1.17.0 版使用者指南

# AWS SimSpace Weaver

# AWS SimSpace Weaver: 1.17.0 版使用者指南

# Table of Contents

終止支援通知： 將於 2026 AWS 年 5 月 20 日結束對 的支援 AWS SimSpace Weaver。2026 年 5 月 20 日之後，您將無法再存取 SimSpace Weaver 主控台或 SimSpace Weaver 資源。如需詳細資訊，請參閱AWS SimSpace Weaver 終止支援。

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 AWS SimSpace Weaver？

AWS SimSpace Weaver 是一項服務，可用來在 中建置和執行大規模空間模擬 AWS 雲端。例如，您可以建立群眾模擬、大型真實世界環境，以及身歷其境的互動式體驗。

使用 SimSpace Weaver，您可以將模擬工作負載分散到多個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。 會為您 SimSpace Weaver 部署基礎 AWS 基礎設施，並處理執行模擬的 Amazon EC2 執行個體之間的模擬資料管理和網路通訊。

## 的主要概念 SimSpace Weaver

模擬或遊戲受到執行該模擬或遊戲的電腦限制。隨著虛擬世界的大小和複雜性不斷增加，處理效能開始下降。計算需要更長的時間、系統記憶體不足，以及用戶端影格率下降。對於不需要即時效能的模擬，這可能只會令人煩惱。或者，這可能是業務關鍵情況，其中增加的處理延遲會導致成本增加。如果您的模擬或遊戲需要即時效能，則效能降低一定是問題。

模擬達到效能限制的常見解決方案是簡化模擬。許多使用者的線上遊戲通常會透過在不同伺服器上製作虛擬世界的副本，並將使用者分散到不同伺服器上，來解決擴展問題。

SimSpace Weaver 透過空間除以虛擬世界，並將片段分散到在 中執行的運算執行個體叢集中，來解決擴展問題 AWS 雲端。運算執行個體可共同處理整個模擬世界。您的模擬世界顯示為其中所有項目的單一整合空間，以及連接至其中的所有用戶端。由於硬體效能限制，您不再需要簡化模擬。您可以改為在雲端中新增更多運算容量。

主題

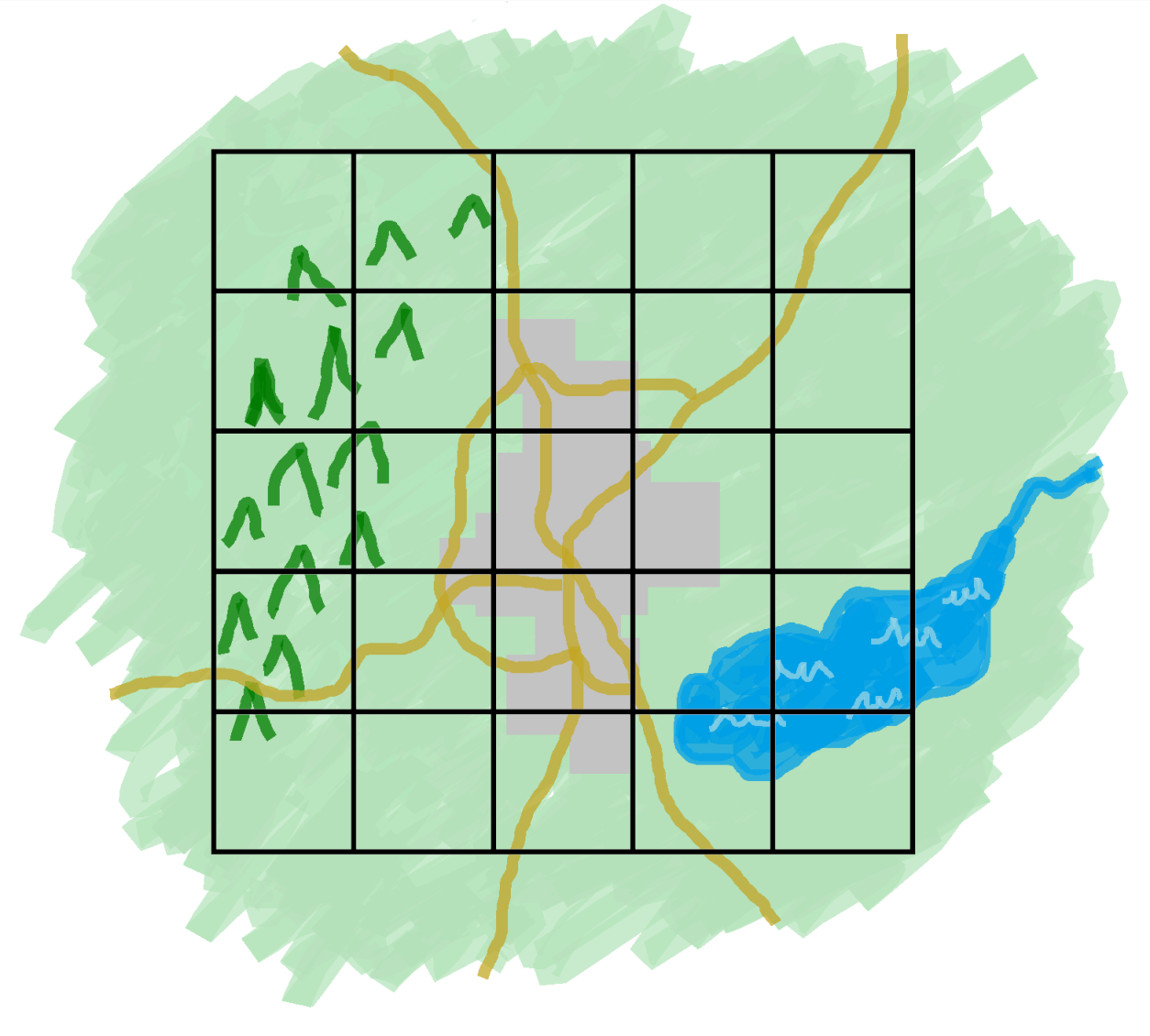
- SimSpace Weaver 運作方式
- 如何使用 SimSpace Weaver
- 模擬結構描述
- 工作者和資源單位
- 模擬時鐘
- 資料分割
- State Fabric
- 實體
- 應用程式

# SimSpace Weaver 運作方式

您的模擬包含一個包含物件的世界。某些物件（例如人物和車輛）會移動並執行動作。其他物件（例如樹木和建築物）是靜態的。在 中 SimSpace Weaver，實體是模擬世界中的物件。

您可以定義模擬世界的邊界，並將其劃分為網格。您可以建立模擬邏輯，在網格的一個儲存格上操作，而不是建立在整個網格上操作的模擬邏輯。在 中 SimSpace Weaver，空間應用程式是您撰寫的程式，可實作網格儲存格的模擬邏輯。這包括該儲存格中所有實體的邏輯。空間應用程式的擁有權區域是空間應用程式控制的網格儲存格。

> **Note**
>
> 在 中 SimSpace Weaver，「應用程式」一詞可以參考應用程式的程式碼或該程式碼的執行中執行個體。



您的模擬世界分為網格

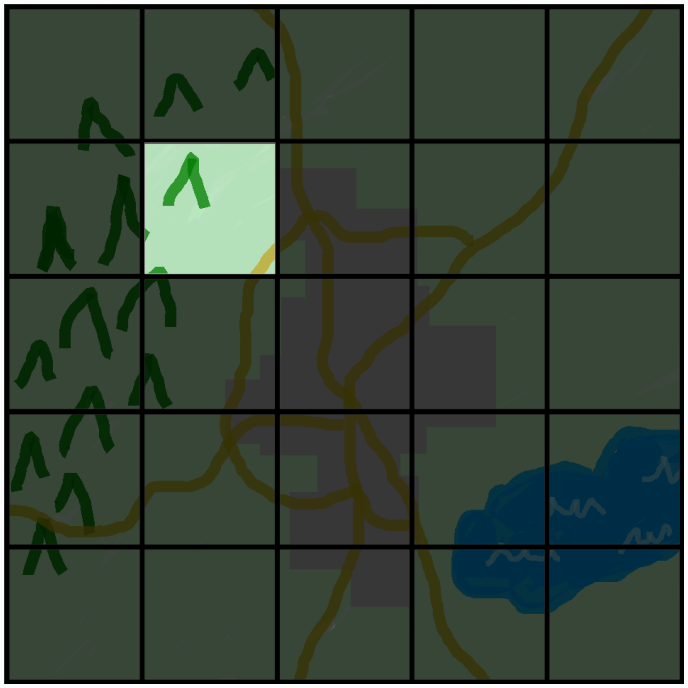
您可以將模擬世界分割為網格。每個空間應用程式都會為該網格中的單一儲存格實作模擬邏輯。

SimSpace Weaver 會為您網格的每個儲存格執行空間應用程式程式碼的執行個體。所有空間應用程式執行個體都會平行執行。基本上，會將您的整體模擬 SimSpace Weaver 分割成多個較小的模擬。每個較小的模擬都會處理整個模擬世界的一部分。 SimSpace Weaver 可以在 中的多個 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體 （稱為工作者) 之間分配和執行這些較小的模擬 AWS 雲端。單一工作者可以執行多個空間應用程式。

實體可以在模擬世界中移動。如果實體進入另一個空間應用程式的擁有區域（網格中的另一個儲存格），則新區域的空間應用程式擁有者會接管實體的控制。如果您的模擬在多個工作者上執行，則實體可以從一個工作者的空間應用程式控制移至不同工作者的空間應用程式。當實體移至不同的工作者時，會 SimSpace Weaver 處理基礎網路通訊。

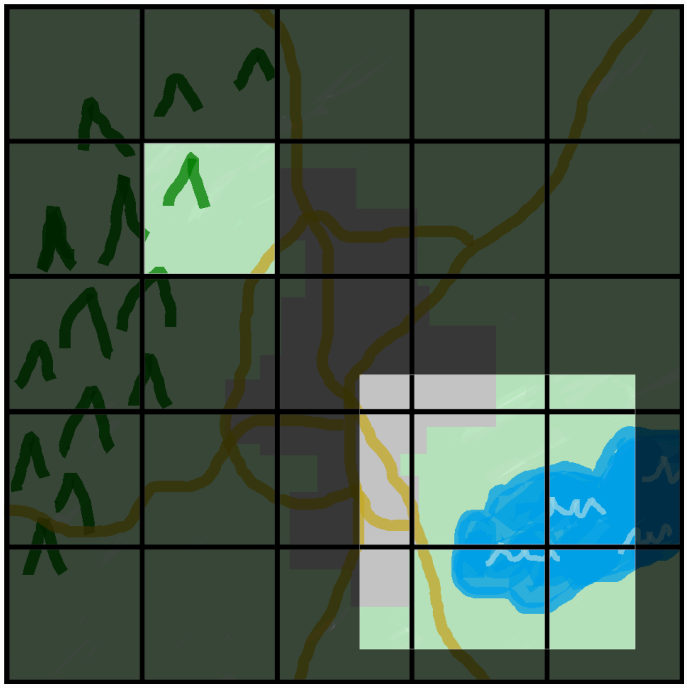
## 訂閱

空間應用程式的世界視觀表是自己的擁有權區域。為了了解模擬世界的另一個部分發生了什麼情況，空間應用程式會建立訂閱。訂閱區域是整體模擬世界區域的子集。訂閱區域可以包含多個擁有區域的一部分，包括空間應用程式自己的擁有權區域。 會 SimSpace Weaver 通知空間應用程式訂閱區域內發生的所有實體事件（例如，進入、退出、建立、更新和刪除）。



空間應用程式的世界檢視

空間應用程式的世界視觀表是其擁有權區域，這是世界網格中的一個儲存格。



新增訂閱區域的空間應用程式檢視

空間應用程式使用訂閱來了解模擬世界的另一個部分發生了什麼事。訂閱區域可以包含多個網格儲存格和部分儲存格。

例如，模擬實體互動的應用程式可能需要知道實體遠超過其擁有區域的空間邊界。為了達成此目的，應用程式可以訂閱其擁有權區域的邊界區域。建立訂閱後，應用程式會收到有關這些區域中實體事件的通知，並且可以讀取實體。另一個範例是自動駕駛車輛，無論哪個應用程式擁有該區域，都需要在前方

200 公尺處看到所有實體。車輛的應用程式可以使用篩選條件建立訂閱，做為(AABB)涵蓋可檢視區域的軸對齊週框方塊。

您可以建立不負責管理模擬空間層面的模擬邏輯。自訂應用程式是在單一工作者上執行的可執行程式。您可以控制自訂應用程式的生命週期（啟動和停止）。模擬用戶端可以連線至自訂應用程式，以檢視模擬或與之互動。您也可以建立在每個 worker 上執行的服務應用程式。在執行模擬的每個 worker 上 SimSpace Weaver 啟動服務應用程式的執行個體。

自訂應用程式和服務應用程式會建立訂閱，以了解實體事件和讀取實體。這些應用程式沒有擁有權區域，因為它們不是空間。使用訂閱是他們能夠了解模擬世界中發生情況的唯一方式。

## 如何使用 SimSpace Weaver

當您使用 時 SimSpace Weaver，以下是您遵循的主要步驟：

1. 撰寫並建置整合C++應用程式 SDK SimSpace Weaver 的應用程式。
   a. 您的應用程式會進行 API 呼叫，以與模擬狀態互動。
2. 撰寫用戶端，透過一些應用程式檢視模擬並與之互動。
3. 在文字檔案中設定模擬。
4. 將您的應用程式套件和模擬組態上傳至 服務。
5. 啟動模擬。
6. 視需要啟動和停止您的自訂應用程式。
7. 將用戶端連接至您的自訂或服務應用程式，以檢視模擬或與之互動。
8. 在 Amazon CloudWatch Logs 中檢查您的模擬日誌。
9. 停止模擬。
10清除模擬。

## 模擬結構描述

模擬結構描述（或結構描述) 是 YAML格式化的文字檔案，其中包含模擬的組態資訊。 會在啟動模擬時 SimSpace Weaver 使用您的結構描述。 SimSpace Weaver 應用程式開發套件可分發套件包含範例專案的結構描述。您可以使用此作為您自己的結構描述的起點。如需模擬結構描述的詳細資訊，請參閱 SimSpace Weaver 模擬結構描述參考。

## 工作者和資源單位

工作者是執行模擬的 Amazon EC2 執行個體。您可以在模擬結構描述中指定工作者類型。將您的工作者類型 SimSpace Weaver 映射至服務使用的特定 Amazon EC2 執行個體類型。 會為您 SimSpace Weaver 啟動和停止工作者，並管理工作者之間的網路通訊。 會為每個模擬 SimSpace Weaver 啟動一組工作者。不同的模擬使用不同的工作者。

工作者上可用的運算（處理器和記憶體）容量分為稱為運算資源單位（或資源單位) 的邏輯單位。資源單位代表固定數量的處理器和記憶體容量。

> **Note**
>
> 我們先前將運算資源單位稱為槽。您可能仍然會在我們的文件中看到此前一個術語。

## 模擬時鐘

每個模擬都有自己的時鐘。您可以使用 API 呼叫或 SimSpace Weaver 主控台來啟動和停止時鐘。模擬只會在時鐘執行時更新。模擬中的所有操作都會在稱為刻度的時段內進行。時鐘會向所有工作者宣告每個刻度的開始時間。

時鐘速率（或刻度率) 是時鐘宣告的每秒刻度數（赫茲或 Hz)。模擬所需的時鐘速率是模擬結構描述的一部分。刻度的所有操作都必須在下一個刻度開始之前完成。因此，有效的時鐘速率可以低於所需的時鐘速率。有效時鐘速率不會高於所需的時鐘速率。

## 資料分割

分割區是工作者上共用記憶體的區段。每個分割區會保留部分模擬狀態資料。

空間應用程式的分割區（也稱為空間應用程式分割區或空間分割區) 包含空間應用程式擁有區域中的所有實體。根據每個實體的空間位置，在空間應用程式分割區中 SimSpace Weaver 輸入實體。這表示 SimSpace Weaver 會嘗試將空間接近彼此的實體放置在同一個工作者上。這可最大限度地減少應用程式需要的實體知識量，而這些實體並非其擁有來模擬其擁有的實體。

## State Fabric

State Fabric 是所有工作者的共用記憶體系統（所有分割區的集合）。它會保留模擬的所有狀態資料。

State Fabric 使用自訂二進位格式，針對實體的每個資料欄位，將實體描述為一組初始資料和更新日誌。使用此格式，您可以在模擬時間的上一個時間點存取實體的狀態，並將其映射回真實世界時間的

某個點。緩衝區的大小有限，無法回到超過緩衝區中的時間。 會針對每個欄位 SimSpace Weaver 使用更新日誌中目前位移的指標，並將其更新為欄位更新的一部分。 會使用共用記憶體將這些更新日誌 SimSpace Weaver 映射到應用程式的處理空間。

此物件格式會導致低額外負荷且沒有序列化成本。 SimSpace Weaver 也會使用此物件格式來剖析和識別索引欄位（例如實體位置）。

## 實體

實體是模擬中最小的資料建置區塊。實體的範例包括演員（例如人物和車輛）和靜態物件（例如建築物和障礙物）。實體具有屬性（例如位置和方向），您可以將其儲存為持久性資料 SimSpace Weaver。實體存在於分割區中。

## 應用程式

A SimSpace Weaver app 是您撰寫的軟體，其中包含執行每個模擬刻度的自訂邏輯。大多數應用程式的目的是在模擬執行時更新實體。您的應用程式會在 SimSpace Weaver 應用程式 SDK 中呼叫 APIs，以對模擬中的實體執行動作（例如讀取和更新）。

您可以將應用程式及其必要資源（例如程式庫）封裝為 .zip 檔案，並將其上傳至其中 SimSpace Weaver。應用程式會在工作者的 Docker 容器中執行。 SimSpace Weaver 會在工作者上為每個應用程式配置固定數量的資源單位。

SimSpace Weaver 會將一個（且只有一個）分割區的擁有權指派給每個應用程式。應用程式及其分割區位於相同的工作者上。每個分割區只有一個應用程式擁有者。應用程式可以在其分割區中建立、讀取、更新和刪除實體。應用程式在其分割區中擁有所有實體。

應用程式有三種類型：空間應用程式、自訂應用程式和服務應用程式。它們因使用案例和生命週期而異。

> **Note**
>
> 在 中 SimSpace Weaver，「應用程式」一詞可以參考應用程式的程式碼或該程式碼的執行中執行個體。

## 空間應用程式

空間應用程式會更新模擬中空間存在的實體狀態。例如，您可以定義Physics應用程式，負責根據每個刻度的速度、形狀和大小移動和碰撞實體。在此情況下， SimSpace Weaver 會平行執行Physics應用程式的多個執行個體，以處理工作負載的大小。

SimSpace Weaver 管理空間應用程式的生命週期。您可以在模擬結構描述中指定空間應用程式分割區的排列。當您啟動模擬時， 會為每個空間應用程式分割區 SimSpace Weaver 啟動空間應用程式。當您停止模擬時， SimSpace Weaver 會關閉您的空間應用程式。

其他類型的應用程式可以建立實體，但只有空間應用程式可以更新實體。其他類型的應用程式必須將他們建立的實體轉移到空間網域。 SimSpace Weaver 會使用實體的空間位置，將實體移至空間應用程式的分割區。這會將實體的所有權轉移到空間應用程式。

## 自訂應用程式

您可以使用自訂應用程式來與模擬互動。自訂應用程式會使用訂閱讀取實體資料。自訂應用程式可以建立實體。不過，應用程式必須將實體轉移到空間應用程式，以在模擬中包含實體並進行更新。您可以讓 將網路端點 SimSpace Weaver 指派給自訂應用程式。模擬用戶端可以連線至網路端點，以與模擬互動。您可以在模擬結構描述中定義自訂應用程式，但您需負責啟動和停止它們（使用 SimSpace Weaver API 呼叫）。在工作者上啟動自訂應用程式執行個體後， SimSpace Weaver 不會將執行個體轉移到另一個工作者。

## 服務應用程式

當您需要在每個工作者上執行的唯讀程序時，可以使用服務應用程式。例如，如果您有大型模擬，而且需要檢視用戶端，在模擬中移動並僅向使用者顯示可見實體，則可以使用服務應用程式。在此情況下，單一自訂應用程式執行個體無法處理模擬中的所有實體。您可以將服務應用程式設定為在每個工作者上啟動。然後，每個服務應用程式都可以在其指派的工作者上篩選實體，並僅將相關實體傳送至其連線的用戶端。然後，您的檢視用戶端可以在模擬空間中移動時連線到不同的服務應用程式。您可以在模擬結構描述中設定服務應用程式。 會為您 SimSpace Weaver 啟動和停止服務應用程式。

## 應用程式摘要

下表摘要說明不同類型的 SimSpace Weaver 應用程式特性。

| | 空間應用程式 | 自訂應用程式 | 服務應用程式 |
|---|---|---|---|
| 讀取實體 | 是 | 是 | 是 |

| | 空間應用程式 | 自訂應用程式 | 服務應用程式 |
|---|---|---|---|
| 更新實體 | 是 | 否 | 否 |
| 建立實體 | 是 | 是* | 是* |
| 生命週期 | 受管 (SimSpace Weaver 控制它）。 | 未受管（您可以控制它）。 | 受管 (SimSpace Weaver 控制它）。 |
| 啟動方法 | SimSpace Weaver 會為每個空間分割區啟動一個應用程式執行個體，如您的結構描述中所指定。 | 您可以啟動每個應用程式執行個體。 | SimSpace Weaver 會在每個工作者上啟動一或多個應用程式執行個體，如您的結構描述中所指定。 |
| 用戶端可以連線 | 否 | 是 | 是 |

* 當自訂應用程式或服務應用程式建立實體時，應用程式必須將實體的所有權轉移到空間應用程式，以便空間應用程式可以更新實體的狀態。

## 網域

A SimSpace Weaver domain 是應用程式執行個體的集合，可執行相同的可執行應用程式程式碼，並具有相同的啟動選項和命令。我們會依包含的應用程式類型來參考網域：空間網域、自訂網域和服務網域。您可以在網域內設定應用程式。

## 訂閱和複寫

應用程式會建立空間區域的訂閱，以了解該區域中的實體事件（例如，進入、退出、建立、更新和刪除）。應用程式會先處理訂閱中的實體事件，再讀取其未擁有之分割區中的實體資料。

分割區可以存在於與應用程式相同的工作者上（這稱為本機分割區)，但另一個應用程式可以擁有該分割區。分割區也可以存在於不同的工作者上（這稱為遠端分割區)。如果訂閱是遠端分割區，工作者會透過稱為複寫的程序建立遠端分割區的本機複本。工作者接著會讀取本機複本（複寫的遠端分割區）。如果工作者上的另一個應用程式需要在相同的刻度上從該分割區讀取，則工作者會讀取相同的本機複本。

# 的範例使用案例 SimSpace Weaver

您可以使用 SimSpace Weaver 進行代理程式型模型和具有空間元件的離散時間步驟模擬。

### 建立大型群眾模擬

您可以使用 SimSpace Weaver 模擬真實環境中的群眾。 SimSpace Weaver 可讓您使用自己的行為，將模擬擴展到數百萬個動態物件。

### 建立城市規模環境

使用 SimSpace Weaver 建立整個城市的數位分身。建立城市規劃、設計流量路由和規劃環境危害回應的模擬。您可以使用自己的地理空間資料來源做為環境的建置區塊。

### 建立沉浸式和互動式體驗

建立多個使用者可以參與和互動的模擬體驗。使用 Unreal Engine 和 Unity 等熱門開發工具來建置 3D (3D) 虛擬世界。使用您自己的內容和行為自訂 3D 體驗。

# AWS SimSpace Weaver 終止支援

在仔細考慮之後，我們決定終止對 的支援 AWS SimSpace Weaver，自 2026 年 5 月 20 日起。 AWS SimSpace Weaver 自 2025 年 5 月 20 日起將不再接受新客戶。身為在 2025 年 5 月 20 日之前註冊服務的現有客戶，您可以繼續使用 AWS SimSpace Weaver 功能。2026 年 5 月 20 日之後，您將無法再使用 AWS SimSpace Weaver。

如需轉換至 AWS Batch 以協助執行容器化模擬的詳細資訊，請參閱此部落格文章。

# 設定 SimSpace Weaver

若要設定 SimSpace Weaver 第一次使用 ，您必須設定 AWS 帳戶 和本機環境。當您完成這些任務時，您將準備好開始教學課程。

設定任務

1. 設定您的 AWS 帳戶 以使用 SimSpace Weaver.
2. 設定 的本機環境 SimSpace Weaver.

# 設定您的 AWS 帳戶 以使用 SimSpace Weaver

完成下列任務，以設定 AWS 帳戶 來使用 SimSpace Weaver。

## 註冊 AWS 帳戶

如果您沒有 AWS 帳戶，請完成以下步驟來建立一個。

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，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。

AWS 會在註冊程序完成後傳送確認電子郵件給您。您可以隨時登錄 https://aws.amazon.com/ 並選擇我的帳戶，以檢視您目前的帳戶活動並管理帳戶。

## 建立具有管理存取權的使用者

註冊 之後 AWS 帳戶，請保護您的 AWS 帳戶根使用者 AWS IAM Identity Center、啟用和建立管理使用者，以免將根使用者用於日常任務。

保護您的 AWS 帳戶根使用者

1. 選擇根使用者並輸入 AWS 帳戶 您的電子郵件地址，以帳戶擁有者AWS Management Console身分登入 。在下一頁中，輸入您的密碼。

   如需使用根使用者登入的說明，請參閱 AWS 登入 使用者指南中的以根使用者身分登入。

2. 若要在您的根使用者帳戶上啟用多重要素驗證 (MFA)。

   如需說明，請參閱《IAM 使用者指南》中的為您的 AWS 帳戶 根使用者 （主控台） 啟用虛擬 MFA 裝置。

建立具有管理存取權的使用者

1. 啟用 IAM Identity Center。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的啟用 AWS IAM Identity Center。

2. 在 IAM Identity Center 中，將管理存取權授予使用者。

   如需使用 IAM Identity Center 目錄 做為身分來源的教學課程，請參閱AWS IAM Identity Center《 使用者指南》中的使用預設值設定使用者存取 IAM Identity Center 目錄。

以具有管理存取權的使用者身分登入

- 若要使用您的 IAM Identity Center 使用者簽署，請使用建立 IAM Identity Center 使用者時傳送至您電子郵件地址的簽署 URL。

  如需使用 IAM Identity Center 使用者登入的說明，請參閱AWS 登入 《 使用者指南》中的登入 AWS 存取入口網站。

指派存取權給其他使用者

1. 在 IAM Identity Center 中，建立一個許可集來遵循套用最低權限的最佳實務。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的建立許可集。

2. 將使用者指派至群組，然後對該群組指派單一登入存取權。

   如需指示，請參閱《AWS IAM Identity Center 使用者指南》中的新增群組。

# 新增要使用的許可 SimSpace Weaver

若要提供存取權，請新增權限至您的使用者、群組或角色：

- 中的使用者和群組 AWS IAM Identity Center：

  建立權限合集。請按照 AWS IAM Identity Center 使用者指南 中的 建立權限合集 說明進行操作。
- 透過身分提供者在 IAM 中管理的使用者：

  建立聯合身分的角色。遵循「IAM 使用者指南」的為第三方身分提供者 (聯合) 建立角色中的指示。
- IAM 使用者：
  - 建立您的使用者可擔任的角色。請按照「IAM 使用者指南」的為 IAM 使用者建立角色中的指示。
  - (不建議) 將政策直接附加至使用者，或將使用者新增至使用者群組。請遵循 IAM 使用者指南的新增許可到使用者 (主控台) 中的指示。

Example 授予使用許可的 IAM 政策 SimSpace Weaver

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateAndRunSimulations",
            "Effect": "Allow",
            "Action": [
                "simspaceweaver:*",
                "iam:GetRole",
                "iam:ListRoles",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:UpdateRole",
                "iam:CreatePolicy",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy",
                "iam:GetRolePolicy",
                "iam:DeleteRolePolicy",
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListAllMyBuckets",
                "s3:PutBucketPolicy",
                "s3:CreateBucket",
                "s3:ListBucket",
```

