鎖定政策，然後將封存上傳至保存庫，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。您可以將保存庫鎖定和保存庫存取政策一起搭配使用。例如，您可以在保存庫鎖定政策 (拒絕刪除) 中實作以時間為基礎的資料保留規則，以及在保存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. 透過將保存庫鎖定政策連接至保存庫來啟動鎖定，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 過期之前驗證保存庫鎖定政策。若要防止保存庫結束進行中狀態，您必須在這 24 小時內完成保存庫鎖定程序。否則，保存庫鎖定政策將遭到刪除。
2. 使用鎖定 ID 來完成鎖定程序。如果保存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。如需如何使用 Amazon Glacier API 鎖定保存庫的詳細資訊，請參閱 使用 Amazon Glacier API 鎖定保存庫。

使用 Amazon Glacier API 鎖定保存庫

若要使用 Amazon Glacier API 鎖定保存庫，您必須先啟動保存庫鎖定 (POST 鎖定政策)使用指定要部署之控制項的保存庫鎖定政策來呼叫 。`Initiate Vault Lock` 作業會將政策連接至保存庫，將保存庫鎖定轉為進行中狀態，並傳回唯一的鎖定 ID。保存庫鎖定進入進行中狀態後，您有 24 小時可透過使用 完成保存庫鎖定 (POST lockId) 呼叫傳回的鎖定 ID 呼叫 `Initiate Vault Lock` 來完成鎖定。

> **Important**
>
> - 我們建議您先建立保存庫、完成保存庫鎖定政策，然後將封存上傳至保存庫，以便將政策套用至其中。
> - 在保存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 小時內完成保存庫鎖定程序，保存庫會自動結束進行中狀態，並且保存庫鎖定政策會遭到移除。您可以再次呼叫 `Initiate Vault Lock` 安裝新的保存庫鎖定政策，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保存庫鎖定政策。保存庫鎖定政策在進行中狀態會充分發揮作用，宛如保存庫已經鎖定，不過您也可以透過呼叫 「中止保存庫鎖定」(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，來驗證保存庫鎖定政策變更。

在您驗證保存庫鎖定政策後，您可以使用最近的鎖定 ID 呼叫 完成保存庫鎖定 (POST lockId)，來完成保存庫鎖定程序。保存庫會轉換到鎖定狀態，其中保存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
- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 完成保存庫鎖定 (POST lockId)
- 取得保存庫鎖定 (GET 鎖定政策)
- 啟動保存庫鎖定 (POST 鎖定政策)

使用 鎖定保存庫 AWS Command Line Interface

您可以使用 鎖定保存庫 AWS Command Line Interface。這會在指定的保存庫上安裝保存庫鎖定政策，並傳回鎖定 ID。您必須在 24 小時內完成保存庫鎖定程序，否則會將保存庫鎖定政策從保存庫中移除。

（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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

     ```
     aws configure list
     ```

1. 使用 initiate-vault-lock 安裝保存庫鎖定政策，並將保存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。

   若要尋找要鎖定的保存庫 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. 啟動保存庫鎖定之後，您應該會看到傳回的 lockId。

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

若要完成保存庫鎖定，您必須在 24 小時內執行 complete-vault-lock，否則會將保存庫鎖定政策從保存庫中移除。

```
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
- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 完成保存庫鎖定 (POST lockId)
- 取得保存庫鎖定 (GET 鎖定政策)

- 啟動保存庫鎖定 (POST 鎖定政策)

使用 Amazon Glacier 主控台鎖定保存庫

Amazon Glacier Vault Lock 可協助您使用保存庫鎖定政策，輕鬆部署和強制執行個別 Amazon Glacier 保存庫的合規控制。如需 Amazon Glacier 保存庫鎖定的詳細資訊，請參閱 Amazon Glacier 存取控制與保存庫鎖定政策。

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

使用 Amazon Glacier 主控台啟動保存庫上的保存庫鎖定政策

您透過將保存庫鎖定政策連接至保存庫來啟動鎖定，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 過期之前驗證保存庫鎖定政策。

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從區域選取器中選取 。
3. 在左側導覽窗格中，選擇保存庫。
4. 在保存庫頁面上，選擇建立保存庫。
5. 建立新的保存庫。

   > **Important**
   >
   > 我們建議您先建立保存庫、完成保存庫鎖定政策，然後將封存上傳至保存庫，以便將政策套用至其中。

6. 從保存庫清單中選擇新的保存庫。
7. 選擇保存庫政策索引標籤。
8. 在保存庫鎖定政策區段中，選擇啟動保存庫鎖定政策。

9. 在啟動保存庫鎖定政策頁面上，在標準文字方塊中，以文字格式指定保存庫鎖定政策中的記錄保留控制項。

> **Note**
>
> 您可以在保存庫鎖定政策中以文字格式指定記錄保留控制，並透過呼叫 `Initiate Vault Lock` API 操作或透過 Amazon Glacier 主控台中的互動式 UI 啟動保存庫鎖定。如需有關格式化保存庫鎖定政策的資訊，請參閱 Amazon Glacier 保存庫鎖定政策範例。

10. 選擇儲存變更。
11. 在記錄保存庫鎖定 ID 對話方塊中，複製鎖定 ID 並將其儲存在安全的位置。

> **Important**
>
> 啟動保存庫鎖定政策後，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 小時內測試保存庫鎖定政策。如果政策如預期運作，請選擇完成保存庫鎖定政策。
14. 在完成保存庫鎖定對話方塊中，選取該核取方塊以確認保存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. 選擇完成保存庫鎖定。

在 Amazon Glacier 中使用封存

封存是存放在保存庫中的任何物件 (如相片、影片或文件)。它是 Amazon Glacier (Amazon Glacier) 中儲存的基本單位。每個封存都有唯一的 ID 以及選擇性說明。當您上傳封存時，Amazon Glacier 會傳回包含封存 ID 的回應。此封存 ID 在存放封存 AWS 的區域中是唯一的。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**
>
> Amazon Glacier 提供管理主控台。您可以使用主控台來建立及刪除保存庫。不過，與 Amazon Glacier 的所有其他互動都需要您使用 AWS Command Line Interface (CLI) 或寫入程式碼。例如，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，直接使用 REST API 或使用 Amazon SDKs。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。

主題

在 Amazon Glacier 中封存操作

Amazon Glacier 支援下列基本封存操作：上傳、下載和刪除。下載封存是一種非同步操作。

在 Amazon 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。如需詳細資訊，請參閱在 Amazon Glacier 中上傳封存。

> **Important**
>
> Amazon Glacier 提供管理主控台。您可以使用主控台來建立及刪除保存庫。不過，與 Amazon Glacier 的所有其他互動都需要您使用 AWS Command Line Interface (CLI) 或寫入程式碼。例如，若要上傳資料，例如相片、影片和其他文件，您必須使用 AWS CLI 或編寫程式碼來提出請求，直接使用 REST API 或使用 Amazon SDKs。如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。

在 Amazon Glacier 中尋找封存 ID

您可以透過為包含封存的保存庫下載保存庫庫存來取得封存 ID。如需下載保存庫庫存的詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

在 Amazon Glacier 中下載封存

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。收到任務請求後，Amazon Glacier 會準備您的封存以供下載。任務完成後，下載封存資料。由於任務的非同步性質，您可以請求 Amazon Glacier 在任務完成時傳送通知至 Amazon Simple Notification Service (Amazon SNS) 主題。您可以為每個個別工作請求指定一個 SNS 主題，或者設定您的保存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 Amazon Glacier 中下載封存。

在 Amazon Glacier 中刪除封存

Amazon Glacier 提供 API 呼叫，可讓您一次刪除一個封存。如需詳細資訊，請參閱在 Amazon Glacier 中刪除封存。

在 Amazon 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，Amazon Glacier 都會傳回唯一的封存 ID。

維護用戶端封存中繼資料

除了選用的封存描述之外，Amazon Glacier 不支援封存的任何其他中繼資料。當您上傳封存時，Amazon Glacier 會指派 ID，這是不透明的字元序列，您無法從中推斷封存的任何意義。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 Amazon Glacier 中，您無法將物件金鑰指派給您上傳的封存。

如果您維護用戶端封存中繼資料，請注意，Amazon Glacier 會維護保存庫庫存，其中包含封存 IDs和您在封存上傳期間提供的任何說明。您可能偶爾會下載保存庫庫存，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。不過，Amazon Glacier 大約每天會取得保存庫庫存。當您請求保存庫庫存時，Amazon 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 Glacier 中上傳封存

Amazon Glacier (Amazon Glacier) 提供管理主控台，可用來建立和刪除保存庫。不過，您無法使用 管理主控台將封存上傳至 Amazon Glacier。若要上傳資料，例如相片、影片和其他文件，您必須直接使用 REST API AWS CLI 或使用 Amazon SDKs，使用 或編寫程式碼來提出請求。

如需有關搭配 使用 Amazon Glacier 的資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。若要安裝 AWS CLI，請前往 AWS Command Line Interface。下列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 將封存上傳至 Amazon Glacier。

主題

- 將封存上傳至 Amazon 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)

將封存上傳至 Amazon Glacier 的選項

視您上傳的資料大小而定，Amazon Glacier 會提供下列選項：

- 在單一作業中上傳封存：在單一作業中，您可以上傳大小為 1 位元組到最大 4 GB 的封存。不過，我們鼓勵 Amazon Glacier 客戶使用分段上傳來上傳大於 100 MB 的封存。如需詳細資訊，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。如需詳細資訊，請參閱上傳分段中的大型封存 (分段上傳)。

> **Important**
>
> Amazon Glacier 保存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 Edge 服務

AWS Snowball Edge 加速 AWS 使用 Amazon 擁有的裝置進出大量資料，繞過網際網路。如需詳細資訊，請參閱 AWS Snowball Edge 的詳細資訊頁面。

若要將現有資料上傳至 Amazon Glacier (Amazon Glacier)，您可以考慮使用其中一種 AWS Snowball Edge 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 Amazon Glacier 儲存類別以進行封存。當您將 Amazon S3 物件轉換為 Amazon Glacier 儲存類別時，Amazon S3 會在內部使用 Amazon Glacier 以較低成本提供耐久儲存。雖然物件存放在 Amazon 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 Amazon Glacier 直接存取。

如需 Amazon S3 生命週期組態和將物件轉換為 Amazon 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 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。不過，我們鼓勵 Amazon Glacier (Amazon Glacier) 客戶使用分段上傳上傳大於 100 MB 的封存。

主題

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

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

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

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

主題

- （先決條件）設定 AWS CLI
- 範例：使用 上傳封存 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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

     ```
     aws configure list
     ```

範例：使用 上傳封存 AWS CLI

若要上傳封存，您必須建立保存庫。如需有關建立保存庫的詳細資訊，請參閱在 Amazon Glacier 中建立保存庫。

1. 使用 upload-archive 命令將封存新增至現有保存庫。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"
}
```

   完成時，命令將輸出 Amazon Glacier 中的封存 ID、檢查總和和和位置。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 在單一操作中上傳封存 適用於 Java 的 AWS SDK

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

使用 的高階 API 上傳封存 適用於 Java 的 AWS SDK

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 的高階 API 上傳封存 適用於 Java 的 AWS SDK

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) 中的保存庫 (examplevault)。如需支援 AWS 的區域和端點清單，請參閱 存取 Amazon Glacier。

如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon Glacier 的 Java 範例。您需要更新程式碼，如所示的要上傳的保存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 的低階 API 在單一操作中上傳封存 適用於 Java 的 AWS SDK

低階 API 提供所有封存操作的方法。以下是使用 適用於 Java 的 AWS SDK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 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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)、保存庫名稱、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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。
3. 以參數形式提供請求物件，以便執行 uploadArchive 方法。

   為了回應，Amazon Glacier (Amazon Glacier) 會傳回新上傳封存的封存 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 的低階 API 在單一操作中上傳封存 適用於 Java 的 AWS SDK

下列 Java 程式碼範例使用 適用於 Java 的 AWS SDK 將封存上傳至保存庫 (examplevault)。如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon Glacier 的 Java 範例。您需要更新程式碼，如所示的要上傳的保存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 Glacier 中使用 在單一操作 適用於 .NET 的 AWS SDK 中上傳封存

適用於 .NET 的 Amazon 開發套件提供的高階和低階 API 都提供了在單一作業中上傳封存的方法。

主題

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

使用 的高階 API 上傳封存 適用於 .NET 的 AWS SDK

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 的高階 API 上傳封存 適用於 .NET 的 AWS SDK

以下 C# 程式碼範例將封存上傳到美國西部 (奧勒岡) 區域中的保存庫 (`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 的低階 API 在單一操作中上傳封存 適用於 .NET 的 AWS SDK

低階 API 提供所有封存操作的方法。以下是使用 適用於 .NET 的 AWS SDK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 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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)、保存庫名稱和帳戶 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。如需詳細資訊，請參閱適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。
3. 以參數形式提供請求物件，以便執行 UploadArchive 方法。

   為了回應，Amazon Glacier 會傳回新上傳封存的封存 ID。

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

下列 C# 程式碼範例描述前述步驟。此範例使用 適用於 .NET 的 AWS SDK 將封存上傳至保存庫 (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 上傳大型封存 AWS CLI
- 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 上傳大型封存 適用於 .NET 的 AWS SDK
- 使用 REST API 上傳分段中的大型封存

分段上傳程序

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

1. 啟動分段上傳

   當您傳送啟動分段上傳的請求時，Amazon Glacier 會傳回分段上傳 ID，這是分段上傳的唯一識別符。任何後續分段上傳操作中都需要此 ID。此 ID 在 Amazon Glacier 完成任務後至少 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。Amazon 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。Amazon Glacier 會根據您提供的內容範圍，以遞增順序串連組件來建立封存。Amazon Glacier 對完整分段上傳請求的回應包含新建立封存的封存 ID。如果您在啟動分段上傳請求中提供選用的封存描述，Amazon 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 Glacier (Amazon Glacier) 提供下列額外的分段上傳 API 呼叫。

- 列出部分：使用此作業時，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。對於每個清單組件請求，Amazon Glacier 會傳回最多 1，000 個組件的資訊。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：使用此作業，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，Amazon 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 上傳大型封存 AWS CLI

您可以使用 AWS Command Line Interface () 在 Amazon Glacier (Amazon Glacier) 中上傳封存AWS CLI。為了改善大型封存的上傳體驗，Amazon Glacier 提供多種 API 操作來支援分段上傳。透過使用這些 API 作業，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 Amazon 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

- （先決條件）設定 AWS CLI
- (先決條件) 安裝 Python
- （先決條件）建立 Amazon Glacier 保存庫
- 範例：使用 分段上傳大型封存 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

設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

   - 若要查看 目前的組態資料 AWS CLI，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 Amazon Glacier 保存庫

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

範例：使用 分段上傳大型封存 AWS CLI

在此範例中，您將建立檔案並使用分段上傳 API 作業，將此檔案以部分的形式上傳到 Amazon 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，您必須建立保存庫、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，命令將輸出分段上傳資源在 Amazon Glacier 中的上傳 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，命令將輸出封存的 ID、檢查總和和和 Amazon Glacier 中的位置。

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 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。根據您上傳的檔案， 方法會在單一操作中上傳封存，或使用 Amazon Glacier (Amazon Glacier) 中的分段上傳支援來分段上傳封存。
- 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 適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用。

主題

- 使用 的高階 API 分段上傳大型封存 適用於 Java 的 AWS SDK
- 使用 的低階 API 分段上傳大型封存 適用於 Java 的 AWS SDK

使用 的高階 API 分段上傳大型封存 適用於 Java 的 AWS SDK

您可以使用高階 API 的相同方法來上傳小型或大型封存。根據封存大小，高階 API 方法會決定在單一操作中上傳封存，還是使用 Amazon Glacier 提供的分段上傳 API。如需詳細資訊，請參閱使用 的高階 API 上傳封存 適用於 Java 的 AWS SDK。

使用 的低階 API 分段上傳大型封存 適用於 Java 的 AWS SDK

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 適用於 Java 的 AWS SDK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 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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。在回應中，Amazon Glacier 會傳回上傳 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) 和保存庫名稱。Amazon Glacier 會從上傳的組件建構封存，並傳回封存 ID。

範例：使用 在組件中上傳大型封存 適用於 Java 的 AWS SDK

下列 Java 程式碼範例使用 適用於 Java 的 AWS SDK 將封存上傳至保存庫 (`examplevault`)。如需執行此範例的逐步說明，請參閱 使用 Eclipse 執行 Amazon 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，Amazon Glacier 支援部分大小最多 4 GB。

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 上傳大型封存 適用於 .NET 的 AWS SDK

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 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。根據您上傳的檔案， 方法會在單一操作中上傳封存，或使用 Amazon Glacier (Amazon Glacier) 中的分段上傳支援來分段上傳封存。
- 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 適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用。

主題

- 使用 的高階 API 分段上傳大型封存 適用於 .NET 的 AWS SDK

- 使用 的低階 API 分段上傳大型封存 適用於 .NET 的 AWS SDK

使用 的高階 API 分段上傳大型封存 適用於 .NET 的 AWS SDK

您可以使用高階 API 的相同方法來上傳小型或大型封存。根據封存大小，高階 API 方法會決定在單一操作中上傳封存，還是使用 Amazon Glacier 提供的分段上傳 API。如需詳細資訊，請參閱使用 的高階 API 上傳封存 適用於 .NET 的 AWS SDK。

使用 的低階 API 分段上傳大型封存 適用於 .NET 的 AWS SDK

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 適用於 .NET 的 AWS SDK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 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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保存庫名稱、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。在回應中，Amazon Glacier 會傳回上傳 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) 和保存庫名稱。Amazon Glacier 會從上傳的組件建構封存，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

下列 C# 程式碼範例使用 適用於 .NET 的 AWS SDK 將封存上傳至保存庫 (`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 Amazon Glacier 中下載封存

Amazon Glacier 提供管理主控台，可用來建立和刪除保存庫。不過，您無法使用 管理主控台從 Amazon Glacier 下載封存。若要下載資料，例如相片、影片和其他文件，您必須使用 AWS Command Line Interface (AWS CLI) 或編寫程式碼來提出請求，方法是直接使用 REST API 或使用 AWS SDKs。

如需搭配 使用 Amazon Glacier 的詳細資訊 AWS CLI，請參閱 AWS CLI Amazon Glacier 的參考。若要安裝 AWS CLI，請參閱 AWS Command Line Interface。下列主題說明如何使用 適用於 Java 的 AWS SDK、 適用於 .NET 的 AWS SDK和 Amazon Glacier REST API 將封存下載至 Amazon Glacier。

主題

擷取 Amazon Glacier Archives

從 Amazon Glacier 擷取封存是一種非同步操作，您會先啟動任務，然後在任務完成後下載輸出。若要啟動封存擷取任務，您可以使用 啟動任務 (POST 任務) REST API 操作或 AWS CLI或 AWS SDKs中的同等操作。

主題

從 Amazon Glacier 擷取封存是一個兩步驟的程序。以下是此程序的概觀。

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 SDKs 取得封存 ID。如需詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

b. 使用 啟動任務 (POST 任務)操作啟動請求 Amazon Glacier 準備整個封存或部分封存以進行後續下載的任務。

當您啟動任務時，Amazon Glacier 會在回應中傳回任務 ID，並以非同步方式執行任務。(如步驟 2 所述，直到工作完成後，您才能下載工作輸出。)

> **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 Amazon 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，您可以還原存放在 Amazon Glacier 中的大量資料區段。如需從 Amazon 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. 工作完成後，使用 「取得任務輸出」 (GET 輸出) 作業下載位元組。

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。

Amazon Glacier 必須先完成任務，才能取得其輸出。完成工作之後至少 24 小時內，工作不會過期，意思是您可以在工作完成後 24 小時內下載輸出。還原可能會在任務完成後 24 小時後隨時過期。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 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 可以在任務完成後發佈通知。Amazon Glacier 只會在完成任務後傳送通知。

啟動工作時，您可指定工作的 Amazon SNS 主題。除了在任務請求中指定 Amazon SNS 主題之外，如果您的保存庫已為封存擷取事件設定通知，則 Amazon Glacier 也會發佈通知至該 SNS 主題。如需詳細資訊，請參閱在 Amazon Glacier 中設定保存庫通知。

- 明確請求任務資訊 – 您也可以使用 Amazon Glacier `Describe Job` API 操作 (描述任務 (GET JobID)) 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 作業所取得的資訊相同。

封存擷取選項

啟動工作來擷取封存時，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。如需擷取定價的資訊，請參閱 Amazon 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的 Amazon Glacier 擷取選項，您可以在一天中以廉價的方式擷取大量資料，甚至是 PB。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 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`請求元素設定為您想要的選項，或 AWS Command Line Interface (AWS CLI) AWS SDKs中的對等項目。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 Amazon Glacier 主控台、購買佈建容量 (POST 佈建的容量)REST API 操作、 AWS SDKs 或 購買佈建容量單位 AWS CLI。如需佈建容量定價資訊，請參閱 Amazon 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 Glacier 主控台購買佈建容量

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon 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 Amazon Glacier 擷取存檔時，您可以選擇性指定要擷取之封存範圍或部分。預設值是擷取整個存檔。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 – Amazon Glacier 允許在擷取請求完成後 24 小時內下載擷取的資料。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 (擷取工作完成後)，您在工作啟動請求的範圍，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 在 Amazon Glacier 中下載封存 適用於 Java 的 AWS SDK

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 的高階 API 下載封存 適用於 Java 的 AWS SDK
- 使用 的低階 API 下載封存 適用於 Java 的 AWS SDK

使用 的高階 API 下載封存 適用於 Java 的 AWS SDK

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 的高階 API 下載封存 適用於 Java 的 AWS SDK

以下 Java 程式碼範例從美國西部 (奧勒岡) 區域 (`us-west-2`) 中的保存庫 (`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 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 的低階 API 下載封存 適用於 Java 的 AWS SDK

以下是使用 適用於 Java 的 AWS SDK 低階 API 來擷取保存庫庫存的步驟。

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

   您需要指定要 AWS 從中下載封存的區域。您使用此用戶端執行的所有操作都會套用到該 AWS 區域。
2. 執行 archive-retrieval 方法以啟動 initiateJob 任務。

   您可以透過建立 InitiateJobRequest類別的執行個體，提供任務資訊，例如您要下載之封存的封存 ID，以及您希望 Amazon Glacier (Amazon Glacier) 發佈任務完成訊息的選用 Amazon SNS 主題。Amazon Glacier 傳回任務 ID 以回應。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();
```

   您可以選擇性地指定位元組範圍，以請求 Amazon Glacier 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 Amazon 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 工作完成。

   您必須等到任務輸出準備好供您下載。如果您已在保存庫上設定通知組態來識別 Amazon Simple Notification Service (Amazon SNS) 主題，或在啟動任務時指定 Amazon SNS 主題，Amazon Glacier 會在完成任務後傳送訊息至該主題。

   您也可以呼叫 `describeJob`方法來輪詢 Amazon 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，如工作 ID 和保存庫名稱。Amazon Glacier 傳回的輸出可在 `GetJobOutputResult` 物件中使用。

```
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 ****");
```

   為了回應您的`GetJobOutput`呼叫，如果符合特定條件，Amazon Glacier 會傳回您下載之資料部分的檢查總和。如需詳細資訊，請參閱[下載資料時接收檢查總和](#)。

若要確認下載中沒有錯誤，您可以計算用戶端的檢查總和，並將其與回應中Amazon Glacier傳送的檢查總和進行比較。

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。您可以將其與 Amazon Glacier 傳回的檢查總和進行比較，以回應您的描述任務請求，以確認您下載的整個位元組範圍與存放在 Amazon Glacier 中的位元組範圍相同。

如需運作範例，請參閱 範例 2：使用 的低階 API 擷取封存 適用於 Java 的 AWS SDK- 在區塊中下載輸出。

範例 1：使用 的低階 API 擷取封存 適用於 Java 的 AWS SDK

以下 Java 程式碼範例從指定的保存庫下載封存。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 的低階 API 擷取封存 適用於 Java 的 AWS SDK- 在區塊中下載輸出。

範例會執行下列任務：

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

  Amazon Glacier 會在完成任務後傳送通知至此主題。
- 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，指定已建立的 Amazon SNS 主題，以便 Amazon Glacier 在完成任務後可以將通知發佈至主題。
- 定期檢查 Amazon SQS 佇列中是否有包含工作 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 的低階 API 擷取封存 適用於 Java 的 AWS SDK- 在區塊中下載輸出

下列 Java 程式碼範例會從 Amazon 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 在 Amazon Glacier 中下載封存 適用於 .NET 的 AWS SDK

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 的高階 API 下載封存 適用於 .NET 的 AWS SDK
- 使用 的低階 API 下載封存 適用於 .NET 的 AWS SDK

使用 的高階 API 下載封存 適用於 .NET 的 AWS SDK

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 的高階 API 下載封存 適用於 .NET 的 AWS SDK

以下 C# 程式碼範例從美國西部 (奧勒岡) 區域中的保存庫 (`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 的低階 API 下載封存 適用於 .NET 的 AWS SDK

以下是使用 的低階 API 下載 Amazon Glacier (Amazon Glacier) 封存的步驟 適用於 .NET 的 AWS SDK。

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 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 AWS 區域。

2. 執行 `archive-retrieval` 方法以啟動 `InitiateJob` 任務。

   您可以透過建立 `InitiateJobRequest`類別的執行個體，提供任務資訊，例如您要下載之封存的封存 ID，以及您希望 Amazon Glacier 發佈任務完成訊息的選用 Amazon SNS 主題。 Amazon Glacier Amazon Glacier 會傳回任務 ID 以回應。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;
```

您可以選擇性地指定位元組範圍，以請求 Amazon Glacier 僅準備封存的一部分，如下列請求所示。請求指定 Amazon Glacier 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 工作完成。

   您必須等到任務輸出準備好供您下載。如果您已在保存庫上設定通知組態來識別 Amazon Simple Notification Service (Amazon SNS) 主題，或在啟動任務時指定 Amazon SNS 主題，Amazon Glacier 會在任務完成後傳送訊息至該主題。下一節中提供的程式碼範例使用 Amazon SNS for Amazon Glacier 來發佈訊息。

   您也可以呼叫 `DescribeJob`方法來輪詢 Amazon 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，如工作 ID 和保存庫名稱。Amazon Glacier 傳回的輸出可在 `GetJobOutputResponse` 物件中使用。

```
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.
```

為了回應您的GetJobOutput呼叫，如果符合特定條件，Amazon Glacier 會傳回您下載之資料部分的檢查總和。如需詳細資訊，請參閱下載資料時接收檢查總和。

若要確認下載中沒有錯誤，您可以在用戶端運算檢查總和，並與回應中傳送的檢查總和 Amazon Glacier 進行比較。

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。您可以將其與 Amazon Glacier 傳回的檢查總和進行比較，以回應您的描述任務請求，以確認您下載的整個位元組範圍與存放在 Amazon Glacier 中的位元組範圍相同。

如需運作範例，請參閱 範例 2：使用 的低階 API 擷取封存 適用於 .NET 的 AWS SDK- 在區塊中下載輸出。

範例 1：使用 的低階 API 擷取封存 適用於 .NET 的 AWS SDK

以下 C# 程式碼範例從指定的保存庫下載封存。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 的低階 API 擷取封存 適用於 .NET 的 AWS SDK- 在區塊中下載輸出。

範例會執行下列任務：

- 設定 Amazon Simple Notification Service (Amazon SNS) 主題

  Amazon Glacier 會在完成任務後傳送通知至此主題。
- 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，範例會指定 Amazon SNS 主題，以便 Amazon Glacier 在完成任務後傳送訊息。
- 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\",&TCX5-2025-waiver;" +
        "    \"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 的低階 API 擷取封存 適用於 .NET 的 AWS SDK- 在區塊中下載輸出

下列 C# 程式碼範例會從 Amazon 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\",&TCX5-2025-waiver;" +
       "    \"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 Python 平行處理下載大型封存

本主題說明如何使用 Python 平行處理從 Amazon S3 Glacier (S3 Glacier) 下載大型封存。此方法可讓您將任何大小的封存可靠地下載，方法是將其分成可獨立處理的較小部分。

概觀

此範例中提供的 Python 指令碼會執行下列任務：

1. 設定通知的必要 AWS 資源 (Amazon SNS 主題和 Amazon SQS 佇列)
2. 使用 Amazon Glacier 啟動封存擷取任務
3. 監控任務完成通知的 Amazon SQS 佇列
4. 將大型封存分割為可管理區塊
5. 使用多個工作者執行緒平行下載區塊
6. 將每個區塊儲存至磁碟，以供日後重組

先決條件

開始之前，請確定您已：

- 已安裝 Python 3.6 或更新版本
- AWS 已安裝適用於 Python (Boto3) 的 SDK
- AWS 針對 Amazon Glacier、Amazon SNS 和 Amazon SQS 設定具有適當許可的憑證
- 有足夠的磁碟空間來存放下載的封存區塊

範例：使用 Python 平行處理下載封存

下列 Python 指令碼示範如何使用平行處理從 Amazon Glacier 下載大型封存：

```
import boto3
import time
import json
import jmespath
import re
import concurrent.futures
import os
```

```
output_file_path = "output_directory_path"
vault_name = "vault_name"

chunk_size = 1000000000 #1gb - size of chunks for parallel download.
notify_queue_name = 'GlacierJobCompleteNotifyQueue' # SQS queue for Glacier recall
 notification
chunk_download_queue_name='GlacierChunkReadyNotifyQueue' # SQS queue for chunks
sns_topic_name = 'GlacierRecallJobCompleted' # the SNS topic to be notified when
 Glacier archive is restored.
chunk_queue_visibility_timeout = 7200 # 2 hours - this may need to be adjusted.
region = 'us-east-1'
archive_id = "archive_id_to_restore"
retrieve_archive = True # set to false if you do not want to restore from Glacier -
 useful for restarting or parallel processing of the chunk queue.
workers = 12 # the number of parallel worker threads for downloading chunks.

def setup_queues_and_topic():
    sqs = boto3.client('sqs')
    sns = boto3.client('sns')

    # Create the SNS topic
    topic_response = sns.create_topic(
        Name=sns_topic_name
    )
    topic_arn = topic_response['TopicArn']
    print("Creating the SNS topic " + topic_arn)

    # Create the notification queue
    notify_queue_response = sqs.create_queue(
        QueueName=notify_queue_name,
        Attributes={
            'VisibilityTimeout': '300',  # 5 minutes
            'ReceiveMessageWaitTimeSeconds': '20'  # Enable long polling
        }
    )
    notify_queue_url = notify_queue_response['QueueUrl']
    print("Creating the archive-retrieval notification queue " + notify_queue_url)

    # Create the chunk download queue
    chunk_queue_response = sqs.create_queue(
        QueueName=chunk_download_queue_name,
        Attributes={
            'VisibilityTimeout': str(chunk_queue_visibility_timeout),  # 5 minutes
            'ReceiveMessageWaitTimeSeconds': '0'
```

```
        }
    )
    chunk_queue_url = chunk_queue_response['QueueUrl']

    print("Creating the chunk ready notification queue " + chunk_queue_url)


   # Get the ARN for the notification queue
    notify_queue_attributes = sqs.get_queue_attributes(
        QueueUrl=notify_queue_url,
        AttributeNames=['QueueArn']
    )
    notify_queue_arn = notify_queue_attributes['Attributes']['QueueArn']

    # Set up the SNS topic policy on the notification queue
    queue_policy = {
        "Version": "2012-10-17",
        "Statement": [{
            "Sid": "allow-sns-messages",
            "Effect": "Allow",
            "Principal": {"AWS": "*"},
            "Action": "SQS:SendMessage",
            "Resource": notify_queue_arn,
            "Condition": {
                "ArnEquals": {
                    "aws:SourceArn": topic_arn
                }
            }
        }]
    }

    # Set the queue policy
    sqs.set_queue_attributes(
        QueueUrl=notify_queue_url,
        Attributes={
            'Policy': json.dumps(queue_policy)
        }
    )

    # Subscribe the notification queue to the SNS topic
    sns.subscribe(
        TopicArn=topic_arn,
        Protocol='sqs',
        Endpoint=notify_queue_arn
```

```
    )

    return {
        'topic_arn': topic_arn,
        'notify_queue_url': notify_queue_url,
        'chunk_queue_url': chunk_queue_url
    }


def split_and_send_chunks(archive_size, job_id,chunk_queue_url):
    ranges = []
    current = 0
    chunk_number = 0

    while current < archive_size:
        chunk_number += 1
        next_range = min(current + chunk_size - 1, archive_size - 1)
        ranges.append((current, next_range, chunk_number))
        current = next_range + 1

    # Send messages to SQS queue
    for start, end, chunk_number in ranges:
        body = {"start": start, "end": end, "job_id": job_id, "chunk_number":
 chunk_number}
        body = json.dumps(body)
        print("Sending SQS message for range:" + str(body))
        response = sqs.send_message(
            QueueUrl=chunk_queue_url,
            MessageBody=str(body)
        )

def GetJobOutputChunks(job_id, byterange, chunk_number):
    glacier = boto3.client('glacier')
    response = glacier.get_job_output(
        vaultName=vault_name,
        jobId=job_id,
        range=byterange,

    )

    with open(os.path.join(output_file_path,str(chunk_number)+".chunk"), 'wb') as
 output_file:
        output_file.write(response['body'].read())
```

```
    return response

def ReceiveArchiveReadyMessages(notify_queue_url,chunk_queue_url):

    response = sqs.receive_message(
        QueueUrl=notify_queue_url,
        AttributeNames=['All'],
        MaxNumberOfMessages=1,
        WaitTimeSeconds=20,
        MessageAttributeNames=['Message']
    )
    print("Polling archive retrieval job ready queue...")
    # Checking that there is a Messages key before proceeding. No 'Messages' key likely
 means the queue is empty

    if 'Messages' in response:
        print("Received a message from the archive retrieval job queue")
        jsonresponse = response
        # Loading the string into JSON and checking that ArchiveSizeInBytes key is
 present before continuing.
        jsonresponse=json.loads(jsonresponse['Messages'][0]['Body'])
        jsonresponse=json.loads(jsonresponse['Message'])
        if 'ArchiveSizeInBytes' in jsonresponse:
            receipt_handle = response['Messages'][0]['ReceiptHandle']
            if jsonresponse['ArchiveSizeInBytes']:
                archive_size = jsonresponse['ArchiveSizeInBytes']

                print(f'Received message: {response}')
                if archive_size > chunk_size:
                    split_and_send_chunks(archive_size,
 jsonresponse['JobId'],chunk_queue_url)

                    sqs.delete_message(
                    QueueUrl=notify_queue_url,
                    ReceiptHandle=receipt_handle)

            else:
                print("No ArchiveSizeInBytes value found in message")
                print(response)

    else:
        print('No messages available in the queue at this time.')

    time.sleep(1)
```

```
def ReceiveArchiveChunkMessages(chunk_queue_url):
    response = sqs.receive_message(
        QueueUrl=chunk_queue_url,
        AttributeNames=['All'],
        MaxNumberOfMessages=1,
        WaitTimeSeconds=0,
        MessageAttributeNames=['Message']
    )
    print("Polling archive chunk queue...")
    print(response)
    # Checking that there is a Messages key before proceeding. No 'Messages' key likely
 means the queue is empty
    if 'Messages' in response:
        jsonresponse = response
        # Loading the string into JSON and checking that ArchiveSizeInBytes key is
 present before continuing.
        jsonresponse=json.loads(jsonresponse['Messages'][0]['Body'])
        if 'job_id' in jsonresponse: #checking that there is a job id before continuing
            job_id = jsonresponse['job_id']
            byterange = "bytes="+str(jsonresponse['start']) + '-' +
 str(jsonresponse['end'])
            chunk_number = jsonresponse['chunk_number']
            receipt_handle = response['Messages'][0]['ReceiptHandle']
            if jsonresponse['job_id']:
                print(f'Received message: {response}')
                GetJobOutputChunks(job_id,byterange,chunk_number)
                sqs.delete_message(
                QueueUrl=chunk_queue_url,
                ReceiptHandle=receipt_handle)
    else:
        print('No messages available in the chunk queue at this time.')

def initiate_archive_retrieval(archive_id, topic_arn):
    glacier = boto3.client('glacier')

    job_parameters = {
        "Type": "archive-retrieval",
        "ArchiveId": archive_id,
        "Description": "Archive retrieval job",
        "SNSTopic": topic_arn,
        "Tier": "Bulk"  # You can change this to "Standard" or "Expedited" based on
 your needs
    }
```

```
    try:
        response = glacier.initiate_job(
            vaultName=vault_name,
            jobParameters=job_parameters
        )

        print("Archive retrieval job initiated:")
        print(f"Job ID: {response['jobId']}")
        print(f"Job parameters: {job_parameters}")
        print(f"Complete response: {json.dumps(response, indent=2)}")

        return response['jobId']

    except Exception as e:
        print(f"Error initiating archive retrieval job: {str(e)}")
        raise

def run_async_tasks(chunk_queue_url, workers):
    max_workers = workers  # Set the desired maximum number of concurrent tasks
    with concurrent.futures.ThreadPoolExecutor(max_workers=max_workers) as executor:
        for _ in range(max_workers):
            executor.submit(ReceiveArchiveChunkMessages, chunk_queue_url)

# One time setup of the necessary queues and topics.
queue_and_topic_atts = setup_queues_and_topic()

topic_arn = queue_and_topic_atts['topic_arn']
notify_queue_url = queue_and_topic_atts['notify_queue_url']
chunk_queue_url = queue_and_topic_atts['chunk_queue_url']

if retrieve_archive:
    print("Retrieving the defined archive... The topic arn we will notify when
 recalling the archive is: "+topic_arn)
    job_id = initiate_archive_retrieval(archive_id, topic_arn)
else:
    print("Retrieve archive is false, polling queues and downloading only.")

while True:
   ReceiveArchiveReadyMessages(notify_queue_url,chunk_queue_url)
   run_async_tasks(chunk_queue_url,workers)
```