```
                "s3:PutEncryptionConfiguration",
                "s3:DeleteBucket",
                "cloudformation:CreateStack",
                "cloudformation:UpdateStack",
                "cloudformation:DescribeStacks"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PassAppRoleToSimSpaceWeaver",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "simspaceweaver.amazonaws.com"
                }
            }
        }
    ]
}
```

# 設定 的本機環境 SimSpace Weaver

SimSpace Weaver 模擬會在容器化 Amazon Linux 2(AL2) 環境中執行。您必須擁有 AL2 環境，才能編譯您的應用程式並將其與 SimSpace Weaver 應用程式 SDK 連結。標準本機開發環境是 中的 AL2 容器Docker。如果您選擇不使用 Docker，我們會提供在 中執行 AL2 環境的替代指示Windows Subsystem for Linux (WSL)。您也可以使用自己的方法來建立本機 AL2 環境。如需在本機執行 AL2 的一些其他方式，請參閱 Amazon EC2 文件。

> **Important**
>
> Docker 上的 Microsoft Windows是標準開發環境。為了方便起見，我們建議您使用其他方法來設定本機開發環境，但它們不是標準且不受支援。

主題

- 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Docker
- 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Windows Subsystem for Linux (WSL)

# 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Docker

本節提供在 中使用 AL2 環境設定本機 SimSpace Weaver 分佈 zip 的說明Docker。如需在 中使用 AL2 設定 的說明Windows Subsystem for Linux (WSL)，請參閱 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Windows Subsystem for Linux (WSL)。

要求

- Microsoft Windows 10 或更高版本，或相容的 Linux 系統
- Microsoft Visual Studio 2019 或更新版本，已安裝Desktop development with C++工作負載
- CMake3
- Git
- Docker Desktop
- AWS CLI
- Python 3.9

在 中使用 AL2 設定 SimSpace Weaver 分佈 zip Docker

1. 如果您尚未設定 的 AWS 登入資料 AWS CLI，請遵循下列指示：設定 AWS CLI。
2. 下載 SimSpace Weaver 應用程式開發套件可分發套件。其中包含下列各項：
   - 用於 SimSpace Weaver 應用程式開發的二進位檔和程式庫
   - 自動化部分開發工作流程的協助程式指令碼
   - 示範 SimSpace Weaver 概念的範例應用程式
3. 將檔案解壓縮到*sdk-folder*您選擇的 。
4. 前往 *sdk-folder*。
5. 輸入下列命令來安裝所需的 Python 套件：

   ```
   pip install -r PackagingTools/python_requirements.txt
   ```

6. 輸入下列命令以使用 Docker 映像設定 SimSpace Weaver 分佈。

   ```
   python setup.py
   ```

   此命令會執行下列動作：

- 建立已安裝建置 SimSpace Weaver 專案的所有需求的 AL2 Docker 映像。
- 建立啟動模擬所需的 CloudFormation 資源。
  - 您可以在 中找到 CloudFormation 堆疊範本範例 *sdk-folder*/PackagingTools/sample-stack-template.yaml
- 使用本機系統的正確路徑來設定提供的範例專案。

### 故障診斷

- Docker 似乎卡住
  - 如果在呼叫 Docker 命令之後，主控台輸出似乎卡住，請嘗試重新啟動 Docker 引擎。如果無法運作，請重新啟動您的電腦。

## 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Windows Subsystem for Linux (WSL)

本節提供在 中使用 AL2 環境設定 SimSpace Weaver 分發 zip 的說明Windows Subsystem for Linux (WSL)。如需在 中設定 AL2 的說明Docker，請參閱 在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Docker。

> **Important**
>
> 本節說明使用非 Amazon 擁有、開發或支援的 AL2 版本的解決方案。如果您選擇不使用 ，此解決方案僅為方便起見而提供Docker。如果您選擇使用此解決方案，Amazon 和 不 AWS 負任何責任。

### 要求

- Hyper-V 上的 Windows 10
- Windows Subsystem for Linux (WSL)
- 的第三方開放原始碼 AL2 分佈 WSL(下載版本 2.0.20200722.0-update.2) （請參閱說明)

> **Important**
>
> 我們的WSL說明使用 2.0.20200722.0-update.2 版的 AL2 分佈WSL。如果您使用任何其他版本，則可能會發生錯誤。

在 中使用 AL2 設定 SimSpace Weaver 分佈 zip WSL

1. 在 Windows 命令提示字元中，在 中啟動您的 AL2 環境WSL。

   ```
   wsl -d Amazon2
   ```

   > **Important**
   >
   > 當您在 中執行時WSL，請在執行位於 的其中一個 `quick-start.py` Python 協助程式指令碼時包含 `--al2`選項`sdky-folder/Samples/sample-name/tools/cloud/quick-start.py`。

2. 在 Linux shell 提示字元中，更新您的 yum 套件管理員。

   ```
   yum update -y
   ```

   > **Important**
   >
   > 如果此步驟逾時，您可能需要切換到 WSL1 並重試這些程序。結束您的 WSL AL2 工作階段，並在 Windows 命令提示中輸入以下內容：
   >
   > ```
   > wsl --set-version Amazon2 1
   > ```

3. 安裝解壓縮工具。

   ```
   yum install -y unzip
   ```

4. 移除任何yum已安裝 AWS CLI 的 。如果您不確定是否已yum安裝 ，請嘗試以下兩個命令 AWS CLI。

```
yum remove awscli
```

```
yum remove aws-cli
```

5. 建立暫存目錄並前往它。

```
mkdir ~/temp
cd ~/temp
```

6. 下載並安裝 AWS CLI：

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
./aws/install
```

7. 您可以移除暫時目錄。

```
cd ~
rm -rf temp
```

8. 重新啟動 shell 工作階段以更新環境中的路徑。

```
exec
```

9. 在 AL2 環境中設定 AWS CLI 的 AWS 登入資料。如需詳細資訊，請參閱設定 AWS CLI。如果您使用 AWS IAM Identity Center，請參閱AWS Command Line Interface 《 使用者指南》中的設定 AWS CLI 以使用 AWS IAM Identity Center 。

```
aws configure
```

10. 安裝 Git。

```
yum install -y git
```

11. 安裝 wget。

```
yum install -y wget
```

12. 為 SimSpace Weaver 應用程式 SDK 建立資料夾。

```
mkdir sdk-folder
```

13. 前往 SDK 資料夾。

```
cd sdk-folder
```

14. 下載 SimSpace Weaver 應用程式開發套件可分發套件。其中包含下列各項：

    - 用於 SimSpace Weaver 應用程式開發的二進位檔和程式庫
    - 自動化部分開發工作流程的協助程式指令碼
    - 示範 SimSpace Weaver 概念的範例應用程式

```
wget https://artifacts.simspaceweaver.us-east-2.amazonaws.com/latest/
SimSpaceWeaverAppSdkDistributable.zip
```

15. 解壓縮檔案。

```
unzip *.zip
```

16. 執行 WSL 設定指令碼。

```
source ./setup-wsl-distro.sh
```

17. 輸入下列命令來安裝所需的 Python 套件：

```
pip install -r PackagingTools/python_requirements.txt
```

18. 執行 SimSpace Weaver 分佈 zip 設定指令碼：

```
python setup.py --samples --cloudformation
```

    此命令會執行下列動作：

    - 建立啟動模擬所需的 CloudFormation 資源。
        - 您可以在 中找到 CloudFormation 堆疊範本範例 *sdk-folder*/PackagingTools/sample-stack-template.yaml
    - 使用本機系統的正確路徑來設定提供的範例專案。

> **Note**
>
> 您只需要在 WSL 中為您的 AL2 環境執行一次此操作。

# 搭配 使用授權軟體 AWS SimSpace Weaver

AWS SimSpace Weaver 可讓您使用您選擇的模擬引擎和內容來建置模擬。在使用 時 SimSpace Weaver，您需負責取得、維護和遵守您在模擬中使用的任何軟體或內容的授權條款。驗證您的授權合約可讓您在虛擬託管環境中部署軟體和內容。

# 入門 SimSpace Weaver

本節提供教學課程，協助您開始使用 SimSpace Weaver。這些教學課程會向您介紹使用 建置模擬的一般工作流程 SimSpace Weaver。這些教學課程示範如何在 中建立、部署和執行模擬 SimSpace Weaver。我們建議您從快速入門教學課程開始，以在幾分鐘內執行模擬。之後請進行其他教學課程，以進一步了解。

這些教學課程使用您在設定程序期間下載 SimSpace Weaver 的應用程式 SDK .zip 檔案中包含的範例應用程式 (`PathfindingSample`)。範例應用程式示範所有 SimSpace Weaver 模擬共用的概念，包括空間分割、跨分割區實體交接、應用程式和訂閱。

在教學課程中，您將建立具有四個空間分割區的模擬。`PathfindingSample` 空間應用程式的個別執行個體會管理每個個別分割區。空間應用程式會在自己的分割區中建立實體。實體會移至模擬世界中的特定位置，避免障礙物移動。您可以使用單獨的用戶端應用程式（包含在 SimSpace Weaver 應用程式 SDK 中）來檢視模擬。

主題

- 的快速入門教學課程 SimSpace Weaver
- 詳細教學課程：了解建置範例應用程式時的詳細資訊

## 的快速入門教學課程 SimSpace Weaver

本教學課程會引導您完成在 上 SimSpace Weaver 建置和執行模擬的程序。我們建議您從本教學課程開始，然後進行詳細的教學課程。

要求

開始之前，請確定您已完成 中的步驟設定 SimSpace Weaver。

> **Note**
>
> 此處使用的指令碼是為了方便起見，並非必要。請參閱詳細教學課程，了解如何手動完成這些步驟。

## 步驟 1：啟用記錄 （選用）

開啟記錄

1. 導覽至：

```
sdk-folder/Samples/PathfindingSample/tools
```

2. 在文字編輯器中開啟結構描述檔案：

```
pathfinding-single-worker-schema.yaml
```

3. 尋找 檔案開頭的 simulation_properties: 區段：

```
simulation_properties:
  default_entity_index_key_type: "Vector3<f32>"
```

4. 在行 後面插入以下 2 行simulation_properties:：

```
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
```

5. 確認您的 simulation_properties: 區段與下列項目相同：

```
simulation_properties:
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
```

6. 儲存檔案並結束您的文字編輯器。

## 步驟 2：使用主控台用戶端快速入門 （選項 1)

導覽至：

```
sdk-folder/Samples/PathfindingSample/tools/cloud
```

請執行下列其中一個命令：

- Docker： python quick-start.py --consoleclient

- WSL： `python quick-start.py —-consoleclient --al2`

根據預設，這將在單一工作者上啟動具有單一分割區的模擬。其他組態可以透過`--schema {file name}.yaml`從 `/Samples/PathfindingSample/tools/` 資料夾傳遞 來啟動。

> **Note**
>
> 詳細教學課程：了解建置範例應用程式時的詳細資訊 如需此指令碼功能的深入說明，請參閱 。

## 步驟 2：使用 Unreal Engine 用戶端快速入門 （ 選項 2)

請參閱 啟動 Unreal Engine 檢視用戶端。

## 停止並刪除模擬

導覽至：

```
sdk-folder/Samples/PathfindingSample/tools/cloud
```

尋找模擬的名稱：

```
aws simspaceweaver list-simulations
```

停止並刪除模擬

```
python stop-and-delete.py --simulation simulation-name
```

## 故障診斷

- FileNotFoundError：cmake

```
subprocess.run('cmake')
...
 FileNotFoundError: The system cannot find the file specified
```

  - 解決方案：指令碼找不到命令 cmake。請確定您已安裝最低建議的 CMake 版本，並且可以使用 PATH 中的 cmake命令來呼叫。使用 命令cmake -version來驗證。

- ImportError：匯入 libweaver_app_sdk_python_v1 時 DLL 載入失敗：找不到指定的模組。
  - 解決方案：當 Python 3.9 未用於啟動 Weaver Python SDK 時，會發生此錯誤。請確保與「python」命令相關聯的 Python 版本為 Python 3.9。您可以執行 `python --version`命令來檢查 。
- 快速入門指令碼在啟動 Docker Build 後卡在 。
  - 解決方法：有時候 Docker 需要幾分鐘的時間來暖機。如果此問題持續超過 ~5 分鐘，請重新啟動 Docker 或您的系統。
- target_compile_features 沒有 CXX 編譯器 "GNU" 的已知功能：
  - 解決方案：清除 Docker 快取、刪除 Weaverappbuilder Docker 映像、刪除專案建置成品，然後重新執行 `setup.py`。這應該會重設 Docker 環境並解決錯誤。

# 詳細教學課程：了解建置範例應用程式時的詳細資訊

快速入門教學課程說明如何使用 `quick-start.py`和 建置、啟動、停止和刪除範例模擬`stop-and-delete.py`。本教學課程將詳細說明這些指令碼的運作方式，以及這些指令碼可以採取的額外參數，以最大限度地提高自訂 Weaver 模擬的靈活性。

要求

開始之前，請確定您已完成 中的步驟設定 SimSpace Weaver。

## 步驟 1：啟用記錄 （選用）

開啟記錄

1. 導覽至：

   ```
   sdk-folder/Samples/PathfindingSample/tools
   ```

2. 在文字編輯器中開啟結構描述檔案：

   ```
   pathfinding-single-worker-schema.yaml
   ```

3. 尋找 檔案開頭的 `simulation_properties:`區段：

   ```
   simulation_properties:
     default_entity_index_key_type: "Vector3<f32>"
   ```

4. 在行 後面插入以下 2 行simulation_properties:：

```
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
```

5. 確認您的simulation_properties:區段與下列項目相同：

```
simulation_properties:
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
```

6. 儲存檔案並結束您的文字編輯器。

## 步驟 2：啟動模擬

如快速入門教學中所示，啟動範例模擬的最基本步驟為：

1. 導覽至：

```
sdk-folder/Samples/PathfindingSample/tools/cloud
```

2. 執行其中一個模糊命令：

   - Docker： `python quick-start.py`
   - WSL： `python quick-start.py --al2`

此指令碼會自動執行常見的終端機命令，所有命令都可以使用 手動執行 AWS CLI。這些步驟包括：

1. 將 Weaver 結構描述上傳至 S3。

   - SimSpace Weaver 使用結構描述來設定您的模擬。結構描述是 YAML 格式的純文字檔案。如需詳細資訊，請參閱 設定模擬。

2. 建置和上傳自訂容器 （選用）。

   - 如果您的結構描述定義了自訂容器，快速入門指令碼將建置 Docker 映像並將其上傳至 Amazon ECR。如需詳細資訊，請參閱自訂容器。如需此功能的範例，請參閱 PythonBubblesSample 結構描述。

3. 建置專案。

- `quick-start.py` 會呼叫 中定義的 `build_project`函數`build.py`。此步驟會根據專案而有所不同。對於 PathfindingSample，會使用 CMake。CMake 和 Docker 命令可在 中找到。`build.py`

4. 將建置成品上傳至 S3。

   - 您可以檢查 S3 儲存貯體，以確保所有上傳都成功。如需使用 Amazon S3 的詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的建立、設定和使用 Amazon S3 儲存貯體。
   - 範例應用程式 zip 和 S3 儲存貯體使用以下名稱格式：
     - `weaver-sample-bucket-account-number-region`
     - 空間應用程式： `ProjectNameSpatial.zip`
     - 檢視（自訂）應用程式： `ProjectNameView.zip`

5. 啟動模擬。

   - 這是`aws simspaceweaver start-simulation` AWS CLI 通話的包裝函式。如需詳細資訊，請參閱 的 AWS CLI 命令參考 SimSpace Weaver。
   - 指令碼會循環，直到模擬狀態為 `STARTED`或 為止`FAILED`。模擬可能需要幾分鐘的時間才會開始。

6. 取得模擬詳細資訊。

   - DescribeSimulation API 提供模擬的詳細資訊，包括其狀態。模擬可以處於下列其中一種狀態：

     模擬生命週期狀態

     1. **`STARTING`** – 呼叫 後的初始狀態 StartSimulation
     2. **`STARTED`** – 所有空間應用程式都會啟動且運作狀態良好
     3. **`STOPPING`** – 呼叫 後的初始狀態 StopSimulation
     4. **`STOPPED`** – 所有運算資源都會停止
     5. **`DELETING`** – 呼叫 後的初始狀態 DeleteSimulation
     6. **`DELETED`** – 指派給模擬的所有資源都會遭到刪除
     7. **`FAILED`** – 模擬發生嚴重錯誤/失敗並停止
     8. **`SNAPSHOT_IN_PROGRESS`** – 快照正在進行中

取得模擬詳細資訊

1. 呼叫 ListSimulations API。

```
aws simspaceweaver list-simulations
```

指令碼應會顯示每個模擬的詳細資訊，如下所示：

```
{
    "Status": "STARTED",
    "CreationTime": 1664921418.09,
    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "Arn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15",
    "TargetStatus": "STARTED"
}
```

2. 呼叫 DescribeSimulation 以取得您的模擬詳細資訊。從上一個步驟Name的輸出中*simulation-name*，以模擬的 取代 。

```
aws simspaceweaver describe-simulation --simulation simulation-name
```

指令碼應會顯示您指定之模擬的更多詳細資訊，如下所示：

```
{
    "Status": "STARTED",
    "CreationTime": 1664921418.09,
    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "Arn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15",
    "TargetStatus": "STARTED"
}
```

7. 啟動自訂應用程式。

- SimSpace Weaver 不會管理自訂應用程式的生命週期。您必須啟動自訂應用程式。最佳實務是在啟動模擬時鐘之前啟動自訂應用程式，但您可以在啟動時鐘之後啟動自訂應用程式。

  您可以呼叫 StartApp API 來啟動自訂應用程式。

```
aws simspaceweaver start-app --simulation simulation-name --name app-name --
domain domain-name
```

StartApp API 呼叫將使用您提供的名稱來建立和啟動自訂應用程式的新執行個體。如果您提供已存在的應用程式名稱，則會收到錯誤。如果您想要重新啟動特定應用程式（執行個體），您必須先停止該應用程式並將其刪除。

> **Note**
>
> 模擬的狀態必須在您可以啟動自訂應用程式STARTED之前。

範例應用程式提供ViewApp自訂應用程式來檢視您的模擬。此應用程式為您提供靜態 IP 地址和連接埠號碼來連接模擬用戶端（您將在本教學稍後的步驟中執行此操作）。您可以將 domain視為具有相同可執行程式碼和啟動選項的應用程式類別。app name 可識別應用程式的執行個體。如需 SimSpace Weaver 概念的詳細資訊，請參閱 的主要概念 SimSpace Weaver。

您可以使用 DescribeApp API 在啟動自訂應用程式後檢查其狀態。

```
aws simspaceweaver describe-app --simulation simulation-name --app app-name --
domain domain-name
```

在本教學課程中啟動檢視應用程式

1. 呼叫 StartApp。 ViewApp

   ```
   aws simspaceweaver start-app --simulation simulation-name --name ViewApp --
   domain MyViewDomain
   ```

2. 呼叫 DescribeApp 以檢查自訂應用程式的狀態。

   ```
   aws simspaceweaver describe-app --simulation simulation-name --app ViewApp --
   domain MyViewDomain
   ```

在自訂應用程式（執行個體）的狀態為 之後STARTED， 的輸出DescribeApp將包含該自訂應用程式的 IP 地址和連接埠號碼（執行個體）。在下列範例輸出中，IP 地址是 的值Address，連接埠號碼是 EndpointInfo 區塊Actual中的 值。

```
{
    "Status": "STARTED",
    "Domain": "MyViewDomain",
    "TargetStatus": "STARTED",
    "Simulation": "MyProjectSimulation_22-10-04_22_10_15",
    "LaunchOverrides": {
        "LaunchCommands": []
    },
    "EndpointInfo": {
        "IngressPortMappings": [
            {
                "Declared": 7000,
                "Actual": 4321
            }
        ],
        "Address": "198.51.100.135"
    },
    "Name": "ViewApp"
}
```

> **Note**
>
> 的值`Declared`是應用程式程式碼應繫結的連接埠號碼。的值`Actual`是向用戶端 SimSpace Weaver 公開的連接埠號碼，以連接到您的應用程式。將`Declared`連接埠 SimSpace Weaver 映射到`Actual`連接埠。

> **Note**
>
> 您可以使用 所述的程序取得自訂應用程式的 IP 地址和連接埠號碼，取得任何已啟動自訂應用程式的 IP 地址和連接埠號碼。

8. 啟動時鐘。

- 當您第一次建立模擬時，它有一個時鐘，但時鐘未執行。當您的時鐘未執行時，模擬不會更新其狀態。啟動時鐘後，它會開始傳送刻度到您的應用程式。每個刻度，您的空間應用程式會逐步查看他們擁有的實體，並將結果遞交給 SimSpace Weaver

  > **Note**
  >
  > 啟動時鐘可能需要 30-60 秒。

  呼叫 StartClock API。

  ```
  aws simspaceweaver start-clock --simulation simulation-name
  ```

  > **Note**
  >
  > StartClock API 使用您的 *simulation-name*，您可以使用 ListSimulations API 找到：
  >
  > ```
  > aws simspaceweaver list-simulations
  > ```

快速入門參數

- -h、 -求助
  - 列出這些參數。
- --clean
  - 在建置之前刪除建置目錄的內容。
- --al2
  - 直接在原生機器而非 Docker 上建置。只有在 Amazon Linux 2 環境中執行，例如 WSL 時，才能使用此功能。
- --uploadonly
  - 僅將結構描述和應用程式 zip 上傳至 Amazon S3，請勿啟動模擬。
- --nobuild
  - 略過重建專案。
- --無容器

  - 略過重建結構描述中列出的模擬容器。
- --consoleclient
  - 自動建置並連接 https：//config.py 中列出的主控台用戶端。
- --結構描述 SCHEMA
  - 此調用將使用的結構描述。預設為 config.py 中的 'SCHEMA' 值。
- --name NAME
  - 模擬的名稱。預設為 config.py 中 'PROJECT_NAME'-date-time 的值。

## 步驟 3：檢查日誌（選用）

SimSpace Weaver 會將模擬管理訊息和主控台輸出從應用程式寫入 Amazon CloudWatch Logs。如需使用日誌的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的使用日誌群組和日誌串流。

您建立的每個模擬在 CloudWatch Logs 中都有自己的日誌群組。日誌群組的名稱是在模擬結構描述中指定。在下列結構描述程式碼片段中， 的值`log_destination_service`為 `logs`。這表示 的值`log_destination_resource_name`是日誌群組的名稱。在此情況下，日誌群組為 `MySimulationLogs`。

```
simulation_properties:
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
```

您也可以在啟動日誌群組之後，使用 DescribeSimulation API 來尋找日誌群組的名稱以進行模擬。

```
aws simspaceweaver describe-simulation --simulation simulation-name
```

下列範例顯示DescribeSimulation描述記錄組態的 輸出部分。日誌群組的名稱會顯示在 的結尾`LogGroupArn`。

```
    "LoggingConfiguration": {
        "Destinations": [
            {
                "CloudWatchLogsLogGroup": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:111122223333:log-
group:MySimulationLogs"
                }
            }
        ]
    },
```

每個模擬日誌群組包含數個日誌串流：

- 管理日誌串流 – 由 SimSpace Weaver 服務產生的模擬管理訊息。

  ```
  /sim/management
  ```

- 錯誤日誌串流 – SimSpace Weaver 服務產生的錯誤訊息。只有在發生錯誤時，此日誌串流才會存在。 會將應用程式寫入的錯誤 SimSpace Weaver 存放在其自己的應用程式日誌串流中 （請參閱下列日誌串流）。

  ```
  /sim/errors
  ```

- 空間應用程式日誌串流 （每個工作者上的每個空間應用程式 1 個） – 空間應用程式產生的主控台輸出。每個空間應用程式都會寫入自己的日誌串流。*spatial-app-id* 是 結尾斜線後面的所有字元*worker-id*。

  ```
  /domain/spatial-domain-name/app/worker-worker-id/spatial-app-id
  ```

- 自訂應用程式日誌串流 （每個自訂應用程式執行個體 1 個） – 自訂應用程式產生的主控台輸出。每個自訂應用程式執行個體都會寫入自己的日誌串流。

  ```
  /domain/custom-domain-name/app/custom-app-name/random-id
  ```

- 服務應用程式日誌串流 （每個服務應用程式執行個體 1 個） – 服務應用程式產生的主控台輸出。每個服務應用程式都會寫入自己的日誌串流。*service-app-id* 是 結尾斜線後面的所有字元*service-app-name*。

  ```
  /domain/service-domain-name/app/service-app-name/service-app-id
  ```

> **Note**
>
> 範例應用程式沒有服務應用程式。

## 步驟 4：檢視您的模擬

SimSpace Weaver 應用程式開發套件提供檢視範例應用程式的不同選項。如果您沒有 Unreal Engine 開發的任何本機支援，您可以使用範例主控台用戶端。Unreal Engine 用戶端的指示假設您使用 Windows。

主控台用戶端會在實體事件發生時顯示其清單。用戶端會從 取得實體事件資訊ViewApp。如果您的主控台用戶端顯示事件清單，則會確認與模擬中 ViewApp和 活動的網路連線。

PathfindingSample 模擬會在二維平面上建立固定和移動的實體。移動中的實體會四處移動固定實體。Unreal Engine 用戶端提供實體事件的視覺化。

### 主控台用戶端

quick-start.py 如果您包含 --consoleclient選項，則可以在使用 啟動範例時自動建置和連接主控台用戶端。若要在已呼叫 quick-start.py 之後建置並連接主控台用戶端，請執行下列動作：

導覽至：

```
sdk-folder/Clients/TCP/CppConsoleClient
```

執行指令碼來建置和連接用戶端：

```
python start_client.py --host ip-address --port port-number
```

指令碼將執行下列動作：

1. 使用 CMake 建置主控台用戶端。
2. 使用指定的 IP 地址和連接埠號碼啟動建置的可執行檔。

```
.\WeaverNngConsoleClient.exe --url tcp://ip-address:port-number
```

### Unreal Engine 用戶端

請參閱 啟動 Unreal Engine 檢視用戶端。

## 步驟 5：停止並刪除模擬

導覽至：

```
sdk-folder/Samples/PathfindingSample/tools/cloud
```

尋找模擬的名稱：

```
aws simspaceweaver list-simulations
```

停止並刪除模擬：

```
python stop-and-delete.py --simulation simulation-name
```

指令碼`stop-and-delete.py`將執行下列動作：

1. 呼叫 AWS CLI 命令以停止模擬。
   - `aws simspaceweaver stop-simulation`
   - 如需詳細資訊，請參閱 的AWS CLI 命令參考 SimSpace Weaver。
2. 呼叫 AWS CLI 命令來刪除模擬。
   - `aws simpaceweaver delete-simulation`
   - 如需詳細資訊，請參閱 的AWS CLI 命令參考 SimSpace Weaver。

stop-and-delete 參數

- -h、 -求助
  - 列出這些參數。
- -- 模擬模擬
  - 要stop-and-delete的模擬名稱
- --stop
  - 僅停止模擬。不會刪除它。
- --刪除

- 僅刪除模擬。只有在模擬為 STOPPED或 時，才會運作FAILED。

## 故障診斷

請參閱快速入門教學故障診斷中的 。

# 使用 SimSpace Weaver

本章提供的資訊和指導可協助您在 中建置自己的應用程式 SimSpace Weaver。

主題



## 設定模擬

模擬結構描述 （或結構描述) 是 YAML格式化的文字檔案，可指定模擬的組態。您可以使用相同的結構描述來啟動多個模擬。結構描述檔案位於模擬的專案資料夾中。您可以使用任何文字編輯器來編輯檔案。 SimSpace Weaver 只會在啟動模擬時讀取您的結構描述。您對結構描述檔案所做的任何編輯，都只會影響您在編輯後開始的新模擬。

若要設定模擬，請編輯模擬結構描述檔案 （針對您的作業系統使用適當的路徑分隔符號）：

```
project-folder\tools\project-name-schema.yaml
```

您在建立新的模擬時上傳模擬結構描述。專案的快速入門協助程式指令碼將上傳結構描述，做為建置模擬程序的一部分：

```
project-folder\tools\windows\quick-start.py
```

如需執行快速入門指令碼的詳細資訊，請參閱本指南開始使用章節詳細教學課程中的 。

## 模擬組態參數

模擬結構描述包含引導資訊，包括：

- 模擬屬性 – SDK 版本和運算組態 (工作者類型和數量 )
- 時鐘 – 刻度率和公差
- 空間分割策略 – 空間拓撲 （ 例如網格 ）、邊界和置放群組 （ 工作者上的空間分割分組 ）
- 網域及其應用程式 – 應用程式儲存貯體、路徑和啟動命令 (s)

SimSpace Weaver 使用您的結構描述組態來設定和安排空間分割區、啟動應用程式，並以您指定的刻度率推進模擬。

> **Note**
>
> SimSpace Weaver 應用程式開發套件中的 create-project 指令碼會根據範例應用程式，自動為您產生模擬結構描述。

下列主題說明模擬結構描述中的參數。如需模擬結構描述的完整說明，請參閱 SimSpace Weaver 模擬結構描述參考。

主題

- SDK 版本
- 模擬屬性
- 工作程序
- 時鐘
- 分割策略

- 網域

## SDK 版本

sdk_version 欄位指定 SimSpace Weaver 結構描述格式化的版本。有效值：1.17、1.16、1.15、1.14、1.13、1.12

> **Important**
>
> 的值sdk_version僅包含主要版本編號和第一個次要版本編號。例如， 值1.12會指定所有版本 1.12.x，例如 1.12.0、 1.12.1和 1.12.2。

## 模擬屬性

結構描述的 simulation_properties區段會指定實體索引欄位（通常是空間位置）的記錄組態和資料類型。

```
simulation_properties:
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
```

的值log_destination_service決定 值的解譯log_destination_resource_name。目前，僅支援的值為 logs。這表示 的值log_destination_resource_name是 Amazon CloudWatch Logs 中日誌群組的名稱

> **Note**
>
> 記錄是選用的。如果您不設定日誌目的地屬性，則模擬不會產生日誌。

需要 default_entity_index_key_type 屬性。唯一有效的值為 Vector3<f32>。

## 工作程序

workers 區段會指定您想要用於模擬的工作者類型和數量。 SimSpace Weaver 會使用自己的工作者類型來映射至 Amazon EC2 執行個體類型。

```
workers:
  MyComputeWorkers:
    type: "sim.c5.24xlarge"
    desired: 1
```

## 啟用多工作者模擬

您可以建立使用超過 1 個工作者的模擬。根據預設，模擬會使用 1 個工作者。您必須先修改模擬結構描述，才能開始模擬。

> **Note**
>
> 您無法變更已啟動的模擬。如果您想要為執行中的模擬啟用多工作者，您必須先停止並刪除模擬。

若要使用多個工作者，請將運算執行個體`desired`的數量設定為大於 1 的值。每個工作者的應用程式數量上限。如需詳細資訊，請參閱 SimSpace Weaver 端點和配額。只有在工作者上的應用程式數量超過此限制時， SimSpace Weaver 才會使用超過 1 個工作者。 SimSpace Weaver 可以在任何可用的工作者上放置應用程式。無法保證在特定工作者上放置應用程式。

下列結構描述程式碼片段示範了請求 2 個工作者的模擬組態。如果應用程式數量超過 1 個工作者的應用程式數量上限， SimSpace Weaver 會嘗試配置第二個工作者。

```
workers:
  MyComputeWorkers:
    type: "sim.c5.24xlarge"
    desired: 2
```

## 時鐘

`clock` 區段指定模擬時鐘的屬性。目前，您只能設定刻度率（時鐘傳送給應用程式的每秒刻度數）。刻度率是最高速率。有效的刻度率可能較低，因為刻度的所有操作（例如實體更新）都必須在下一個刻度開始之前完成。刻度率也稱為時鐘率。

的有效值`tick_rate`取決於結構描述中`sdk_version`指定的 。

刻度率的有效值

- 早於 的版本"1.14"：
  - 10
  - 15
  - 30
- 版本 "1.14" 或更新版本：
  - "10"
  - "15"
  - "30"
  - "unlimited"

    如需詳細資訊，請參閱無限制刻度率。

> **Important**
>
> - 對於 值sdk_version早於 "1.14"的結構描述， tick_rate 是整數，例如 30。
> - 對於 "1.14"或更高sdk_version版本的結構描述， 的值tick_rate是字串，例如 "30"。值必須包含雙引號。
>
>  如果您將版本"1.12"或結構描述轉換為版本 "1.13" "1.14"或更新版本，則必須以tick_rate雙引號括住 的值。

## 無限制刻度率

您可以將 tick_rate 設定為 "unlimited" ，讓您的模擬能夠以程式碼執行的速度執行。使用無限制的刻度率， 會在所有應用程式完成目前刻度的遞交後立即 SimSpace Weaver 傳送下一個刻度。

> **Important**
>
> 1.14.0 之前的 SimSpace Weaver 版本不支援無限制的刻度率。結構描述sdk_version中 的最小值為 "1.14"。

中的無限制刻度率 SimSpace Weaver Local

SimSpace Weaver Local 實作 ，"unlimited"就好像結構描述指定了 10 kHz (10000) 的刻度率。效果與 中的無限制刻度率相同 AWS 雲端。您仍然會在結構描述tick_rate: "unlimited"中指定 。如需有關 SimSpace Weaver Local 的詳細資訊，請參閱 中的本機開發 SimSpace Weaver。

## 有關時鐘的常見問題

Q1. (問題 1)：我可以變更 STARTED 模擬以使用不同的刻度率嗎？

您無法變更已存在於 AWS 雲端 生命週期任何階段之模擬的刻度率。您也無法變更在 中執行之模擬的刻度率SimSpace Weaver Local。您可以在結構描述tick_rate中設定 ，並從該結構描述啟動新的模擬。

Q2. (問題 2): 我可以在 1.14 之前的版本中以無限制的刻度率執行模擬嗎？

否，1.14.0 之前的版本不支援無限制的刻度率。

## 故障診斷時鐘錯誤

如果您的模擬無法啟動，您可以在 DescribeSimulation API 的輸出"StartError"中檢查 的值。結構描述中的無效tick_rate值會產生下列錯誤。

> **Note**
>
> 此處顯示的錯誤輸出會顯示在多行上，以改善可讀性。實際錯誤輸出是單行。

- sdk_version 早於 ，"1.14"而 的值tick_rate是無效的整數。有效值：10、15、30

```
"[{\"errorType\":\"SchemaFormatInvalid\",\"errorMessage\":
    \"$.clock.tick_rate: does not have a value in the enumeration [10, 15, 30]\"}]"
```

- sdk_version 早於 ，"1.14"而 的值tick_rate是字串。有效值：10、15、30

```
"[{\"errorType\":\"SchemaFormatInvalid\",\"errorMessage\":
    \"$.clock.tick_rate: does not have a value in the enumeration [10, 15, 30]\"},
    {\"errorType\":\"SchemaFormatInvalid\",
    \"errorMessage\":\"$.clock.tick_rate: string found, integer expected\"}]"
```

- sdk_version 是 "1.14"或更新版本，而 的值tick_rate是無效的字串。有效值："10"、"15"、"30"、"unlimited"

```
"[{\"errorType\":\"SchemaFormatInvalid\",\"errorMessage\":
```

```
    \"$.clock.tick_rate: does not have a value in the enumeration [10, 15, 30,
 unlimited]\"}]"
```

- sdk_version 是 "1.14"或更新版本，而 的值tick_rate是整數。有效值："10"、"15"、"30"、"unlimited"

```
"[{\"errorType\":\"SchemaFormatInvalid\",\"errorMessage\":
    \"$.clock.tick_rate: does not have a value in the enumeration [10, 15, 30,
 unlimited]\"},
    {\"errorType\":\"SchemaFormatInvalid\",
    \"errorMessage\":\"$.clock.tick_rate: integer found, string expected\"}]"
```

## 分割策略

partitioning_strategies 區段指定空間應用程式的分割區組態屬性。您可以為分割策略提供自己的名稱（本節中的一組屬性），並將其用於空間應用程式組態。

```
partitioning_strategies:
  MyGridPartitioning:
    topology: "Grid"
    aabb_bounds:
      x: [0, 1000]
      y: [0, 1000]
    grid_placement_groups:
      x: 1
      y: 1
```

topology 屬性指定模擬使用的座標系統類型。值Grid指定二維 (2D) 網格。

對於Grid拓撲，模擬空間會建模為軸對齊週框方塊 (AABB)。您可以在 aabb_bounds 屬性中指定 AABB 每個軸的座標邊界。模擬中存在空間中的所有實體都必須在 AABB 中具有位置。

### 網格置放群組

置放群組是 SimSpace Weaver 您要放置在相同工作者上的空間應用程式分割區集合。您可以在 grid_placement_groups 屬性中指定置放群組（在網格中）的數量和配置。 SimSpace Weaver 會嘗試將分割區平均分散到置放群組。在相同置放群組中具有分割區的空間應用程式的擁有權區域將在空間上相鄰。

我們建議 x * y 等於您想要的工作者數量。如果不相等， SimSpace Weaver 會嘗試平衡可用工作者之間的置放群組。

如果您未指定置放群組組態， SimSpace Weaver 會為您計算一個。

## 網域

您可以為網域的一組組態屬性提供名稱。網域中應用程式的啟動設定會決定網域的類型：

- **`launch_apps_via_start_app_call`** – 自訂網域
- **`launch_apps_by_partitioning_strategy`** – 空間網域
- **`launch_apps_per_worker`**（不包含在範例應用程式中） – 服務網域

> **Important**
>
> SimSpace Weaver 每個模擬最多支援 5 個網域。這包括所有空間、自訂和服務網域。

```
domains:
  MyViewDomain:
    launch_apps_via_start_app_call: {}
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MyViewApp.zip"
      launch_command: ["MyViewApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 7000
  MySpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 2
        y: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MySpatialApp.zip"
      launch_command: ["MySpatialApp"]
      required_resource_units:
```

```
        compute: 1
```

> **Note**
>
> SimSpace Weaver 應用程式開發套件 1.12.x 版專案針對應用程式 .zip 檔案和結構描述使用不同的儲存貯體：
>
> - Weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - weaver-*lowercase-project-name-account-number*-schemas-*region*

主題



## 應用程式組態

您可以將應用程式 (app_config) 的組態指定為其網域組態的一部分。所有類型的網域都使用這些相同的應用程式組態屬性。

```
app_config:
  package: "s3://weaver-myproject-111122223333-us-west-2/MyViewApp.zip"
  launch_command: ["MyViewApp"]
  required_resource_units:
    compute: 1
```

> **Note**
>
> SimSpace Weaver 應用程式開發套件 1.12.x 版專案針對應用程式 .zip 檔案和結構描述使用不同的儲存貯體：
>
> - weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - Weaver-*lowercase-project-name-account-number*-schemas-*region*

package 屬性指定 S3 儲存貯體中 zip 檔案的 S3 URI。zip 檔案包含應用程式可執行檔 （也稱為二進位）及其所需的任何其他資源 （例如程式庫）。應用程式可執行檔的每個執行個體都會在工作者的Docker容器中執行。

launch_command 屬性會指定可執行檔的名稱，以及執行應用程式的任何命令列選項。的值launch_command是陣列。整個啟動命令字串的每個字符都是陣列中的 元素。

範例

- 對於啟動命令： `MyTestApp --option1 value1`
- 指定： `launch_command: ["MyTestApp", "-option1", "value1"]`

required_resource_units 屬性指定 SimSpace Weaver 應配置給此應用程式的運算資源單位數量。運算資源單位是工作者(RAM)上固定的處理容量(vCPU)和記憶體數量。您可以增加此值，以增加應用程式在工作者上執行時可用的運算能力。每個工作者的運算資源單位數量有限。如需詳細資訊，請參閱SimSpace Weaver 端點和配額。

### 設定空間網域

對於空間網域，您必須指定 partitioning_strategy。此屬性的值是您為結構描述另一個部分中定義的分割策略指定的名稱。

```
MySpatialDomain:
  launch_apps_by_partitioning_strategy:
    partitioning_strategy: "MyGridPartitioning"
    grid_partition:
      x: 2
      y: 2
  app_config:
    package: "s3://weaver-myproject-111122223333-us-west-2/MySpatialApp.zip"
    launch_command: ["MySpatialApp"]
    required_resource_units:
      compute: 1
```

> **Note**
>
> SimSpace Weaver 應用程式開發套件 1.12.x 版專案針對應用程式 .zip 檔案和結構描述使用不同的儲存貯體：

- weaver-*lowercase-project-name*-*account-number*-app-zips-*region*
- weaver-*lowercase-project-name*-*account-number*-schemas-*region*

具有Grid拓撲（此版本中唯一支援的拓撲）的分割策略 SimSpace Weaver 會指示 在網格中排列此網域的空間應用程式分割區。grid_partition 屬性指定分割區網格的數字列和資料欄。

SimSpace Weaver 將為分割區網格中的每個儲存格啟動 1 個空間應用程式的執行個體。例如，如果空間網域具有grid_partition值 x： 2和 y： 2，空間網域中有 2 * 2 = 4 個分割區。 SimSpace Weaver 將啟動空間網域中設定的應用程式 4 個執行個體，並為每個應用程式執行個體指派 1 個分割區。

主題



空間網域的資源需求

您可以為每個工作者指派最多 17 個運算資源單位。您可以在空間網域的 app_config區段中指定每個空間應用程式使用的運算資源單位數量。

Example 結構描述程式碼片段，顯示空間應用程式的運算資源單位

```
  MySpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 2
        y: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-artifacts-us-west-2/
MySpatialApp.zip"
      launch_command: ["MySpatialApp"]
      required_resource_units:
        compute: 1
```

若要計算網域所需的運算資源單位數量，請將網格中的儲存格數量 （在您的中grid_partition，x*y) 乘以指派給空間應用程式的運算資源單位數量。

對於先前的範例，網域會MySpatialDomain指定：

- x: 2
- y: 2
- compute: 1

的網格MySpatialDomain具有 2 * 2 = 4 個儲存格。空間網域需要 4 * 1 = 4 個運算資源單位。

結構描述中指定之所有網域的運算資源單位總數，必須小於或等於工作者desired數乘以每個工作者的運算資源單位數上限 (17)。

多個空間網域

您可以將模擬設定為使用超過 1 個空間網域。例如，您可以使用 1 個空間網域來控制模擬中的主要演員 （例如人物和汽車），並使用不同的空間網域來控制環境。

您也可以使用多個空間網域，將不同的資源指派給模擬的不同部分。例如，如果您的模擬具有的實體類型比其他類型多 10 倍的實體執行個體，您可以建立不同的網域來處理每個實體類型，並為具有更多實體的網域配置更多資源。

> **Important**
>
> SimSpace Weaver 1.14.0 之前的版本不支援多個空間網域。

> **Important**
>
> AWS SimSpace Weaver Local 目前不支援多個空間網域。如需有關 SimSpace Weaver Local 的詳細資訊，請參閱 中的本機開發 SimSpace Weaver。

> **Important**
>
> SimSpace Weaver 每個模擬最多支援 5 個網域。這包括所有空間、自訂和服務網域。

### 設定多個空間網域

若要設定超過 1 個空間網域，請在結構描述中將其他空間網域定義新增為個別的命名區段。每個網域都必須指定 launch_apps_by_partitioning_strategy金鑰。請參閱下列範例結構描述。

```
sdk_version: "1.14"
workers:
  MyComputeWorkers:
    type: "sim.c5.24xlarge"
    desired: 1
clock:
  tick_rate: "30"
partitioning_strategies:
  MyGridPartitioning:
    topology: Grid
    aabb_bounds:
      x: [0, 1000]
      y: [0, 1000]
domains:
  MySpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 2
        y: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-artifacts-us-west-2/
MySpatialApp.zip"
      launch_command: ["MySpatialApp"]
      required_resource_units:
        compute: 1
  MySecondSpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 2
        y: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-artifacts-us-west-2/
MySpatialApp2.zip"
      launch_command: ["MySpatialApp2"]
      required_resource_units:
        compute: 1
```

### 將空間網域放在一起

在某些情況下，您可能想要將空間網域的分割區放置在工作者上另一個網域的分割區旁。如果這些分割區彼此建立跨網域訂閱，這可以改善效能特性。

將最上層金鑰`placement_constraints`新增至您的結構描述，以指定哪些網域 SimSpace Weaver 應放在一起。必要的`on_workers`金鑰必須參考結構描述中的具名`workers`組態。

Example 結構描述程式碼片段，顯示放在一起的空間網域

```
workers:
  MyComputeWorkers:
    type: "sim.c5.24xlarge"
    desired: 2
placement_constraints:
  - placed_together: ["MySpatialDomain", "MySecondSpatialDomain"]
    on_workers: ["MyComputeWorkers"]
```

**⚠ Important**

- 如果您使用置放群組：
  - 請確定 x * y 是工作者數目的倍數。
  - 確定置放群組值是您放在一起之網域的網格維度的通用除數。
- 如果您不使用置放群組：
  - 請確定空間網域網格的 1 個軸具有等於工作者數量的常見除數。

如需置放群組的詳細資訊，請參閱 分割策略。

### 有關空間網域的常見問題

Q1. (問題 1)：如何將另一個空間網域新增至現有的模擬？

- 針對執行中的模擬 – 您無法變更執行中模擬的組態。變更結構描述中的網域組態、上傳結構描述和應用程式壓縮，然後啟動新的模擬。
- 對於新的模擬 – 將網域組態新增至結構描述、上傳結構描述和應用程式壓縮，然後啟動新的模擬。

### 對空間網域進行故障診斷

當您嘗試啟動模擬但網域組態無效時， 可能會收到下列錯誤。

```
"StartError": "[{\"errorType\":\"SchemaFormatInvalid\",\"errorMessage\":
    \"We were unable to determine an arrangement of your domains that would fit
    within the provided set of workers. This can generally be resolved by
    increasing the number of workers if able, decreasing your domains\u0027
    [\u0027\u0027grid_partition\u0027\u0027] values, or adjusting the
    dimensions of your [\u0027\u0027grid_placement_groups\u0027\u0027].\"}]"
```

#### 可能的原因

- 結構描述為應用程式配置的運算資源單位比工作者上可用的更多。
- SimSpace Weaver 無法判斷將網域放在工作者上的安排。當您指定多個空間網域，但網域網格之間沒有通用的除數或多個，例如 2x4 網格和 3x5 網格之間） 時，就會發生這種情況。

## 網路端點

自訂和服務應用程式可以有外部用戶端可以連線的網路端點。您可以在 `ingress_ports`中指定連接埠號碼清單做為 的值`endpoint_config`。這些連接埠號碼皆為 TCP 和 UDP。自訂 或服務應用程式應該繫結至您在 中指定的連接埠號碼`ingress_ports`。 會在執行時間 SimSpace Weaver 動態配置連接埠號碼，並將這些連接埠映射至動態連接埠。您可以在應用程式開始尋找動態 （實際） 連接埠號碼之後呼叫 describe-app API。如需詳細資訊，請參閱快速入門教學取得自訂應用程式的 IP 地址和連接埠號碼中的 。

```
domains:
  MyViewDomain:
    launch_apps_via_start_app_call: {}
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MyViewApp.zip"
      launch_command: ["MyViewApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 7000
```

> **Note**
>
> SimSpace Weaver 應用程式開發套件 1.12.x 版專案針對應用程式 .zip 檔案和結構描述使用不同的儲存貯體：
>
> - Weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - Weaver-*lowercase-project-name-account-number*-schemas-*region*

> **Note**
>
> `endpoint_config` 是自訂應用程式和服務應用程式的選用屬性。如果您未指定 `endpoint_config`，則應用程式不會有網路端點。

## 設定服務網域

網域組態`launch_apps_per_worker:`中存在 ，表示它是具有服務 app 的服務網域。 會為您 SimSpace Weaver 啟動和停止服務應用程式。當 SimSpace Weaver 啟動和停止應用程式時，應用程式會被視為具有受管生命週期 。 SimSpace Weaver 目前支援在每個工作者上啟動 1 或 2 個服務應用程式。

Example 設定為在每個工作者上啟動 1 個服務應用程式的網域範例

```
domains:
  MyServiceDomain:
    launch_apps_per_worker:
      count: 1
    app_config:
      package: "s3://weaver-myproject-111122223333-app-zips-us-west-2/
PlayerConnectionServiceApp.zip"
      launch_command: ["PlayerConnectionServiceApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 9000
          - 9001
```

Example 設定為在每個工作者上啟動 2 個服務應用程式的網域範例

```
domains:
  MyServiceDomain:
    launch_apps_per_worker:
      count: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-app-zips-us-west-2/
PlayerConnectionServiceApp.zip"
      launch_command: ["PlayerConnectionServiceApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 9000
          - 9001
```

# 模擬的持續時間上限

中的每個模擬 AWS SimSpace Weaver 都有一個最長持續時間設定，指定模擬可以執行的時間上限。當您啟動模擬時，您會提供最長持續時間做為參數。StartSimulation 應用程式程式設計界面 (API) 具有選用參數 MaximumDuration。參數的值為分鐘 (m 或 M)、小時 (h 或 H) 或天數 (d 或 D)。例如， 1h或 1H表示 1 小時。 會在達到此限制時 SimSpace Weaver 停止模擬。

## 最大值

的最高有效值MaximumDuration為 14D，或其對等值，以小時 (336H) 或分鐘 () 為單位20160M。

## 預設值

MaximumDuration 為選用參數。如果您未提供值， SimSpace Weaver 會使用 的值14D。

## 最小值

的最低有效值MaximumDuration是數值相當於 的值0。例如，值 0M、 0H和 0D，在數值上都相當於 0。

如果您提供最長持續時間的最小值，模擬會在達到 STOPPING 狀態後立即轉換為 STARTED 狀態。

## 使用主控台啟動模擬

您可以在SimSpace Weaver 主控台中啟動模擬時，提供最長持續時間的值。當您選擇開始模擬時，請在模擬設定表單的持續時間上限欄位中輸入值。

> **Important**
>
> 如果您未提供最長持續時間的值， SimSpace Weaver 會使用預設值 (14D)。

## 達到其最長持續時間的模擬狀態

當 SimSpace Weaver 自動停止達到其最長持續時間的模擬時，模擬的狀態為 STOPPING（如果進行中）或 STOPPED。在 SimSpace Weaver 主控台中，模擬的目標狀態仍為 STARTED，因為這是使用者請求的最後一個狀態。

# 開發應用程式

SimSpace Weaver 開發需要Amazon Linux 2 (AL2)環境來建置應用程式，因為您的模擬會在 Amazon Linux中於 上執行AWS Cloud。如果您使用的是 Windows，則可以使用 SimSpace Weaver 應用程式 SDK 中的指令碼來建立和啟動容器，該Docker容器AL2執行了您建置 SimSpace Weaver 應用程式所需的相依性。您也可以使用 啟動AL2環境Windows Subsystem for Linux (WSL)，或使用原生AL2系統。如需詳細資訊，請參閱設定 的本機環境 SimSpace Weaver。

> **Note**
>
> 無論您如何設定本機開發環境，當您上傳應用程式以在 中執行時，您的應用程式都會在Docker容器中執行 AWS 雲端。您的應用程式無法直接存取主機作業系統。

SimSpace Weaver 應用程式的一般流程

1. 建立 應用程式。
2. 迴圈：
    a. 透過建立 開始更新`Transaction`。
        - 如果模擬正在關閉，請退出迴圈。

b. 處理訂閱和擁有權實體事件。

c. 更新模擬。

d. 遞交 Transaction以結束更新。

3. 銷毀應用程式。

## 空間應用程式

每個空間應用程式都有一個擁有區域，這是模擬世界的空間區域。位於空間應用程式擁有區域的實體會存放在應用程式指派的分割區中。單一空間應用程式對其指派的分割區中的所有實體擁有完整所有權（讀取和寫入許可）。沒有其他應用程式可以寫入這些實體。空間應用程式會提升其實體的狀態。每個空間應用程式只擁有 1 個分割區。 SimSpace Weaver 會使用實體的空間位置來編製索引，並將其指派給空間應用程式分割區。

SimSpace Weaver 應用程式開發套件提供範例應用程式。您可以在下列資料夾中找到範例應用程式空間應用程式的原始程式碼（使用適用於您作業系統的正確路徑分隔符號）：

```
sdk-folder\Samples\PathfindingSample\src\SpatialApp
```

## 自訂應用程式

您可以建立並使用自訂應用程式來與模擬互動。

自訂應用程式可以

- 建立實體
- 訂閱其他分割區
- 遞交變更

自訂應用程式的一般流程

1. 建立 應用程式。
2. 在模擬中訂閱特定區域：

   a. 建立 Transaction以開始第一次更新。

   b. 為特定區域建立訂閱。

   c. 遞交 Transaction以結束第一次更新。

3. 迴圈：

    a. 建立 Transaction以開始更新。

        - 如果模擬正在關閉，請退出迴圈。

    b. 程序狀態變更。

    c. 遞交 Transaction以結束更新。

4. 銷毀應用程式。

自訂應用程式建立實體之後，必須將實體轉移到空間網域，實體才能在模擬中空間內存在。 SimSpace Weaver 會使用實體的空間位置，將實體放置在適當的空間應用程式分割區中。建立實體的自訂應用程式無法在將實體轉移至空間網域後更新或刪除實體。

SimSpace Weaver 應用程式開發套件提供範例應用程式。您可以使用範例應用程式中包含的自訂應用程式，做為您自己自訂應用程式的模型。您可以在下列資料夾中找到範例應用程式的檢視應用程式（自訂應用程式）的原始碼（使用適用於您作業系統的正確路徑分隔符號）：

```
sdk-folder\Samples\PathfindingSample\src\ViewApp
```

# 開發用戶端應用程式

您可能想要將用戶端連線至模擬的一些原因包括：

- 將即時流量資訊注入城市規模模擬。
- 建立human-in-the-loop模擬，其中人工運算子控制模擬的某些層面。
- 讓使用者能夠與模擬互動，例如用於訓練模擬。

這些範例中的自訂應用程式充當模擬狀態與外部世界之間的界面。用戶端會連線至自訂應用程式，以與模擬互動。

SimSpace Weaver 不會處理用戶端應用程式及其與您的自訂應用程式的通訊。您負責用戶端應用程式的設計、建立、操作和安全性，以及其與自訂應用程式的通訊。 SimSpace Weaver 只會公開每個自訂應用程式的 IP 地址和連接埠號碼，以便用戶端可以與其連線。

SimSpace Weaver 應用程式開發套件提供用戶端用於其範例應用程式。您可以使用這些用戶端做為您自己的用戶端應用程式的模型。您可以在下列資料夾中找到範例應用程式用戶端的原始程式碼：

Docker

```
sdk-folder\packaging-tools\clients\PathfindingSampleClients
```

WSL

> **Important**
>
> 為了您的方便，我們提供這些說明。它們適用於 Windows Subsystem for Linux (WSL)，不支援。如需詳細資訊，請參閱設定 的本機環境 SimSpace Weaver。

```
sdk-folder/packaging-tools/clients/PathfindingSampleClients
```

如需建置和使用範例應用程式用戶端的詳細資訊，請參閱中的教學課程入門 SimSpace Weaver。

# 取得自訂應用程式的 IP 地址和連接埠號碼

若要檢視模擬，您可以建立自訂應用程式，並與用戶端連線。如需詳細資訊，請參閱 中的教學課程入門 SimSpace Weaver。您可以使用下列程序來取得自訂應用程式的 IP 地址和連接埠號碼。為您的作業系統使用適當的路徑分隔符號 （ 例如，\在 Windows 和 Linux /中 ）。

取得您的 IP 地址和連接埠號碼

1. 使用 ListSimulations API 取得模擬的名稱。

   ```
   aws simspaceweaver list-simulations
   ```

   輸出範例：

   ```
   {
       "Simulations": [
           {
               "Status": "STARTED",
               "CreationTime": 1664921418.09,
               "Name": "MyProjectSimulation_22-10-04_22_10_15",
   ```

```
            "Arn": "arn:aws:simspaceweaver:us-west-2: 111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15",
            "TargetStatus": "STARTED"
        }
    ]

}
```

2. 使用 DescribeSimulation API 取得模擬中的網域清單。

```
aws simspaceweaver describe-simulation --simulation simulation-name
```

在輸出的 Domains區段中尋找 LiveSimulationState區段。

輸出範例：

```
    "LiveSimulationState": {
        "Domains": [
            {
                "Type": "",
                "Name": "MySpatialSimulation",
                "Lifecycle": "Unknown"
            },
            {
                "Type": "",
                "Name": "MyViewDomain",
                "Lifecycle": "ByRequest"
            }
        ],
```

3. 使用 ListApps API 取得網域中的自訂應用程式清單。例如，範例專案中檢視（自訂）應用程式的網域名稱為 MyViewDomain。在輸出中尋找應用程式名稱。

```
aws simspaceweaver list-apps --simulation simulation-name --domain domain-name
```

輸出範例：

```
{
    "Apps": [
        {
            "Status": "STARTED",
            "Domain": "MyViewDomain",
            "TargetStatus": "STARTED",
            "Name": "ViewApp",
            "Simulation": "MyProjectSimulation_22-10-04_22_10_15"
        }
    ]
}
```

4. 使用 DescribeApp API 取得 IP 地址和連接埠號碼。對於範例專案，網域名稱為 MyViewDomain，應用程式名稱為 ViewApp。

```
aws simspaceweaver describe-app --simulation simulation-name --domain domain-name
 --app app-name
```

IP 地址和連接埠號碼位於輸出的 EndpointInfo 區塊中。IP 地址是 的值Address，連接埠號碼是 的值Actual。

輸出範例：

```
{
    "Status": "STARTED",
    "Domain": "MyViewDomain",
    "TargetStatus": "STARTED",
    "Simulation": "MyProjectSimulation_22-10-04_22_10_15",
    "LaunchOverrides": {
        "LaunchCommands": []
    },
    "EndpointInfo": {
        "IngressPortMappings": [
            {
                "Declared": 7000,
                "Actual": 4321
            }
        ],
        "Address": "198.51.100.135"
```

```
    },
    "Name": "ViewApp"
}
```

> **Note**
>
> 的值`Declared`是應用程式程式碼應繫結的連接埠號碼。的值`Actual`是向用戶端 SimSpace Weaver 公開以連接到您 app 的連接埠號碼。將`Declared`連接埠 SimSpace Weaver 映射至`Actual`連接埠。

# 啟動 Unreal Engine 檢視用戶端

導覽至：

```
sdk-folder/Samples/PathfindingSample/tools/cloud
```

1. 請執行下列其中一個命令：
   - Docker： `python quick-start.py`
   - WSL： `python quick-start.py --al2`
2. 取得 IP 地址和「實際」連接埠號碼。這些將位於執行 quick-start.py 的主控台輸出中，或依照 中的程序取得取得自訂應用程式的 IP 地址和連接埠號碼。
3. 導覽至：

   ```
   sdk-folder/Clients/TCP/UnrealClient/lib
   ```
4. 執行下列命令來建置 NNG 程式庫：

   ```
   cmake -S . -B build
   cmake --build build --config RelWithDebInfo
   cmake --install build
   ```
5. 在文字編輯器中，開啟 `view_app_url.txt`。
6. 使用檢視應用程式的 IP 地址和連接埠號碼更新 URL： `tcp://ip-address:actual-port-number`（看起來應該像 `tcp://198.51.100.135:1234`）。
7. 在 Unreal 編輯器中，選擇播放。

## 故障診斷

- NNG CMake 安裝步驟失敗，並顯示「可能需要管理權限」：

```
CMake Error at build/_deps/nng-build/src/cmake_install.cmake:39 (file):
  file cannot create directory: C:/Program Files
  (x86)/ThirdPartyNngBuild/lib.  Maybe need administrative privileges.
Call Stack (most recent call first):
  build/_deps/nng-build/cmake_install.cmake:37 (include)
  build/cmake_install.cmake:73 (include)
```