使用指令碼

若要使用此指令碼，請遵循下列步驟：

1. 將指令碼中的預留位置值取代為您的特定資訊：
   - *output_file_path*：儲存區塊檔案的目錄
   - *vault_name*：S3 Glacier 保存庫的名稱
   - *notify_queue_name*：任務通知佇列的名稱
   - *chunk_download_queue_name*：區塊下載佇列的名稱
   - *sns_topic_name*：SNS 主題的名稱
   - *region*：region 您的保存庫所在的 AWS 區域
   - *archive_id*：要擷取的封存 ID
2. 執行 指令碼：

```
python download_large_archive.py
```

3. 下載所有區塊之後，您可以使用下列命令將它們合併為單一檔案：

```
cat /path/to/chunks/*.chunk > complete_archive.file
```

重要考量

使用此指令碼時，請記住下列事項：

- 從 S3 Glacier 擷取封存可能需要數小時才能完成，具體取決於選取的擷取層。
- 指令碼會無限期執行，持續輪詢佇列。您可能想要根據您的特定需求新增終止條件。
- 請確定您有足夠的磁碟空間來存放封存的所有區塊。
- 如果指令碼中斷，您可以使用 重新啟動指令碼`retrieve_archive=False`，以繼續下載區塊，而無需啟動新的擷取任務。
- 根據您的網路頻寬和系統資源調整 *chunk_size* 和###參數。
- Amazon S3 擷取、Amazon SNS 和 Amazon SQS 用量需支付標準 AWS 費用。

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。如需詳細資訊，請參閱啟動任務 (POST 任務)。
2. 工作完成後，下載封存資料。如需詳細資訊，請參閱「取得任務輸出」 (GET 輸出)。

使用 在 Amazon Glacier 中下載封存 AWS CLI

您可以使用 AWS Command Line Interface () 在 Amazon Glacier (Amazon Glacier) 中下載封存AWS CLI。

主題

- （先決條件）設定 AWS CLI
- 範例：使用 下載封存 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

   設定 AWS Command Line Interface
2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

- 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

```
aws configure list
```

範例：使用 下載封存 AWS CLI

> **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` 命令開始庫存擷取工作。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 工作命令的狀態。

```
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 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組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在任務完成後傳送訊息至主題。

   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 Glacier 中設定保存庫通知。Amazon Glacier 會在發生特定事件時傳送訊息至指定的 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，從保存庫開始每個封存的擷取程序。您將需要將工作參數指定為 `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 工作完成。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. 當上述工作完成後，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
```

在 Amazon Glacier 中刪除封存

您無法使用 Amazon Glacier (Amazon Glacier) 管理主控台刪除封存。若要刪除封存，您必須使用 AWS Command Line Interface (CLI) 或編寫程式碼，直接使用 REST API 或 適用於 Java 的 AWS SDK 和 .NET 包裝函式程式庫提出刪除請求。下列主題說明如何使用 適用於 Java 的 AWS SDK 和 .NET 包裝函式程式庫、REST API 和 AWS CLI。

主題

- 使用 在 Amazon Glacier 中刪除封存 適用於 Java 的 AWS SDK
- 使用 在 Amazon Glacier 中刪除封存 適用於 .NET 的 AWS SDK
- 使用 REST API 刪除 Amazon Glacier Archive
- 使用 在 Amazon Glacier 中刪除封存 AWS Command Line Interface

您可以一次從保存庫刪除一個封存。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保存庫下載保存庫庫存來取得封存 ID。如需下載保存庫庫存的詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

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 Glacier 收到刪除封存請求時主動準備資料以供下載，則封存擷取操作可能會失敗。
- 如果封存擷取任務在 Amazon Glacier 收到刪除封存請求時已成功準備要下載的封存，則您將能夠下載輸出。

如需封存擷取的詳細資訊，請參閱 在 Amazon 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。

刪除封存之後，如果您立即下載保存庫庫存，它可能會在清單中包含已刪除的封存，因為 Amazon Glacier 每天只會準備一次保存庫庫存。

> **Note**
>
> 如需自動刪除保存庫封存，請參閱 Amazon S3 Glacier 中的自動刪除保存庫封存。

使用 在 Amazon Glacier 中刪除封存 適用於 Java 的 AWS SDK

以下是使用 適用於 Java 的 AWS SDK 低階 API 刪除封存的步驟。

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

   您需要指定儲存您要刪除的封存 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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、保存庫名稱和您的帳戶 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。如需詳細資訊，請參閱適用於 Java 的 AWS SDK 搭配 Amazon Glacier 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 刪除封存 適用於 Java 的 AWS SDK

下列 Java 程式碼範例使用 適用於 Java 的 AWS SDK 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 Glacier 的 Java 範例。您需要如所示，使用保存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 在 Amazon Glacier 中刪除封存 適用於 .NET 的 AWS SDK

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 的高階 API 刪除封存 適用於 .NET 的 AWS SDK
- 使用低階 API 刪除封存 適用於 .NET 的 AWS SDK

使用 的高階 API 刪除封存 適用於 .NET 的 AWS SDK

高階 API 的 `ArchiveTransferManager` 類別提供可用來刪除封存的 `DeleteArchive` 方法。

範例：使用 的高階 API 刪除封存 適用於 .NET 的 AWS SDK

下列 C# 程式碼範例使用 的高階 API 適用於 .NET 的 AWS SDK 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();
    }
  }
}
```

使用低階 API 刪除封存 適用於 .NET 的 AWS SDK

以下是使用 適用於 .NET 的 AWS SDK刪除保存庫的步驟。

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

   您需要指定儲存您要刪除的封存 AWS 的區域。您使用此用戶端執行的所有操作都會套用到該 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、保存庫名稱和您的帳戶 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。如需詳細資訊，請參閱搭配 Amazon Glacier 使用 AWS SDKs。
3. 以參數形式提供請求物件，以便執行 `DeleteArchive` 方法。

範例：使用 的低階 API 刪除封存 適用於 .NET 的 AWS SDK

下列 C# 範例描述前述步驟。此範例使用 的低階 API 適用於 .NET 的 AWS SDK 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 Glacier Archive

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 Glacier 的 API 參考。

使用 在 Amazon Glacier 中刪除封存 AWS Command Line Interface

您可以使用 AWS Command Line Interface () 刪除 Amazon Glacier (Amazon Glacier) 中的封存AWS CLI。

主題

- （先決條件）設定 AWS CLI
- 範例：使用 刪除封存 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

   設定 AWS Command Line Interface

2. 在命令提示中輸入下列命令來驗證您的 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的 Amazon Glacier 保存庫清單，請使用 `list-vaults`命令。將 *123456789012* 取代為您的 AWS 帳戶 ID。

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

- 若要查看 目前的組態資料 AWS CLI，請使用 `aws configure list`命令。

```
aws configure list
```

範例：使用 刪除封存 AWS CLI

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 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組態，或在啟動任務時指定 Amazon Simple Notification Service (Amazon SNS) 主題，Amazon Glacier 會在任務完成後傳送訊息至主題。

   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 Glacier 中設定保存庫通知。Amazon Glacier 會在發生特定事件時傳送訊息至指定的 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 ***
```

搭配 Amazon Glacier 使用 AWS SDKs

AWS 為您提供SDKs，以開發 Amazon Glacier 的應用程式。開發套件程式庫包裝基礎 Amazon Glacier API，簡化您的程式設計任務。例如，對於傳送至 Amazon Glacier 的每個請求，您必須包含簽章以驗證您的請求。使用 SDK 程式庫時，您只需在程式碼中提供 AWS 安全登入資料，程式庫會計算必要的簽章，並將其包含在傳送至 Amazon Glacier 的請求中。 AWS SDKs 提供對應至基礎 REST API 的程式庫，並提供可用來輕鬆建構請求和處理回應的物件。

主題

AWS Command Line Interface (AWS CLI) 是管理 的統一工具 AWS 服務，包括 Amazon Glacier。如需下載 的資訊 AWS CLI，請參閱 AWS Command Line Interface。如需 Amazon Glacier CLI 命令的清單，請參閱 AWS CLI 命令參考。

AWS 適用於 Java 和 .NET 的 SDK 程式庫

適用於 Java 和 .NET AWS SDKs 提供高階和低階包裝函式程式庫。

您可以使用本開發人員指南 適用於 .NET 的 AWS SDK 中的 適用於 Java 的 AWS SDK 和 ，找到使用 Amazon Glacier 的範例。

什麼是低階 API？

低階包裝函式程式庫緊密對應 Amazon Glacier 支援的基礎 REST API (Amazon Glacier 的 API 參考)。對於每個 Amazon Glacier REST 操作，低階 API 會提供對應的方法、請求物件，讓您提供請求資訊和回應物件，讓您處理 Amazon Glacier 回應。低階包裝函式程式庫是基礎 Amazon Glacier 操作最完整的實作。

如需開發套件程式庫的詳細資訊，請參閱 適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用 和 適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 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。您需要撰寫其他程式碼來設定 Amazon Glacier 的 Amazon Simple Notification Service (Amazon SNS) 主題，以便在任務完成時通知您。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 適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用 和 適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 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 SDK 使用 Amazon Glacier

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。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
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例

SDK 文件	代碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 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 Amazon Glacier 特有的範例，請參閱 使用 AWS SDKs Amazon Glacier 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。

適用於 Java 的 AWS SDK 搭配 Amazon Glacier 使用

適用於 Java 的 AWS SDK 同時提供 Amazon Glacier (Amazon Glacier) 的高階和低階 APIs，如中所述搭配 Amazon Glacier 使用 AWS SDKs。如需下載 的相關資訊 適用於 Java 的 AWS SDK，請參閱適用於 Java 的 Amazon 開發套件。

> **Note**
>
> 適用於 Java 的 AWS SDK 提供安全執行緒用戶端來存取 Amazon Glacier。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`類別提供映射至 Amazon Glacier 基礎 REST 操作的所有方法 ()Amazon Glacier 的 API 參考。呼叫任何這些方法時，您必須建立對應的請求物件，並提供回應物件，其中方法可以傳回 Amazon Glacier 回應給 操作。

例如，`AmazonGlacierClient` 類別提供 `createVault` 方法來建立保存庫。此方法對應到底層建立保存庫 REST 作業 (請參閱 建立保存庫 (PUT 保存庫))。若要使用此方法，您必須建立接收 Amazon Glacier 回應的`CreateVaultResult`物件執行個體，如下列 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`。不過，使用 時 適用於 Java 的 AWS SDK，請求`AccountId`中的 是選用的，因此本指南中的所有低階範例都不會設定此值。`AccountId` 是 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶

ID 相符。您可以指定 AWS 帳戶 ID 或選擇性的 '-'，在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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。使用 時 適用於 Java 的 AWS SDK，如果您不提供帳戶 ID，程式庫會將帳戶 ID 設定為 '-'。

使用高階 API

為了進一步簡化您的應用程式開發， 適用於 Java 的 AWS SDK 提供 `ArchiveTransferManager`類別，可針對低階 API 中的某些方法實作更高層級的抽象。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會套用到您在建立`ArchiveTransferManager`物件時指定的 AWS 區域。如果您未指定任何 AWS 區域， 會將 適用於 Java 的 AWS SDK 設定為`us-east-1`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 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	如 Amazon SDK for Java 中的提供登入資料 適用於 Java 的 AWS SDK 主題所述，為您的 AWS 登入資料建立預設登入資料設定檔。 AWS
2	在 Eclipse 中建立新的 AWS Java 專案。專案是使用 適用於 Java 的 AWS SDK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 管理主控台來建立物件。如需 的詳細資訊 AWS 管理主控台，請前往 https：//http://aws.amazon.com/console/。

設定終端節點

根據預設， 適用於 Java 的 AWS SDK 會使用端點 `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 的區域和端點清單，請參閱 存取 Amazon Glacier。

適用於 .NET 的 AWS SDK 搭配 Amazon Glacier 使用

適用於 .NET 的 AWS SDK API 可在 中使用`AWSSDK.dll`。如需下載 的相關資訊 適用於 .NET 的 AWS SDK，請前往範例程式碼程式庫。如中所述搭配 Amazon Glacier 使用 AWS SDKs， 同時 適用於 .NET 的 AWS SDK 提供高階和低階 APIs。

> **Note**
>
> 低階 API 和高階 API 提供用於存取 Amazon Glacier 的執行緒安全用戶端。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`類別提供映射至 Amazon Glacier (Amazon Glacier) 基礎 REST 操作的所有方法 ()Amazon Glacier 的 API 參考。呼叫任何這些方法時，您必須建立對應的請求物件，並提供回應物件，其中方法可以傳回 Amazon Glacier 回應給 操作。

例如，`AmazonGlacierClient` 類別提供 `CreateVault` 方法來建立保存庫。此方法對應到底層建立保存庫 REST 作業 (請參閱 建立保存庫 (PUT 保存庫))。若要使用此方法，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 Amazon Glacier 回應，如下列 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。不過，使用 時 適用於 .NET 的 AWS SDK，請求AccountId中的 是選用的，因此本指南中的所有低階範例都不會設定此值。AccountId 是 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性的 '-'，在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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。使用 時 適用於 .NET 的 AWS SDK，如果您不提供帳戶 ID，程式庫會將帳戶 ID 設定為 '-'。

使用高階 API

為了進一步簡化您的應用程式開發， 適用於 .NET 的 AWS SDK 提供 ArchiveTransferManager類別，可針對低階 API 中的某些方法實作更高層級的抽象。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會套用到您在建立ArchiveTransferManager物件時指定的 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 適用於 .NET 的 AWS SDK。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	建立登入 AWS 資料的登入資料設定檔，如 Amazon SDK for .NET 主題設定 AWS 登入資料中所述。
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 管理主控台來建立物件。如需詳細資訊 AWS 管理主控台，請前往 https：//http://aws.amazon.com/console/。

設定終端節點

根據預設， 適用於 .NET 的 AWS SDK 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，如下列 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);
```

如需目前支援 AWS 的區域和端點清單，請參閱 存取 Amazon Glacier。

使用 AWS SDKs Amazon Glacier 程式碼範例

下列程式碼範例示範如何使用 Amazon Glacier 搭配 AWS 軟體開發套件 (SDK)。

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中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- 將檔案存檔至 Amazon Glacier、取得通知，以及使用 AWS SDK 啟動任務
- 使用 AWS SDK 取得 Amazon Glacier 封存內容並刪除封存

使用 AWS SDKs Amazon Glacier 基本範例

下列程式碼範例示範如何搭配 AWS SDKs 使用 Amazon Glacier 的基本概念。

範例

- 您好，Amazon Glacier
- Amazon Glacier AWS SDKs的動作
  - AddTagsToVault 搭配 AWS SDK 或 CLI 使用
  - CreateVault 搭配 AWS SDK 或 CLI 使用
  - DeleteArchive 搭配 AWS SDK 或 CLI 使用
  - DeleteVault 搭配 AWS SDK 或 CLI 使用
  - DeleteVaultNotifications 搭配 AWS SDK 或 CLI 使用
  - DescribeJob 搭配 AWS SDK 或 CLI 使用
  - DescribeVault 搭配 AWS SDK 或 CLI 使用
  - GetJobOutput 搭配 AWS SDK 或 CLI 使用
  - GetVaultNotifications 搭配 AWS SDK 或 CLI 使用
  - InitiateJob 搭配 AWS SDK 或 CLI 使用
  - ListJobs 搭配 AWS SDK 或 CLI 使用
  - ListTagsForVault 搭配 AWS SDK 或 CLI 使用
  - ListVaults 搭配 AWS SDK 或 CLI 使用
  - SetVaultNotifications 搭配 AWS SDK 或 CLI 使用
  - UploadArchive 搭配 AWS SDK 或 CLI 使用
  - UploadMultipartPart 搭配 AWS SDK 或 CLI 使用

您好，Amazon Glacier

下列程式碼範例示範如何開始使用 Amazon Glacier。

.NET

適用於 .NET 的 SDK

> **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。

```
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《適用於 .NET 的 AWS SDK API 參考》中的 ListVaults。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon Glacier AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Amazon Glacier 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

這些摘錄會呼叫 Amazon Glacier API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 使用 AWS SDKs Amazon Glacier 案例 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱 Amazon Glacier API 參考。