  - 解決方法：如果 UnrealClient/lib 目錄中nng.so存在 nng.lib或 ，則可以安全地忽略此錯誤。如果沒有，請嘗試在具有管理員權限的終端機中執行 cmake 建置命令。
- 「CMake to find a package configuration file provided by nng」：

```
CMake Error at CMakeLists.txt:23 (find_package):
By not providing "Findnng.cmake" in CMAKE_MODULE_PATH this project has
 asked CMake to find a package configuration file provided by "nng", but
 CMake did not find one.
```

  - 解決方法：CMake 在尋找Findnng.cmake檔案時遇到問題。使用 CMake 建置時，請新增引數 -DTHIRD_PARTY_LIB_PATH sdk-folder/ThirdParty。在重新執行 CMake 建置之前，請確定 Findnng.cmake 檔案仍在 ThirdParty目錄中。

```
cmake -S . -B build -DTHIRD_PARTY_LIB_PATH sdk-folder/ThirdParty
cmake --build build --config RelWithDebInfo
cmake --install build
```

# 中的本機開發 SimSpace Weaver

您可以在本機部署 SimSpace Weaver 應用程式，以進行快速測試和偵錯。

要求

- 完成「設定 SimSpace Weaver」中的步驟。

主題

- 步驟 1：啟動本機模擬

- 步驟 2：檢視本機模擬
- 步驟 3：停止本機模擬（在 Windows 上選用）
- 故障診斷 中的本機開發 SimSpace Weaver

## 步驟 1：啟動本機模擬

1. 導覽至

   ```
   cd sdk-folder/Samples/sample-name/tools/local
   ```

2. 執行下列命令，在本機建置和啟動模擬。

   ```
   python quick-start.py
   ```

   此指令碼將執行下列動作：

   1. 建置專案。
      - `quick-start.py` 會呼叫 https：//build.py. 中定義的 `build_project` 函數。此步驟會根據專案而有所不同。對於 PathfindingSample，會使用 CMake。CMake 和 Docker 命令，可在 build.py：//。
   2. 啟動本機模擬
      - 指令碼將針對結構描述中定義的每個空間分割區啟動一個本機程序。
      - 指令碼將為結構描述中定義的每個自訂應用程式啟動一個程序。
      - 空間應用程式會先啟動，接著是自訂應用程式，每個應用程式都會依照顯示在結構描述中的順序。

> **Important**
>
> 在不支援 GUI 的環境中啟動時，例如主控台 SSH 工作階段，請使用 `--noappwindow`選項將所有輸出重新導向至目前的終端機。

> **Important**
>
> 對於 Linux 使用者，指令碼假設您的系統具有 `xterm`命令。如果您的 Linux 發行版本沒有 `xterm`命令，請使用 `--noappwindow`選項將所有輸出重新導向至目前的終端機。

- -h、 -求助
  - 列出這些參數。
- --clean
  - 在建置之前刪除建置目錄的內容。
- --nobuild
  - 略過重建專案。
- --noappwindow
  - 請勿為每個應用程式開啟新視窗。反之，請將 stdout 重新導向至目前的終端機。
- --logfile
  - 將主控台輸出寫入日誌檔案。
- --consoleclient
  - 自動連接組態中列出的主控台用戶端。
- --結構描述 SCHEMA
  - 此調用將使用的結構描述。預設為 https：//config.py 中的「SCHEMA」。

## 步驟 2：檢視本機模擬

若要檢視本機模擬，您可以使用 SimSpaceWeaverAppSdkDistributable 隨附的任何用戶端。如需建置和使用範例用戶端的詳細資訊，請參閱 中的教學課程入門 SimSpace Weaver。

您必須更新用戶端中的 IP 地址和連接埠號碼，才能連線至本機模擬的檢視應用程式。一律搭配 使用下列值SimSpace Weaver Local：

```
tcp://127.0.0.1:7000
```

根據您選取的用戶端，您可以更新 IP 地址和連接埠號碼，如下所示：

- Unreal – 變更第 1 行的 URL `view_app_url.txt`

- 主控台 – 使用 IP 地址和連接埠號碼 URL 做為參數啟動用戶端

## 步驟 3：停止本機模擬 （ 在 Windows 上選用 ）

> **Note**
>
> Linux 需要此步驟，但 Windows 則為選用。

1. 導覽至：

   ```
   sdk-folder/Samples/sample-name/tools/local
   ```

2. 執行下列命令來停止本機模擬並刪除任何共用記憶體資源。

   ```
   python stop-and-delete.py
   ```

   此指令碼將執行下列動作：

   - 停止本機程序。
   - 刪除共用記憶體物件 （ 僅在 Linux 上需要 ）。

stop-and-delete.py 參數

- -h、 -求助
  - 列出這些參數。
- --stop
  - 僅嘗試停止程序。
- --delete
  - 僅嘗試刪除共用記憶體資源。
- -- 程序
  - 要停止的程序名稱。如果您的程序名稱與結構描述中的套件名稱不相符，請使用此選項。
- --結構描述 SCHEMA
  - 此調用將使用的結構描述。預設為 https：//www.config.py 中的 'SCHEMA' 值。

## 故障診斷 中的本機開發 SimSpace Weaver

- Linux：找不到 xterm 命令 / 無法開啟
  - 在 Linux 上執行時，本機指令碼會假設 xterm 命令存在。如果您沒有 xterm 命令或在不支援 GUI 的環境中執行，請在執行快速入門指令碼時使用 `--noappwindow`選項。
- 沒有開啟的應用程式視窗！
  - 當本機模擬立即當機時，就會發生這種情況。若要在當機後查看主控台輸出，請在執行快速入門指令碼時使用 `--noappwindow`或 `--logfile`選項。
- 檢視應用程式啟動或檢視用戶端連線後，模擬不會勾選！
  - 使用 –`noappwindow`選項執行 通常會解決這類問題。否則，重新啟動幾次也會成功 （ 雖然速率較低 ）。

# AWS SimSpace Weaver 應用程式開發套件

SimSpace Weaver 應用程式開發套件提供 APIs，可讓您用來控制模擬中的實體並回應 SimSpace Weaver 事件。它包含下列命名空間：

- API – API 及其使用的核心定義

與下列程式庫連結：

- `libweaver_app_sdk_cxx_v1_full.so`

> **Important**
>
> 當您在 中執行應用程式時，程式庫可用於動態連結 AWS 雲端。您不需要使用應用程式上傳它。

> **Note**
>
> SimSpace Weaver 應用程式開發套件 APIs 會控制模擬中的資料。這些 APIs與 SimSpace Weaver 服務 APIs，可在其中控制您的 SimSpace Weaver 服務資源 （ 例如模擬、應用程式和時鐘 ） AWS。如需詳細資訊，請參閱SimSpace Weaver API 參考。

主題



## API 方法會傳回 Result

大多數 SimSpace Weaver API 函數都有傳回類型 `Aws::WeaverRuntime::Result<T>`。如果函數已成功執行，則 `Result`包含 `T`。否則， `Result`包含`Aws::WeaverRuntime::ErrorCode`代表來自 錯誤碼的 Rust App SDK。

Example 範例

```
Result<Transaction> BeginUpdate(Application& app)
```

此方法：

- `Transaction` 如果 成功`BeginUpdate()`執行，則傳回 。
- `Aws::WeaverRuntime::ErrorCode` 如果`BeginUpdate()`失敗，傳回 。

## 在頂層與應用程式 SDK 互動

生命週期

- SimSpace Weaver 應用程式開發套件會管理應用程式生命週期。您不需要讀取或寫入應用程式的生命週期狀態。

資料分割

- 使用 `Result <PartitionSet> AssignedPartitions(Transaction& txn);`取得擁有的分割區。
- 使用 `Result <PartitionSet> AllPartitions(Transaction& txn);`取得模擬中的所有分割區。

## 模擬管理

本節說明常見模擬管理任務的解決方案。

主題

- 開始模擬
- 更新模擬
- 終止模擬

### 開始模擬

使用 `CreateApplication()`建立應用程式。

Example 範例

```
Result<Application> applicationResult = Api::CreateApplication();

if (!applicationResult)
{
    ErrorCode errorCode = WEAVERRUNTIME_EXPECT_ERROR(applicationResult);

    std::cout << "Failed to create application. Error code " <<
        static_cast<std::underlying_type_t<ErrorCode>>(errorCode) <<
        " Last error message "<< Api::LastErrorMessage() << ".";

    return 1;
}

/**
* Run simulation
*/
RunSimulation(std::move(applicationResult.assume_value()));
```

## 更新模擬

使用下列BeginUpdate函數更新應用程式：

- `Result<Transaction> BeginUpdate(Application& app)`
- `Result<bool> BeginUpdateWillBlock(Application& app)` – 告訴您 `BeginUpdate()`是否會封鎖。

使用 `Result<void> Commit(Transaction& txn)`遞交變更。

Example 範例

```
Result<void> AppDriver::RunSimulation(Api::Application app) noexcept
{
    while (true)
    {
        {
            bool willBlock;

            do
            {
                WEAVERRUNTIME_TRY(willBlock, Api::BeginUpdateWillBlock(m_app));
            } while (willBlock);
        }

        WEAVERRUNTIME_TRY(Transaction transaction, Api::BeginUpdate(app));

        /**
         * Simulate app.
         */
        WEAVERRUNTIME_TRY(Simulate(transaction));
        WEAVERRUNTIME_TRY(Api::Commit(std::move(transaction)));
    }

    return Success();
}
```

## 終止模擬

使用 `Result<void> DestroyApplication(Application&& app)`來終止應用程式和模擬。

其他應用程式發現，從ErrorCode::ShuttingDown對 BeginUpdateWillBlock()或 的呼叫接收到模擬時，模擬正在關閉BeginUpdate()。當應用程式收到 時ErrorCode::ShuttingDown，它可以呼叫 Result<void> DestroyApplication(Application&& app) 來終止自己。

Example 範例

```
Result<void> AppDriver::EncounteredAppError(Application&& application) noexcept
{
    const ErrorCode errorCode = WEAVERRUNTIME_EXPECT_ERROR(runAppResult);

    switch (errorCode)
    {
    case ErrorCode::ShuttingDown:
        {
            // insert custom shutdown process here.

            WEAVERRUNTIME_TRY(Api::DestroyApplication(std::move(application)));
            return Success();
        }
    default:
        {
            OnAppError(errorCode);
            return errorCode;
        }
    }
}
```

**Important**

僅在 Result<void> DestroyApplication(Application&& app)之後呼叫 Api::Commit()。在更新期間銷毀應用程式可能會導致未定義的行為。

**Important**

您必須在程式結束DestroyApplication()之前呼叫 ，以確保應用程式回報為成功終止。當程式結束DestroyApplication()時，無法呼叫 將導致狀態被視為 FATAL。

# 訂閱

您可以使用訂閱區域和網域 ID 建立訂閱。網域 ID 代表擁有該訂閱區域的網域。BoundingBox2F32 描述訂閱區域。使用下列函數建立訂閱：

```
Result<SubscriptionHandle> CreateSubscriptionBoundingBox2F32(Transaction& txn, DomainId
 id, const BoundingBox2F32& boundingBox)
```

Example 範例

```
Result<void> CreateSubscriptionInSpatialDomain(Transaction& transaction)
{
    WEAVERRUNTIME_TRY(Api::PartitionSet partitionSet, Api::AllPartitions(transaction));


    Api::DomainId spatialDomainId;

    for (const Api::Partition& partition : partitionSet.partitions)
    {
        if (partition.domain_type == Api::DomainType::Spatial)
        {
            /**
            * Get the spatial domain ID.
            */
            spatialDomainId = partition.domain_id;
            break;
        }
    }

    constexpr Api::BoundingBox2F32 subscriptionBounds {
        /* min */ { /* x */ 0, /* y */ 0 },
        /* max */ { /* x */ 1000, /* y */ 1000 } }

    WEAVERRUNTIME_TRY(
        Api::SubscriptionHandle subscriptionHandle,
        Api::CreateSubscriptionBoundingBox2F32(
        transaction,
        spatialDomainId,
        subscriptionBounds));

    return Success();
}
```

您可以使用 Api::SubscriptionHandle傳回的 CreateSubscriptionBoundingBox2F32()來修改訂閱。您可以將它做為引數傳遞給下列函數：

```
Result<void> ModifySubscriptionBoundingBox2F32(Transaction& txn, SubscriptionHandle
 handle, const BoundingBox2F32& boundingBox)
```

```
Result<void> DeleteSubscription(Transaction& txn, SubscriptionHandle handle)
```

## 實體

您可以使用從 Result<Api::Entity>傳回Api:Entity的 的 CreateEntity()，或實體進入應用程式的訂閱區域時，從所有權變更事件呼叫 Store和 Load APIs （如需詳細資訊，請參閱 實體事件)。我們建議您追蹤Api::Entity物件，以便使用這些 APIs。

主題

- 建立實體
- 將實體轉移到空間網域
- 寫入和讀取實體欄位資料
- 儲存實體的位置
- 載入實體的位置

### 建立實體

使用 CreateEntity()建立實體。您可以定義Api::TypeId您傳遞至此函數的 意義。

```
Namespace
{
    constexpr Api::TypeId k_entityTypeId { /* value */ 512 };
}

Result<void> CreateEntity(Transaction& transaction)
{
    WEAVERRUNTIME_TRY(
        Api::Entity entity,
        Api::CreateEntity(
            transaction, Api::BuiltinTypeIdToTypeId(k_entityTypeId )));
}
```

> **Note**
>
> Api::BuiltinTypeId 為 保留 0-511 的值。您的實體 TypeID(k_entityTypeId在此範例中為 ) 必須具有 512 或更高的值。

## 將實體轉移到空間網域

自訂應用程式或服務應用程式建立實體後，應用程式必須將實體轉移到空間網域，實體才能在模擬中空間上存在。空間網域中的實體可由其他應用程式讀取，並由空間應用程式更新。使用 ModifyEntityDomain( ) API 將實體轉移到空間網域。

```
AWS_WEAVERRUNTIME_API Result<void> ModifyEntityDomain(Transaction& txn, const Entity&
 entity, DomainId domainId) noexcept;
```

如果 DomainId 不符合呼叫應用程式指派Partition的 DomainType::Spatial ，則 DomainId 必須是 。 Domain所有權轉移至新的 會在 期間Domain發生Commit(Transaction&&)。

參數

txn

目前的 Transaction。

entity

變更 Entity的目標Domain。

domainId

DomainId Domain 目的地的 Entity。

Success 如果實體網域已成功變更，此 API 會傳回 。

## 寫入和讀取實體欄位資料

所有實體資料欄位都是 Blob 類型。您最多可以將 1，024 個位元組的資料寫入實體。我們建議您盡可能減少 Blob，因為較大的大小會降低效能。當您寫入 Blob 時，您會將 SimSpace Weaver 指標傳遞至資料及其長度。當您從 Blob 讀取時， SimSpace Weaver 會為您提供要讀取的指標和長度。在應用程式呼叫 之前，所有讀取都必須完成Commit( )。當應用程式呼叫 時，從讀取呼叫傳回的指標會失效Commit( )。

> **Important**
>
> - 在Commit()不支援 之後從快取 Blob 指標讀取，可能會導致模擬失敗。
> - 不支援寫入讀取呼叫傳回的 Blob 指標，並可能導致模擬失敗。

主題



### 儲存實體的欄位資料

下列範例示範如何存放 （寫入狀態結構） 應用程式所擁有實體的欄位資料。這些範例使用以下函數：

```
AWS_WEAVERRUNTIME_API Result<void> StoreEntityField(
    Transaction& txn,
    const Entity& entity,
    TypeId keyTypeId,
    FieldIndex index,
    std::int8_t* src,
    std::size_t length) noexcept;
```

Api::TypeId keyTypeId 參數代表傳入資料的 資料類型。

Api::TypeId keyTypeId 參數應該Api::TypeId會從 接收對應的 Api::BuiltinTypeId。如果沒有適當的轉換，您可以使用 Api::BuiltinTypeId::Dynamic。

對於複雜的資料類型，請使用 Api::BuiltInTypeId::Dynamic。

> **Note**
>
> 的值FieldIndex index必須大於 0。值 0 保留給索引鍵 （請參閱 StoreEntityIndexKey()）。

Example 使用基本資料類型的範例

```
namespace
```

```
{
    constexpr Api::FieldIndex k_isTrueFieldId { /* value */ 1 };
}

Result<void> SetEntityFields(
    Api::Entity& entity,
    Transaction& transaction)
{
    bool value = true;

    auto* src = reinterpret_cast<std::int8_t*>(value);
    size_t length = sizeof(*value);

    WEAVERRUNTIME_TRY(Api::StoreEntityField(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Bool),
        k_isTrueFieldId,
        src,
        length));
}
```

Example 使用 struct保留資料的範例

```
namespace
{
    constexpr Api::FieldIndex k_dataFieldId { /* value */ 1 };
}

struct Data
{
    bool boolData;
    float floatData;
};

Result<void> SetEntityFields(
    Api::Entity& entity,
    Transaction& transaction)
{
    Data data = { /* boolData */ false, /* floatData */ -25.93 };

    auto* src = reinterpret_cast<std::int8_t*>(data);
```

```
    size_t length = sizeof(*data);

    WEAVERRUNTIME_TRY(Api::StoreEntityField(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Dynamic),
        k_dataFieldId,
        src,
        length));
}
```

### 載入實體的欄位資料

下列範例示範如何載入 實體的欄位資料 （從狀態結構讀取）。這些範例使用以下函數：

```
Result<std::size_t> LoadEntityField(
    Transaction& txn,
    const Entity& entity,
    TypeId keyTypeId,
    FieldIndex index,
    std::int8_t** dest) noexcept;
```

`Api::TypeId keyTypeId` 參數應該`Api::TypeId`會從 接收對應的 `Api::BuiltinTypeId`。如果沒有適當的轉換，您可以使用 `Api::BuiltinTypeId::Dynamic`。

> **Note**
>
> `FieldIndex` 索引的值必須大於 0。值 0 保留給索引鍵 （請參閱 `StoreEntityIndexKey()`）。

#### Example 使用基本資料類型的範例

```
namespace
{
    constexpr Api::FieldIndex k_isTrueFieldId { /* value */ 1 };
}

Result<void> LoadEntityFields(
    Api::Entity& entity,
    Transaction& transaction)
```

```
{
    std::int8_t* dest = nullptr;

    WEAVERRUNTIME_TRY(Api::LoadEntityField(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Bool),
        k_isTrueFieldId,
        &dest));

    bool isTrueValue = *reinterpret_cast<bool*>(dest);
}
```

### Example 使用 struct保留資料的範例

```
namespace
{
    constexpr Api::FieldIndex k_dataFieldId { /* value */ 1 };
}

struct Data
{
    bool boolData;
    float floatData;
};

Result<void> LoadEntityFields(
    Api::Entity& entity,
    Transaction& transaction)
{
    std::int8_t* dest = nullptr;

    WEAVERRUNTIME_TRY(Api::LoadEntityField(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Dynamic),
        k_dataFieldId,
        &dest));

    Data dataValue = *reinterpret_cast<Data*>(dest);
}
```

### 載入已移除實體的欄位資料

您無法載入（從狀態結構讀取）實體欄位資料，用於已從應用程式的擁有權和訂閱區域移除的實體。下列範例會導致 錯誤，因為它因為 而Api::LoadIndexKey()呼叫實體Api::ChangeListAction::Remove。第二個範例顯示直接在應用程式中存放和載入實體資料的正確方式。

Example 不正確程式碼的範例

```
Result<void> ProcessSubscriptionChanges(Transaction& transaction)
{
    /* ... */

    WEAVERRUNTIME_TRY(Api::SubscriptionChangeList subscriptionChangeList,
        Api::AllSubscriptionEvents(transaction));

    for (const Api::SubscriptionEvent& event :
        subscriptionChangeList.changes)
    {
        switch (event.action)
        {
        case Api::ChangeListAction::Remove:
            {
                std::int8_t* dest = nullptr;

                /**
                 * Error!
                 * This calls LoadEntityIndexKey on an entity that
                 * has been removed from the subscription area.
                 */
                WEAVERRUNTIME_TRY(Api::LoadEntityIndexKey(
                    transaction,
                    event.entity,
                    Api::BuiltinTypeIdToTypeId(
                        Api::BuiltinTypeId::Vector3F32),
                    &dest));

                AZ::Vector3 position =
                    *reinterpret_cast<AZ::Vector3*>(dest);
                break;
            }
        }
```

```
    }

    /* ... */
}
```

Example 在應用程式中存放和載入實體資料的正確方式範例

```
Result<void> ReadAndSaveSubscribedEntityPositions(Transaction& transaction)
{
    static std::unordered_map<Api::EntityId, AZ::Vector3>
        positionsBySubscribedEntity;

    WEAVERRUNTIME_TRY(Api::SubscriptionChangeList subscriptionChangeList,
        Api::AllSubscriptionEvents(transaction));

    for (const Api::SubscriptionEvent& event :
        subscriptionChangeList.changes)
    {
        switch (event.action)
        {
        case Api::ChangeListAction::Add:
            {
                std::int8_t* dest = nullptr;

                /**
                 * Add the position when the entity is added.
                 */
                WEAVERRUNTIME_TRY(Api::LoadEntityIndexKey(
                    transaction,
                    event.entity,
                    Api::BuiltinTypeIdToTypeId(
                        Api::BuiltinTypeId::Vector3F32),
                    &dest));

                AZ::Vector3 position =
                    *reinterpret_cast<AZ::Vector3*>(dest);
                positionsBySubscribedEntity.emplace(
                    event.entity.descriptor->id, position);

                break;
            }
        case Api::ChangeListAction::Update:
            {
```

```
                std::int8_t* dest = nullptr;

                /**
                 * Update the position when the entity is updated.
                 */
                WEAVERRUNTIME_TRY(Api::LoadEntityIndexKey(
                    transaction,
                    event.entity,
                    Api::BuiltinTypeIdToTypeId(
                        Api::BuiltinTypeId::Vector3F32),
                    &dest));

                AZ::Vector3 position =
                    *reinterpret_cast<AZ::Vector3*>(dest);
                positionsBySubscribedEntity[event.entity.descriptor->id] =
                    position;

                break;
            }
        case Api::ChangeListAction::Remove:
            {
                /**
                 * Load the position when the entity is removed.
                 */
                AZ::Vector3 position = positionsBySubscribedEntity[
                    event.entity.descriptor->id];

                /**
                 * Do something with position...
                 */
                break;
            }
        }
    }

    /* ... */
}
```

## 儲存實體的位置

您可以使用整數資料結構來存放（寫入狀態結構）實體的位置。這些範例使用以下函數：

```
Result<void> StoreEntityIndexKey(
```

```
    Transaction& txn,
    const Entity& entity,
    TypeId keyTypeId,
    std::int8_t* src,
    std::size_t length)
```

> **Note**
>
> 您必須提供 `Api::BuiltinTypeId::Vector3F32` 給 `Api::StoreEntityIndexKey()`，如下列範例所示。

Example 使用陣列代表位置的範例

```
Result<void> SetEntityPositionByFloatArray(
    Api::Entity& entity,
    Transaction& transaction)
{
    std::array<float, 3> position = { /* x */ 25, /* y */ 21, /* z */ 0 };

    auto* src = reinterpret_cast<std::int8_t*>(position.data());
    std::size_t length = sizeof(position);

    WEAVERRUNTIME_TRY(Api::StoreEntityIndexKey(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(Api::BuiltinTypeId::Vector3F32),
        src,
        length));
}
```

Example 使用 struct代表位置的範例

```
struct Position
{
   float x;
   float y;
   float z;
};
```

```
Result<void> SetEntityPositionByStruct(
    Api::Entity& entity,
    Transaction& transaction)
{
    Position position = { /* x */ 25, /* y */ 21, /* z */ 0 };

    auto* src = reinterpret_cast<std::int8_t*>(&position);
    std::size_t length = sizeof(position);

    WEAVERRUNTIME_TRY(Api::StoreEntityIndexKey(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(Api::BuiltinTypeId::Vector3F32),
        src,
        length));
}
```

## 載入實體的位置

您可以使用整數資料結構載入（從狀態結構讀取）實體的位置。這些範例使用以下函數：

> **Note**
>
> 您必須提供 `Api::BuiltinTypeId::Vector3F32` 給 `Api::LoadEntityIndexKey()`，如下列範例所示。

### Example 使用陣列代表位置的範例

```
Result<void> GetEntityPosition(Api::Entity& entity,
    Transaction& transaction)
{
    std::int8_t* dest = nullptr;

    WEAVERRUNTIME_TRY(Aws::WeaverRuntime::Api::LoadEntityIndexKey(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Vector3F32),
        &dest));

    std::array<float, 3> position =
```

```
        *reinterpret_cast<std::array<float, 3>*>(dest);
}
```

Example 使用 struct代表位置的範例

```
struct Position
{struct
   float x;
   float y;
   float z;
};

Result<void> GetEntityPosition(Api::Entity& entity, Transaction& transaction)
{
    std::int8_t* dest = nullptr;

    WEAVERRUNTIME_TRY(Aws::WeaverRuntime::Api::LoadEntityIndexKey(
        transaction,
        entity,
        Api::BuiltinTypeIdToTypeId(
            Aws::WeaverRuntime::Api::BuiltinTypeId::Vector3F32),
        &dest));

    Position position = *reinterpret_cast<Position*>(dest);
}
```

## 實體事件

您可以在 SimSpace Weaver 應用程式 SDK 中使用下列函數，以取得所有擁有權和訂閱事件：

- `Result<OwnershipChangeList> OwnershipChanges(Transaction& txn)`
- `Result<SubscriptionChangeList> AllSubscriptionEvents(Transaction& txn)`

如果您需要回呼驅動實體事件處理，您可以使用 SimSpace Weaver 示範架構。如需詳細資訊，請參閱下列標頭檔案：

- *sdk-folder*`/packaging-tools/samples/ext/DemoFramework/include/DemoFramework/EntityEventProcessor.h`

您也可以建立自己的實體事件處理。

主題



## 逐一查看擁有實體的事件

使用 OwnershipChanges()取得擁有實體 （應用程式擁有區域中的實體） 的事件清單。函數具有下列簽章：

```
Result<OwnershipChangeList> OwnershipChanges(Transaction& txn)
```

然後，使用迴圈逐一查看實體，如下列範例所示。

Example 範例

```
WEAVERRUNTIME_TRY(Result<Api::OwnershipChangeList> ownershipChangesResult,
 Api::OwnershipChanges(transaction));

for (const Api::OwnershipChange& event : ownershipChangeList.changes)
{
    Api::Entity entity = event.entity;
    Api::ChangeListAction action = event.action;

    switch (action)
    {
    case Api::ChangeListAction::None:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Remove:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Add:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Update:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Reject:
        // insert code to handle the event
```

```
        break;
    }
}
```

事件類型

- None – 實體位於 區域，且其位置和欄位資料未修改。
- Remove – 實體已從 區域移除。
- Add – 實體已新增至 區域。
- Update – 實體位於 區域並已修改。
- Reject – 應用程式無法從 區域移除實體。

> **Note**
>
> 如果發生Reject事件，應用程式將在下次刻度再次嘗試傳輸。

## 逐一查看已訂閱實體的事件

使用 AllSubscriptionEvents()取得已訂閱實體 （ 應用程式訂閱區域中的實體 ） 的事件清單。函數具有下列簽章：

```
Result<SubscriptionChangeList> AllSubscriptionEvents(Transaction& txn)
```

然後，使用迴圈逐一查看實體，如下列範例所示。

Example 範例

```
WEAVERRUNTIME_TRY(Api::SubscriptionChangeList subscriptionChangeList,
 Api::AllSubscriptionEvents(transaction));

for (const Api::SubscriptionEvent& event : subscriptionChangeList.changes)
{
    Api::Entity entity = event.entity;
    Api::ChangeListAction action = event.action;

    switch (action)
    {
    case Api::ChangeListAction::None:
```

```
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Remove:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Add:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Update:
        // insert code to handle the event
        break;
    case Api::ChangeListAction::Reject:
        // insert code to handle the event
        break;
    }
}
```

事件類型

- None – 實體位於 區域，且其位置和欄位資料未修改。
- Remove – 實體已從 區域移除。
- Add – 實體已新增至 區域。
- Update – 實體位於 區域並已修改。
- Reject – 應用程式無法從 區域移除實體。

> **Note**
>
> 如果發生Reject事件，應用程式將在下次刻度再次嘗試傳輸。

## 逐一查看實體的所有權變更事件

若要取得實體在擁有權區域和訂閱區域之間移動的事件，請比較目前和先前實體擁有權與訂閱事件之間的變更。

您可以透過閱讀來處理這些事件：

- Api::SubscriptionChangeList
- Api::OwnershipEvents

然後，您可以將變更與先前儲存的資料進行比較。

下列範例顯示如何處理實體所有權變更事件。此範例假設，對於在已訂閱實體和擁有實體之間轉換的實體（任一方向），所有權移除/新增事件會先發生，接著是下一個刻度中的訂閱移除/新增事件。

Example 範例

```
Result<void> ProcessOwnershipEvents(Transaction& transaction)
{
    using EntityIdsByAction =
        std::unordered_map<Api::ChangeListAction,
        std::vector<Api::EntityId>>;
    using EntityIdSetByAction =
        std::unordered_map<Api::ChangeListAction,
        std::unordered_set<Api::EntityId>>;

    static EntityIdsByAction m_entityIdsByPreviousOwnershipAction;

    EntityIdSetByAction entityIdSetByAction;

    /**
     * Enumerate Api::SubscriptionChangeList items
     * and store Add and Remove events.
     */
    WEAVERRUNTIME_TRY(Api::SubscriptionChangeList subscriptionEvents,
        Api::AllSubscriptionEvents(transaction));

    for (const Api::SubscriptionEvent& event : subscriptionEvents.changes)
    {
        const Api::ChangeListAction action = event.action;

        switch (action)
        {
        case Api::ChangeListAction::Add:
        case Api::ChangeListAction::Remove:

            {
                entityIdSetByAction[action].insert(
                    event.entity.descriptor->id);
                break;
            }
        case Api::ChangeListAction::None:
        case Api::ChangeListAction::Update:
        case Api::ChangeListAction::Reject:
```

```
        {
            break;
        }
    }
}

EntityIdsByAction entityIdsByAction;

/**
 * Enumerate Api::OwnershipChangeList items
 * and store Add and Remove events.
 */

WEAVERRUNTIME_TRY(Api::OwnershipChangeList ownershipChangeList,
    Api::OwnershipChanges(transaction));

for (const Api::OwnershipChange& event : ownershipChangeList.changes)
{
    const Api::ChangeListAction action = event.action;

    switch (action)
    {
    case Api::ChangeListAction::Add:
    case Api::ChangeListAction::Remove:
        {
            entityIdsByAction[action].push_back(
                event.entity.descriptor->id);
            break;
        }
    case Api::ChangeListAction::None:
    case Api::ChangeListAction::Update:
    case Api::ChangeListAction::Reject:
        {
            break;
        }
    }

}

std::vector<Api::EntityId> fromSubscribedToOwnedEntities;
std::vector<Api::EntityId> fromOwnedToSubscribedEntities;

/**
 * Enumerate the *previous* Api::OwnershipChangeList Remove items
```

```
     * and check if they are now in
     * the *current* Api::SubscriptionChangeList Add items.
     *
     * If true, then that means
     * OnEntityOwnershipChanged(bool isOwned = false)
     */
    for (const Api::EntityId& id : m_entityIdsByPreviousOwnershipAction[
        Api::ChangeListAction::Remove])
    {
        if (entityIdSetBySubscriptionAction[
            Api::ChangeListAction::Add].find(id) !=
                entityIdSetBySubscriptionAction[
                Api::ChangeListAction::Add].end())
        {
            fromOwnedToSubscribedEntities.push_back(id);
        }
    }


    /**
     * Enumerate the *previous* Api::OwnershipChangeList Add items
     * and check if they are now in
     * the *current* Api::SubscriptionChangeList Remove items.
     *
     * If true, then that means
     * OnEntityOwnershipChanged(bool isOwned = true)
     */
    for (const Api::EntityId& id : m_entityIdsByPreviousOwnershipAction[
        Api::ChangeListAction::Add])
    {
        if (entityIdSetBySubscriptionAction[
            Api::ChangeListAction::Remove].find(id) !=

                entityIdSetBySubscriptionAction[
                Api::ChangeListAction::Remove].end())
        {
            fromSubscribedToOwnedEntities.push_back(id);
        }
    }

    m_entityIdsByPreviousOwnershipAction = entityIdsByOwnershipAction;

    return Success();
```

```
}
```

## Result 和錯誤處理

Aws::WeaverRuntime::Result<T> 類別使用第三方Outcome程式庫。您可以使用下列模式來檢查 API 呼叫傳回的 Result和 擷取錯誤。

```
void DoBeginUpdate(Application& app)
{
    Result<Transaction> transactionResult = Api::BeginUpdate(app);

    if (transactionResult)
    {
        Transaction transaction =
            std::move(transactionResult).assume_value();

        /**
         * Do things with transaction ...
         */
    }
    else
    {
        ErrorCode errorCode = WEAVERRUNTIME_EXPECT_ERROR(transactionResult);
        /**
         * Macro compiles to:
         * ErrorCode errorCode = transactionResult.assume_error();
         */
    }
}
```

### Result 控制陳述式巨集

在傳回類型為 的函數內Aws::WeaverRuntime::Result<T>，您可以使用WEAVERRUNTIME_TRY巨集，而不是先前的程式碼模式。巨集將執行傳遞給它的函數。如果傳遞的函數失敗，巨集會讓封裝函數傳回錯誤。如果傳遞的函數成功，則執行會繼續進行到下一行。下列範例顯示先前DoBeginUpdate()函數的重寫。此版本使用WEAVERRUNTIME_TRY巨集，而非if-else控制項結構。請注意，函數的傳回類型為 Aws::WeaverRuntime::Result<void>。

```
Aws::WeaverRuntime::Result<void> DoBeginUpdate(Application& app)
{
    /**
     * Execute Api::BeginUpdate()
```

```
     * and return from DoBeginUpdate() if BeginUpdate() fails.
     * The error is available as part of the Result.
     */
    WEAVERRUNTIME_TRY(Transaction transaction, Api::BeginUpdate(m_app));

    /**
     * Api::BeginUpdate executed successfully.
     *
     * Do things here.
     */

    return Aws::Success();
}
```

如果BeginUpdate()失敗，則巨集會提早DoBeginUpdate()傳回失敗。您可以使用 WEAVERRUNTIME_EXPECT_ERROR巨集從 取得 Aws::WeaverRuntime::ErrorCode BeginUpdate()。下列範例顯示 Update()函數如何在失敗時呼叫 DoBeginUpdate() 並取得錯誤代碼。

```
void Update(Application& app)
{
    Result<void> doBeginUpdateResult = DoBeginUpdate(app);

    if (doBeginUpdateResult)
    {
        /**
         * Successful.
         */
    }
    else
    {
        /**
         * Get the error from Api::BeginUpdate().
         */
        ErrorCode errorCode = WEAVERRUNTIME_EXPECT_ERROR(doBeginUpdateResult);

    }
}
```

您可以將 的傳回類型變更為 ，Update()將 的錯誤碼BeginUpdate()提供給呼叫 Update() 的函數Aws::WeaverRuntime::Result<void>。您可以重複此程序，讓錯誤碼繼續沿著呼叫堆疊傳送。

## 一般和網域類型

SimSpace Weaver 應用程式 SDK 提供單精度資料類型 `Api::Vector2F32`和 `Api::BoundingBox2F32`，以及雙精度 `Api::Vector2F64`和 `Api::BoundingBox2F64`。這些資料類型是沒有便利方法的被動資料結構。請注意，API 僅使用 `Api::Vector2F32`和 `Api::BoundingBox2F32`。您可以使用這些資料類型來建立和修改訂閱。

SimSpace Weaver 示範架構提供最小版本的AzCore數學程式庫，其中包含 `Vector3`和 `Aabb` 。如需詳細資訊，請參閱 中的標頭檔案：

- *sdk-folder*/packaging-tools/samples/ext/DemoFramework/include/AzCore/Math

## 其他應用程式 SDK 操作

主題

- AllSubscriptionEvents 和 OwnershipChanges包含上次呼叫的事件
- 處理後釋放讀取鎖定 SubscriptionChangeList
- 建立獨立的應用程式執行個體進行測試

### AllSubscriptionEvents 和 OwnershipChanges包含上次呼叫的事件

傳回對 的呼叫值，`Api::AllSubscriptionEvents()`並`Api::OwnershipChanges()`包含來自上次呼叫的事件，而非上次刻度。在下列範例中， `secondSubscriptionEvents`和 `secondOwnershipChangeList`是空的，因為其函數會在第一次呼叫後立即呼叫。

如果您等待 10 個刻度，然後呼叫 `Api::AllSubscriptionEvents()`和 `Api::OwnershipChanges()`，則其結果將同時包含事件和過去 10 個刻度 （ 而不是最後一個刻度 ）的變更。

Example 範例

```
Result<void> ProcessOwnershipChanges(Transaction& transaction)
{
    WEAVERRUNTIME_TRY(
        Api::SubscriptionChangeList firstSubscriptionEvents,
        Api::AllSubscriptionEvents(transaction));
    WEAVERRUNTIME_TRY(
```

```
        Api::OwnershipChangeList firstOwnershipChangeList,
        Api::OwnershipChanges(transaction));

    WEAVERRUNTIME_TRY(
        Api::SubscriptionChangeList secondSubscriptionEvents,
        Api::AllSubscriptionEvents(transaction));
    WEAVERRUNTIME_TRY(
        Api::OwnershipChangeList secondOwnershipChangeList,
        Api::OwnershipChanges(transaction));

    /**
     * secondSubscriptionEvents and secondOwnershipChangeList are
     * both empty because there are no changes since the last call.
     */
}
```

> **Note**
>
> 函數AllSubscriptionEvents()已實作，但函數SubscriptionEvents()未實作。

## 處理後釋放讀取鎖定 SubscriptionChangeList

當您開始更新時，在先前刻度的其他分割區中有遞交資料的共用記憶體區段。這些共用記憶體區段可能會被讀取器鎖定。應用程式無法完全遞交，直到所有讀取器都釋放鎖定為止。作為最佳化，應用程式應該在處理Api::Subscriptionchangelist項目後呼叫 Api::ReleaseReadLeases()以釋出鎖定。這可減少遞交時間的爭用。 預設會Api::Commit()釋出讀取租用，但最佳實務是在處理訂閱更新後手動釋出。

### Example 範例

```
Result<void> ProcessSubscriptionChanges(Transaction& transaction)
{
    WEAVERRUNTIME_TRY(ProcessSubscriptionChanges(transaction));

    /**
     * Done processing Api::SubscriptionChangeList items.
     * Release read locks.
     */

    WEAVERRUNTIME_EXPECT(Api::ReleaseReadLeases(transaction));
```

```
    ...
}
```

### 建立獨立的應用程式執行個體進行測試

您可以使用 在實際模擬中執行程式碼之前，Api::CreateStandaloneApplication()建立獨立的應用程式來測試應用程式邏輯。

Example 範例

```
int main(int argc, char* argv[])
{
    Api::StandaloneRuntimeConfig config = {
        /* run_for_seconds (the lifetime of the app) */ 3,
        /* tick_hertz (the app clock rate) */ 10 };

    Result<Application> applicationResult =
        Api::CreateStandaloneApplication(config);

    ...
}
```

# AWS SimSpace Weaver 示範架構

AWS SimSpace Weaver 示範架構 （ 示範架構 ） 是公用程式程式庫，可用來開發 SimSpace Weaver 應用程式。

示範架構提供

- 程式碼範例和程式設計模式供您使用和檢查
- 簡化簡單應用程式開發的抽象和公用程式函數
- 測試 SimSpace Weaver 應用程式 SDK 實驗性功能的簡單方法

我們設計 SimSpace Weaver 了應用程式 SDK 搭配低階 SimSpace Weaver APIs 存取，以提供更高的效能。相反地，我們設計了示範架構，以提供更高階的抽象和 API 存取，讓 SimSpace Weaver APIs 更容易使用。與直接使用 SimSpace Weaver 應用程式 SDK 相比，易用性的成本較低。可以容忍較低效能的模擬 （ 例如沒有即時效能需求的模擬 ） 可能是使用示範架構的理想候選者。我們建議您將

SimSpace Weaver 應用程式 SDK 中的原生功能用於複雜的應用程式，因為示範架構不是完整的工具組。

示範架構包含

- 支援並示範以下項目的工作程式碼範例：
  - 應用程式流程管理
  - 回呼驅動實體事件處理
- 一組第三方公用程式程式庫：
  - spdlog （記錄程式庫）
  - (AZCore數學程式庫）的最小版本，僅包含：
    - Vector3
    - Aabb
  - cxxopts （命令列選項剖析器程式庫）
- 專屬於 的公用程式函數 SimSpace Weaver

示範架構包含程式庫、來源檔案和 CMakeLists。檔案包含在 SimSpace Weaver 應用程式 SDK 可分發套件中。

# 使用服務配額

本節說明如何使用 的服務配額 SimSpace Weaver。配額也稱為限制。如需服務配額清單，請參閱 SimSpace Weaver 端點和配額。本節中的 APIs 來自一組應用程式 APIs。應用程式 APIs與服務 APIs 不同。應用程式 APIs 是 SimSpace Weaver 應用程式 SDK 的一部分。您可以在本機系統的應用程式 SDK 資料夾中找到應用程式 APIs 的文件：

```
sdk-folder\SimSpaceWeaverAppSdk-sdk-version\documentation\index.html
```

主題

- 取得應用程式的限制
- 取得應用程式使用的資源量
- 重設指標
- 超過限制
- 記憶體不足

- 最佳實務

## 取得應用程式的限制

您可以使用RuntimeLimits應用程式 API 來查詢應用程式的限制。

```
Result<Limit> RuntimeLimit(Application& app, LimitType type)
```

參數

Application& 應用程式

應用程式參考。

LimitType 類型

具有下列限制類型的列舉：

```
enum LimitType {
    Unset = 0,
    EntitiesPerPartition = 1,
    RemoteEntityTransfers = 2,
    LocalEntityTransfers = 3
};
```

下列範例會查詢實體計數限制。

```
WEAVERRUNTIME_TRY(auto entity_limit,
    Api::RuntimeLimit(m_app, Api::LimitType::EntitiesPerPartition))
Log::Info("Entity count limit", entity_limit.value);
```

## 取得應用程式使用的資源量

您可以呼叫RuntimeMetrics應用程式 API 來取得應用程式使用的資源量：

```
Result<std::reference_wrapper<const AppRuntimeMetrics>> RuntimeMetrics(Application&
 app) noexcept
```

參數

Application& 應用程式

應用程式參考。

API 會傳回struct包含指標的 參考。計數器指標會保留執行中的總值，且只會增加。計量指標會保留可增加或減少的值。應用程式執行時間會在事件增加值時更新計數器。執行時間只會在您呼叫 API 時更新量測值。 SimSpace Weaver 保證參考在應用程式的生命週期內有效。對 API 重複呼叫不會變更參考。

```
struct AppRuntimeMetrics {
    uint64_t total_committed_ticks_gauge,

    uint32_t active_entity_gauge,
    uint32_t ticks_since_reset_counter,

    uint32_t load_field_counter,
    uint32_t store_field_counter,

    uint32_t created_entity_counter,
    uint32_t deleted_entity_counter,

    uint32_t entered_entity_counter,
    uint32_t exited_entity_counter,

    uint32_t rejected_incoming_transfer_counter,
    uint32_t rejected_outgoing_transfer_counter
}
```

## 重設指標

ResetRuntimeMetrics 應用程式 API 會重設 AppRuntimeMetrics 中的值struct。

```
Result<void> ResetRuntimeMetrics(Application& app) noexcept
```

下列範例示範如何在ResetRuntimeMetrics應用程式中呼叫 。

```
if (ticks_since_last_report > 100)
{
    auto metrics = WEAVERRUNTIME_EXPECT(Api::RuntimeMetrics(m_app));
```

```
    Log::Info(metrics);

    ticks_since_last_report = 0;

    WEAVERRUNTIME_EXPECT(Api::ResetRuntimeMetrics(m_app));
}
```

## 超過限制

超過限制的應用程式 API 呼叫將傳回 `ErrorCode::CapacityExceeded`，但實體傳輸除外。SimSpace Weaver 會在遞交和BeginUpdate應用程式 API 操作中以非同步方式處理實體傳輸，因此，如果傳輸因實體傳輸限制而失敗，則不會傳回錯誤的特定操作。若要偵測傳輸失敗，您可以將 `rejected_incoming_transfer_counter`和 `rejected_outgoing_transfer_counter`（在 `AppRuntimeMetrics` 中struct) 的目前值與其先前的值進行比較。拒絕的實體不會在分割區中，但應用程式仍然可以模擬它們。

## 記憶體不足

SimSpace Weaver 使用垃圾收集器程序來清理和釋放釋放的記憶體。寫入資料的速度可能比垃圾收集器釋放記憶體的速度快。如果發生這種情況，寫入操作可能會超過應用程式的預留記憶體限制。SimSpace Weaver 會傳回內部錯誤，其中包含訊息 `OutOfMemory`（和其他詳細資訊）。如需詳細資訊，請參閱將寫入分散到不同時間。

## 最佳實務

下列最佳實務是設計應用程式以避免超過限制的一般準則。它們可能不適用於您的特定應用程式設計。

### 經常監控並放慢速度

您應該經常監控指標，並減慢接近限制的操作。

### 避免超過訂閱限制和轉移限制

如果可能，請設計您的模擬，以減少遠端訂閱和實體傳輸的數量。您可以使用置放群組在同一個工作者上放置多個分割區，並減少工作者之間遠端實體傳輸的需求。

### 將寫入分散到不同時間

刻度中的更新數量和大小可能會對遞交交易所需的時間和記憶體產生重大影響。記憶體需求過大可能會導致應用程式執行期用盡記憶體。您可以將寫入分散到不同時間，以降低每個刻度的平均更新總大小。

這有助於改善效能並避免超過限制。建議您不要在每個刻度上寫入超過 12 MB 的平均值，或在每個實體上寫入超過 1.5 KB 的平均值。

# 偵錯模擬

您可以使用下列方法來取得模擬的相關資訊。

主題



## 使用SimSpace Weaver Local並查看主控台輸出

建議您先在本機開發模擬，然後在 中執行它們 AWS 雲端。當您使用 執行 時，可以直接檢視主控台輸出SimSpace Weaver Local。如需詳細資訊，請參閱中的本機開發 SimSpace Weaver。

## 查看 Amazon CloudWatch Logs 中的日誌

當您在應用程式的主控台輸出中執行模擬 AWS 雲端 時， 會傳送至 Amazon CloudWatch Logs 中的日誌串流。您的模擬也會寫入其他日誌資料。如果您希望模擬寫入日誌資料，則必須在模擬結構描述中啟用記錄。如需詳細資訊，請參閱SimSpace Weaver Amazon CloudWatch Logs 中的日誌。

> **Warning**
>
> 您的模擬可以產生大量日誌資料。日誌資料可以快速成長。您應該密切監看日誌，並在不再需要它們時停止模擬。記錄可能會產生大量成本。

## 使用 describe API 呼叫

您可以使用下列服務 APIs在 中取得模擬的相關資訊 AWS 雲端。

- ListSimulations – 取得 中所有模擬的清單 AWS 雲端。

Example 範例

```
aws simspaceweaver list-simulations
```

- DescribeSimulation – 取得模擬的詳細資訊。

  Example 範例

  ```
  aws simspaceweaver describe-simulation --simulation MySimulation
  ```

- DescribeApp – 取得應用程式的詳細資訊。

  Example 範例

  ```
  aws simspaceweaver describe-app --simulation MySimulation --domain MyCustomDomain --
  app MyCustomApp
  ```

如需 SimSpace Weaver APIs的詳細資訊，請參閱 SimSpace Weaver API 參考。

## 連接用戶端

您可以將用戶端連線至執行中的自訂或服務應用程式，而您在模擬結構描述endpoint_config中使用定義。 SimSpace Weaver 應用程式 SDK 包含範例用戶端，您可以用來檢視範例應用程式。您可以查看這些範例用戶端的原始程式碼和範例應用程式，了解如何建立自己的用戶端。如需如何建置和執行範例用戶端的詳細資訊，請參閱 中的教學課程入門 SimSpace Weaver。

您可以在下列資料夾中找到範例用戶端的原始程式碼：

- *sdk-folder*\packaging-tools\clients\PathfindingSampleClients\

## 偵錯本機模擬

您可以使用 偵錯您的SimSpace Weaver Local應用程式Microsoft Visual Studio。如需如何使用 偵錯的詳細資訊Visual Studio，請參閱 Microsoft Visual Studio documentation。

偵錯本機模擬

1. 請確定您的 schema.yaml 位於您的工作目錄中。

2. 在 中Visual Studio，開啟您要偵錯的每個應用程式的內容選單（例如 `PathfindingSampleLocalSpatial`或 `PathfindingSampleLocalView`），並在偵錯區段中設定工作目錄。
3. 開啟您要偵錯之應用程式的內容選單，然後選取設定為啟動專案。
4. 選擇 F5以開始偵錯應用程式。

偵錯模擬的要求與正常執行模擬的要求相同。您必須開始結構描述中指定的空間應用程式數量。例如，如果您的結構描述指定了 2x2 網格，而且您在偵錯模式下啟動空間應用程式，則在您啟動另外 3 個空間應用程式（在偵錯模式下或不在偵錯模式下）之前，模擬不會執行。

若要偵錯自訂應用程式，您必須先啟動空間應用程式，然後在偵錯工具中啟動自訂應用程式。

請注意，您的模擬會在鎖定步驟中執行。一旦應用程式達到中斷點，所有其他應用程式都會暫停。從該中斷點繼續之後，其他應用程式將會繼續。

# 自訂容器

AWS SimSpace Weaver 應用程式會在容器化 Amazon Linux 2(AL2) 環境中執行。在 中 AWS 雲端， 會在從 Amazon Elastic Container Registry (Amazon ECR) 提供的`amazonlinux:2`映像建置的 Docker 容器中 SimSpace Weaver 執行模擬。您可以建立自訂 Docker 映像檔，將其存放在 Amazon ECR 中，並將該映像檔用於模擬，而不是我們提供的預設 Docker 映像檔。

您可以使用自訂容器來管理您的軟體相依性，並包含不在標準 Docker 映像中的其他軟體元件。例如，您可以將應用程式使用的公開可用的軟體程式庫新增至容器，並且只將自訂程式碼放入應用程式 zip 檔案中。

> **Important**
>
> 我們僅支援 Amazon ECR 儲存庫中託管的 AL2 Docker 映像，無論是在 Amazon ECR Public Gallery 或您的私有 Amazon ECR 登錄檔中。我們不支援在 Amazon ECR 外部託管的 Docker 映像。如需 Amazon ECR 的詳細資訊，請參閱 Amazon Elastic Container Registry 文件。

主題

- 關於自訂容器的常見問答集
- 對自訂容器進行故障診斷

## 建立自訂容器

這些指示假設您知道如何使用 Docker 和 Amazon Elastic Container Registry (Amazon ECR)。如需 Amazon ECR 的詳細資訊，請參閱《Amazon ECR 使用者指南》。

先決條件

- 您用來執行這些動作的 IAM 身分 （ 使用或角色 ） 具有使用 Amazon ECR 的正確許可
- Docker 安裝在您的本機系統上

建立自訂容器

1. 建立 `Dockerfile`。

   `Dockerfile` 執行 AWS SimSpace Weaver 應用程式的 會從 Amazon ECR 中的Amazon Linux 2 映像開始。

   ```
   # parent image required to run AWS SimSpace Weaver apps
   FROM public.ecr.aws/amazonlinux/amazonlinux:2
   ```

2. 建置您的 `Dockerfile`。
3. 將容器映像上傳至 Amazon ECR。
   - 使用 AWS Management Console。
   - 使用 AWS Command Line Interface。

   > **Note**
   >
   > 如果您在嘗試將容器映像上傳至 Amazon ECR `AccessDeniedException`時發生錯誤，您的 IAM 身分 （ 使用者或角色 ） 可能沒有使用 Amazon ECR 的必要許可。您可以將 `AmazonEC2ContainerRegistryPowerUser` AWS 受管政策連接至 IAM 身分，然後再試一次。如需如何連接政策的詳細資訊，請參閱AWS Identity and Access Management 《 使用者指南》中的新增和移除 IAM 身分許可。

# 修改專案以使用自訂容器

這些指示假設您已知道如何使用 ， AWS SimSpace Weaver 並希望讓您的應用程式儲存和開發工作流程更有效率 AWS 雲端 。

先決條件

- 您在 Amazon Elastic Container Registry (Amazon ECR) 中有自訂容器。如需建立自訂容器的詳細資訊，請參閱 建立自訂容器。

修改專案以使用自訂容器

1. 將許可新增至專案的模擬應用程式角色，以使用 Amazon ECR。

   a. 如果您還沒有具有下列許可的 IAM 政策，請建立政策。我們建議使用政策名稱 `simspaceweaver-ecr`。如需如何建立 IAM 政策的詳細資訊，請參閱AWS Identity and Access Management 《 使用者指南》中的建立 IAM 政策。

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "Statement",
               "Effect": "Allow",
               "Action": [
                   "ecr:BatchGetImage",
                   "ecr:GetDownloadUrlForLayer",
                   "ecr:GetAuthorizationToken"
               ],
               "Resource": "*"
           }
       ]
   }
   ```

   b. 尋找專案模擬應用程式角色的名稱：

      i. 在文字編輯器中，開啟 AWS CloudFormation 範本：

      ```
      sdk-folder\PackagingTools\sample-stack-template.yaml
      ```

      ii. 在 下尋找 `RoleName` 屬性`WeaverAppRole`。值是專案模擬應用程式角色的名稱。

Example

```
AWSTemplateFormatVersion: "2010-09-09"
Resources:
  WeaverAppRole:
    Type: 'AWS::IAM::Role'
    Properties:
      RoleName: 'weaver-MySimulation-app-role'
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: Allow
          Principal:
            Service:
              - 'simspaceweaver.amazonaws.com'
```

c. 將simspaceweaver-ecr政策連接至專案的模擬應用程式角色。如需如何連接政策的詳細資訊，請參閱AWS Identity and Access Management 《 使用者指南》中的新增和移除 IAM 身分許可。

d. 導覽至 *sdk-folder*並執行下列命令來更新範例 SimSpace Weaver 堆疊：

```
python setup.py --cloudformation
```

2. 在專案的模擬結構描述中指定容器映像。

- 您可以在 下新增選用default_image屬性simulation_properties，以為所有網域指定預設自訂容器映像。
- 針對您要使用自訂容器映像app_config的網域，在 中新增 image 屬性。指定 Amazon ECR 儲存庫 URI 做為值。您可以為每個網域指定不同的映像。
  - 如果 image 未針對網域指定 且default_image已指定 ，則該網域中的應用程式會使用預設映像。
  - 如果 image 未指定網域，default_image且未指定 ，則該網域中的應用程式會在標準 SimSpace Weaver 容器中執行。

Example 包含自訂容器設定的結構描述程式碼片段

```
sdk_version: "1.17.0"
simulation_properties:
```

```
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
  default_image: "111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-
repository:latest" # image to use if no image specified for a domain
domains:
  MyCustomDomain:
    launch_apps_via_start_app_call: {}
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MyViewApp.zip"
      launch_command: ["MyViewApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 7000
      image: "111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-
repository:latest" # custom container image to use for this domain
  MySpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 2
        y: 2
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MySpatialApp.zip"
      launch_command: ["MySpatialApp"]
      required_resource_units:
        compute: 1
      image: "111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-
repository:latest" # custom container image to use for this domain
```

3. 照常建置和上傳您的專案。

## 關於自訂容器的常見問答集

### Q1. (問題 1)：如果我想要變更容器的內容，該怎麼辦？

- 針對執行中的模擬 – 您無法變更執行中模擬的容器。您必須建立新的容器，並啟動使用該容器的新模擬。
- 對於新的模擬 – 建立新的容器，將其上傳至 Amazon Elastic Container Registry (Amazon ECR)，並啟動使用該容器的新模擬。

### Q2. (問題 2): 如何變更模擬的容器映像？

- 針對執行中的模擬 – 您無法變更執行中模擬的容器。您必須啟動使用新容器的新模擬。
- 對於新的模擬 – 在專案的模擬結構描述中指定新的容器映像。如需詳細資訊，請參閱修改專案以使用自訂容器。

## 對自訂容器進行故障診斷

主題

- 將映像上傳至 Amazon Elastic Container Registry (Amazon ECR) 時的 AccessDeniedException
- 使用自訂容器的模擬無法啟動

### 將映像上傳至 Amazon Elastic Container Registry (Amazon ECR) 時的 AccessDeniedException

如果您在嘗試將容器映像上傳至 Amazon ECR `AccessDeniedException`時發生錯誤，您的 IAM 身分（使用者或角色）可能沒有使用 Amazon ECR 的必要許可。您可以將 `AmazonEC2ContainerRegistryPowerUser` AWS 受管政策連接至 IAM 身分，然後再試一次。如需如何連接政策的詳細資訊，請參閱AWS Identity and Access Management 《使用者指南》中的新增和移除 IAM 身分許可。

### 使用自訂容器的模擬無法啟動

對秘訣進行故障診斷

- 如果您的模擬已啟用記錄，請檢查您的錯誤日誌。
- 在沒有自訂容器的情況下測試模擬。
- 在本機測試模擬。如需詳細資訊，請參閱中的本機開發 SimSpace Weaver。

# 使用 Python

您可以針對 SimSpace Weaver 應用程式和用戶端使用 Python。Python 軟體開發套件 (Python SDK) 包含在標準 SimSpace Weaver 應用程式開發套件可分發套件中。使用 Python 開發的運作方式與其他支援語言的開發類似。

> **Important**
>
> SimSpace Weaver 僅支援 Python 3.9 版。

> **Important**
>
> SimSpace Weaver 支援 Python 需要 1.15.0 SimSpace Weaver 版或更新版本。

主題



# 建立 Python 專案

## Python 自訂容器

若要在 中執行 Python 型 SimSpace Weaver 模擬 AWS 雲端，您可以建立包含必要相依性的自訂容器。如需詳細資訊，請參閱自訂容器。

Python 自訂容器必須包含下列項目：

- gcc
- openssl-devel
- bzip2-devel
- libffi-devel
- wget
- tar
- gzip
- make
- Python (3.9 版)

如果您使用 PythonBubblesSample 範本來建立專案，您可以執行quick-start.py指令碼（位於專案的 tools 資料夾）來建立具有必要相依性的 Docker 映像。指令碼會將映像上傳至 Amazon Elastic Container Registry (Amazon ECR)。

quick-start.py 指令碼使用以下 Dockerfile：

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2
RUN yum -y install gcc openssl-devel bzip2-devel libffi-devel
RUN yum -y install wget
RUN yum -y install tar
RUN yum -y install gzip
RUN yum -y install make
WORKDIR /opt
RUN wget https://www.python.org/ftp/python/3.9.0/Python-3.9.0.tgz
RUN tar xzf Python-3.9.0.tgz
WORKDIR /opt/Python-3.9.0
RUN ./configure --enable-optimizations
RUN make altinstall
COPY requirements.txt ./
RUN python3.9 -m pip install --upgrade pip
RUN pip3.9 install -r requirements.txt
```