範例

- AddTagsToVault 搭配 AWS SDK 或 CLI 使用
- CreateVault 搭配 AWS SDK 或 CLI 使用
- DeleteArchive 搭配 AWS SDK 或 CLI 使用
- DeleteVault 搭配 AWS SDK 或 CLI 使用
- DeleteVaultNotifications 搭配 AWS SDK 或 CLI 使用
- DescribeJob 搭配 AWS SDK 或 CLI 使用
- DescribeVault 搭配 AWS SDK 或 CLI 使用
- GetJobOutput 搭配 AWS SDK 或 CLI 使用
- GetVaultNotifications 搭配 AWS SDK 或 CLI 使用
- InitiateJob 搭配 AWS SDK 或 CLI 使用
- ListJobs 搭配 AWS SDK 或 CLI 使用
- ListTagsForVault 搭配 AWS SDK 或 CLI 使用
- ListVaults 搭配 AWS SDK 或 CLI 使用
- SetVaultNotifications 搭配 AWS SDK 或 CLI 使用
- UploadArchive 搭配 AWS SDK 或 CLI 使用
- UploadMultipartPart 搭配 AWS SDK 或 CLI 使用

`AddTagsToVault` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `AddTagsToVault`。

.NET

適用於 .NET 的 SDK

> **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。

```
    /// <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《適用於 .NET 的 AWS SDK API 參考》中的 AddTagsToVault。

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》中的 AddTagsToVault。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`CreateVault` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 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

適用於 .NET 的 SDK

> **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。

```
/// <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《適用於 .NET 的 AWS SDK API 參考》中的 CreateVault。

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》中的 CreateVault。

Java

SDK for Java 2.x

> **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.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 的詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 CreateVault。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **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。

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《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 的詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateVault。

適用於 JavaScript (v2) 的 SDK

> **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。

```
// 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《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 的詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 CreateVault。

PowerShell

Tools for PowerShell V4

範例 1：為使用者帳戶建立新的保存庫。由於未將任何值提供給 -AccountId 參數，Cmdlet 會使用預設值 "-" 來表示目前的帳戶。

```
New-GLCVault -VaultName myvault
```

輸出：

```
/01234567812/vaults/myvault
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V4)》中的 CreateVault。

Tools for PowerShell V5

範例 1：為使用者帳戶建立新的保存庫。由於未將任何值提供給 -AccountId 參數，Cmdlet 會使用預設值 "-" 來表示目前的帳戶。

```
New-GLCVault -VaultName myvault
```

輸出：

```
/01234567812/vaults/myvault
```

- 如需 API 詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考 (V5)》中的 CreateVault。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 CreateVault。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteArchive` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 DeleteArchive。

Java

SDK for Java 2.x

> **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.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》中的 DeleteArchive。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS 開發套件 API 參考》中的 DeleteArchive。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DeleteVault` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 DeleteVault。

Java

SDK for Java 2.x

> **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.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 的詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 DeleteVault。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 DeleteVault。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DeleteVaultNotifications 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 DeleteVaultNotifications。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的 AWS 開發套件 API 參考》中的 DeleteVaultNotifications。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeJob` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 DescribeJob。

PowerShell

Tools for PowerShell V4

範例 1：傳回指定任務的詳細資訊。當任務成功完成時，可使用 Read-GCJobOutput cmdlet 將任務的內容 (封存或庫存清單) 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 Cmdlet 參考 (V4)》中的 DescribeJob。

Tools for PowerShell V5

範例 1：傳回指定任務的詳細資訊。當任務成功完成時，可使用 Read-GCJobOutput cmdlet 將任務的內容 (封存或庫存清單) 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 Cmdlet 參考 (V5)》中的 DescribeJob。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 DescribeJob。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeVault` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DescribeVault`。

.NET

適用於 .NET 的 SDK

> **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。

```
/// <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《適用於 .NET 的 AWS SDK API 參考》中的 DescribeVault。

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》中的 DescribeVault。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetJobOutput 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 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》中的 GetJobOutput。

PowerShell

Tools for PowerShell V4

範例 1：下載排程於指定任務中擷取的封存內容，並將內容存放在磁碟上的檔案中。如果有的話，下載會為您驗證檢查總和。如果需要，可以透過指定 **-Select '*'** 傳回包含檢查總和的整體回應。

```
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 Cmdlet 參考 (V4)》中的 GetJobOutput。

Tools for PowerShell V5

範例 1：下載排程於指定任務中擷取的封存內容，並將內容存放在磁碟上的檔案中。如果有的話，下載會為您驗證檢查總和。如果需要，可以透過指定 `-Select '*'` 傳回包含檢查總和的整體回應。

```
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 Cmdlet 參考 (V5)》中的 GetJobOutput。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 GetJobOutput。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`GetVaultNotifications` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 GetVaultNotifications。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 GetVaultNotifications。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

InitiateJob 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 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

適用於 .NET 的 SDK

> **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。

從保存庫擷取封存。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《適用於 .NET 的 AWS SDK API 參考》中的 InitiateJob。

CLI

AWS CLI

下列命令會啟動任務，以取得保存庫 my-vault 的庫存：

```
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。

下列命令會啟動任務，從保存庫 `my-vault` 擷取封存：

```
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，其中會指定任務類型、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"
}
```

封存 ID 可在 `aws glacier upload-archive` 和 `aws glacier get-job-output` 的輸出中使用。

輸出：

```
{
    "location": "/011685312445/vaults/mwunderl/jobs/l7IL5-
EkXyEY9Ws95fClzIbk2O5uLYaFdAYOi-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav",
    "jobId": "l7IL5-EkXy2O5uLYaFdAYOiEY9Ws95fClzIbk-
azsX_Z8V6NH4yERHzars8wTKYQMX6nBDI9cMNHzyZJO59-8N9aHWav"
}
```

如需任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》中的 InitiateJob。

Java

SDK for Java 2.x

> **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。

擷取保存庫庫存。

```
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》中的 InitiateJob。

PowerShell

Tools for PowerShell V4

範例 1：啟動任務，從使用者擁有的指定保存庫擷取封存。您可以使用 Get-GLCJob cmdlet 檢查任務的狀態。當任務成功完成時，可使用 Read-GCJobOutput cmdlet 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 Cmdlet 參考 (V4)》中的 InitiateJob。

Tools for PowerShell V5

範例 1：啟動任務，從使用者擁有的指定保存庫擷取封存。您可以使用 Get-GLCJob cmdlet 檢查任務的狀態。當任務成功完成時，可使用 Read-GCJobOutput cmdlet 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 Cmdlet 參考 (V5)》中的 InitiateJob。

Python

適用於 Python 的 SDK (Boto3)

> **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。

擷取保存庫庫存。

```
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
```

從保存庫擷取封存。

```
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_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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 InitiateJob。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListJobs` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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

適用於 .NET 的 SDK

> **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。

```
    /// <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《適用於 .NET 的 AWS SDK API 參考》中的 ListJobs。

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》中的 ListJobs。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListTagsForVault` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `ListTagsForVault`。

.NET

適用於 .NET 的 SDK

> **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。

```
    /// <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《適用於 .NET 的 AWS SDK API 參考》中的 ListTagsForVault。

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》中的 ListTagsForVault。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`ListVaults` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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

適用於 .NET 的 SDK

> **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。

```
/// <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《適用於 .NET 的 AWS SDK API 參考》中的 ListVaults。

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》中的 ListVaults。

Java

SDK for Java 2.x

> **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.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 的詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 ListVaults。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 ListVaults。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`SetVaultNotifications` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `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》中的 SetVaultNotifications。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 SetVaultNotifications。

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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadArchive 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 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

適用於 .NET 的 SDK

> **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。

```
   /// <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《適用於 .NET 的 AWS SDK API 參考》的 UploadArchive。

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 的詳細資訊，請參閱《AWS CLI 命令參考》的 UploadArchive。

Java

SDK for Java 2.x

> **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。

JavaScript

適用於 JavaScript (v3) 的 SDK

> **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。

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《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 的詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》的 UploadArchive。

適用於 JavaScript (v2) 的 SDK

> **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。

```
// 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《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 的詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》的 UploadArchive。

PowerShell

Tools for PowerShell V4

範例 1：將單一檔案上傳到指定的保存庫，傳回封存 ID 和運算的檢查總和。

```
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，上傳至使用者帳戶中指定的保存庫。對於每個上傳的檔案，Cmdlet 會發出檔案名稱、對應的封存 ID 和封存的運算檢查總和。

```
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 Cmdlet 參考 (V4)》的 UploadArchive。

Tools for PowerShell V5

範例 1：將單一檔案上傳到指定的保存庫，傳回封存 ID 和運算的檢查總和。

```
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，上傳至使用者帳戶中指定的保存庫。對於每個上傳的檔案，Cmdlet 會發出檔案名稱、對應的封存 ID 和封存的運算檢查總和。

```
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 Cmdlet 參考 (V5)》的 UploadArchive。

Python

適用於 Python 的 SDK (Boto3)

> **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。

```
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《適用於 Python (Boto3) 的AWS 開發套件 API 參考》中的 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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

UploadMultipartPart 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 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。

如需使用 CLI 將分段上傳至 Amazon Glacier AWS 的詳細資訊，請參閱《AWS CLI 使用者指南》中的使用 Amazon Glacier。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 UploadMultipartPart。

JavaScript

適用於 JavaScript (v2) 的 SDK

> **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。

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《適用於 JavaScript 的 AWS SDK 開發人員指南》。
- 如需 API 詳細資訊，請參閱《適用於 JavaScript 的 AWS SDK API 參考》中的 UploadMultipartPart。

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 SDK 使用 Amazon Glacier。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 SDKs Amazon Glacier 案例

下列程式碼範例示範如何在 Amazon Glacier AWS SDKs 中實作常見案例。這些案例說明如何透過在 Amazon Glacier 中呼叫多個函數或與其他函數結合，來完成特定任務 AWS 服務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執行程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- 將檔案存檔至 Amazon Glacier、取得通知，以及使用 AWS SDK 啟動任務
- 使用 AWS SDK 取得 Amazon Glacier 封存內容並刪除封存

將檔案存檔至 Amazon Glacier、取得通知，以及使用 AWS SDK 啟動任務

以下程式碼範例顯示做法：

- 建立 Amazon 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 的 SDK (Boto3)

> **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。

建立包裝 Amazon Glacier 操作的類別。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》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 SDK 使用 Amazon Glacier。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 SDK 取得 Amazon Glacier 封存內容並刪除封存

以下程式碼範例顯示做法：

- 列出 Amazon Glacier 保存庫的任務並取得任務狀態。
- 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 的 SDK (Boto3)

> **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。

建立包裝 Amazon Glacier 操作的類別。

```
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 詳細資訊，請參閱《適用於 Python (Boto3) 的AWS SDK API 參考》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 SDK 使用 Amazon Glacier。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon Glacier 中的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為符合最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模型 將此描述為雲端的安全和雲端內的安全：

- 雲端的安全性 – AWS 負責保護在 中執行 AWS 服務的基礎設施 AWS 雲端。 AWS 也為您提供可安全使用的服務。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 Glacier (Amazon Glacier) 的合規計劃，請參閱AWS 合規計劃範圍內的服務。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。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。

本文件將協助您了解如何在使用 Amazon Glacier 時套用共同責任模型。下列主題說明如何設定 Amazon Glacier 以符合您的安全與合規目標。您也將了解如何使用其他 AWS 服務，協助您監控和保護 Amazon Glacier 資源。

主題

- Amazon Glacier 中的資料保護
- Amazon Glacier 的 Identity and Access Management
- 在 Amazon Glacier 中記錄和監控
- Amazon Glacier 的合規驗證
- Amazon Glacier 中的彈性
- Amazon Glacier 中的基礎設施安全

Amazon Glacier 中的資料保護

Amazon Glacier (Amazon Glacier) 為資料封存和長期備份提供高耐用性的雲端儲存。Amazon Glacier 旨在提供 99.999999999% 的耐用性，並提供全方位的安全性和合規功能，可協助您滿足嚴格的法規要求。Amazon Glacier 會以備援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。為了提高耐用性，Amazon Glacier 會在確認上傳成功之前，同步跨多個AZs存放您的資料。

如需 AWS 全球雲端基礎設施的詳細資訊，請參閱 全球基礎設施。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並僅提供個別使用者、群組或角色完成其任務所需的許可。

如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-2 驗證的密碼編譯模組，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 Glacier 時）和靜態（存放在 AWS 資料中心時）時保護資料。您可以使用 Secure Sockets Layer (SSL) 或用戶端加密來保護直接上傳至 Amazon Glacier 的傳輸中資料。

您也可以透過 Amazon S3 存取 Amazon Glacier。 Amazon S3 Amazon S3 支援 Amazon S3 儲存貯體的生命週期組態，可讓您將物件轉換為 Amazon Glacier 儲存類別以進行封存。透過生命週期政策在 Amazon S3 和 Amazon Glacier 之間傳輸的資料會使用 SSL 加密。

存放在 Amazon Glacier 中的靜態資料會自動使用 256 位元進階加密標準 (AES-256) 搭配 維護的金鑰進行伺服器端加密 AWS。如果您偏好管理自己的金鑰，也可以在將資料儲存在 Amazon Glacier 之前使用用戶端加密。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，也就是說，Amazon Glacier 會在將資料寫入資料中心時加密資料，並在您存取資料時將其解密。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 Amazon Glacier 中的靜態資料會使用 維護的金鑰，使用 AES-256 自動進行伺服器端加密 AWS。作為額外的保護， 會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

透過網路存取 Amazon Glacier 是透過 AWS 發佈APIs。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 服務和採用 AWS PrivateLink 技術的 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 Amazon Glacier 不支援 VPC 端點，但如果您將 Amazon Glacier 作為與 Amazon S3 整合的儲存層存取，則可以利用 Amazon S3) VPC 端點。

如需 Amazon S3 生命週期組態和將物件轉換為 Amazon 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 Glacier 的 Identity and Access Management

AWS Identity and Access Management (IAM) 是一種 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證 （登入） 和授權 （具有許可），以使用 Amazon Glacier 資源。IAM 是您可以免費使用 AWS 服務 的 。

主題

- 目標對象
- 使用身分驗證
- 使用政策管理存取權
- Amazon Glacier 如何與 IAM 搭配使用
- Amazon Glacier 的身分型政策範例
- Amazon Glacier 的資源型政策範例
- 對 Amazon Glacier 身分和存取進行故障診斷
- API 許可參考

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

- 服務使用者 — 若無法存取某些功能，請向管理員申請所需許可 (請參閱 對 Amazon Glacier 身分和存取進行故障診斷)
- 服務管理員 — 負責設定使用者存取權並提交相關許可請求 (請參閱 Amazon Glacier 如何與 IAM 搭配使用)
- IAM 管理員 — 撰寫政策以管理存取控制 (請參閱 Amazon Glacier 的身分型政策範例)

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

對於程式設計存取， AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的 API 請求的AWS 第 4 版簽署程序。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分可完整存取所有 AWS 服務 和 資源。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，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

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取 。

聯合身分是來自您的企業目錄、Web 身分提供者的使用者，或使用來自身分來源的 AWS 服務 憑證存取 Directory Service 。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，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https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。

IAM 群組會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

IAM 角色

IAM 角色https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html的身分具有特定許可權，其可以提供臨時憑證。您可以透過從使用者切換到 IAM 角色（主控台）或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS ， 會評估這些政策。大多數政策會以 JSON 文件 AWS 形式存放在 中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的 JSON 政策概觀。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的在受管政策與內嵌政策之間選擇。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策與 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。您必須在資源型政策中指定主體。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《 IAM 使用者指南》中的 IAM 實體許可界限。
- 服務控制政策 (SCP) — 為 AWS Organizations中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的資源控制政策 (RCP)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參《IAM 使用者指南》中的工作階段政策。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多個政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

Amazon Glacier 如何與 IAM 搭配使用

在您使用 IAM 管理 Amazon Glacier 的存取權之前，請先了解哪些 IAM 功能可與 Amazon Glacier 搭配使用。

您可以搭配 Amazon Glacier 使用的 IAM 功能

IAM 功能	Amazon Glacier 支援
身分型政策	是
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

IAM 功能	Amazon Glacier 支援
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

若要全面了解 Amazon Glacier 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱《AWS IAM 使用者指南》中的與 IAM 搭配使用的 服務。

Amazon Glacier 的身分型政策

支援身分型政策：是

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。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

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。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。

Amazon Glacier 的身分型政策範例

若要檢視 Amazon Glacier 身分型政策的範例，請參閱 Amazon Glacier 的身分型政策範例。

Amazon Glacier 內的資源型政策

支援資源型政策：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。委託人可以包括帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

Amazon Glacier 服務僅支援一種名為保存庫政策的資源型政策類型，該政策連接到保存庫。此政策定義哪些主體可以對保存庫執行動作。

Amazon 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。

Amazon Glacier 中的資源型政策範例

若要檢視 Amazon Glacier 資源型政策的範例，請參閱 Amazon Glacier 的資源型政策範例。

Amazon 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。政策會使用動作來授予執行相關聯動作的許可。

若要查看 Amazon Glacier 動作的清單，請參閱《服務授權參考》中的 Amazon Glacier 定義的動作。

Amazon Glacier 中的政策動作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 Amazon Glacier 身分型政策的範例，請參閱 Amazon Glacier 的身分型政策範例。

Amazon 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。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。若動作不支援資源層級許可，使用萬用字元 (*) 表示該陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 Amazon Glacier 資源類型及其 ARNs的清單，請參閱《服務授權參考》中的 Amazon Glacier 定義的資源。若要了解您可以指定每個資源的 ARN 的動作，請參閱 Amazon Glacier 定義的動作。

在 Amazon Glacier 中，主要資源是保存庫。 Amazon 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 AWS 區域中的特定保存庫或一組保存庫。Amazon Glacier 不支援封存層級權限。

對於所有 Amazon 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。

Amazon Glacier 提供一組操作，以使用 Amazon Glacier 資源。如需有關可用操作的詳細資訊，請參閱 Amazon Glacier 的 API 參考。

有些 Amazon 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","
]
```

Amazon 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` 元素會根據定義的條件，指定陳述式的執行時機。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。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的AWS 全域條件內容索引鍵。

若要查看 Amazon Glacier 條件索引鍵的清單，請參閱《服務授權參考》中的 Amazon Glacier 的條件索引鍵。若要了解您可以使用條件金鑰的動作和資源，請參閱 Amazon 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。

Amazon Glacier 中的 ACLs

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 Amazon 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，依據稱為標籤的屬性來定義許可。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在委託人的標籤符合資源上的標籤時允許操作。

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》中的使用 ABAC 授權定義許可。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)。