您可以將自己的相依性新增至 Dockerfile：

```
RUN yum -y install dependency-name
```

requirements.txt 檔案包含PythonBubblesSample範例模擬所需的 Python 套件清單：

```
Flask==2.1.1
```

您可以將自己的 Python 套件相依性新增至 requirements.txt：

```
package-name==version-number
```

Dockerfile 和 requirements.txt位於專案的 tools資料夾中。

> **Important**
>
> 在技術上，您不需要搭配 Python 模擬使用自訂容器，但強烈建議您使用自訂容器。我們提供的標準 Amazon Linux 2 (AL2) 容器沒有 Python。因此，如果您不使用具有 Python 的自

訂容器，則必須在上傳到的每個應用程式 zip 檔案中包含 Python 和所需的相依性 SimSpace Weaver。

## 啟動 Python 模擬

您可以在 中SimSpace Weaver Local和 SimSpace Weaver 中，以與一般模擬相同的方式啟動 Python 型 SimSpace Weaver 模擬 AWS 雲端。如需詳細資訊，請參閱 中的教學課程入門 SimSpace Weaver。

PythonBubblesSample 包含自己的 Python 範例用戶端。如需詳細資訊，請參閱範例 Python 用戶端。

## 範例 Python 用戶端

如果您使用 PythonBubblesSample 範本建立專案，則您的專案會包含 Python 範例用戶端。您可以使用範例用戶端來檢視PythonBubblesSample模擬。您也可以使用範例用戶端做為起點來建立自己的 Python 用戶端。

下列程序假設您已建立PythonBubblesSample專案並啟動其模擬。

啟動 Python 用戶端

1. 在命令提示字元視窗中，前往PyBubbleClient範例專案資料夾。

   ```
   cd sdk-folder\Clients\HTTP\PyBubbleClient
   ```

2. 執行 Python 用戶端。

   ```
   python tkinter_client.py --host ip-address --port port-number
   ```

   參數

   host

   模擬的 IP 地址。對於在 中啟動的模擬 AWS 雲端，您可以在 SimSpace Weaver 主控台中找到模擬的 IP 地址，或使用快速入門教學課程中取得自訂應用程式的 IP 地址和連接埠號碼的 中的程序。對於本機模擬，請使用 127.0.0.1做為 IP 地址。

port

模擬的連接埠號碼。對於在 中啟動的模擬 AWS 雲端，這是Actual連接埠號碼。您可以在 SimSpace Weaver 主控台中找到模擬的連接埠號碼，或使用快速入門教學課程中取得自訂應用程式的 IP 地址和連接埠號碼的程序。對於本機模擬，請使用 7000做為連接埠號碼。

simsize

要在用戶端中顯示的實體數量上限。

## 有關使用 Python 的常見問題

### Q1. (問題 1)：支援哪些版本的 Python？

SimSpace Weaver 僅支援 Python 3.9 版。

## 故障診斷 Python 相關問題

主題

- 自訂容器建立期間失敗
- 您的 Python 模擬無法啟動
- Python 模擬或檢視用戶端擲回 ModuleNotFound 錯誤

### 自訂容器建立期間失敗

如果您在執行no basic auth credentials後收到錯誤quick-start.py，則 Amazon ECR 的臨時登入資料可能會出現問題。使用您的 AWS 區域 ID 和 AWS 帳戶號碼執行下列命令：

```
aws ecr get-login-password --region region | docker login --username AWS --password-
stdin account_id.dkr.ecr.region.amazonaws.com
```

Example

```
aws ecr get-login-password --region us-west-2 | docker login --username AWS --password-
stdin 111122223333.dkr.ecr.region.amazonaws.com
```

> **Important**
>
> 請確定 AWS 區域 您指定的 與您用於模擬的相同。使用 AWS 區域 SimSpace Weaver 支援的其中一個 。如需詳細資訊，請參閱SimSpace Weaver 端點和配額。

執行 `aws ecr`命令後，請`quick-start.py`再次執行 。

要檢查的其他疑難排解資源

- 對自訂容器進行故障診斷
- 《Amazon ECR 使用者指南》中的 Amazon ECR 疑難排解
- 《Amazon ECR 使用者指南》中的使用 Amazon ECR 設定

### 您的 Python 模擬無法啟動

您可能會在模擬的管理日誌中看到`Unable to start app`錯誤。如果您的自訂容器建立失敗，可能會發生這種情況。如需詳細資訊，請參閱自訂容器建立期間失敗。如需日誌的詳細資訊，請參閱SimSpace Weaver Amazon CloudWatch Logs 中的日誌。

如果您確定您的容器沒有任何問題，請檢查應用程式的 Python 原始程式碼。您可以使用 SimSpace Weaver Local 來測試您的應用程式。如需詳細資訊，請參閱中的本機開發 SimSpace Weaver。

### Python 模擬或檢視用戶端擲回 ModuleNotFound 錯誤

當找不到所需的 Python 套件時，Python 會擲回`ModuleNotFound`錯誤。

如果您的模擬位於 中 AWS 雲端，請確定您的自訂容器具有 中列出的所有必要相依性`requirements.txt`。如果您編輯 ，請記得`quick-start.py`再次執行 `requirements.txt`。

如果`PythonBubblesSample`用戶端發生錯誤，請使用 `pip` 安裝指定的套件：

```
pip install package-name==version-number
```

# 支援其他引擎

您可以搭配 使用自己的自訂C++引擎 SimSpace Weaver。我們目前正在開發下列引擎的支援。這些引擎各有不同的文件。

> **Important**
>
> 與此處列出的引擎整合是實驗性的。它們可供預覽。

引擎

- Unity（最低版本為 2022.3.19.F1）
- Unreal Engine（最低版本 5.0）

## Unity

您必須先安裝Unity開發環境，才能使用 Unity 建置 SimSpace Weaver 模擬。如需詳細資訊，請參閱不同的指示：

```
sdk-folder\Unity-Guide.pdf
```

## Unreal Engine

您必須從原始碼建置Unreal Engine專用伺服器。SimSpaceWeaverAppSdkDistributable 包含適用於 的 PathfindingSample 版本Unreal Engine。如需詳細資訊，請參閱不同的指示：

```
sdk-folder\Unreal-Engine-Guide.pdf
```

# 搭配 使用授權軟體 AWS SimSpace Weaver

AWS SimSpace Weaver 可讓您使用您選擇的模擬引擎和內容來建置模擬。在使用 時 SimSpace Weaver，您需負責取得、維護和遵守您在模擬中使用的任何軟體或內容的授權條款。驗證您的授權合約可讓您在虛擬託管環境中部署軟體和內容。

# 使用 管理您的 資源 AWS CloudFormation

您可以使用 AWS CloudFormation 來管理您的 AWS SimSpace Weaver 資源。 AWS CloudFormation 是一項獨立的 AWS 服務，可協助您以程式碼形式指定、佈建和管理 AWS 基礎設施。 AWS CloudFormation 使用 建立 JSON 或 YAML 檔案，稱為範本。您的範本會指定基礎設施的詳細資訊。AWS CloudFormation 使用您的範本將基礎設施佈建為單一單位，稱為堆疊。刪除堆疊時，您可以同時 AWS CloudFormation 刪除堆疊中的所有項目。您可以使用標準原始碼管理程序來管理範本（例

如，在 Git 等版本控制系統中追蹤範本）。如需 的詳細資訊 AWS CloudFormation，請參閱AWS CloudFormation 《使用者指南》。

您的模擬資源

在 中 AWS，資源是您可以使用的實體。範例包括 Amazon EC2 執行個體、Amazon S3 儲存貯體或 IAM 角色。您的 SimSpace Weaver 模擬是 資源。在組態中，您通常會以 的形式指定 AWS 資源`AWS::service::resource`。對於 SimSpace Weaver，您可以將模擬資源指定為 `AWS::SimSpaceWeaver::Simulation`。如需 中模擬資源的詳細資訊 AWS CloudFormation，請參閱AWS CloudFormation 《使用者指南》中的SimSpace Weaver一節。

如何 AWS CloudFormation 搭配 使用 SimSpace Weaver？

您可以建立 AWS CloudFormation 範本，指定您要佈建 AWS 的資源。您的範本可以指定整個架構、架構的一部分，或小型解決方案。例如，您可以為 SimSpace Weaver 解決方案指定 架構，其中包含 Amazon S3 儲存貯體、IAM 許可、Amazon Relational Database Service 或 Amazon DynamoDB 中的支援資料庫，以及您的 `Simulation` 資源。然後，您可以使用 AWS CloudFormation 將所有這些資源佈建為單位，並同時佈建。

Example 範本，可建立 IAM 資源並啟動模擬

下列範例範本會建立 IAM 角色和許可， SimSpace Weaver 這些角色和許可需要在您的帳戶中執行動作。 SimSpace Weaver 應用程式開發套件指令碼會在您建立專案 AWS 區域 時，於特定 中建立角色和許可，但您可以使用 AWS CloudFormation 範本將模擬部署至另一個 AWS 區域 ，而無需再次執行指令碼。例如，您可以這樣做來設定備份模擬，以用於災難復原。

在此範例中，原始模擬名稱為 `MySimulation`。結構描述的儲存貯體已存在於 AWS 區域 中，其中 AWS CloudFormation 將建置堆疊。儲存貯體包含已正確設定在其中執行模擬的結構描述版本 AWS 區域。請記住，結構描述會指定應用程式 zip 檔案的位置，這是與模擬相同 AWS 區域 中的 Amazon S3 儲存貯體。當 AWS CloudFormation 建置堆疊 AWS 區域 時，應用程式 zip 儲存貯體和檔案必須已存在於 中，否則模擬不會啟動。請注意，此範例中的儲存貯體名稱包含 AWS 區域，但這無法判斷儲存貯體的實際位置。您必須確定儲存貯體實際位於 AWS 區域 （您可以在 Amazon S3 主控台、使用 Amazon S3 APIs 或 中的 Amazon S3 命令來檢查儲存貯體屬性 AWS CLI)。

此範例使用 中的一些內建函數和參數 AWS CloudFormation 來執行變數替換。如需詳細資訊，請參閱AWS CloudFormation 《使用者指南》中的內部函數參考和虛擬參數參考。

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  WeaverAppRole:
```

```
    Type: AWS::IAM::Role
    Properties:
      RoleName: SimSpaceWeaverAppRole
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal:
            Service:
              - simspaceweaver.amazonaws.com
          Action:
            - sts:AssumeRole
      Path: /
      Policies:
        - PolicyName: SimSpaceWeaverAppRolePolicy
          PolicyDocument:
            Version: 2012-10-17
            Statement:
            - Effect: Allow
              Action:
                - logs:PutLogEvents
                - logs:DescribeLogGroups
                - logs:DescribeLogStreams
                - logs:CreateLogGroup
                - logs:CreateLogStream
              Resource: *
            - Effect: Allow
              Action:
                - cloudwatch:PutMetricData
              Resource: *
            - Effect: Allow
              Action:
                - s3:ListBucket
                - s3:PutObject
                - s3:GetObject
              Resource: *
  MyBackupSimulation:
    Type: AWS::SimSpaceWeaver::Simulation
    Properties:
      Name: !Sub 'mySimulation-${AWS::Region}'
      RoleArn: !GetAtt WeaverAppRole.Arn
      SchemaS3Location:
        BucketName: !Sub 'weaver-mySimulation-${AWS::AccountId}-schemas-${AWS::Region}'
```

```
        ObjectKey: !Sub 'schema/mySimulation-${AWS::Region}-schema.yaml'
```

## 搭配 使用快照 AWS CloudFormation

快照是模擬的備份。下列範例會從快照而非結構描述啟動新的模擬。此範例中的快照是從 SimSpace Weaver 應用程式 SDK 專案 simulation 建立。 會 AWS CloudFormation 建立新的模擬資源，並使用快照中的資料來初始化它。新的模擬可以具有`MaximumDuration`與原始模擬不同的 。

我們建議您製作並使用原始模擬應用程式角色的副本。如果您刪除模擬的堆疊，則可以刪除原始模擬 AWS CloudFormation 的應用程式角色。

```
Description: "Example - Start a simulation from a snapshot"
Resources:
  MyTestSimulation:
    Type: "AWS::SimSpaceWeaver::Simulation"
    Properties:
      MaximumDuration: "2D"
      Name: "MyTestSimulation_from_snapshot"
      RoleArn: "arn:aws:iam::111122223333:role/weaver-MyTestSimulation-app-role-copy"

      SnapshotS3Location:
        BucketName: "weaver-mytestsimulation-111122223333-artifacts-us-west-2"
        ObjectKey: "snapshot/MyTestSimulation_22-12-15_12_00_00-230428-1207-13.zip"
```

# 快照

您可以隨時建立快照來備份模擬實體資料。 會在 Amazon S3 儲存貯體中 SimSpace Weaver 建立 .zip 檔案。您可以使用快照建立新的模擬。使用快照中存放的實體資料 SimSpace Weaver 初始化新模擬的狀態結構、啟動建立快照時正在執行的空間和服務應用程式，並將時鐘設定為適當的 tick。從快照而非結構描述檔案 SimSpace Weaver 取得模擬的組態。您的應用程式 .zip 檔案必須位於 Amazon S3 中與原始模擬相同的位置。您必須分別啟動任何自訂應用程式。

主題

- 快照的使用案例
- 使用 SimSpace Weaver 主控台來使用快照
- 使用 AWS CLI 處理快照

- 搭配 使用快照 AWS CloudFormation
- 有關快照的常見問答集

# 快照的使用案例

## 返回先前的狀態並探索分支案例

您可以建立模擬的快照，將其儲存為特定狀態。然後，您可以從該快照建立多個新的模擬，並探索可從該狀態分支的不同案例。

## 災難復原和安全性最佳實務

我們建議您定期備份模擬，特別是對於執行超過 1 小時或使用多個工作者的模擬。備份可協助您從災難和安全事件中復原。快照可讓您備份模擬。快照需要您的應用程式 .zip 檔案與之前位於 Amazon S3 的相同位置。如果您需要能夠將應用程式 .zip 檔案移至另一個位置，則必須使用自訂備份解決方案。

如需其他最佳實務的詳細資訊，請參閱 使用 時的最佳實務 SimSpace Weaver和 的安全最佳實務 SimSpace Weaver。

## 延長模擬的持續時間

您的模擬資源是模擬中的表示法 SimSpace Weaver。所有模擬資源都有一個`MaximumDuration`設定。模擬資源到達其 時會自動停止`MaximumDuration`。的最大值`MaximumDuration`為 14D(14 天）。

如果您需要模擬的持續時間超過其模擬資源`MaximumDuration`的 ，您可以在模擬資源到達其之前建立快照`MaximumDuration`。您可以使用快照啟動新的模擬 （建立新的模擬資源）。 會從快照 SimSpace Weaver 初始化實體資料、啟動之前執行的相同空間和服務應用程式，以及還原時鐘。您可以啟動自訂應用程式，並執行任何其他自訂初始化。您可以在啟動時，將新模擬資源`MaximumDuration`的 設定為不同的值。

# 使用 SimSpace Weaver 主控台來使用快照

您可以使用 SimSpace Weaver 主控台來建立模擬的快照。

主題

- 使用 主控台建立快照
- 使用主控台從快照啟動模擬

## 使用 主控台建立快照

建立快照

1. 登入 AWS Management Console 並連線至 SimSpace Weaver 主控台。
2. 從導覽窗格中選擇模擬。
3. 選取模擬名稱旁的選項按鈕。必須啟動模擬的狀態。
4. 在頁面頂端，選擇建立快照。
5. 在快照設定下，針對快照目的地，輸入 SimSpace Weaver 您要建立快照之儲存貯體或儲存貯體和資料夾的 Amazon S3 URI。如果您想要瀏覽可用的儲存貯體並選取位置，可以選擇瀏覽 S3。

   > **Important**
   >
   > Amazon S3 儲存貯體必須與 AWS 區域 模擬位於相同的 中。

   > **Note**
   >
   > SimSpace Weaver 在您選取的快照目的地內建立`snapshot`資料夾。 會在該`snapshot`資料夾中 SimSpace Weaver 建立快照 .zip 檔案。

6. 選擇建立快照。

## 使用主控台從快照啟動模擬

若要從快照啟動模擬，您的快照 .zip 檔案必須存在於模擬可存取的 Amazon S3 儲存貯體中。您的模擬使用您在啟動模擬時所選取之應用程式角色中定義的許可。來自原始模擬的所有應用程式 .zip 檔案都必須存在於與建立快照時相同的位置。

從快照啟動模擬

1. 登入 AWS Management Console 並連線至 SimSpace Weaver 主控台。
2. 從導覽窗格中選擇模擬。
3. 在頁面頂端，選擇開始模擬。
4. 在模擬設定下，輸入模擬的名稱和選用描述。您的模擬名稱在 中必須是唯一的 AWS 帳戶。
5. 針對模擬開始方法，選擇在 Amazon S3 中使用快照。

6. 針對快照的 Amazon S3 URI，輸入快照檔案的 Amazon S3 URI，或選擇瀏覽 S3 以瀏覽並選取檔案。

   > **Important**
   >
   > Amazon S3 儲存貯體必須與 AWS 區域 模擬位於相同的 中。

7. 針對 IAM 角色，選取模擬將使用的應用程式角色。
8. 在持續時間上限中，輸入模擬資源應執行的時間上限。最大值為 14D. 如需最長持續時間的詳細資訊，請參閱 。
9. 在標籤 - 選用下，如果您想要新增標籤，請選擇新增標籤。
10. 選擇開始模擬。

## 使用 AWS CLI 處理快照

您可以使用 AWS CLI 從命令提示字元呼叫 SimSpace Weaver APIs。您必須正確 AWS CLI 安裝和設定 。如需詳細資訊，請參閱《 第 AWS 2 版使用者指南》中的安裝或更新最新版本的 CLI。 AWS Command Line Interface

主題

- 使用 AWS CLI 建立快照
- 使用 從快照 AWS CLI 啟動模擬

### 使用 AWS CLI 建立快照

建立快照

- 在命令提示字元中，呼叫 `CreateSnapshot` API。

  ```
  aws simspaceweaver create-snapshot --simulation simulation-name –destination s3-destination
  ```

  參數

模擬

已啟動模擬的名稱。您可以使用 `aws simspaceweaver list-simulations` 查看模擬的名稱和狀態。

目的地

指定快照檔案目的地 Amazon S3 儲存貯體和選用物件金鑰字首的字串。您的物件金鑰字首通常是儲存貯體中的資料夾。 會在此目的地的`snapshot`資料夾內 SimSpace Weaver 建立快照。

> **Important**
>
> Amazon S3 儲存貯體必須與 AWS 區域 模擬位於相同的 中。

範例

```
aws simspaceweaver create-snapshot –simulation
 MyProjectSimulation_23-04-29_12_00_00 –destination BucketName=weaver-
myproject-111122223333-artifacts-us-west-2,ObjectKeyPrefix=myFolder
```

如需 `CreateSnapshot` API 的詳細資訊，請參閱 AWS SimSpace Weaver API 參考中的 CreateSnapshot。

## 使用 從快照 AWS CLI 啟動模擬

從快照啟動模擬

- 在命令提示字元中，呼叫 `StartSimulation` API。

```
aws simspaceweaver start-simulation --name simulation-name --role-arn role-arn --
snapshot-s3-location s3-location
```

參數

name

新模擬的名稱。模擬名稱在您的 中必須是唯一的 AWS 帳戶。您可以使用 `aws simspaceweaver list-simulations` 查看現有模擬的名稱。

role-arn

模擬將使用的應用程式角色的 Amazon Resource Name (ARN)。

snapshot-s3-location

指定快照檔案之 Amazon S3 儲存貯體和物件金鑰的字串。

> **Important**
>
> Amazon S3 儲存貯體必須與 AWS 區域 模擬位於相同的 中。

**範例**

```
aws simspaceweaver start-simulation –name MySimulation –role-arn
 arn:aws:iam::111122223333:role/weaver-MyProject-app-role –snapshot-s3-location
 BucketName=weaver-myproject-111122223333-artifacts-us-west-2,ObjectKey=myFolder/
snapshot/MyProjectSimulation_23-04-29_12_00_00-230429-1530-27.zip
```

如需 `StartSimulation` API 的詳細資訊，請參閱AWS SimSpace Weaver 《 API 參考》中的 StartSimulation。

## 有關快照的常見問答集

**我的模擬是否在快照期間繼續執行？**

您的模擬資源會在快照期間繼續執行，而且您會繼續收到當時的帳單費用。時間會計入模擬的最長持續時間。您的應用程式在快照進行時不會收到刻度。如果您的時鐘狀態是在STARTED快照建立開始時，您的時鐘仍會指出STARTED狀態。快照完成後，您的應用程式會再次收到刻度。如果您的時鐘狀態為 STOPPED，則您的時鐘狀態將保持 STOPPED。請注意，即使其時鐘STARTED狀態為 ，狀態為 的模擬仍會執行STOPPED。

**如果快照正在進行中且我的模擬達到其最長持續時間，會發生什麼情況？**

您的模擬將完成快照，然後在快照程序結束時立即停止（成功或失敗）。我們建議您事先測試快照程序，以了解快照程序需要多久、您可以預期的快照檔案大小，以及是否應成功完成。

如果我停止的模擬正在進行快照，會發生什麼情況？

當您停止模擬時，進行中的快照會立即停止。它不會建立快照檔案。

如何停止進行中的快照？

停止進行中的快照的唯一方法是停止模擬。您無法在停止模擬後重新啟動模擬。

完成快照需要多長時間？

建立快照所需的時間取決於您的模擬。我們建議您事先測試快照程序，以了解模擬需要多長時間。

我的快照檔案會有多大？

快照檔案的大小取決於您的模擬。我們建議您事先測試快照程序，以了解檔案對您的模擬而言的大小。

# 簡訊

訊息 API 可簡化應用程式與模擬內應用程式通訊的程序。傳送和接收訊息的 APIs是 SimSpace Weaver 應用程式 SDK 的一部分。訊息目前使用盡最大努力來傳送和接收訊息。SimSpace Weaver 會嘗試在下一個模擬刻度上傳送/接收訊息，但沒有交付、訂購或抵達時間保證。

主題



## 訊息的使用案例

模擬應用程式之間的通訊

使用傳訊 API 來在模擬中的應用程式之間進行通訊。使用它來變更遠距離實體的狀態、變更實體行為，或將資訊廣播到整個模擬。

確認收到訊息

傳送的訊息包含訊息標頭中寄件者的相關資訊。使用此資訊，在收到訊息時傳回確認回覆。

將自訂應用程式收到的資料轉送至模擬中的其他應用程式

訊息無法取代用戶端連線到在 中執行之自訂應用程式的方式 SimSpace Weaver。不過，簡訊允許使用者將資料從接收用戶端資料的自訂應用程式轉送到沒有外部連線的其他應用程式。訊息流程也可以反向運作，允許沒有外部連線的應用程式將資料轉送至自訂應用程式，然後轉送至用戶端。

## 使用傳訊 APIs

訊息 APIs 包含在 SimSpace Weaver 應用程式 SDK 中 （最低版本 1.16.0)。C++、Python 以及與 Unreal Engine 5 和 Unity 的整合支援傳訊。

有兩種處理訊息交易的函數： `SendMessage`和 `ReceiveMessages`。所有傳送的訊息都包含目的地和承載。`ReceiveMessages` API 會傳回目前在應用程式傳入訊息佇列中的訊息清單。

C++

傳送訊息

```
AWS_WEAVERRUNTIME_API Result<void> SendMessage(
    Transaction& txn,
    const MessagePayload& payload,
    const MessageEndpoint& destination,
    MessageDeliveryType deliveryType = MessageDeliveryType::BestEffort
    ) noexcept;
```

接收訊息

```
AWS_WEAVERRUNTIME_API Result<MessageList> ReceiveMessages(
    Transaction& txn) noexcept;
```

Python

傳送訊息

```
api.send_message(
 txn, # Transaction
 payload, # api.MessagePayload
 destination, # api.MessageDestination
```

```
 api.MessageDeliveryType.BestEffort # api.MessageDeliveryType
)
```

接收訊息

```
api.receive_messages(
 txn, # Transaction
) -> api.MessageList
```

主題

- 傳送訊息
- 接收訊息
- 回覆寄件者

## 傳送訊息

訊息包含交易（類似其他 Weaver API 呼叫）、承載和目的地。

訊息承載

訊息承載是高達 256 個位元組的彈性資料結構。我們建議建立訊息承載的最佳實務如下。

建立訊息承載

1. 建立定義訊息內容的資料結構（例如 C++ `struct`中的 )。
2. 建立訊息承載，其中包含要在訊息中傳送的值。
3. 建立 `MessagePayload` 物件。

訊息目的地

訊息的目的地由 `MessageEndpoint` 物件定義。這包括端點類型和端點 ID。目前唯一支援的端點類型是 `Partition`，可讓您將訊息定址到模擬中的其他分割區。端點 ID 是您目標目的地的分割區 ID。

您只能在訊息中提供 1 個目的地地址。如果您想要同時傳送訊息到多個分割區，請建立並傳送多個訊息。

如需如何從位置解析訊息端點的指引，請參閱 使用簡訊時的提示。

### 傳送訊息

您可以在建立目的地和承載物件之後使用 SendMessage API。

C++

```
Api::SendMessage(transaction, payload, destination,
 MessageDeliveryType::BestEffort);
```

Python

```
api.send_message(txn, payload, destination, api.MessageDeliveryType.BestEffort)
```

### 傳送訊息的完整範例

下列範例示範如何建構和傳送一般訊息。此範例會傳送 16 則個別訊息。每個訊息都包含值為 0 和 15 的承載，以及目前的模擬刻度。

Example

C++

```
// Message struct definition
struct MessageTickAndId
{
    uint32_t id;
    uint32_t tick;
};

Aws::WeaverRuntime::Result<void> SendMessages(Txn& txn) noexcept
{
     // Fetch the destination MessageEndpoint with the endpoint resolver
    WEAVERRUNTIME_TRY(
        Api::MessageEndpoint destination,
        Api::Utils::MessageEndpointResolver::ResolveFromPosition(
        txn,
            "MySpatialSimulation",
            Api::Vector2F32 {231.3, 654.0}
        )
    );
    Log::Info("destination: ", destination);
```

```
    WEAVERRUNTIME_TRY(auto tick, Api::CurrentTick(txn));

    uint16_t numSentMessages = 0;
    for (std::size_t i=0; i<16; i++)
    {
        // Create the message that'll be serialized into payload
        MessageTickAndId message {i, tick.value};

        // Create the payload out of the struct
        const Api::MessagePayload& payload = Api::Utils::CreateMessagePayload(
            reinterpret_cast<const std::uint8_t*>(&message),
            sizeof(MessageTickAndId)
        );

        // Send the payload to the destination
        Result<void> result = Api::SendMessage(txn, payload, destination);
        if (result.has_failure())
        {
            // SendMessage has failure modes, log them
            auto error = result.as_failure().error();
            std::cout<< "SendMessage failed, ErrorCode: " << error << std::endl;
            continue;
        }

        numSentMessages++;
    }

    std::cout << numSentMessages << " messages is sent to endpoint"
       << destination << std::endl;
    return Aws::WeaverRuntime::Success();
}
```

## Python

```
# Message data class
@dataclasses.dataclass
class MessageTickAndId:
    tick: int = 0
    id: int = 0

# send messages
def _send_messages(self, txn):
    tick = api.current_tick(txn)
```

```
    num_messages_to_send = 16

    # Fetch the destination MessageEndpoint with the endpoint resolver
    destination = api.utils.resolve_endpoint_from_domain_name_position(
       txn,
       "MySpatialSimulation",
       pos
   )
    Log.debug("Destination_endpoint = %s", destination_endpoint)

   for id in range(num_messages_to_send):
       # Message struct that'll be serialized into payload
        message_tick_and_id = MessageTickAndId(id = id, tick = tick.value)

       # Create the payload out of the struct
        message_tick_and_id_data = struct.pack(
           '<ii',
           message_tick_and_id.id,
           message_tick_and_id.tick
       )
        payload = api.MessagePayload(list(message_tick_and_id_data))

        # Send the payload to the destination
        Log.debug("Sending message: %s, endpoint: %s",
           message_tick_and_id,
           destination
       )
        api.send_message(
           txn,
           payload,
           destination,
           api.MessageDeliveryType.BestEffort
       )

    Log.info("Sent %s messages to %s", num_messages_to_send, destination)
    return True
```

## 接收訊息

SimSpace Weaver 會將訊息傳遞至分割區的傳入訊息佇列。使用 `ReceiveMessages` API 取得包含佇列訊息的`MessageList`物件。使用 `ExtractMessage` API 處理每則訊息，以取得訊息資料。

### Example

#### C++

```
Result<void> ReceiveMessages(Txn& txn) noexcept
{
     // Fetch all the messages sent to the partition owned by the app
    WEAVERRUNTIME_TRY(auto messages, Api::ReceiveMessages(txn));
    std::cout << "Received" << messages.messages.size() << " messages" << std::endl;
    for (Api::Message& message : messages.messages)
    {
        std::cout << "Received message: " << message << std::endl;

         // Deserialize payload to the message struct
        const MessageTickAndId& receivedMessage
            = Api::Utils::ExtractMessage<MessageTickAndId>(message);
        std::cout << "Received MessageTickAndId, Id: " << receivedMessage.id
            <<", Tick: " << receivedMessage.tick << std::endl;
    }

    return Aws::WeaverRuntime::Success();
}
```

#### Python

```
# process incoming messages
def _process_incoming_messages(self, txn):
    messages = api.receive_messages(txn)
    for message in messages:
        payload_list = message.payload.data
        payload_bytes = bytes(payload_list)
        message_tick_and_id_data_struct
           = MessageTickAndId(*struct.unpack('<ii', payload_bytes))

        Log.debug("Received message. Header: %s, message: %s",
                    message.header, message_tick_and_id_data_struct)

    Log.info("Received %s messages", len(messages))
    return True
```

## 回覆寄件者

每個收到的訊息都包含訊息標頭，其中包含訊息原始寄件者的相關資訊。您可以使用 message.header.source_endpoint 傳送回覆。

Example

C++

```
Result<void> ReceiveMessages(Txn& txn) noexcept
{
     // Fetch all the messages sent to the partition owned by the app
    WEAVERRUNTIME_TRY(auto messages, Api::ReceiveMessages(txn));
    std::cout << "Received" << messages.messages.size() << " messages" << std::endl;
    for (Api::Message& message : messages.messages)
    {
        std::cout << "Received message: " << message << std::endl;

         // Deserialize payload to the message struct
        const MessageTickAndId& receivedMessage
            = Api::Utils::ExtractMessage<MessageTickAndId>(message);
        std::cout << "Received MessageTickAndId, Id: " << receivedMessage.id
            <<", Tick: " << receivedMessage.tick << std::endl;

        // Get the sender endpoint and payload to bounce the message back
        Api::MessageEndpoint& sender = message.header.source_endpoint;
        Api::MessagePayload& payload = message.payload;
        Api::SendMessage(txn, payload, sender);
    }

    return Aws::WeaverRuntime::Success();
}
```

Python

```
# process incoming messages
def _process_incoming_messages(self, txn):
    messages = api.receive_messages(txn)
    for message in messages:
        payload_list = message.payload.data
        payload_bytes = bytes(payload_list)
        message_tick_and_id_data_struct
           = MessageTickAndId(*struct.unpack('<ii', payload_bytes))
```

```
        Log.debug("Received message. Header: %s, message: %s",
                    message.header, message_tick_and_id_data_struct)
        # Get the sender endpoint and payload
        # to bounce the message back
        sender = message.header.source_endpoint
        payload = payload_list
        api.send_message(
            txn,
            payload_list,
            sender,
            api.MessageDeliveryType.BestEffort

    Log.info("Received %s messages", len(messages))
    return True
```

## 何時使用簡訊

中的傳訊 SimSpace Weaver 提供另一種模式，用於在模擬應用程式之間交換資訊。訂閱提供從特定應用程式或模擬區域讀取資料的提取機制；訊息提供推送機制，將資料傳送至特定應用程式或模擬區域。

以下是兩個使用案例，其中使用傳訊推送資料，而不是透過訂閱提取或讀取資料會更有幫助。

Example 1：將命令傳送至另一個應用程式以變更實體位置

```
// Message struct definition
struct MessageMoveEntity
{
     uint64_t entityId;
    std::array<float, 3> destinationPos;
};

// Create the message
MessageMoveEntity message {45, {236.67, 826.22, 0.0} };

// Create the payload out of the struct
const Api::MessagePayload& payload = Api::Utils::CreateMessagePayload(
    reinterpret_cast<const std::uint8_t*>(&message),
    sizeof(MessageTickAndId)
);

// Grab the MessageEndpoint of the recipient app.
```

```
Api::MessageEndpoint destination = ...

// One way is to resolve it from the domain name and position
WEAVERRUNTIME_TRY(
    Api::MessageEndpoint destination,
    Api::Utils::MessageEndpointResolver::ResolveFromPosition(
    txn,
        "MySpatialSimulation",
        Api::Vector2F32 {200.0, 100.0}
    )
);

// Then send the message
Api::SendMessage(txn, payload, destination);
```

在接收端，應用程式會更新實體的位置，並將其寫入 State Fabric。

```
Result<void> ReceiveMessages(Txn& txn) noexcept
{
    WEAVERRUNTIME_TRY(auto messages, Api::ReceiveMessages(txn));
    for (Api::Message& message : messages.messages)
    {
        std::cout << "Received message: " << message << std::endl;
         // Deserialize payload to the message struct
        const MessageMoveEntity& receivedMessage
            = Api::Utils::ExtractMessage<MessageMoveEntity>(message);

        ProcessMessage(txn, receivedMessage);
    }

    return Aws::WeaverRuntime::Success();
}

void ProcessMessage(Txn& txn, const MessageMoveEntity& receivedMessage)
{
     // Get the entity corresponding to the entityId
    Entity entity = EntityFromEntityId (receivedMessage.entityId);

    // Update the position and write to StateFabric
    WEAVERRUNTIME_TRY(Api::StoreEntityIndexKey(
            txn,
            entity,
            k_vector3f32TypeId, // type id of the entity
```

```
            reinterpret_cast<std::int8_t*>(&receivedMessage.destinationPos),
            sizeof(receivedMessage.destinationPos)));

}
```

Example 2：將建立實體訊息傳送至空間應用程式

```
struct WeaverMessage
{
    const Aws::WeaverRuntime::Api::TypeId messageTypeId;
};

const Aws::WeaverRuntime::Api::TypeId k_createEntityMessageTypeId = { 1 };

struct CreateEntityMessage : WeaverMessage
{
    const Vector3 position;
   const Aws::WeaverRuntime::Api::TypeId typeId;
};


CreateEntityMessage messageData {
    k_createEntityMessageTypeId,
    Vector3{ position.GetX(), position.GetY(), position.GetZ() },
    Api::TypeId { 0 }
}

WEAVERRUNTIME_TRY(Api::MessageEndpoint destination,
 Api::Utils::MessageEndpointResolver::ResolveFromPosition(
    transaction, "MySpatialDomain", DemoFramework::ToVector2F32(position)
));

Api::MessagePayload payload = Api::Utils::CreateMessagePayload(
    reinterpret_cast<const uint8_t*>(&messageData),
    sizeof(CreateEntityMessage));

Api::SendMessage(transaction, payload, destination);
```

在接收端，應用程式會在 State Fabric 中建立新的實體，並更新其位置。

```
Result<void> ReceiveMessages(Txn& txn) noexcept
{
    WEAVERRUNTIME_TRY(auto messageList, Api::ReceiveMessages(transaction));
```

```
    WEAVERRUNTIME_TRY(auto tick, Api::CurrentTick(transaction));
    for (auto& message : messageList.messages)
    {
        // cast to base WeaverMessage type to determine MessageTypeId
        WeaverMessage weaverMessageBase =
 Api::Utils::ExtractMessage<WeaverMessage>(message);
        if (weaverMessageBase.messageTypeId == k_createEntityMessageTypeId)
        {
            CreateEntityMessage createEntityMessageData =
                Api::Utils::ExtractMessage<CreateEntityMessage>(message);
        CreateActorFromMessage(transaction, createEntityMessageData));
        }
        else if (weaverMessageBase.messageTypeId == k_tickAndIdMessageTypeId)
        {
            ...
        }
    }
}

void ProcessMessage(Txn& txn, const CreateEntityMessage& receivedMessage)
{
    // Create entity
    WEAVERRUNTIME_TRY(
        Api::Entity entity,
        Api::CreateEntity(transaction, receivedMessage.typeId)
    );

    // Update the position and write to StateFabric
    WEAVERRUNTIME_TRY(Api::StoreEntityIndexKey(
        transaction,
        entity,
        receivedMessage.typeId,
        reinterpret_cast<std::int8_t*>(&receivedMessage.position),
        sizeof(receivedMessage.position)));
}
```

# 使用簡訊時的提示

## 從位置或應用程式名稱解析端點

您可以使用 `AllPartitions`函數來取得空間邊界和網域 ID，以判斷訊息分割區 IDs和訊息目的地。不過，如果您知道要傳送訊息的位置，但不知道其分割區 ID，您可以使用 MessageEndpointResolver 函數。

```
/**
* Resolves MessageEndpoint's from various inputs
**/
class MessageEndpointResolver
{
    public:
    /**
    * Resolves MessageEndpoint from position information
    **/
    Result<MessageEndpoint> ResolveEndpointFromPosition(
        const DomainId& domainId,
        const weaver_vec3_f32_t& pos);

    /**
    * Resolves MessageEndpoint from custom app name
    **/
    Result<MessageEndpoint> ResolveEndpointFromCustomAppName(
        const DomainId& domainId,
        const char* agentName);
};
```

## 序列化和還原序列化訊息承載

您可以使用下列函數來建立和讀取訊息承載。如需詳細資訊，請參閱本機系統應用程式 SDK 程式庫中的 MessagingUtils.h。

```
/**
* Utility function to create MessagePayload from a custom type
*
* @return The @c MessagePayload.
*/
template <class T>
AWS_WEAVERRUNTIME_API MessagePayload CreateMessagePayload(const T& message) noexcept
{
    const std::uint8_t* raw_data = reinterpret_cast<const std::uint8_t*>(&message);

    MessagePayload payload;
    std::move(raw_data, raw_data + sizeof(T), std::back_inserter(payload.data));

    return payload;
}
```

```
/**
* Utility function to convert MessagePayload to custom type
*/
template <class T>
AWS_WEAVERRUNTIME_API T ExtractMessage(const MessagePayload& payload) noexcept
{
    return *reinterpret_cast<const T*>(payload.data.data());
}
```

## 訊息錯誤和疑難排解

使用簡訊 APIs 時，您可能會遇到下列錯誤。

### 端點解析錯誤

這些錯誤可能會在應用程式傳送訊息之前發生。

#### 網域名稱檢查

傳送訊息至無效的端點會導致下列錯誤：

```
ManifoldError::InvalidArgument {"No DomainId found for the given domain name" }
```

當您嘗試傳送訊息至自訂應用程式，且該自訂應用程式尚未加入模擬時，可能會發生這種情況。使用 `DescribeSimulation` API 確保您的自訂應用程式在您傳送訊息給它之前已啟動。此行為在 SimSpace Weaver Local和 中相同 AWS 雲端。

#### 位置檢查

嘗試解析具有有效網域名稱但位置無效的端點會導致下列錯誤。

```
ManifoldError::InvalidArgument {"Could not resolve endpoint from domain : DomainId
 { value: domain-id } and position: Vector2F32 { x: x-position, y: y-position}" }
```

我們建議在 SimSpace Weaver 應用程式 SDK `MessageEndpointResolver`中包含的`MessageUtils`程式庫中使用 。

### 訊息傳送錯誤

當應用程式傳送訊息時，可能會發生下列錯誤。

超過每個應用程式、每個刻度的訊息傳送限制

每個模擬刻度每個應用程式可傳送的訊息數量目前限制為 128。對相同刻度的後續呼叫將會失敗，並顯示下列錯誤：

```
ManifoldError::CapacityExceeded {"At Max Outgoing Message capacity: {}", 128}
```

SimSpace Weaver 會嘗試在下一個刻度傳送未傳送的訊息。降低傳送頻率以解決此問題。合併小於 256 位元組限制的訊息承載，以降低傳出訊息的數量。

此行為在 SimSpace Weaver Local和 中相同 AWS 雲端。

超過訊息承載大小限制

訊息承載大小的目前限制為 和 SimSpace Weaver Local中的 256 個位元組 AWS 雲端。傳送承載大於 256 個位元組的訊息會導致下列錯誤：

```
ManifoldError::CapacityExceeded {"Message data too large! Max size: {}", 256}
```

SimSpace Weaver 會檢查每個訊息，並僅拒絕超過限制的訊息。例如，如果您的應用程式嘗試傳送 10 則訊息，而 1 則檢查失敗，則只會拒絕 1 則訊息。 SimSpace Weaver 會傳送其他 9 則訊息。

此行為在 SimSpace Weaver Local和 中相同 AWS 雲端。

目的地與來源相同

應用程式無法傳送訊息至他們擁有的分割區。如果應用程式傳送訊息到其擁有的分割區，您會收到下列錯誤。

```
ManifoldError::InvalidArgument { "Destination is the same as source" }
```

此行為在 SimSpace Weaver Local和 中相同 AWS 雲端。

盡最大努力傳訊

SimSpace Weaver 不保證訊息交付。服務會嘗試在後續模擬刻度上完成訊息交付，但訊息可能會遺失或延遲。

# 使用 時的最佳實務 SimSpace Weaver

我們建議在使用 時採用下列最佳實務 SimSpace Weaver。

主題



## 設定帳單警示

在 中佈建資源很容易， AWS 即使不再需要資源，也能讓資源隨時執行。這可能會導致在您收到帳單時產生意外的流失成本。您可以在 Amazon CloudWatch 中設定警示，該警示會在成本超過您設定的閾值時觸發並通知您。您可以使用成本管理工具來檢查成本。如需詳細資訊，請參閱：

- 建立帳單警示以監控您的預估 AWS 費用
- 什麼是 AWS Cost Management

## 使用 SimSpace Weaver Local

建議您在將模擬上傳至 中的 SimSpace Weaver 服務之前，使用 SimSpace Weaver Local來開發和測試模擬 AWS 雲端。使用 開發 的優點SimSpace Weaver Local包括：

- 不需要等待大型上傳
- 您可以建立的本機模擬數目沒有限制
- 您不需要為本機電腦上的運算時間付費
- 從您的應用程式直接存取主控台輸出
- 修改、重建和重新啟動本機模擬，而無需在 中重新建立 AWS 雲端

# 停止您不需要的模擬

當模擬執行時，您會收到模擬的帳單費用。您必須停止模擬，才能停止收取費用。執行模擬也會計入模擬數量上限的配額。已設定記錄的執行中模擬也會產生大量日誌，您也會收到這些日誌的帳單費用。您應該停止不需要的任何模擬，才能停止收取額外費用。

> **Important**
>
> 停止模擬時鐘不會停止模擬，時鐘只會停止將刻度發佈到您的應用程式。停止模擬之後，就無法重新啟動模擬。

# 刪除您不需要的資源

您在 中建立的每個模擬 SimSpace Weaver 也會在其他 AWS 服務中建立資源。您可以針對這些其他服務中的資源和資料取得帳單費用。執行中和失敗的模擬會計入模擬數量上限的配額。您應該刪除不需要的失敗模擬，以便啟動新的模擬。當您刪除模擬時，可能不會刪除存在於 AWS 其他服務中的模擬資源。例如，Amazon CloudWatch Logs 中的任何模擬日誌資料都會保留在那裡，直到您將其刪除為止。您會收到該日誌資料的帳單費用。如果您不再需要模擬的所有相關資源，您應該清除這些資源。

# 擁有備份

最好為所有項目制定備份和備份計畫。您不應該只因為資料位於您不必備份的資料 AWS 而假設。如果您需要備份模擬狀態，則必須建立自己的系統。考慮使用多個 AWS 區域 並制定計劃， AWS 區域 以便在需要時快速將生產工作負載切換到另一個工作負載。如需 AWS 區域 該支援的詳細資訊 SimSpace Weaver，請參閱 SimSpace Weaver 端點和配額。

# 中的安全性 AWS SimSpace Weaver

的雲端安全 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，這些架構是為了滿足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。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 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。作為AWS 合規計畫的一部分，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用的合規計劃 AWS SimSpace Weaver，請參閱AWS 合規計劃的 服務範圍。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 時套用共同責任模型 SimSpace Weaver。下列主題說明如何設定 SimSpace Weaver 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 SimSpace Weaver 資源。

主題

- 中的資料保護 AWS SimSpace Weaver
- 的 Identity and Access Management AWS SimSpace Weaver
- 中的安全事件記錄和監控 AWS SimSpace Weaver
- 的合規驗證 AWS SimSpace Weaver
- 中的彈性 AWS SimSpace Weaver
- 中的基礎設施安全 AWS SimSpace Weaver
- 中的組態和漏洞分析 AWS SimSpace Weaver
- 的安全最佳實務 SimSpace Weaver

## 中的資料保護 AWS SimSpace Weaver

AWS 共同責任模型適用於 中的資料保護 AWS SimSpace Weaver。如此模型所述， AWS 負責保護執行所有 的 全球基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 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) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 SimSpace Weaver 或使用 主控台、API AWS CLI或其他 AWS 服務 AWS SDKs 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

## 靜態加密

當資料位於非揮發性 （ 持久性 ） 資料儲存體，例如磁碟時，會將其視為靜態資料。位於揮發性資料儲存體中的資料，例如記憶體和註冊，不會被視為靜態。

當您使用 時 SimSpace Weaver，唯一的靜態資料為：

- 您上傳至 Amazon Simple Storage Service (Amazon S3) 的應用程式和結構描述
- 儲存在 Amazon CloudWatch 中的模擬日誌資料

在您停止模擬之後，其他在內部 SimSpace Weaver 使用 的資料不會保留。

若要了解如何加密靜態資料，請參閱：

- 在 Amazon S3 中加密您的資料
- 加密您的日誌資料

## 傳輸中加密

透過 AWS Command Line Interface (AWS CLI)、 AWS SDK 和 SimSpace Weaver 應用程式 SDK 與 SimSpace Weaver API 的連線，搭配 Signature 第 4 版簽署程序使用 TLS 加密。 會使用 IAM 定義的存取政策來 AWS 管理您用來連線的安全登入資料的身分驗證。

在內部， SimSpace Weaver 會使用 TLS 連線到其使用的其他 AWS 服務。

> **Important**
>
> 您的應用程式與其用戶端之間的通訊不涉及 SimSpace Weaver。如有需要，您有責任加密與模擬用戶端的通訊。建議您建立解決方案，以加密跨用戶端連線傳輸中的所有資料。

若要進一步了解可支援加密解決方案的 AWS 服務，請參閱 AWS 安全部落格。

## 網際網路流量隱私權

SimSpace Weaver 運算資源位於所有 SimSpace Weaver 客戶共用的 1 個 Amazon VPC 內。所有內部 SimSpace Weaver 服務流量都會保留在 AWS 網路中，且不會透過網際網路傳輸。模擬用戶端與您的應用程式之間的通訊會通過網際網路。

# 的 Identity and Access Management AWS SimSpace Weaver

AWS Identity and Access Management (IAM) 是 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驗證 （登入） 和授權 （具有許可） 來使用 SimSpace Weaver 資源。IAM 是您可以免費使用 AWS 服務 的 。

主題

- 預防跨服務混淆代理人
- 對 AWS SimSpace Weaver 身分和存取進行故障診斷

## 目標對象

使用 AWS Identity and Access Management (IAM) 的方式會有所不同，取決於您在其中執行的工作 SimSpace Weaver。

服務使用者 – 如果您使用 SimSpace Weaver 服務來執行任務，您的管理員會為您提供所需的登入資料和許可。當您使用更多 SimSpace Weaver 功能來執行工作時，您可能需要額外的許可。了解存取的管理方式可協助您向管理員請求正確的許可。若您無法存取 SimSpace Weaver中的某項功能，請參閱 對 AWS SimSpace Weaver 身分和存取進行故障診斷。

服務管理員 – 如果您負責公司 SimSpace Weaver 的資源，您可能擁有 的完整存取權 SimSpace Weaver。您的任務是判斷服務使用者應存取哪些 SimSpace Weaver 功能和資源。接著，您必須將請求提交給您的 IAM 管理員，來變更您服務使用者的許可。檢閱此頁面上的資訊，了解 IAM 的基本概念。若要進一步了解貴公司如何搭配 使用 IAM SimSpace Weaver，請參閱 AWS SimSpace Weaver 如何使用 IAM。

IAM 管理員：如果您是 IAM 管理員，建議您掌握如何撰寫政策以管理 SimSpace Weaver存取權的詳細資訊。若要檢視您可以在 IAM 中使用的以 SimSpace Weaver 身分為基礎的政策範例，請參閱 的身分型政策範例 AWS SimSpace Weaver。

## 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 帳戶根使用者身分、IAM 使用者身分或擔任 IAM 角色來驗證 （ 登入 AWS)。

您可以使用透過身分來源提供的憑證，以聯合身分 AWS 身分身分登入 。 AWS IAM Identity Center (IAM Identity Center) 使用者、您公司的單一登入身分驗證，以及您的 Google 或 Facebook 登入資料，都是聯合身分的範例。您以聯合身分登入時，您的管理員先前已設定使用 IAM 角色的聯合身分。當您使用聯合 AWS 身分存取 時，您會間接擔任角色。

根據您的使用者類型，您可以登入 AWS Management Console 或 AWS 存取入口網站。如需登入的詳細資訊 AWS，請參閱AWS 登入 《 使用者指南》中的如何登入您的 AWS 帳戶 。

如果您以 AWS 程式設計方式存取 ， AWS 會提供軟體開發套件 (SDK) 和命令列界面 (CLI)，以使用您的 憑證以密碼編譯方式簽署您的請求。如果您不使用 AWS 工具，則必須自行簽署請求。如需

使用建議的方法自行簽署請求的詳細資訊，請參閱《IAM 使用者指南》中的適用於 API 請求的AWS Signature 第 4 版。

無論您使用何種身分驗證方法，您可能都需要提供額外的安全性資訊。例如， AWS 建議您使用多重驗證 (MFA) 來提高帳戶的安全性。如需更多資訊，請參閱《AWS IAM Identity Center 使用者指南》中的多重要素驗證和《IAM 使用者指南》中的 IAM 中的AWS 多重要素驗證。

### AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個登入身分開始，該身分可完整存取帳戶中的所有 AWS 服務 和資源。此身分稱為 AWS 帳戶 Theroot 使用者，可透過使用您用來建立帳戶的電子郵件地址和密碼登入來存取。強烈建議您不要以根使用者處理日常任務。保護您的根使用者憑證，並將其用來執行只能由根使用者執行的任務。如需這些任務的完整清單，了解需以根使用者登入的任務，請參閱 IAM 使用者指南中的需要根使用者憑證的任務。

### 聯合身分

最佳實務是， 要求人類使用者，包括需要管理員存取權的使用者，使用臨時 AWS 服務 憑證與身分提供者聯合來存取 。

聯合身分是來自您的企業使用者目錄、Web 身分提供者、 AWS Directory Service、 Identity Center 目錄或任何使用透過身分來源提供的登入資料 AWS 服務 存取的使用者。當聯合身分存取時 AWS 帳戶，它們會擔任 角色，而角色會提供臨時登入資料。

對於集中式存取權管理，我們建議您使用 AWS IAM Identity Center。您可以在 IAM Identity Center 中建立使用者和群組，也可以連接並同步到您自己的身分來源中的一組使用者 AWS 帳戶 和群組，以便在所有 和應用程式中使用。如需 IAM Identity Center 的詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

### IAM 使用者和群組

IAM 使用者是 中的身分 AWS 帳戶 ，具有單一人員或應用程式的特定許可。建議您盡可能依賴臨時憑證，而不是擁有建立長期憑證 (例如密碼和存取金鑰) 的 IAM 使用者。但是如果特定使用案例需要擁有長期憑證的 IAM 使用者，建議您輪換存取金鑰。如需更多資訊，請參閱 IAM 使用者指南中的為需要長期憑證的使用案例定期輪換存取金鑰。

IAM 群組是一種指定 IAM 使用者集合的身分。您無法以群組身分簽署。您可以使用群組來一次為多名使用者指定許可。群組可讓管理大量使用者許可的程序變得更為容易。例如，您可以擁有一個名為 IAMAdmins 的群組，並給予該群組管理 IAM 資源的許可。

使用者與角色不同。使用者只會與單一人員或應用程式建立關聯，但角色的目的是在由任何需要它的人員取得。使用者擁有永久的長期憑證，但角色僅提供臨時憑證。如需更多資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

## IAM 角色

IAM 角色是 中具有特定許可 AWS 帳戶 的身分。它類似 IAM 使用者，但不與特定的人員相關聯。若要暫時在 中擔任 IAM 角色 AWS Management Console，您可以從使用者切換至 IAM 角色（主控台）。您可以透過呼叫 AWS CLI 或 AWS API 操作或使用自訂 URL 來擔任角色。如需使用角色的方法詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

使用臨時憑證的 IAM 角色在下列情況中非常有用：

- 聯合身分使用者存取 — 如需向聯合身分指派許可，請建立角色，並為角色定義許可。當聯合身分進行身分驗證時，該身分會與角色建立關聯，並獲授予由角色定義的許可。如需有關聯合角色的相關資訊，請參閱《IAM 使用者指南》中的為第三方身分提供者 (聯合) 建立角色。如果您使用 IAM Identity Center，則需要設定許可集。為控制身分驗證後可以存取的內容，IAM Identity Center 將許可集與 IAM 中的角色相關聯。如需有關許可集的資訊，請參閱 AWS IAM Identity Center 使用者指南中的許可集。
- 暫時 IAM 使用者許可 – IAM 使用者或角色可以擔任 IAM 角色來暫時針對特定任務採用不同的許可。
- 跨帳戶存取權：您可以使用 IAM 角色，允許不同帳戶中的某人 (信任的主體) 存取您帳戶的資源。角色是授予跨帳戶存取權的主要方式。不過，對於某些 AWS 服務，您可以直接將政策連接到資源（而不是使用角色做為代理）。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。
- 跨服務存取 – 有些 AWS 服務 使用其他 中的功能 AWS 服務。例如，當您在服務中進行呼叫時，該服務通常會在 Amazon EC2 中執行應用程式或將物件儲存在 Amazon Simple Storage Service (Amazon S3) 中。服務可能會使用呼叫主體的許可、使用服務角色或使用服務連結角色來執行此作業。
  - 轉送存取工作階段 (FAS) – 當您使用 IAM 使用者或角色在其中執行動作時 AWS，您會被視為委託人。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 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。只有在服務收到需要與其他 AWS 服務 或 資源互動才能完成的請求時，才會提出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱《轉發存取工作階段》。
  - 服務角色 – 服務角色是服務擔任的 IAM 角色，可代表您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可權給 AWS 服務。

- 服務連結角色 – 服務連結角色是一種連結至 的服務角色。 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。
- 在 Amazon EC2 上執行的應用程式 – 您可以使用 IAM 角色來管理在 EC2 執行個體上執行之應用程式的臨時登入資料，以及提出 AWS CLI 或 AWS API 請求。這是在 EC2 執行個體內儲存存取金鑰的較好方式。若要將 AWS 角色指派給 EC2 執行個體並將其提供給其所有應用程式，您可以建立連接至執行個體的執行個體描述檔。執行個體設定檔包含該角色，並且可讓 EC2 執行個體上執行的程式取得臨時憑證。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM 角色來授予許可權給 Amazon EC2 執行個體上執行的應用程式。

## 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。政策是 中的物件，當與身分或資源相關聯時， AWS 會定義其許可。當委託人 （使用者、根使用者或角色工作階段） 發出請求時， 會 AWS 評估這些政策。政策中的許可決定是否允許或拒絕請求。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需 JSON 政策文件結構和內容的詳細資訊，請參閱 IAM 使用者指南中的 JSON 政策概觀。

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

預設情況下，使用者和角色沒有許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

IAM 政策定義該動作的許可，無論您使用何種方法來執行操作。例如，假設您有一個允許 `iam:GetRole` 動作的政策。具有該政策的使用者可以從 AWS Management Console AWS CLI、 或 AWS API 取得角色資訊。

### 身分型政策

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。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。

身分型政策可進一步分類成內嵌政策或受管政策。內嵌政策會直接內嵌到單一使用者、群組或角色。受管政策是獨立的政策，您可以連接到 中的多個使用者、群組和角色 AWS 帳戶。受管政策包括 AWS 受管政策和客戶受管政策。如需了解如何在受管政策及內嵌政策之間選擇，請參閱《IAM 使用者指南》中的在受管政策和內嵌政策間選擇。

## 資源型政策

資源型政策是連接到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

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

## 存取控制清單 (ACL)

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

Amazon S3 AWS WAF和 Amazon VPC 是支援 ACLs的服務範例。如需進一步了解 ACL，請參閱 Amazon Simple Storage Service 開發人員指南中的存取控制清單 (ACL) 概觀。

## 其他政策類型

AWS 支援其他較不常見的政策類型。這些政策類型可設定較常見政策類型授予您的最大許可。

- 許可界限 – 許可範圍是一種進階功能，可供您設定身分型政策能授予 IAM 實體 (IAM 使用者或角色) 的最大許可。您可以為實體設定許可界限。所產生的許可會是實體的身分型政策和其許可界限的交集。會在 `Principal` 欄位中指定使用者或角色的資源型政策則不會受到許可界限限制。所有這類政策中的明確拒絕都會覆寫該允許。如需許可界限的詳細資訊，請參閱 IAM 使用者指南中的 IAM 實體許可界限。
- 服務控制政策 SCPs) – SCPs是 JSON 政策，可指定 中組織或組織單位 (OU) 的最大許可 AWS Organizations。 AWS Organizations 是一種用於分組和集中管理您企業擁有 AWS 帳戶 之多個 的服務。若您啟用組織中的所有功能，您可以將服務控制政策 (SCP) 套用到任何或所有帳戶。SCP 會限制成員帳戶中實體的許可，包括每個實體 AWS 帳戶根使用者。如需 Organizations 和 SCP 的詳細資訊，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) - RCP 是 JSON 政策，可用來設定您帳戶中資源的可用許可上限，採取這種方式就不需要更新附加至您所擁有的每個資源的 IAM 政策。RCP 會限制成員帳戶中資源的許可，並可能影響身分的有效許可，包括 AWS 帳戶根使用者，無論它們是否屬於您的組織。如需 Organizations 和 RCPs的詳細資訊，包括 AWS 服務 支援 RCPs 清單，請參閱AWS Organizations 《 使用者指南》中的資源控制政策 (RCPs)。
- 工作階段政策 – 工作階段政策是一種進階政策，您可以在透過撰寫程式的方式建立角色或聯合使用者的暫時工作階段時，做為參數傳遞。所產生工作階段的許可會是使用者或角色的身分型政策和工作

階段政策的交集。許可也可以來自資源型政策。所有這類政策中的明確拒絕都會覆寫該允許。如需詳細資訊，請參閱 IAM 使用者指南中的工作階段政策。

### 多種政策類型

將多種政策類型套用到請求時，其結果形成的許可會更為複雜、更加難以理解。若要了解如何 AWS 在涉及多種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。

## AWS SimSpace Weaver 如何使用 IAM

在您使用 IAM 管理對 的存取之前 SimSpace Weaver，請先了解可使用哪些 IAM 功能 SimSpace Weaver。

您可以搭配 使用的 IAM 功能 AWS SimSpace Weaver

| IAM 功能 | SimSpace Weaver 支援 |
| --- | --- |
| 身分型政策 | 是 |
| 資源型政策 | 否 |
| 政策動作 | 是 |
| 政策資源 | 是 |
| 政策條件索引鍵 (服務特定) | 是 |
| ACL | 否 |
| ABAC (政策中的標籤) | 是 |
| 臨時憑證 | 是 |
| 主體許可 | 是 |
| 服務角色 | 是 |
| 服務連結角色 | 否 |

若要全面了解 SimSpace Weaver 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。

## 的身分型政策 SimSpace Weaver

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。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

的身分型政策範例 SimSpace Weaver

若要檢視 SimSpace Weaver 身分型政策的範例，請參閱 的身分型政策範例 AWS SimSpace Weaver。

## 中的資源型政策 SimSpace Weaver

支援資源型政策：否

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。委託人可以包含帳戶、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，您可以指定在其他帳戶內的所有帳戶或 IAM 實體，做為資源型政策的主體。新增跨帳戶主體至資源型政策，只是建立信任關係的一半。當委託人和資源位於不同位置時 AWS 帳戶，信任帳戶中的 IAM 管理員也必須授予委託人實體 （使用者或角色） 存取資源的許可。其透過將身分型政策連接到實體來授與許可。不過，如果資源型政策會為相同帳戶中的主體授予存取，這時就不需要額外的身分型政策。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

## 的政策動作 SimSpace Weaver

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，哪個主體在什麼条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。政策動作通常具有與相關聯 AWS API 操作相同的名稱。有一些例外狀況，例如沒有相符的 API 操作的僅限許可動作。也有一些作業需要政策中的多個動作。這些額外的動作稱為相依動作。

政策會使用動作來授予執行相關聯動作的許可。

若要查看 SimSpace Weaver 動作清單，請參閱《服務授權參考》中的 定義的動作 AWS SimSpace Weaver。

中的政策動作在動作之前 SimSpace Weaver 使用以下字首：