搭配 Amazon Glacier 使用臨時登入資料

支援臨時憑證：是

臨時登入資料提供 AWS 資源的短期存取權，當您使用聯合或切換角色時，會自動建立。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的臨時安全憑證與可與 IAM 搭配運作的AWS 服務。

Amazon Glacier 的跨服務主體許可

支援轉寄存取工作階段 (FAS)：否

轉送存取工作階段 (FAS) 使用呼叫 的委託人許可 AWS 服務，結合 AWS 服務 請求向下游服務提出請求。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。

Amazon 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。

> **Warning**
>
> 變更服務角色的許可可能會中斷 Amazon Glacier 功能。只有在 Amazon Glacier 提供指引時，才能編輯服務角色。

Amazon 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 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。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。在資料表中尋找服務，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 Glacier 的身分型政策範例

根據預設，使用者和角色沒有建立或修改 Amazon Glacier 資源的許可。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。

如需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 (主控台)。

如需 Amazon Glacier 定義之動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱《服務授權參考》中的 Amazon Glacier 的動作、資源和條件索引鍵。

以下是範例政策，使用識別us-west-2 AWS 區域中所有保存庫的 Amazon Resource Name (ARN)，授予 資源上三個 Amazon Glacier 保存庫相關動作 (glacier:CreateVaultglacier:DescribeVault和) 的許可。 glacier:ListVaultsARNs可唯一識別 AWS 資源。如需與 Amazon Glacier 搭配使用ARNs 的詳細資訊，請參閱 Amazon Glacier 的政策資源。

JSON

```
{
   "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 Amazon 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 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。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。如果透過特定 等使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 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，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的透過 MFA 的安全 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 Amazon Glacier 主控台

若要存取 Amazon Glacier 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視 中 Amazon Glacier 資源的詳細資訊 AWS 帳戶。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。

對於僅呼叫 AWS CLI 或 AWS API 的使用者，您不需要允許最低主控台許可。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。

Amazon Glacier 主控台提供整合式環境，供您建立和管理 Amazon Glacier 保存庫。您建立的 IAM 身分至少必須獲得許可，才能讓 `glacier:ListVaults`動作檢視 Amazon Glacier 主控台，如下列範例所示。

JSON

```
{
      "Version":"2012-10-17",
      "Statement": [
         {
            "Action": [
            "glacier:ListVaults"
            ],
            "Effect": "Allow",
            "Resource": "*"
         }
      ]
   }
```

AWS 提供由 建立和管理的獨立 IAM 政策，以解決許多常見的使用案例 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 受管政策是 Amazon Glacier 特有的，您可以連接到您帳戶中的使用者：

- AmazonGlacierReadOnlyAccess – 透過 授予對 Amazon Glacier 的唯讀存取權 AWS 管理主控台。
- AmazonGlacierFullAccess – 透過 授予 Amazon Glacier 的完整存取權 AWS 管理主控台。

您也可以建立自己的自訂 IAM 政策，以允許 Amazon Glacier API 動作和資源的許可。您可以將這些自訂政策連接到您為 Amazon Glacier 保存庫建立的自訂 IAM 角色。

下一節討論的兩個 Amazon Glacier AWS 受管政策都會授予 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。此政策包含在主控台或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
    "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

在本節中，您可以找到授予各種 Amazon Glacier 動作許可的使用者政策範例。當您使用 Amazon Glacier REST API、Amazon SDKs、 AWS CLI或 Amazon Glacier 管理主控台時，這些政策會運作。

> **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 Glacier 主控台取得保存庫 ARN，或以程式設計方式呼叫 描述保存庫 (GET 保存庫)或 「列出保存庫」(GET 保存庫) API 動作。

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 Glacier 中設定保存庫通知。此政策也會授予許可，以列出 AWS 區域中的保存庫，並取得特定的保存庫描述。

> **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。

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。

範例 4：允許使用者授與特定保存庫的完整許可

下列範例政策授予名為 的保存庫上所有 Amazon Glacier 動作的許可`examplevault`。

Amazon Glacier 的資源型政策範例

Amazon Glacier 保存庫可以有一個與之相關聯的保存庫存取政策和一個保存庫鎖定政策。Amazon Glacier 保存庫存取政策是以資源為基礎的政策，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 Glacier 保存庫存取政策是以資源為基礎的政策，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。Amazon Glacier 也在每個保存庫上支援保存庫鎖定政策，這些政策在您鎖定後就無法變更。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 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 Glacier 動作的跨帳戶許可

下列範例政策 AWS 帳戶 會針對名為 的保存庫上的一組 Amazon Glacier 操作，將跨帳戶許可授予兩個 `examplevault`。

> **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。

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) 來保護 Amazon 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。

範例政策會授予 AWS 帳戶 具有臨時登入資料的 許可，以從名為 examplevault 的保存庫中刪除封存，前提是請求已使用 MFA 裝置進行身分驗證。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。

保存庫鎖定政策

Amazon Glacier (Amazon Glacier) 保存庫可以連接一個資源型保存庫存取政策和一個保存庫鎖定政策。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 Glacier 為您提供一組 API 操作來管理保存庫鎖定政策，請參閱 使用 Amazon Glacier API 鎖定保存庫。

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 Amazon Glacier API AWS CLI、Amazon SDKs 或 Amazon Glacier 主控台來建立和管理保存庫鎖定政策。如需保存庫資源型政策允許的 Amazon 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。此政策使用 Amazon Glacier 特定條件金鑰`ArchiveAgeInDays`來強制執行一年保留要求。

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。

對 Amazon Glacier 身分和存取進行故障診斷

使用以下資訊來協助您診斷和修正使用 Amazon Glacier 和 IAM 時可能遇到的常見問題。

主題

- 我無權在 Amazon Glacier 中執行動作
- 我未獲得執行 iam:PassRole 的授權
- 我想要允許 以外的人員 AWS 帳戶 存取我的 Amazon Glacier 資源

我無權在 Amazon 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。

我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，告知您無權執行 iam:PassRole動作，您的政策必須更新，以允許您將角色傳遞給 Amazon Glacier。

有些 AWS 服務 可讓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 的 IAM marymajor 使用者嘗試使用主控台在 Amazon Glacier 中執行動作時，會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞給服務的許可。

```
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。

我想要允許 以外的人員 AWS 帳戶 存取我的 Amazon 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 Amazon Glacier 是否支援這些功能，請參閱 Amazon Glacier 如何與 IAM 搭配使用。
- 若要了解如何在您擁有 AWS 帳戶 的 資源之間提供存取權，請參閱《IAM 使用者指南》中的在您擁有 AWS 帳戶 的另一個 IAM 使用者中提供存取權。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《IAM 使用者指南》中的將存取權提供給第三方 AWS 帳戶 擁有。

- 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)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，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 Glacier 如何與 IAM 搭配使用 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。包含每個 Amazon Glacier API 操作、您可以授予執行動作許可的對應動作，以及您可以授予許可 AWS 的資源。

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 Amazon 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Amazon 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。

Amazon Glacier API 和動作的必要許可

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/*`

Amazon Glacier 條件金鑰：

「中止保存庫鎖定」(DELETE 鎖定政策)

所需許可 (API 動作)： glacier:AbortVaultLock

資源：

Amazon Glacier 條件金鑰：

新增標籤至保存庫 (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/*

Amazon Glacier 條件金鑰： 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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

完成保存庫鎖定 (POST lockId)

所需許可 (API 動作)：glacier:CompleteVaultLock

資源：

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

建立保存庫 (PUT 保存庫) *

所需許可 (API 動作)：glacier:CreateVault

資源：

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

[刪除保存庫存取政策 (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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

[刪除保存庫通知 (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/*

Amazon Glacier 條件金鑰： 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/*

Amazon Glacier 條件金鑰：

描述保存庫 (GET 保存庫)

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/*

Amazon Glacier 條件金鑰：

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

Amazon Glacier 條件金鑰：

「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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：

取得保存庫鎖定 (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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

啟動保存庫鎖定 (POST 鎖定政策)

所需許可 (API 動作)：glacier:InitiateVaultLock

資源：

Amazon Glacier 條件金鑰： 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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：

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/*

Amazon Glacier 條件金鑰：

「列出保存庫」(GET 保存庫)

所需許可 (API 動作)：glacier:ListVaults

資源：

Amazon Glacier 條件金鑰：

從保存庫移除標籤 (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/*

Amazon Glacier 條件金鑰： 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

Amazon Glacier 條件金鑰：

設定保存庫存取政策 (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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

設定保存庫通知組態 (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/*

Amazon Glacier 條件金鑰： 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/*

Amazon Glacier 條件金鑰： 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/*

Amazon Glacier 條件金鑰： glacier:ResourceTag/*TagKey*

在 Amazon Glacier 中記錄和監控

監控是維護 Amazon Glacier (Amazon Glacier) 和 AWS 解決方案可靠性、可用性和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生故障時更輕鬆地識別和偵錯故障來源。AWS 提供下列工具來監控 Amazon Glacier 資源並回應潛在事件：

Amazon CloudWatch 警示

透過 Amazon S3 使用 Amazon Glacier 時，您可以使用 Amazon CloudWatch 警示，在您指定的期間內監看單一指標。 Amazon S3 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調用動作。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 提供使用者、角色或 AWS 服務在 Amazon Glacier 中所採取動作的記錄。CloudTrail 會將 Amazon Glacier 的所有 API 呼叫擷取為事件，包括來自 Amazon Glacier 主控台的呼叫，以及來自對 Amazon Glacier APIs 的程式碼呼叫。如需詳細資訊，請參閱使用 記錄 Amazon Glacier API 呼叫 AWS CloudTrail。

AWS Trusted Advisor

Trusted Advisor 利用從為數十萬 AWS 客戶提供服務的最佳實務。 會 Trusted Advisor 檢查您的 AWS 環境，然後在有機會節省成本、改善系統可用性和效能，或協助填補安全漏洞時提出建議。所有 AWS 客戶都可以存取五個 Trusted Advisor 檢查。擁有商業或企業支援計劃的客戶可以檢視所有 Trusted Advisor 檢查。

如需詳細資訊，請參閱《支援 使用者指南》中的 AWS Trusted Advisor。

Amazon Glacier 的合規驗證

Amazon Glacier (Amazon Glacier) 的安全與合規由第三方稽核人員評估，做為多個 AWS 合規計畫的一部分，包括下列項目：

- 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 在合規計劃範圍內 AWS 的 Services in 提供特定合規計劃範圍內的經常更新AWS 服務清單。

可使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱AWS Artifact 《 使用者指南》中的在 AWS 成品中下載報告。

如需 AWS 合規計劃的詳細資訊，請參閱AWS 合規計劃。

您使用 Amazon Glacier 時的合規責任取決於資料的敏感度、組織的合規目標，以及適用的法律和法規。如果您使用 Amazon Glacier 符合 HIPAA、PCI 或 FedRAMP 等標準， AWS 會提供資源來協助：

- Amazon Glacier 保存庫鎖定 可讓您使用保存庫鎖定政策，輕鬆部署和強制執行個別 Amazon Glacier 保存庫的合規控制。您可以在保存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。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討論部署以安全與合規為重心的基準環境的架構考量和步驟 AWS。
- HIPAA Security and Compliance 架構概述了公司如何使用 AWS 來協助他們滿足 HIPAA 要求。
- AWS Well-Architected Tool (AWS WA Tool) 是一種雲端服務，提供一致的程序，讓您使用 AWS 最佳實務來檢閱和測量架構。 AWS WA 工具提供建議，讓您的工作負載更可靠、安全、有效率且符合成本效益。
- AWS 合規資源提供數種不同的工作手冊和指南，可能適用於您的產業和據點。
- 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 可讓您全面檢視 內的安全狀態， AWS 並協助您檢查是否符合安全產業標準和最佳實務。

Amazon Glacier 中的彈性

AWS 全球基礎設施是以區域和可用區域為基礎建置。 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。Amazon Glacier 會以備援方式將資料存放在跨至少三個可用區域的多個裝置中。為了提高耐用性，Amazon Glacier 會在確認上傳成功之前，同步跨多個AZs存放您的資料。

如需 AWS 區域和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

Amazon Glacier 中的基礎設施安全

作為受管服務，Amazon Glacier (Amazon Glacier) 受到 Amazon Web Services：安全程序概觀中所述 AWS 的全球網路安全程序的保護。

透過網路存取 Amazon Glacier 是透過 AWS 發佈APIs。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 服務和採用 AWS PrivateLink 技術的 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 Amazon Glacier 不支援 VPC 端點，但如果您將 Amazon Glacier 作為與 Amazon S3 整合的儲存層存取，則可以利用 Amazon S3 VPC 端點。 Amazon Glacier

如需 Amazon S3 生命週期組態和將物件轉換為 Amazon 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 Glacier 資料擷取政策

使用 Amazon Glacier 資料擷取政策，您可以輕鬆地設定資料擷取配額，並管理每個 AWS 帳戶 中跨的資料擷取活動 AWS 區域。如需 Amazon Glacier 資料擷取費用的詳細資訊，請參閱 Amazon Glacier 定價。

> **Important**
>
> 資料擷取政策僅適用於標準擷取和管理直接向 Amazon Glacier 提出的擷取請求。
> 如需 Amazon Glacier 儲存類別的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的用於封存物件和轉換物件的儲存類別。

主題

選擇 Amazon Glacier 資料擷取政策

您可以選擇三種類型的 Amazon Glacier 資料擷取政策：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。

透過使用僅限免費方案政策，您可以將擷取保持在每日 AWS 免費方案限額內，而不會產生任何資料擷取費用。如果您想要擷取的資料超過 AWS 免費方案額度，您可以使用最高擷取率政策來設定bytes-per-hour擷取率配額。最高擷取率政策可確保 AWS 區域 中所有擷取工作的最高擷取率不超過所設定的每小時位元組配額。

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。如果您使用僅限免費方案政策，Amazon Glacier 會同步拒絕超過免費 AWS 方案限額的擷取請求。如果您使用最高擷取率政策，Amazon Glacier 會拒絕導致進行中任務的最高擷取率超過政策設定的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使用標準擷取從 Amazon Glacier 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 設定一個資料擷取政策 AWS 區域，這將管理您帳戶 AWS 區域 下 中的所有資料擷取活動。資料擷取政策專屬於特定 ， AWS 區域 因為資料擷取成本會有所不同 AWS 區域。如需詳細資訊，請參閱 Amazon 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限免費方案，以確保您的擷取一律保持在 AWS 免費方案限額內，因此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高擷取率，透過指定具有每小時最大位元組的資料擷取配額來控制最高擷取率。將資料擷取政策設定為最高擷取率時，如果會導致進行中工作的最高擷取率超過政策指定的每小時位元組配額，則將拒絕新的擷取請求。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。

將資料擷取政策設定為最高擷取率政策可能會影響您一天可使用的 AWS 免費方案限額。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 免費方案額度，您可以先將政策設定為僅限免費方案，然後視需要切換到最高擷取率政策。如需如何計算擷取額度的詳細資訊，請參閱 Amazon Glacier FAQs。

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 Amazon Glacier 主控台設定資料擷取政策

使用 Amazon Glacier 主控台建立資料擷取政策

1. 登入 AWS 管理主控台 ，並在 https://console.aws.amazon.com/glacier/home：// 開啟 Amazon Glacier 主控台。
2. 在選取區域下， AWS 區域 從下拉式選單中選擇 。您可以為每個 設定資料擷取政策 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。
   - 如果您只選擇 免費方案，則不接受超過 AWS 免費方案的資料擷取請求。
   - 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 Glacier API 設定資料擷取政策

您可以使用 Amazon Glacier REST API 或使用 AWS SDKs 來檢視和設定資料擷取政策。

使用 Amazon Glacier REST API 設定資料擷取政策

您可以使用 Amazon 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 SDKs設定資料擷取政策

AWS 為您提供SDKs，以開發 Amazon Glacier 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。如需詳細資訊，請參閱搭配 Amazon Glacier 使用 AWS SDKs。

標記 Amazon 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 Glacier (Amazon Glacier) 保存庫資源。使用標籤是一種簡單但強大的方法來管理 AWS 資源和組織資料，包括帳單資料。

主題

標記基本概念

您可以使用 Amazon Glacier 主控台、 AWS Command Line Interface (AWS CLI) 或 Amazon Glacier API 來完成下列任務：

- 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 標記您的 Amazon Glacier 保存庫。

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。
- 標籤索引鍵不能以 開頭，`aws:`因為此字首保留給 使用 AWS。 AWS 可以代表您建立以此字首開頭的標籤，但您無法編輯或刪除它們。
- 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和追蹤您的 AWS 成本。當您將標籤套用至任何 AWS 資源時，包括保存庫，AWS 成本分配報告會包含依標籤彙總的用量和成本。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

- 標記您的 Amazon Glacier 保存庫

使用 記錄 Amazon Glacier API 呼叫 AWS CloudTrail

Amazon Glacier (Amazon Glacier) 已與 服務整合 AWS CloudTrail，此服務可提供使用者、角色或 Amazon Glacier 中 AWS 服務所採取動作的記錄。CloudTrail 會將 Amazon Glacier 的所有 API 呼叫擷取為事件，包括來自 Amazon Glacier 主控台的呼叫，以及來自對 Amazon Glacier APIs 的程式碼呼叫。如果您建立線索，則可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 Amazon Glacier 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台中的事件歷史記錄檢視最新事件。使用 CloudTrail 收集的資訊，您可以判斷向 Amazon Glacier 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，請參閱「AWS CloudTrail 使用者指南」。

CloudTrail 中的 Amazon Glacier 資訊

當您建立帳戶 AWS 帳戶 時，您的 上會啟用 CloudTrail。當活動在 Amazon Glacier 中發生時，該活動會與事件歷史記錄中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中。您可以在 中檢視、搜尋和下載最近的事件 AWS 帳戶。如需詳細資訊，請參閱《使用 CloudTrail 事件歷史記錄檢視事件》https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html。

若要持續記錄 中的事件 AWS 帳戶，包括 Amazon Glacier 的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台中建立線索時，線索會套用至所有 AWS 區域。線索會記錄 AWS 分割區中所有 AWS 區域的事件，並將日誌檔案傳送到您指定的 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 Amazon Glacier 動作，並記錄在 中Amazon Glacier 的 API 參考。例如，對 建立保存庫 (PUT 保存庫)、刪除文件庫 (DELETE 文件庫) 以及 「列出保存庫」(GET 保存庫) 動作發出的呼叫會在 CloudTrail 日誌檔案中產生項目。

每一筆事件或日誌專案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。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 CloudTrail userIdentity 元素。

了解 Amazon 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專案。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 建立保存庫 (PUT 保存庫)、刪除文件庫 (DELETE 文件庫)、「列出保存庫」(GET 保存庫) 及 描述保存庫 (GET 保存庫) 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 Glacier 的 API 參考

Amazon Glacier 支援一組操作，特別是一組 RESTful API 呼叫，可讓您與服務互動。

您可以使用任何可以傳送 HTTP 請求的程式設計程式庫，將您的 REST 請求傳送至 Amazon Glacier。傳送 REST 請求時，Amazon Glacier 會要求您透過簽署請求來驗證每個請求。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。如需詳細資訊，請參閱簽署請求。

如果發生錯誤，您需要知道 Amazon Glacier 在錯誤回應中傳送什麼，以便處理它。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 適用於 Java 的 AWS SDK 和 .NET 的基本 Amazon Glacier 操作的工作範例。如需詳細資訊，請參閱 搭配 Amazon Glacier 使用 AWS SDKs。

主題

常見請求標題

Amazon Glacier (Amazon Glacier) REST 請求包含包含請求基本資訊的標頭。下表說明所有 Amazon Glacier REST 請求可以使用的標頭。

標頭名稱	說明	必要
Authorization	簽署請求所需的標頭。Amazon Glacier 需要 Signature 第 4 版。如需詳細資訊，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，下列日期/時間Wed, 10 Feb 2017 12:00:00 GMT是與 Amazon 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

標頭名稱	說明	必要
Host	此標頭指定您傳送請求的服務終端節點。值的格式必須是 "glacier.*region*.amazonaws.com "，其中##會取代為 AWS 區域名稱，例如 us-west-2 。 類型：字串	是
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 x-amz-date 適用於 Amazon Glacier。 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

標頭名稱	說明	必要
x-amz-glacier-version	要使用的 Amazon Glacier API 版本。目前版本是 2012-06-01 。 類型：字串	是
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 Glacier (Amazon 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，可能不符合保存庫的庫存清單中為封存顯示的日期。 類型：字串
x-amzn-RequestId	Amazon Glacier 建立的值，可唯一識別您的請求。如果 Amazon Glacier 發生問題， AWS 可以使用此值對問題進行故障診斷。建議您記錄這些值。

名稱	描述
	類型：字串
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

Amazon Glacier 要求您透過簽署請求來驗證您傳送的每個請求。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，Amazon Glacier 會使用您用來簽署請求的相同雜湊函數和輸入來重新計算簽章。如果產生的簽章符合請求中的簽章，Amazon Glacier 會處理請求。否則，請求會遭到拒絕。

Amazon 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。使用正式表單是必要的，因為 Amazon Glacier 在重新計算簽章以與您傳送的簽章進行比較時，會使用相同的正式表單。
- 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。登入資料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，此簽署步驟中使用的雜湊函數不是用於上傳資料的 Amazon Glacier APIs 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 建立保存庫 (PUT 保存庫) 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

下列範例會逐步解說為 建立簽章的詳細資訊上傳封存 (POST 封存)，這是 Amazon Glacier 中兩個串流 APIs 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)。
- 內容承載是字串「歡迎使用 Amazon Glacier」。

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。

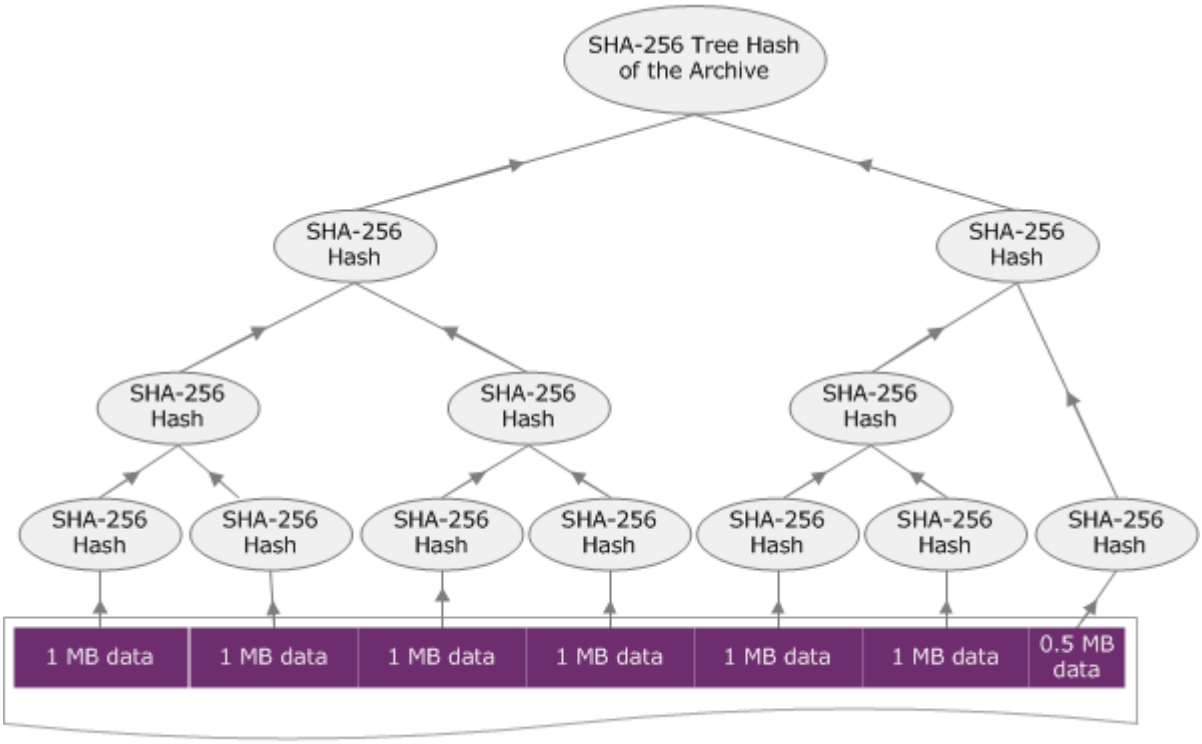
您運算 SHA-256 樹雜湊，如下所示：

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

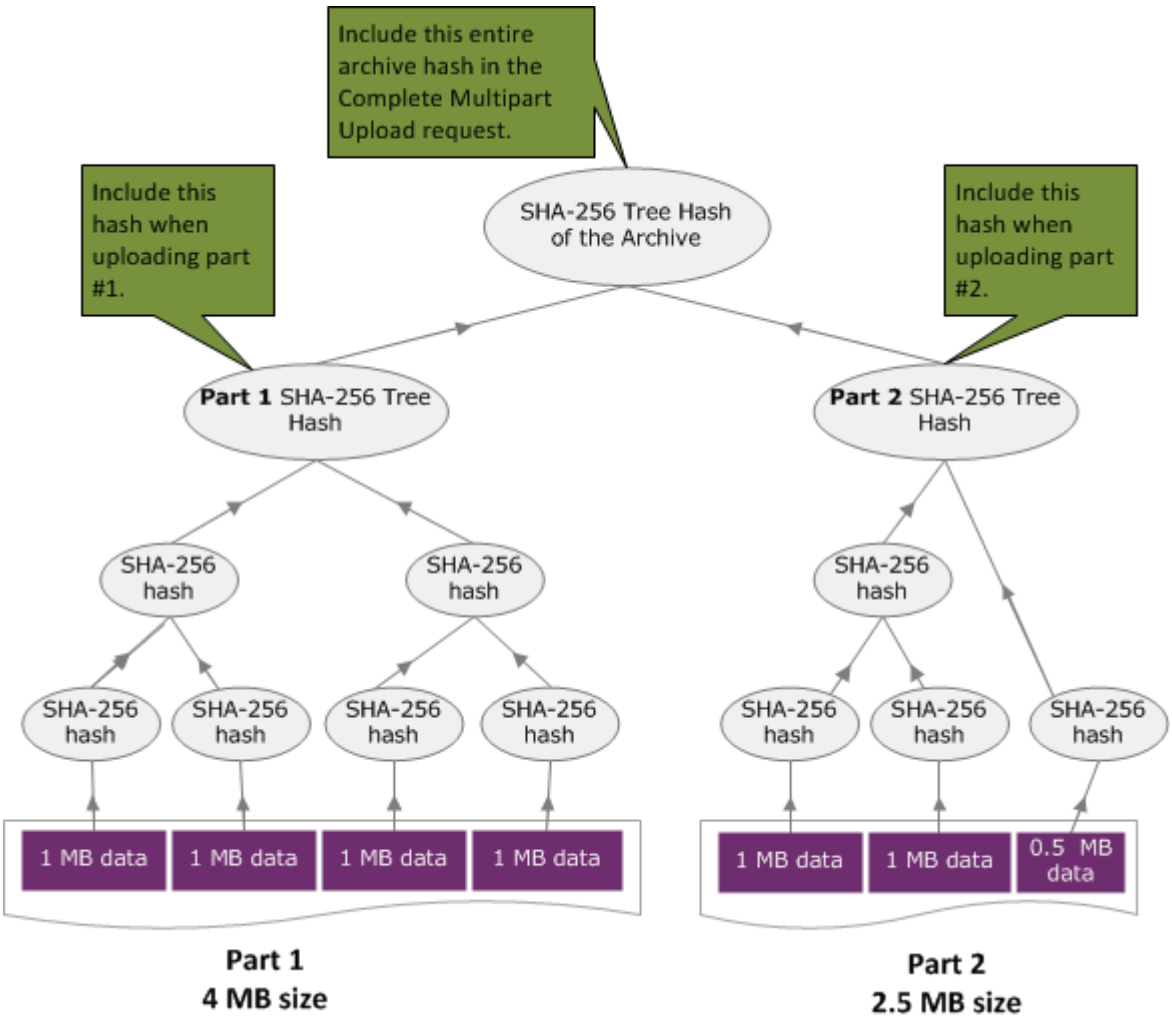
在單一請求中使用上傳封存 API 上傳封存時 (請參閱 上傳封存 (POST 封存)) 請求承載包含整個封存。因此，您必須在 x-amz-sha256-tree-hash 請求標頭中包含整個封存的樹雜湊。假設您想要上傳 6.5 MB 的封存。下圖說明建立封存的 SHA-256 雜湊的程序。您讀取封存並運算每 1 MB 區塊的 SHA-256 雜湊。您也運算剩餘 0.5 MB 資料的雜湊，然後依所述的樹狀結構建置程序。



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 SDK 針對 Amazon Glacier (Amazon Glacier) 進行程式設計，則會為您完成樹雜湊計算，而且您只需要提供檔案參考。

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：

- 符合 MB：當 StartBytes 可以被 1 MB 整除，而 EndBytes 加 1 可以被 1 MB 整除，或等於指定的封存結尾時 (封存位元組大小減 1)，範圍 [StartByte, EndBytes] 是符合 MB (1024* 1024)。如果有指定，啟動任務 API 中使用的範圍需要符合 MB。
- 符合樹雜湊 - 範圍 [StartBytes 、 EndBytes] 是與封存相關的符合樹雜湊，如果且只有在樹雜湊根目錄的建置範圍等同於整個封存的樹雜湊節點。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。

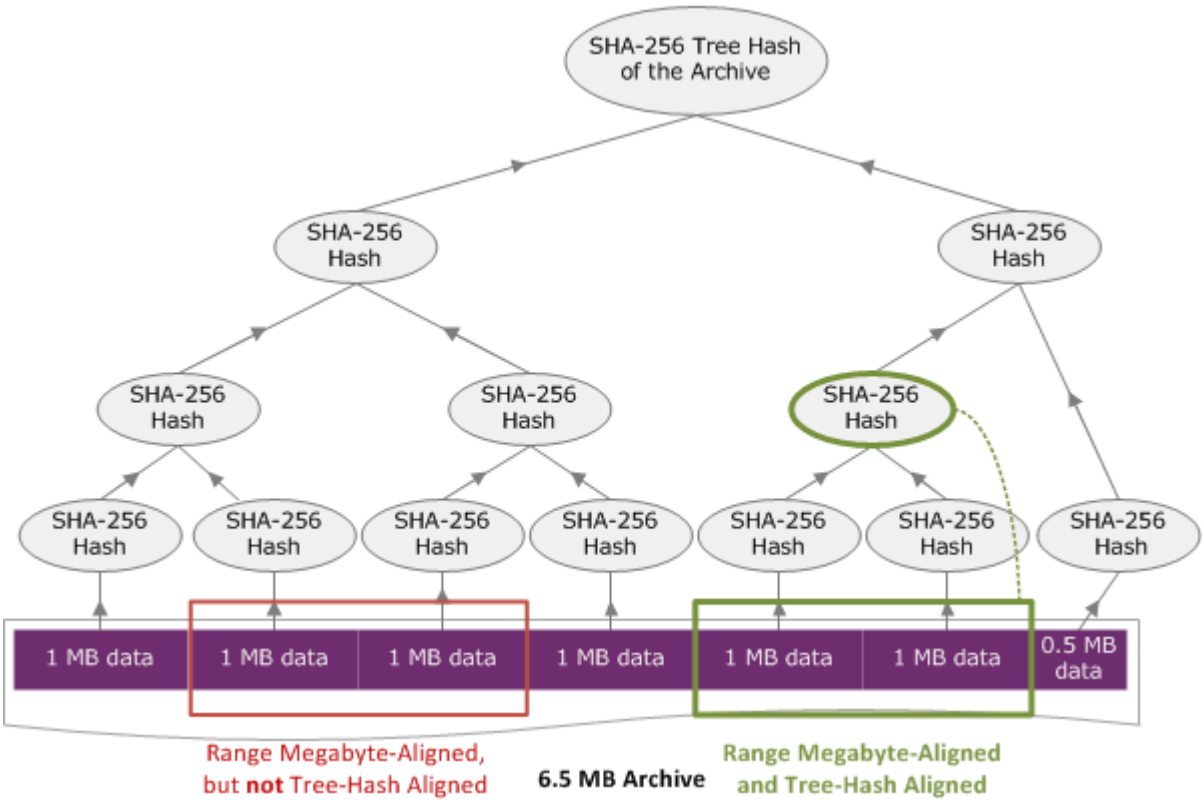
以下案例描述您何時在下載封存資料接收檢查總和值：

- 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：

Code	Description	HTTP 狀態碼	Type
AccessDeniedException	如果嘗試存取 AWS Identity and Access Management (IAM) 政策不允許的資源，或在請求 URI 中使用不正確的 AWS 帳戶 ID，則傳回。如需詳細資訊，請參閱Amazon 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	Server
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：保存庫限制、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

Code	Description	HTTP 狀態碼	Type
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 Amazon Glacier 資料擷取政策。	400 Bad Request	用戶端
ResourceNotFoundException	如果指定的資源不存在，例如保存庫、上傳 ID 或工作 ID，則傳回。	404 Not Found	用戶端
RequestTimeoutException	如果上傳封存，且 Amazon Glacier (Amazon Glacier) 在收到上傳時逾時，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	Server
ThrottlingException	如果您需要降低對 Amazon Glacier 的請求率，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 Amazon Glacier APIs都會傳回相同的例外狀況，但會有不同的例外狀況訊息，協助您疑難排解遇到的特定錯誤。

Amazon Glacier 傳回回應內文中的錯誤資訊。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
```

為了回應，Amazon Glacier 會傳回下列錯誤回應。

```
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，Amazon Glacier 會傳回 4xx和 的狀態碼值5xx。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保存庫，而不是錯誤的工作 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 的保存庫工作，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
```

Amazon Glacier 會傳回InvalidParameterValueException具有適當訊息的 。

```
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"
}
```

保存庫作業

以下是 Amazon Glacier 中可用的保存庫操作。

主題

- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 新增標籤至保存庫 (POST 標籤新增)
- 建立保存庫 (PUT 保存庫)
- 完成保存庫鎖定 (POST lockId)
- 刪除文件庫 (DELETE 文件庫)
- 刪除保存庫存取政策 (DELETE 存取政策)
- 刪除保存庫通知 (DELETE 通知的組態)
- 描述保存庫 (GET 保存庫)
- 取得文件庫存取政策 (GET 存取政策)
- 取得保存庫鎖定 (GET 鎖定政策)
- 取得文件庫通知 (GET 通知的組態)
- 啟動保存庫鎖定 (POST 鎖定政策)
- 列出文件庫的標籤 (GET 標籤)
- 「列出保存庫」(GET 保存庫)
- 從保存庫移除標籤 (POST tags remove)
- 設定保存庫存取政策 (PUT 存取政策)
- 設定保存庫通知組態 (PUT 通知的組態)

「中止保存庫鎖定」(DELETE 鎖定政策)

說明

如果保存庫鎖定不是 Locked 狀態，此作業會停止保存庫鎖定程序。如果在請求此作業時，保存庫鎖定處於 Locked 狀態，則該作業將傳回 AccessDeniedException 錯誤。停止保存庫鎖定程序，會將保存庫鎖定政策從指定的保存庫中移除。

透過呼叫 InProgress，保存庫鎖定進入 啟動保存庫鎖定 (POST 鎖定政策) 狀態。透過呼叫 Locked，保存庫鎖定進入 完成保存庫鎖定 (POST lockId) 狀態。您可以透過呼叫 取得保存庫鎖定 (GET 鎖定政策) 來取得保存庫鎖定的狀態。如需保存庫鎖定程序的詳細資訊，請參閱Amazon Glacier 保存庫鎖定。如需保存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。如果保存庫鎖定處於 InProgress 狀態或者沒有與保存庫關聯的政策，則可以多次成功呼叫此作業。

請求

若要刪除保存庫鎖定政策，請將 HTTP DELETE 請求傳送到保存庫 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

如果成功刪除政策，Amazon Glacier 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何停止保存庫鎖定程序。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 **examplevault** 的保存庫的 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

如果政策已成功刪除，Amazon Glacier 會傳回HTTP 204 No Content回應，如下列範例所示。

```
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

- 完成保存庫鎖定 (POST lockId)
- 取得保存庫鎖定 (GET 鎖定政策)
- 啟動保存庫鎖定 (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

新增標籤至保存庫 (POST 標籤新增)

該作業將指定的標籤新增到保存庫。每個標籤皆包含鍵與值。每個保存庫最多可擁有 50 個標籤。如果您的請求會導致超出保存庫的標籤限制，則該作業會擲出 LimitExceededException 錯誤。

如果標籤已存在於指定金鑰下的保存庫，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 Glacier 資源。

請求語法

若要將標籤新增到保存庫，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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 從保存庫移除標籤 (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)

此標籤新增到保存庫。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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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 以新增至保存庫。

```
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，Amazon Glacier 會傳回 HTTP 204 No Content，如下列範例所示。

```
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)

- 從保存庫移除標籤 (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

建立保存庫 (PUT 保存庫)

說明

此作業將使用指定的名稱建立新的保存庫。 保存庫的名稱在 的 AWS 區域內必須是唯一的 AWS 帳戶。您可以為每個帳戶建立最多 1,000 個保存庫。如需建立更多保存庫的資訊，請前往 Amazon Glacier 產品詳細資訊頁面。

命名保存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)。

此操作是等冪的，您可以多次傳送相同的請求，而且在 Amazon Glacier (Amazon Glacier) 第一次建立指定的保存庫之後不會進一步生效。

請求

語法

若要建立保存庫，請將 HTTP PUT 請求傳送到要建立的保存庫 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**
>
> `AccountId` 值為 AWS 帳戶 ID。此值必須符合與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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作業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	建立的保存庫的相對 URI 路徑。 類型：字串

回應內文

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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 的保存庫。

```
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

Amazon Glacier 會建立保存庫，並在 Location標頭中傳回保存庫的相對 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

- 「列出保存庫」(GET 保存庫)
- 刪除文件庫 (DELETE 文件庫)
- Amazon 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

完成保存庫鎖定 (POST lockId)

說明

此作業透過將保存庫鎖定從 InProgress 狀態轉移為 Locked 狀態來完成保存庫鎖定程序，這會導致保存庫鎖定政策無法變更。透過呼叫 InProgress，保存庫鎖定進入 啟動保存庫鎖定 (POST 鎖定政策) 狀態。您可以透過呼叫 取得保存庫鎖定 (GET 鎖定政策) 來獲得保存庫鎖定的狀態。如需保存庫鎖定程序的詳細資訊，請參閱Amazon Glacier 保存庫鎖定。

此為等冪操作。如果保存庫鎖定處於 Locked 狀態並且提供的鎖定 ID 與最初用於鎖定保存庫的鎖定 ID 符合，則該請求一律成功。

如果保存庫鎖定處於 Locked 狀態時在請求中傳遞了無效的鎖定 ID，則該作業將傳回 AccessDeniedException 錯誤。如果保存庫鎖定處於 InProgress 狀態時在請求中傳遞了無效的鎖定 ID，則該作業將擲出 InvalidParameter 錯誤。

請求

要完成保存庫鎖定程序，請使用有效的鎖定 ID 將HTTP POST 請求傳送到保存庫的 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 啟動保存庫鎖定 (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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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 請求來完成保存庫鎖定程序。

```
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，Amazon Glacier (Amazon Glacier) 會傳回HTTP 204 No Content回應，如下列範例所示。

```
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

- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 取得保存庫鎖定 (GET 鎖定政策)
- 啟動保存庫鎖定 (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)

說明

這個作業會刪除保存庫。Amazon Glacier (Amazon Glacier) 只有在保存庫中沒有根據上次清查的封存，且自上次清查以來沒有寫入保存庫時，才會刪除保存庫。如果未滿足上述任一條件，則保存庫刪除會失敗（即不會移除保存庫），Amazon Glacier 會傳回錯誤。

您可以使用提供保存庫資訊描述保存庫 (GET 保存庫)的操作，包括保存庫中的封存數量；不過，資訊是以 Amazon Glacier 上次產生的保存庫庫存為基礎。

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。如需保存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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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

- 建立保存庫 (PUT 保存庫)
- 「列出保存庫」(GET 保存庫)
- 啟動任務 (POST 任務)
- Amazon 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

刪除保存庫存取政策 (DELETE 存取政策)

Description

這個作業會刪除與指定保存庫關聯的存取政策。操作最終會一致，也就是說，Amazon Glacier (Amazon Glacier) 可能需要一些時間才能完全移除存取政策，而且傳送刪除請求後，您可能會在短時間內看到政策的效果。

此為等冪操作。即使沒有與保存庫關聯的政策，也可以多次叫用刪除。如需保存庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

要求

若要刪除目前保存庫存取政策，請將 HTTP DELETE 請求傳送到保存庫 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Amazon 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何刪除保存庫存取政策。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 **examplevault** 的保存庫的 `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

為了回應，如果成功刪除政策，Amazon Glacier 會傳回 `204 No Content`，如下列範例所示。

```
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)
- 設定保存庫存取政策 (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

刪除保存庫通知 (DELETE 通知的組態)

Description

此操作將刪除為 設定保存庫通知組態 (PUT 通知的組態) 設定的通知組態。操作最終會一致，也就是說，Amazon Glacier (Amazon Glacier) 可能需要一些時間才能完全停用通知，而且傳送刪除請求後，您可能會在短時間內收到一些通知。

要求

要刪除保存庫的通知組態，請向保存庫的 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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何刪除保存庫的通知組態。

範例請求

在這個範例中，將 DELETE 請求傳送到名為 notification-configuration 的保存庫的 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)
- 設定保存庫通知組態 (PUT 通知的組態)
- Amazon 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

描述保存庫 (GET 保存庫)

說明

這個作業會傳回有關保存庫的資訊，包括 Amazon Resource Name (ARN)、保存庫建立日期、保存庫內含的封存數量，以及保存庫中所有封存的大小總計。封存數量及其總大小截至 Amazon Glacier (Amazon Glacier) 產生的最後一個保存庫庫存為止 （ 請參閱 在 Amazon Glacier 中使用保存

庫)。Amazon Glacier 大約每天產生保存庫庫存。這表示如果您新增或移除保存庫的封存，然後立即傳送描述保存庫請求、回應可能不會反映變更。

請求

若要取得有關保存庫的資訊，請傳送 GET 請求至特定保存庫資源的 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

建立保存庫時的 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

Amazon Glacier 完成最後一個保存庫庫存的 UTC 日期。如需有關啟動保存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

根據最後一個保存庫的庫存的保存庫封存數量。如果保存庫尚未執行庫存，此欄位將傳回 Null，例如，您只建立保存庫。

類型：數字

SizeInBytes

保存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。如果保存庫尚未執行庫存，此欄位將傳回 null，例如，您只建立保存庫。

類型：數字

VaultARN

保存庫的 Amazon Resource Name (ARN)。

類型：字串

VaultName

在建立時指定的保存庫名稱。保存庫名稱也包含在保存庫的 ARN 中。

類型：字串

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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` 之保存庫的資訊。

```
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

- 建立保存庫 (PUT 保存庫)
- 「列出保存庫」(GET 保存庫)
- 刪除文件庫 (DELETE 文件庫)
- 啟動任務 (POST 任務)
- Amazon 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)

Description

此作業會擷取保存庫上設定的 `access-policy` 子資源，如需設定此子資源的詳細資訊，請參閱 設定保存庫存取政策 (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。如需保存庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

要求

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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

為了回應，Amazon Glacier (Amazon Glacier) 會在回應內文中傳回 JSON 格式的保存庫存取政策。

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

保存庫存取政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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 請求傳送到保存庫的 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，Amazon Glacier 會在回應內文中以 JSON 字串的形式傳回保存庫存取政策。傳回的 JSON 字串使用 "\" 做為逸出字元，如 設定保存庫存取政策 (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

- 刪除保存庫存取政策 (DELETE 存取政策)
- 設定保存庫存取政策 (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

取得保存庫鎖定 (GET 鎖定政策)

Description

此操作從指定的保存的 `lock-policy` 子資源集中擷取以下屬性：

- 保存庫上設定的保存庫鎖定政策。
- 保存庫鎖定的狀態，它是 InProgess 或 Locked。
- 當鎖定 ID 過期時。鎖定 ID 用於完成保存庫鎖定程序。
- 當保存庫鎖定啟動並進入 InProgress 狀態時。

透過呼叫 InProgress，保存庫鎖定進入 啟動保存庫鎖定 (POST 鎖定政策) 狀態。透過呼叫 Locked，保存庫鎖定進入 完成保存庫鎖定 (POST lockId) 狀態。您可呼叫 「中止保存庫鎖定」 (DELETE 鎖定政策)，停止保存庫鎖定程序。如需保存庫鎖定程序的詳細資訊，請參閱Amazon Glacier 保存庫鎖定。

如果在保存庫沒有設定保存庫鎖定政策，則該作業將傳回 404 Not found 錯誤。如需保存庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

要求

若要傳回目前保存庫鎖定政策和其他屬性，請將 HTTP GET 請求傳送到保存庫的 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

為了回應，Amazon Glacier (Amazon Glacier) 會在回應內文中傳回 JSON 格式的保存庫存取政策。

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

保存庫鎖定政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

州

保存庫鎖定的狀態。

類型：字串

有效值：InProgress|Locked

ExpirationDate

鎖定 ID 過期的 UTC 日期和時間。如果保存庫鎖定處於 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

保存庫鎖定進入 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 Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何取得保存庫鎖定政策。

範例請求

在這個範例中，將 GET 請求傳送到保存庫的 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，Amazon Glacier 會在回應內文中以 JSON 字串的形式傳回保存庫存取政策。傳回的 JSON 字串使用 "\" 做為逸出字元，如 啟動保存庫鎖定 (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

- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 完成保存庫鎖定 (POST lockId)

- 啟動保存庫鎖定 (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)

Description

此操作擷取在文件庫 (請參閱 `notification-configuration` 上設定的 設定保存庫通知組態 (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 Glacier 中設定保存庫通知。

要求

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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

一或多個事件的清單，Amazon Glacier (Amazon Glacier) 會將通知傳送至指定的 Amazon SNS 主題。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 設定保存庫通知組態 (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 Glacier 例外狀況和錯誤訊息的相關資訊，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

- 刪除保存庫通知 (DELETE 通知的組態)
- 設定保存庫通知組態 (PUT 通知的組態)
- Amazon 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

啟動保存庫鎖定 (POST 鎖定政策)

說明

這個作業透過執行以下動作來啟動保存庫鎖定程序：

- 在指定的保存庫上安裝保存庫鎖定政策。
- 將保存庫鎖定的鎖定狀態設定為 `InProgress`。
- 傳回用於完成保存庫鎖定程序的鎖定 ID。

您可以為每個保存庫設定保存庫鎖定政策，並且該政策的大小最多可達 20 KB。如需保存庫鎖定政策的詳細資訊，請參閱保存庫鎖定政策。

保存庫鎖定進入 `InProgress` 狀態後，您必須在 24 小時內完成保存庫鎖定程序。24 小時視窗結束後，鎖定 ID 到期，保存庫自動結束 `InProgress` 狀態，並且從保存庫中刪除保存庫鎖定政策。透過將保存庫鎖定狀態設定為 完成保存庫鎖定 (POST lockId)，呼叫 `Locked` 以完成保存庫鎖定程序。

> **Note**
>
> 保存庫鎖定處於 `Locked` 狀態後，您無法為保存庫啟動新的保存庫鎖定。

您可呼叫「中止保存庫鎖定」(DELETE 鎖定政策)，停止保存庫鎖定程序。您可以透過呼叫 取得保存庫鎖定 (GET 鎖定政策) 來取得保存庫鎖定的狀態。如需保存庫鎖定程序的詳細資訊，請參閱Amazon Glacier 保存庫鎖定。

如果在保存庫鎖定處於 InProgress 狀態時呼叫此作業，該作業將傳回 AccessDeniedException 錯誤。當保存庫鎖定處於 InProgress 狀態時，必須先呼叫 「中止保存庫鎖定」(DELETE 鎖定政策)，才能啟動新的保存庫鎖定政策。

請求

若要啟動保存庫鎖定政策，請將 HTTP POST 請求傳送到保存庫的 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

保存庫鎖定政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

必要：是

回應

如果接受政策，Amazon Glacier (Amazon 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作業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 用於完成保存庫鎖定程序。 類型：字串

回應內文

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

範例請求

以下範例是發送 HTTP PUT 請求至保存庫的`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，Amazon Glacier 會傳回`HTTP 201 Created`回應，如下列範例所示。

```
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

- 「中止保存庫鎖定」(DELETE 鎖定政策)
- 完成保存庫鎖定 (POST lockId)
- 取得保存庫鎖定 (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 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Glacier 例外狀況和錯誤訊息的相關資訊，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，Amazon Glacier (Amazon Glacier) 會傳回保存庫的標籤HTTP 200 OK清單，如下列範例所示。

```
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

- 新增標籤至保存庫 (POST 標籤新增)

- 從保存庫移除標籤 (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

「列出保存庫」(GET 保存庫)

說明

此作業列出了呼叫使用者帳戶擁有的所有保存庫。回應中傳回的清單是 ASCII 依保存庫名稱排序。

在預設情況下，這個操作會傳回每個請求的最多 10 個項目。如果要列出更多的保存庫，則回應內文中的 `marker` 欄位包含保存庫 Amazon Resource Name (ARN)，以便在該列表中繼續使用新的「列出保存庫」請求，否則 `marker` 欄位是 `null`。在下一個 List Vaults 請求中，您將 `marker` 參數設定為 Amazon Glacier (Amazon Glacier) 在先前 List Vaults 請求的回應中傳回的值。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

若要取得保存庫清單，請向GET保存庫 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**
>
> `AccountId` 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

這個作業會傳回以下請求參數。

名稱	描述	必要
limit	所要傳回的保存庫數量上限。預設限制為 10。傳回的保存庫數量可能少於指定的限制，但傳回的保存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 指定保存庫 ARN，之後應該開始保存庫清單。(由 marker 指定的保存庫不包括在傳回的清單中。) 從之前的「列出保存庫」回應中取得 marker 值。只有在您繼續對之前的「列出保存庫」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 (「」) 會傳回從第一個保存庫開始的保存庫清單。 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

建立保存庫的日期，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

上次保存庫庫存的日期，以國際標準時間 (UTC) 為準。如果保存庫尚未執行庫存，此欄位可以為 null，例如，您剛剛建立保存庫。如需有關啟動保存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「列出保存庫」請求中使用 marker 來取得清單中的更多保存庫。如果沒有更多的保存庫，則此值為 null。

類型：字串

NumberOfArchives

截至上次庫存日期的保存庫中的封存數量。

類型：數字

SizeInBytes

保存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

保存庫的 Amazon Resource Name (ARN)。

類型：字串

VaultList

物件陣列，每個物件提供保存庫的說明。

類型：陣列

VaultName

此保存庫名稱。

類型：字串

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

範例：列出所有保存庫

下列範例列出保存庫。由於在請求中未指定 `marker` 和d `limit` 參數，因此傳回最多 10 個保存庫。

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 表明沒有更多的保存庫列出。

```
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：部分保存庫清單

以下範例傳回從 marker 指定的保存庫開始的兩個保存庫。

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

清單中傳回兩個保存庫。Marker 包含保存庫 ARN 以在另一個「列出保存庫」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

- 建立保存庫 (PUT 保存庫)
- 刪除文件庫 (DELETE 文件庫)
- 啟動任務 (POST 任務)
- Amazon 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

從保存庫移除標籤 (POST tags remove)

此作業會從附加到保存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 Glacier 資源。

此為等冪操作。作業將會成功，即使沒有標籤附加到保存庫。

請求語法

若要從保存庫移除標籤，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**
>
> `AccountId` 值為 AWS 帳戶 ID。此值必須符合與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 新增標籤至保存庫 (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保存庫移除。

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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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，Amazon Glacier (Amazon Glacier) 會傳回 HTTP 204 No Content，如下列範例所示。

```
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

- 新增標籤至保存庫 (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

設定保存庫存取政策 (PUT 存取政策)

Description

這個作業將設定保存庫的存取政策，並會覆寫現有的政策。要設定保存庫存取政策，傳送 PUT 請求到保存庫的 `access-policy` 子資源。您可以為每個保存庫設定存取政策，並且該政策的大小最多可達 20 KB。如需保存庫存取政策的詳細資訊，請參閱「保存庫存取政策」。

要求

語法

若要設定保存庫存取政策，請將 HTTP PUT 請求傳送到保存庫的 `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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

保存庫存取政策做為 JSON 字串，使用 "\" 做為逸出字元。

類型：字串

必要：是

回應

為了回應，Amazon Glacier 會在接受政策204 No Content時傳回 。

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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

範例請求

以下範例是發送 HTTP PUT 請求至保存庫的`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，Amazon Glacier (Amazon Glacier) 會傳回 `HTTP 204 No Content`，如下列範例所示。

```
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

- 刪除保存庫存取政策 (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

設定保存庫通知組態 (PUT 通知的組態)

Description

擷取封存和保存庫庫存是 Amazon Glacier (Amazon Glacier) 中的非同步操作，您必須先啟動任務並等待任務完成，才能下載任務輸出。您可以設定保存庫，以在這些工作完成時，將訊息發佈到 Amazon Simple Notification Service (Amazon SNS) 主題。可以使用此作業在保存庫上設定通知組態。如需詳細資訊，請參閱在 Amazon Glacier 中設定保存庫通知。

要設定保存庫通知，傳送 PUT 請求到保存庫的 `notification-configuration` 子資源。通知組態特定於保存庫；因此，也被稱為保存庫子資源。請求應包含提供 Amazon Simple Notification Service (Amazon SNS) 主題的 JSON 文件，以及您希望 Amazon Glacier 將通知傳送至主題的事件。

可以設定保存庫以發佈以下保存庫事件的通知：

- **`ArchiveRetrievalCompleted`**：當為封存擷取啟動的工作完成時，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工作完成時，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 主題必須授予對該保存庫的許可，才能將通知發布到該主題。

要求

要在保存庫上設定通知組態，請將 PUT 請求傳送到保存庫的 `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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Amazon Glacier 傳送通知的一或多個事件陣列。

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

為了回應，`204 No Content`如果接受通知組態，Amazon Glacier (Amazon Glacier) 會傳回 。

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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何設定保存庫通知。

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)
- 刪除保存庫通知 (DELETE 通知的組態)
- Amazon 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

以下是可用於 Amazon Glacier 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)

說明

此作業會從保存庫刪除封存。您可以一次從保存庫刪除一個封存。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保存庫下載保存庫庫存來取得封存 ID。如需下載保存庫庫存的詳細資訊，請參閱在 Amazon Glacier 中下載保存庫庫存。

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 Glacier (Amazon Glacier) 收到刪除封存請求時主動準備資料以供下載，則封存擷取操作可能會失敗。
- 如果封存擷取任務在 Amazon Glacier 收到刪除封存請求時已成功準備要下載的封存，您將可以下載輸出。

如需封存擷取的詳細資訊，請參閱 在 Amazon 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

以下範例示範如何從稱為 examplevault 之保存庫刪除封存。

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，Amazon Glacier 會回應 204 No Content ，指出已刪除封存。

```
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 Glacier 的 Identity and Access Management

上傳封存 (POST 封存)

說明

此作業將封存新增到保存庫。如需成功上傳，您的資料要長期保留。為了回應，Amazon Glacier (Amazon Glacier) 會在回應的 x-amz-archive-id標頭中傳回封存 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 時不需要 AWS CLI。

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保存庫庫存時，Amazon Glacier 會傳回封存描述。Amazon Glacier 不會以任何方式解譯描述。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。

除了選用的封存描述之外，Amazon Glacier 不支援封存的任何其他中繼資料。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。如需詳細資訊，請參閱在 Amazon 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保存庫的 `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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

除了所有作業通用的請求標頭之外，此作業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。當您擷取保存庫庫存 (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 Amazon Glacier 計算承載的不同檢查總和，則會拒絕請求。 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

為了回應，Amazon Glacier 會永久存放封存，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作業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	Amazon Glacier 計算的封存檢查總和。 類型：字串

回應內文

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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

以下成功回應的`Location`標頭可讓您取得 Amazon 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 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 Glacier 的 Identity and Access Management

分段上傳操作

以下是可用於 Amazon Glacier 的分段上傳操作。

主題

中止分段上傳 (DELETE uploadID)

Description

此分段上傳作業命令會停止由上傳 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)。

要求

若要停止分段上傳，請將 HTTP DELETE 請求傳送至保存庫 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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 Glacier 的 Identity and Access Management

完成分段上傳 (POST uploadID)

說明

您呼叫此分段上傳操作，以通知 Amazon Glacier (Amazon Glacier) 所有封存組件都已上傳，Amazon Glacier 現在可從上傳的組件組合封存。

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)。

將封存合併並儲存到保存庫之後，Amazon Glacier 會傳回新建立的封存資源的封存 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，Amazon Glacier 也會建構組合封存的 SHA256 樹雜湊。如果值相符，Amazon Glacier 會將封存儲存至保存庫；否則會傳回錯誤，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，Amazon Glacier 也會檢查是否有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。組合最終封存時，Amazon Glacier 會檢查是否有任何遺失的內容範圍，以及是否有任何遺失的內容範圍，Amazon Glacier 會傳回錯誤，且完成分段上傳操作會失敗。

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 請求傳送至 Amazon Glacier 為回應啟動分段上傳請求而建立的上傳 ID 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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

除了所有作業通用的請求標頭之外，此作業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。如果您在請求中指定的值不符合 Amazon Glacier 計算的最終組合封存的 SHA256 樹雜湊，Amazon Glacier 會傳回錯誤，且請求會失敗。 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 Glacier (Amazon Glacier) 會建立整個封存的 SHA256 樹雜湊。如果值符合您在請求中指定之整個封存的 SHA256 樹雜湊，Amazon Glacier 會將封存新增至保存庫。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，Amazon Glacier 將傳回錯誤，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，最終 Amazon 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作業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 Amazon Glacier 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

此作業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

下列範例回應顯示 Amazon Glacier 已成功從您上傳的部分建立封存。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 Glacier 的 Identity and Access Management

啟動分段上傳 (POST 分段 - 上傳)

Description

這個作業會啟動分段上傳 (請參閱上傳分段中的大型封存 (分段上傳))。Amazon Glacier (Amazon Glacier) 會建立分段上傳資源，並在回應中傳回其 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**
>
> 當您開始分段上傳時，不需要知道封存的大小，因為 Amazon Glacier 不需要您指定整體封存大小。

完成分段上傳後，Amazon Glacier 會移除 ID 參考的分段上傳資源。如果您取消分段上傳，或者如果 24 小時內沒有活動，Amazon Glacier 也會移除分段上傳資源。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。

要求

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保存庫的 `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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

除了所有作業通用的請求標頭之外，此作業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。當您擷取保存庫庫存 (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。	否

名稱	描述	必要
	類型：字串 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，Amazon 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作業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 Amazon 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

名稱	描述
x-amz-multipart-upload-id	分段上傳的 ID。此值也包含在 Location 標頭中。 類型：字串

回應內文

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，請參閱 錯誤回應。

範例

範例請求

以下範例透過名為 POST 的保存庫的 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

Amazon Glacier 會建立分段上傳資源，並將其新增至保存庫的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 Glacier 的 Identity and Access Management

清單部分 (GET uploadID)

Description

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。Amazon Glacier 會傳回按您在每個部分上傳中指定的範圍排序的部分清單。如果您在完成分段上傳後傳送列出組件請求，Amazon Glacier (Amazon 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，若要擷取您傳送的另一組列出組件請求，並將`marker`請求參數設定為 Amazon Glacier 傳回的標記值，以回應先前的列出組件請求。

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。

要求

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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

Amazon Glacier 為組件計算的 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 Glacier 例外狀況和錯誤訊息的相關資訊，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

在回應中，Amazon Glacier 會傳回與指定分段上傳 ID 相關聯的上傳組件清單。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

在回應中，Amazon Glacier 會傳回與指定的進行中分段上傳 ID 相關聯的上傳組件清單。

```
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 Glacier 的 Identity and Access Management

列出分段上傳 (GET 分段 - 上傳)

Description

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，請傳送另一個「列出分段上傳」請求，並將 `marker` 請求參數設定為 Amazon Glacier (Amazon Glacier) 傳回的標記值，以回應先前的「列出分段上傳」請求。

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)。

要求

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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Glacier 例外狀況和錯誤訊息的相關資訊，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，Amazon Glacier 會傳回指定保存庫的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 Glacier (Amazon Glacier) 會傳回指定保存庫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 Glacier 的 Identity and Access Management

分段上傳 (PUT uploadID)

說明

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)。

如果以下任何條件成立，Amazon Glacier (Amazon Glacier) 會拒絕您的上傳組件請求：

- 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。收到組件資料時，Amazon Glacier 也會計算 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。Amazon Glacier 使用上傳 ID 將分段上傳與特定分段上傳建立關聯。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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

除了所有作業通用的請求標頭之外，此作業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。Amazon Glacier 會使用此資訊以適當的順序組合封存。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

成功上傳組件後，Amazon Glacier 會傳回`204 No Content`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作業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	Amazon Glacier 為上傳部分計算的 SHA256 樹雜湊。 類型：字串

回應內文

此作業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
```

組件可以按任何順序上傳；Amazon Glacier 會使用每個組件的範圍規格來決定組合組件的順序。

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 Glacier 的 Identity and Access Management

任務操作

以下是 Amazon 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)

Description

此操作會傳回您先前啟動之任務的相關資訊，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 Glacier (Amazon SNS) 主題。 Amazon Glacier 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。不過，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 Amazon Glacier 可以在完成任務後通知該主題。

在 Amazon Glacier 完成任務後至少 24 小時內，任務 ID 不會過期。

要求

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 Amazon Glacier 在您啟動任務時傳回的路徑相同。

```
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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Amazon 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 Glacier 例外狀況和錯誤訊息的相關資訊，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 Glacier 的 Identity and Access Management

「取得任務輸出」 (GET 輸出)

Description

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，預期的大小是上傳到 Amazon Glacier (Amazon 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。

如果是封存擷取任務，Amazon Glacier 會根據您指定的位元組範圍，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。

在 Amazon Glacier 完成任務後至少 24 小時內，任務 ID 不會過期。也就是說，您可以在 Amazon Glacier 完成任務後的 24 小時內下載任務輸出。

要求

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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

除了所有作業通用的請求標頭之外，此作業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。回應包含存放在 Amazon 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	Description
Content-Range	Amazon Glacier 傳回的位元組範圍。如果只下載部分輸出，回應會提供 Amazon 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	Description
	• 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

Amazon Glacier 會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。即使 Amazon Glacier 每天準備一次保存庫庫存，只有在自上次庫存以來保存庫有存檔新增或刪除時，才會更新庫存日期。

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 Glacier 例外狀況和錯誤訊息的相關資訊，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

此範例會擷取 Amazon 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

此範例只會擷取 Amazon Glacier 為回應啟動封存擷取任務請求而準備的一部分封存。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。在此情況下，回應也包含一個Content-Range標頭，指定 Amazon 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 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 Glacier (Amazon Glacier) 任務：

- `archive-retrieval`：擷取封存
- `inventory-retrieval`：清點保存庫

主題

- 初始化封存或保存庫庫存擷取工作
- 要求
- 回應
- 範例
- 相關章節

初始化封存或保存庫庫存擷取工作

擷取封存或保存庫庫存是非同步作業，需要您啟動工作。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 Amazon 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。當您起始檢索任務時，Amazon Glacier 會建立任務，並在回應中傳回任務 ID。當 Amazon 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 Amazon Glacier 可以在任務完成後發佈通知的 Amazon SNS 主題。 Amazon Glacier 您可以為每個任務請求指定一個 SNS 主題。只有在 Amazon Glacier 完成任務後，才會傳送通知。除了指定每個工作請求的 SNS 主題外，還可以為保存庫設定保存庫通知，以便為所有擷取傳送工作通知。如需詳細資訊，請參閱設定保存庫通知組態 (PUT 通知的組態)。
- 取得工作詳細資訊：工作進行中時，您可以提出 描述任務 (GET JobID) 請求以取得工作狀態資訊。但是，使用 Amazon SNS 通知來判斷工作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。

如果針對特定事件，您可以在保存庫上新增通知組態，並在啟動任務請求中指定 SNS 主題，Amazon Glacier 會傳送這兩個通知。如需詳細資訊，請參閱設定保存庫通知組態 (PUT 通知的組態)。

此保存庫庫存

Amazon Glacier 大約每天更新一次保存庫庫存，從您第一次將封存上傳至保存庫的那一天開始。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。當您啟動保存庫庫存的任務時，Amazon Glacier 會傳回其產生的最後一個庫存，這是point-in-time快照，而不是即時資料。

Amazon Glacier 為保存庫建立第一個庫存後，通常需要半天到一天的時間才能擷取該庫存。

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，假設您在用戶端維護資料庫，關聯您上傳到 Amazon Glacier 之封存的相關中繼資料。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

您可以透過篩選在保存庫建立日期或設定限制來限制擷取到的保存庫庫存項目的數量。

透過封存建立日期進行篩選

您可以透過在啟動工作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

您可以透過在啟動工作請求中設定 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工作輸出中的標記重複建立啟動工作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

當啟動封存擷取工作時，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 Amazon Glacier 成本最低的方案，可讓您在一天內以經濟實惠的價格擷取大量資料，甚至 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 擷取 Amazon Glacier Archives。

要求

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保存庫的 `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 Amazon Glacier 在完成任務後可發佈通知的 Amazon SNS 主題。Amazon Glacier

> **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。Amazon Glacier 不會為您建立主題。主題必須具有從 Amazon Glacier 保存庫接收出版物的許可。Amazon Glacier 不會驗證保存庫是否具有發佈至主題的許可。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**
>
> `AccountId` 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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

Amazon Glacier 會建立任務。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	Description
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。 類型：字串 預設：無

回應內文

此作業不會傳回任何回應內文。

錯誤

除了所有 Amazon Glacier 操作常見的可能錯誤之外，此操作還包含下列錯誤。如需 Amazon Glacier 錯誤的資訊和錯誤代碼清單，請參閱 錯誤回應。

Code	Description	HTTP 狀態碼	Type
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	Server

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工作，以便從 examplevault 保存庫中取得封存清單。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保存庫庫存擷取工作。

```
{
    "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 Glacier 的 Identity and Access Management

列出工作 (GET 工作)

Description

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 Glacier (Amazon Glacier) 會在刪除任務之前保留最近完成的任務一段時間；不過，最終會移除已完成的任務。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。

要求

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**
>
> AccountId 值是擁有保存庫的帳戶 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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	否

名稱	描述	必要
	限制條件：無	
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 Glacier 例外狀況和錯誤訊息的相關資訊，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 Glacier 的 Identity and Access Management

在任務操作中使用的資料類型

以下是與 Amazon Glacier 中的任務操作搭配使用的資料類型。

主題

- CSVInput
- CSVOutput
- 加密

- GlacierJobDescription
- 授權
- 承授者
- InputSerialization
- InventoryRetrievalJobInput
- jobParameters
- OutputLocation
- OutputSerialization
- S3Location
- SelectParameters

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 中儲存工作結果的加密資訊。

目錄

加密

將工作結果儲存在 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

用於物件加密的 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 Glacier (Amazon Glacier) 任務的描述。

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 Amazon 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

如果在工作啟動 (啟動任務 (POST 任務)) 中設定了通知，則 Amazon Resource Name (ARN) 代表傳送工作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

該工作是子資源的保存庫 ARN。

類型：字串

詳細資訊

- 啟動任務 (POST 任務)

授權

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

保存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)

保存庫庫存清單的輸出格式，該清單是在啟動工作以擷取保存庫庫存時由 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

每個保存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，表示其繼續分頁的保存庫庫存擷取結果。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

保存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：必須是您從先前對 Amazon Glacier (Amazon Glacier) 的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 Glacier 在任務完成且輸出準備好供您下載時傳送通知的 Amazon SNS 主題的 Amazon Resource Name (ARN)。 Amazon Glacier 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，Amazon Glacier 不會為您建立。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 中儲存工作結果之位置的資訊。

目錄

AccessControlList

控制對儲存結果的存取之授權清單。

類型：授權 物件陣列

必要：否

BucketName

工作結果儲存所在之 Amazon S3 儲存貯體的名稱。儲存貯體必須與包含輸入封存物件的保存庫位於相同的 AWS 區域。

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

包含有關用於儲存工作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

要隨工作結果儲存於 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 Amazon Glacier 中可用的資料擷取相關操作。

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)

說明

此操作會傳回GET請求中指定之 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 Amazon 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況

下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Amazon Glacier 資料擷取政策。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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)

此作業會列出指定 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須符合與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID。您可以指定 AWS 帳戶 ID 或選擇性指定單一 '-'（連字號），在這種情況下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Glacier 例外狀況和錯誤訊息的相關資訊，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，Amazon Glacier (Amazon Glacier) 會傳回 HTTP 200 OK，其中包含帳戶的佈建容量單位清單，如下列範例所示。

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 限制兩個佈建容量單位 AWS 帳戶。

要求

若要購買 的佈建容量單位，請將 HTTP POST請求 AWS 帳戶 傳送至佈建容量 URI。

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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況

下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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作業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

此作業不會傳回任何回應內文。

錯誤

除了所有 Amazon Glacier 操作常見的可能錯誤之外，此操作還包含下列錯誤。如需 Amazon Glacier 錯誤的資訊和錯誤代碼清單，請參閱 錯誤回應。

Code	Description	HTTP 狀態碼	Type
`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，Amazon Glacier (Amazon Glacier) 會傳回`HTTP 201 Created`回應，如下列範例所示。

```
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)

說明

此操作會在PUT請求中指定的 AWS 區域中設定並制定資料擷取政策。您可以為每個 AWS 區域設定一個政策 AWS 帳戶。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 Amazon 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**
>
> AccountId 值為 AWS 帳戶 ID。此值必須與用來簽署請求的登入資料相關聯的 AWS 帳戶 ID 相符。您可以指定 AWS 帳戶 ID 或選擇性地指定單一 '-'（連字號），在這種情況

下，Amazon Glacier 會使用與用來簽署請求之登入資料相關聯的 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 Amazon 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

此作業不會傳回任何回應內文。

錯誤

如需 Amazon Glacier 例外狀況和錯誤訊息的相關資訊，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，Amazon Glacier (Amazon Glacier) 會設定政策並傳回 `HTTP 204 No Content`，如下列範例所示。

```
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 Amazon 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 Glacier，更能反映 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 RSS 摘要，以接收 Amazon 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 Amazon Glacier 開發人員指南每個版本的重要變更。

變更	Description	版本日期
快速和大量資料擷取	除了標準擷取之外，Amazon Glacier 現在還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	Amazon Glacier 現在支援保存庫鎖定，可讓您使用保存庫鎖定政策，在個別 Amazon Glacier 保存庫上輕鬆部署和強制執行合規控制。如需詳細資訊，請參閱Amazon Glacier 保存庫鎖定及保存庫鎖定政策。	2015 年 7 月 8 日
保存庫標籤	Amazon Glacier 現在可讓您標記 Amazon Glacier 保存庫，以便於資源和成本管理。標籤是您可以定義並與保存庫建立關聯的標籤，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 Glacier 資源及標記您的 Amazon Glacier 保存庫。	2015 年 6 月 22 日
文件庫存取政策	Amazon Glacier 現在支援使用保存庫存取政策來管理對個別 Amazon Glacier 保存庫的存取。現在，您可以直接在保存庫上定義存取政策，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保存庫存取權限。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	Amazon 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。您可以在 中按一下或使用 Amazon Glacier API AWS 管理主控台 來定義自己的資料擷取限制。如需詳細資訊，請參閱Amazon Glacier 資料擷取政策。 此外，Amazon Glacier 現在支援使用 稽核記錄 AWS CloudTrail，這會記錄您帳戶的 Amazon Glacier API 呼叫，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。如需詳細資訊，請參閱使用 記錄 Amazon Glacier API 呼叫 AWS CloudTrail。	2014 年 12 月 11 日
更新 Java 範例	更新本指南中使用 適用於 Java 的 AWS SDK的 Java 程式碼範例。	2014 年 6 月 27 日

變更	Description	版本日期
限制保存庫庫存擷取	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保存庫庫存項目的數量。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	Amazon Glacier 現在支援擷取封存的特定範圍。您可以啟動請求 Amazon Glacier 準備整個封存或一部分封存以進行後續下載的任務。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 Amazon Glacier 中下載封存。	2012 年 11 月 13 日
新的指南	這是 Amazon 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。