```
simspaceweaver
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
      "simspaceweaver:action1",
      "simspaceweaver:action2"
         ]
```

若要檢視 SimSpace Weaver 身分型政策的範例，請參閱 的身分型政策範例 AWS SimSpace Weaver。

## 的政策資源 SimSpace Weaver

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。陳述式必須包含 `Resource` 或 `NotResource` 元素。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。您可以針對支援特定資源類型的動作 (稱為資源層級許可) 來這麼做。

對於不支援資源層級許可的動作 (例如列出操作)，請使用萬用字元 (*) 來表示陳述式適用於所有資源。

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

若要查看 SimSpace Weaver 資源類型及其 ARNs，請參閱《服務授權參考》中的 定義的資源 AWS SimSpace Weaver。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 AWS SimSpace Weaver定義的動作。

若要檢視 SimSpace Weaver 身分型政策的範例，請參閱 的身分型政策範例 AWS SimSpace Weaver。

## 的政策條件索引鍵 SimSpace Weaver

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 (或 Condition 區塊) 可讓您指定使陳述式生效的條件。Condition 元素是選用項目。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。

若您在陳述式中指定多個 Condition 元素，或是在單一 Condition 元素中指定多個索引鍵， AWS 會使用邏輯 AND 操作評估他們。如果您為單一條件索引鍵指定多個值， 會使用邏輯OR操作 AWS 評估條件。必須符合所有條件，才會授與陳述式的許可。

您也可以在指定條件時使用預留位置變數。例如，您可以只在使用者使用其 IAM 使用者名稱標記時，將存取資源的許可授予該 IAM 使用者。如需更多資訊，請參閱 IAM 使用者指南中的 IAM 政策元素：變數和標籤。

AWS 支援全域條件金鑰和服務特定的條件金鑰。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 SimSpace Weaver 條件索引鍵的清單，請參閱《服務授權參考》中的 的條件索引鍵 AWS SimSpace Weaver。若要了解您可以使用條件金鑰的動作和資源，請參閱 定義的動作 AWS SimSpace Weaver。

若要檢視 SimSpace Weaver 身分型政策的範例，請參閱 的身分型政策範例 AWS SimSpace Weaver。

## SimSpace Weaver中的存取控制清單 (ACL)

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

## 使用 的屬性型存取控制 (ABAC) SimSpace Weaver

支援 ABAC (政策中的標籤)：是

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在 中 AWS，這些屬性稱為標籤。您可以將標籤連接至 IAM 實體 （使用者或角色） 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的使用 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 使用臨時登入資料 SimSpace Weaver

支援臨時憑證：是

當您使用臨時登入資料登入時，有些 AWS 服務 無法運作。如需詳細資訊，包括哪些 AWS 服務 使用臨時登入資料，請參閱《AWS 服務 IAM 使用者指南》中的 使用 IAM。

如果您 AWS Management Console 使用使用者名稱和密碼以外的任何方法登入 ，則會使用暫時登入資料。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取 時，該程序會自動建立臨時登入資料。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱《IAM 使用者指南》中的從使用者切換至 IAM 角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

## 的跨服務主體許可 SimSpace Weaver

支援轉寄存取工作階段 (FAS)：是

當您使用 IAM 使用者或角色在 中執行動作時 AWS，您會被視為委託人。使用某些服務時，您可能會執行某個動作，進而在不同服務中啟動另一個動作。FAS 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。只有當服務收到需要與其他 AWS 服務 或 資源互動才能完成的請求時，才會提出 FAS 請求。在此情況下，您必須具有執行這兩個動作的許可。如需提出 FAS 請求時的政策詳細資訊，請參閱轉發存取工作階段。

### SimSpace Weaver的服務角色

支援服務角色：是

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權給 AWS 服務。

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

SimSpace Weaver 應用程式 SDK 指令碼會使用 AWS CloudFormation 範本在其他 AWS 服務中建立資源，以支援您的模擬。其中一個資源是模擬的應用程式角色。 SimSpace Weaver 會擔任應用程式角色， AWS 帳戶 代表您在 中執行動作，例如將日誌資料寫入 CloudWatch Logs。如需應用程式角色的詳細資訊，請參閱 為您 SimSpace Weaver 建立的許可。

### 的服務連結角色 SimSpace Weaver

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。在表格中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

## 的身分型政策範例 AWS SimSpace Weaver

根據預設，使用者和角色不具備建立或修改 SimSpace Weaver 資源的權限。他們也無法使用 AWS Management Console、 AWS Command Line Interface (AWS CLI) 或 AWS API 來執行任務。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。然後，管理員可以將 IAM 政策新增至角色，使用者便能擔任這些角色。

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

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 SimSpace Weaver 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 AWS CloudFormation。如需詳細資訊，請參閱 IAM 使用者指南中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您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》https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_configure-api-require.html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 SimSpace Weaver 主控台

若要存取 AWS SimSpace Weaver 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 SimSpace Weaver 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。

為了確保使用者和角色仍然可以使用 SimSpace Weaver 主控台，也請將 SimSpace Weaver *ConsoleAccess*或 *ReadOnly* AWS 受管政策連接到實體。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": "*"
        }
    ]
}
```

## 允許使用者建立和執行模擬

此範例 IAM 政策提供在其中建立和執行模擬所需的基本許可 SimSpace Weaver。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateAndRunSimulations",
            "Effect": "Allow",
            "Action": [
                "simspaceweaver:*",
                "iam:GetRole",
                "iam:ListRoles",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:UpdateRole",
                "iam:CreatePolicy",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy",
                "iam:GetRolePolicy",
                "iam:DeleteRolePolicy",
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListAllMyBuckets",
                "s3:PutBucketPolicy",
                "s3:CreateBucket",
                "s3:ListBucket",
                "s3:PutEncryptionConfiguration",
                "s3:DeleteBucket",
                "cloudformation:CreateStack",
                "cloudformation:UpdateStack",
```

```
                "cloudformation:DescribeStacks"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PassAppRoleToSimSpaceWeaver",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "simspaceweaver.amazonaws.com"
                }
            }
        }
    ]
}
```

# 為您 SimSpace Weaver 建立的許可

當您建立 SimSpace Weaver 專案時，服務會建立名稱為 的 AWS Identity and Access Management (IAM) 角色`weaver-`*`project-name`*`-app-role`和 IAM 信任政策。信任政策允許 SimSpace Weaver 擔任該角色，使其可以為您執行操作。

## 應用程式角色許可政策

模擬應用程式角色具有下列許可政策。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:DescribeLogGroups",
                "logs:DescribeLogStreams",
                "logs:CreateLogGroup",
                "logs:CreateLogStream"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "*"
        }
    ]
}
```

## 應用程式角色信任政策

SimSpace Weaver 會將信任關係新增至模擬應用程式角色做為信任政策。 會為每個模擬 SimSpace Weaver 建立信任政策，類似於下列範例。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "simspaceweaver.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "ArnLike": {
          "aws:SourceArn":
     "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/MySimName*"
        }
      }
    }
  ]
}
```

> **Note**
>
> 在此範例中，帳戶號碼為 111122223333，模擬名稱為 MySimName。這些值在您的信任政策中不同。

## 預防跨服務混淆代理人

混淆代理人問題是一種安全問題，其中沒有執行動作許可的實體可能會誘使更特權的實體執行動作。在中 AWS，跨服務模擬可能會導致混淆代理人問題。在某個服務 (呼叫服務) 呼叫另一個服務 (被呼叫服務) 時，可能會發生跨服務模擬。可以操縱呼叫服務來使用其許可，以其不應有存取許可的方式對其他客戶的資源採取動作。為了預防這種情況， AWS 提供的工具可協助您保護所有服務的資料，而這些服務主體已獲得您帳戶中資源的存取權。

我們建議在資源政策中使用 aws:SourceArn和 aws:SourceAccount全域條件內容索引鍵，以限制將另一個 服務 AWS SimSpace Weaver 提供給資源的許可。如果 aws:SourceArn 值不包含帳戶 ID (例如 Amazon S3 儲存貯體 Amazon Resource Name (ARN))，則您必須使用這兩個全域條件內容索引鍵來限制許可。如果同時使用這兩個全域條件內容索引鍵，且 aws:SourceArn 值包含帳戶 ID，則在相同政策陳述式中使用 aws:SourceAccount 值和 aws:SourceArn 值中的帳戶時，必須使用相同的帳戶 ID。如果您想要僅允許一個資源與跨服務存取相關聯，則請使用 aws:SourceArn。如果您想要允許該帳戶中的任何資源與跨服務使用相關聯，請使用 aws:SourceAccount。

aws:SourceArn 的值必須使用擴充功能的 ARN。

防範混淆代理人問題的最有效方法是使用 aws:SourceArn 全域條件內容索引鍵，以及資源的完整 ARN。如果您不知道擴充功能的完整 ARN，或者如果您指定了多個擴充功能，請使用 aws:SourceArn 全域內容條件索引鍵搭配萬用字元 (*) 來表示 ARN 的未知部分。例如 arn:aws:*simspaceweaver*:*:*111122223333*:*。

下列範例示範如何在 中使用 aws:SourceArn和 aws:SourceAccount全域條件內容索引鍵 SimSpace Weaver ，以防止混淆代理人問題。此政策將僅允許在請求來自指定的來源帳戶，並隨附指定的 ARN 時 SimSpace Weaver 擔任角色。在此情況下， SimSpace Weaver 只能擔任請求者自有帳戶 (*111122223333*) 中模擬請求的角色，且只能在指定的區域 (*us-west-2*) 中擔任該角色。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": [
          "simspaceweaver.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        },
        "StringLike": {
          "aws:SourceArn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/*"
        }
      }
    }
  ]
}
```

撰寫此政策的更安全方法是在 中包含模擬名稱aws:SourceArn，如下列範例所示，其會將政策限制為名為 的模擬MyProjectSimulation_22-10-04_22_10_15：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "simspaceweaver.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "111122223333"
        },
        "StringLike": {
          "aws:SourceArn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15"
        }
      }
    }
  ]
}
```

當您aws:SourceArn明確包含 帳戶號碼時，您可以離開 的 Condition 元素測試 aws:SourceAccount（如需詳細資訊，請參閱 IAM 使用者指南），例如下列簡化政策：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "simspaceweaver.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringLike": {
          "aws:SourceArn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15"
        }
      }
    }
  ]
}
```

# 對 AWS SimSpace Weaver 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 SimSpace Weaver 和 IAM 時可能遇到的常見問題。

主題

- 我無權在 中執行動作 SimSpace Weaver
- 我未獲得執行 iam:PassRole 的授權
- 我想要檢視我的存取金鑰
- 我是管理員，想要允許其他人存取 SimSpace Weaver
- 我想要允許 以外的人員 AWS 帳戶 存取我的 SimSpace Weaver 資源

## 我無權在 中執行動作 SimSpace Weaver

如果 AWS Management Console 告訴您未獲授權執行 動作，則必須聯絡管理員尋求協助。您的管理員是提供您使用者名稱和密碼的人員。

以下範例錯誤會在 mateojackson IAM 使用者嘗試使用主控台檢視虛構 *my-example-widget* 資源的詳細資訊，但卻沒有虛構 simspaceweaver:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 simspaceweaver:GetWidget on resource: my-example-widget
```

在此情況下，Mateo 會請求管理員更新他的政策，允許他使用 *my-example-widget* 動作存取 simspaceweaver:*GetWidget* 資源。

## 我未獲得執行 iam:PassRole 的授權

如果您收到錯誤，告知您未獲授權執行 iam:PassRole 動作，您的政策必須更新，允許您將角色傳遞給 SimSpace Weaver。

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。

名為 marymajor 的 IAM 使用者嘗試使用主控台在 SimSpace Weaver中執行動作時，發生下列範例錯誤。但是，動作要求服務具備服務角色授予的許可。Mary 沒有將角色傳遞至該服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 iam:PassRole 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的登入憑證。

## 我想要檢視我的存取金鑰

在您建立 IAM 使用者存取金鑰後，您可以隨時檢視您的存取金鑰 ID。但是，您無法再次檢視您的私密存取金鑰。若您遺失了密碼金鑰，您必須建立新的存取金鑰對。

存取金鑰包含兩個部分：存取金鑰 ID (例如 AKIAIOSFODNN7EXAMPLE) 和私密存取金鑰 (例如 wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY)。如同使用者名稱和密碼，您必須一起使用存取金鑰 ID 和私密存取金鑰來驗證您的請求。就如對您的使用者名稱和密碼一樣，安全地管理您的存取金鑰。

> **Important**
>
> 請勿將您的存取金鑰提供給第三方，甚至是協助尋找您的標準使用者 ID。透過這樣做，您可以讓某人永久存取您的 AWS 帳戶。

建立存取金鑰對時，您會收到提示，要求您將存取金鑰 ID 和私密存取金鑰儲存在安全位置。私密存取金鑰只會在您建立它的時候顯示一次。若您遺失了私密存取金鑰，您必須將新的存取金鑰新增到您的 IAM 使用者。您最多可以擁有兩個存取金鑰。若您已有兩個存取金鑰，您必須先刪除其中一個金鑰對，才能建立新的金鑰對。若要檢視說明，請參閱《IAM 使用者指南》中的管理存取金鑰。

## 我是管理員，想要允許其他人存取 SimSpace Weaver

若要允許其他人存取 SimSpace Weaver，您必須將許可授予需要存取的人員或應用程式。如果您使用 AWS IAM Identity Center 管理人員和應用程式，您可以將許可集指派給使用者或群組，以定義其存取層級。許可集會自動建立 IAM 政策，並將其指派給與該人員或應用程式相關聯的 IAM 角色。如需詳細資訊，請參閱AWS IAM Identity Center 《 使用者指南》中的許可集。

如果您不是使用 IAM Identity Center，則必須為需要存取的人員或應用程式建立 IAM 實體（使用者或角色）。您接著必須將政策連接到實體，在 SimSpace Weaver中授予他們正確的許可。授予許可後，請將登入資料提供給使用者或應用程式開發人員。他們將使用這些登入資料來存取 AWS。若要進一步了解如何建立 IAM 使用者、群組、政策和許可，請參閱《IAM 使用者指南》中的 IAM 身分和政策和許可。

## 我想要允許 以外的人員 AWS 帳戶 存取我的 SimSpace Weaver 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 是否 SimSpace Weaver 支援這些功能，請參閱 AWS SimSpace Weaver 如何使用 IAM。
- 若要了解如何 AWS 帳戶 在您擁有的 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 中提供存取權給 IAM 使用者。
- 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。

# 中的安全事件記錄和監控 AWS SimSpace Weaver

監控是維護 和 AWS 解決方案的可靠性、可用性 SimSpace Weaver 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，以便在發生多點失敗時更輕鬆地偵錯。

AWS 和 SimSpace Weaver 提供數種工具來監控模擬資源並回應潛在事件。

Amazon CloudWatch 中的日誌

SimSpace Weaver 會將日誌儲存在 CloudWatch 中。您可以使用這些日誌來監控模擬中的事件 （例如啟動和停止應用程式），以及偵錯。如需詳細資訊，請參閱SimSpace Weaver Amazon CloudWatch Logs 中的日誌。

Amazon CloudWatch 警示

使用 Amazon CloudWatch 警示，您可在自己指定的一段時間內監看單一指標。如果指標超過指定的閾值，則會將通知傳送至 Amazon SNS 主題或 AWS Auto Scaling 政策。CloudWatch 警示會在其狀態變更時觸發，並且會維持一段指定的期間，而不是處於特定狀態。如需詳細資訊，請參閱SimSpace Weaver 使用 Amazon CloudWatch 進行監控。

AWS CloudTrail 日誌

CloudTrail 會提供使用者、角色或服務 AWS 在其中採取之動作的記錄 SimSpace Weaver。您可以使用 CloudTrail 所收集的資訊來判斷提出的請求 SimSpace Weaver、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。如需詳細資訊，請參閱使用 記錄 AWS SimSpace Weaver API 呼叫 AWS CloudTrail。

# 的合規驗證 AWS SimSpace Weaver

SimSpace Weaver 不在任何 AWS 合規計劃的範圍內。

在多個合規計畫中 AWS ，第三方稽核人員會評估其他服務的安全性和 AWS 合規性。這些計劃包括 SOC、PCI、FedRAMP、HIPAA 等等。

若要了解 是否 AWS 服務 在特定合規計劃範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱 AWS Compliance Programs。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱下載 中的 AWS Artifact報告。

您使用 時的合規責任 AWS 服務 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。AWS 提供下列資源來協助合規：

- 安全合規與治理 - 這些解決方案實作指南內容討論了架構考量，並提供部署安全與合規功能的步驟。
- HIPAA 合格服務參考 – 列出 HIPAA 合格服務。並非所有 AWS 服務 都符合 HIPAA 資格。
- AWS 合規資源 – 此工作手冊和指南的集合可能適用於您的產業和位置。
- AWS 客戶合規指南 – 透過合規的角度了解共同責任模型。本指南摘要說明跨多個架構 （包括國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 (ISO)) 保護 AWS 服務和映射指南至安全控制的最佳實務。
- 《AWS Config 開發人員指南》中的使用 規則評估資源 – AWS Config 服務會評估資源組態符合內部實務、產業準則和法規的程度。
- AWS Security Hub – 這 AWS 服務 可讓您全面檢視其中的安全狀態 AWS。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱「Security Hub 控制參考」。
- Amazon GuardDuty – 這會監控您的環境是否有可疑和惡意活動，以 AWS 服務 偵測對您 AWS 帳戶、工作負載、容器和資料的潛在威脅。GuardDuty 可滿足特定合規架構所規定的入侵偵測需求，以協助您因應 PCI DSS 等各種不同的合規需求。
- AWS Audit Manager – 這 AWS 服務 可協助您持續稽核 AWS 用量，以簡化您管理風險的方式，以及符合法規和業界標準的方式。

# 中的彈性 AWS SimSpace Weaver

AWS 全球基礎設施是以 AWS 區域 和 可用區域為基礎建置。 AWS 區域 提供多個實體分隔和隔離的可用區域，這些可用區域與低延遲、高輸送量和高備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

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。

# 中的基礎設施安全 AWS SimSpace Weaver

作為受管服務， AWS SimSpace Weaver 受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的相關資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務來設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 已發佈的 API 呼叫， SimSpace Weaver 透過網路存取 。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以透過 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

## 網路連線安全模型

您的模擬會在您選取 AWS 區域內 Amazon VPC 內的運算執行個體上執行。Amazon VPC 是 AWS 雲端中的虛擬網路，可依工作負載或組織實體隔離基礎設施。Amazon VPC 內的運算執行個體之間的通訊會保留在 AWS 網路中，而且不會透過網際網路傳輸。有些內部服務通訊會跨網際網路並加密。在相同 AWS 區域中執行的所有客戶的模擬都會共用相同的 Amazon VPC。不同客戶的模擬會在相同的 Amazon VPC 中使用不同的運算執行個體。

模擬用戶端與透過網際網路 SimSpace Weaver 傳輸中執行之模擬之間的通訊。 SimSpace Weaver 不會處理這些連線。保護用戶端連線是您的責任。

您對 SimSpace Weaver 服務的連線會透過網際網路進行加密。這包括使用 AWS Management Console、 AWS Command Line Interface (AWS CLI)、 AWS 軟體開發套件 (SDK) 和 SimSpace Weaver 應用程式 SDK 的連線。

# 中的組態和漏洞分析 AWS SimSpace Weaver

組態和 IT 控制是 AWS 與您之間的共同責任。如需詳細資訊，請參閱 AWS 共同責任模型。 AWS 會處理基礎基礎設施的基本安全任務，例如在運算執行個體上修補作業系統、防火牆組態和 AWS 基礎設施災難復原。這些程序已由適當的第三方進行檢閱並認證。如需詳細資訊，請參閱安全性、身分和合規的最佳實務。

您必須負責模擬軟體的安全性：

- 維護您的應用程式程式碼，包括更新和安全性修補程式。
- 驗證並加密模擬用戶端與其連線的應用程式之間的通訊。
- 更新您的模擬以使用最新的 SDK 版本，包括 AWS SDK 和 SimSpace Weaver 應用程式 SDK。

> **Note**
>
> SimSpace Weaver 不支援在執行中的模擬中更新應用程式。如果您需要更新應用程式，您必須停止並刪除模擬，然後使用更新的應用程式程式碼建立新的模擬。我們建議您將模擬狀態儲存在外部資料存放區中，以便在需要重新建立模擬時進行還原。

# 的安全最佳實務 SimSpace Weaver

本節說明特定的安全最佳實務 SimSpace Weaver。若要進一步了解 中的安全最佳實務 AWS，請參閱安全性、身分和合規的最佳實務。

主題



## 加密應用程式與其用戶端之間的通訊

SimSpace Weaver 不會管理應用程式與其用戶端之間的通訊。您應該為用戶端工作階段實作某種形式的身分驗證和加密。

## 定期備份模擬狀態

SimSpace Weaver 不會儲存您的模擬狀態。停止的模擬（由於 API 呼叫、主控台選項或系統當機）不會儲存其狀態，也沒有復原狀態的固有方法。停止的模擬無法重新啟動。執行相當於重新啟動的唯一方法是使用相同的組態和資料重新建立模擬。您可以使用模擬狀態的備份來初始化新的模擬。 AWS 提供高度可靠且可用的雲端儲存和資料庫服務，可用來儲存模擬狀態。

## 維護您的應用程式和SDKs

維護您的應用程式、 AWS 軟體開發套件 (SDKs) 的本機安裝，以及 SimSpace Weaver 應用程式 SDK。您可以下載並安裝新版本的 AWS SDKs。使用非生產應用程式建置測試 SimSpace Weaver 應用程式 SDK 的新版本，以確保您的應用程式繼續如預期般執行。您無法在執行中的模擬中更新您的應用程式。若要更新您的應用程式：

1. 在本機（或在測試環境中）更新和測試應用程式碼。

2. 停止變更模擬狀態並儲存（如有必要）。
3. 停止模擬（一旦停止，就無法重新啟動）。
4. 刪除模擬（未刪除的已停止模擬會計入您的服務限制）。
5. 使用相同的組態和更新的應用程式程式碼重新建立模擬。
6. 使用儲存的狀態資料（如果可用）初始化模擬。
7. 啟動新的模擬。

> **Note**
>
> 使用相同組態建立的新模擬與舊模擬不同。它將具有新的模擬 ID，並將日誌傳送至 Amazon CloudWatch 中的新日誌串流。

# 在 中記錄和監控 SimSpace Weaver

監控是維護 SimSpace Weaver 及其他 AWS 解決方案的可靠性、可用性和效能的重要部分。 AWS 提供下列監控工具，讓您監看 SimSpace Weaver、回報錯誤，並適時採取自動動作：

- Amazon CloudWatch AWS 會即時監控您的 AWS 資源和您在 上執行的應用程式。您可以收集和追蹤指標、建立自訂儀板表，以及設定警示，在特定指標達到您指定的閾值時通知您或採取動作。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。
- Amazon CloudWatch Logs 可讓您監控、存放和存取 SimSpace Weaver 工作者、CloudTrail 和其他來源的日誌資料。CloudWatch Logs 可以監控日誌資料中的資訊，並在達到特定閾值時通知您。您也可以將日誌資料存檔在高耐用性的儲存空間。如需詳細資訊，請參閱 Amazon CloudWatch Logs 使用者指南。
- AWS CloudTrail 擷取您 AWS 帳戶 發出或代表發出的 API 呼叫和相關事件，並傳送日誌檔案至您指定的 Amazon S3 儲存貯體。您可以找出哪些使用者和帳戶呼叫 AWS、發出呼叫的來源 IP 地址，以及呼叫的發生時間。如需詳細資訊，請參閱 AWS CloudTrail 使用者指南。

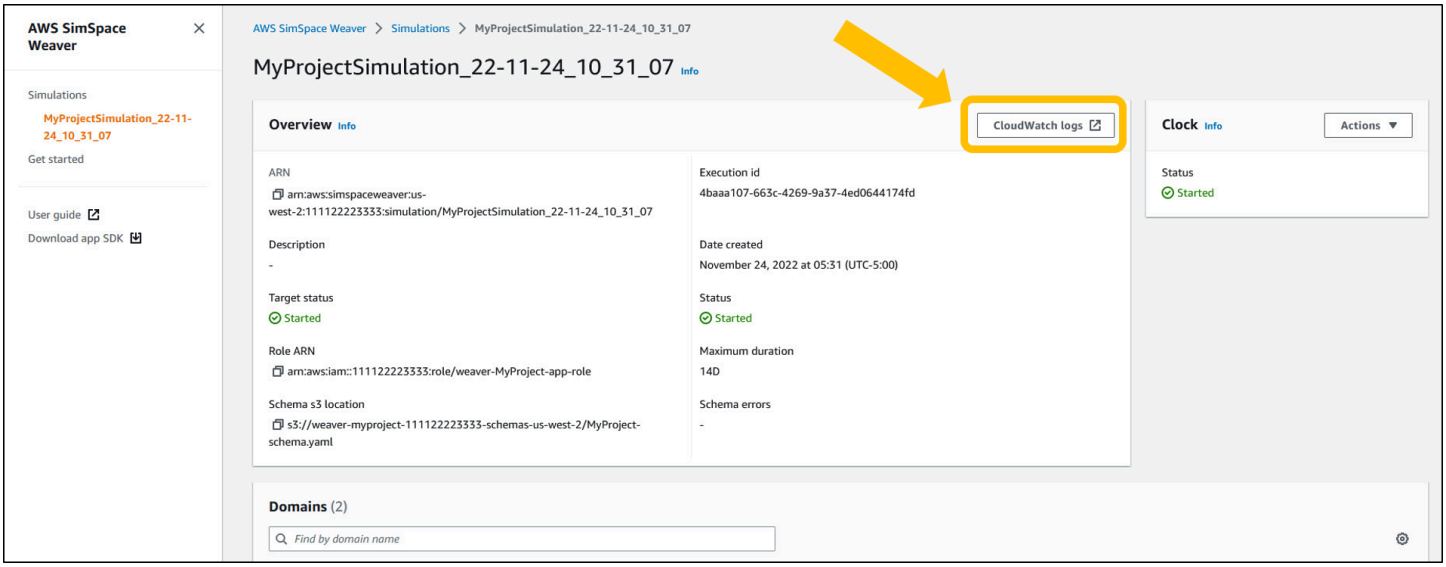
主題



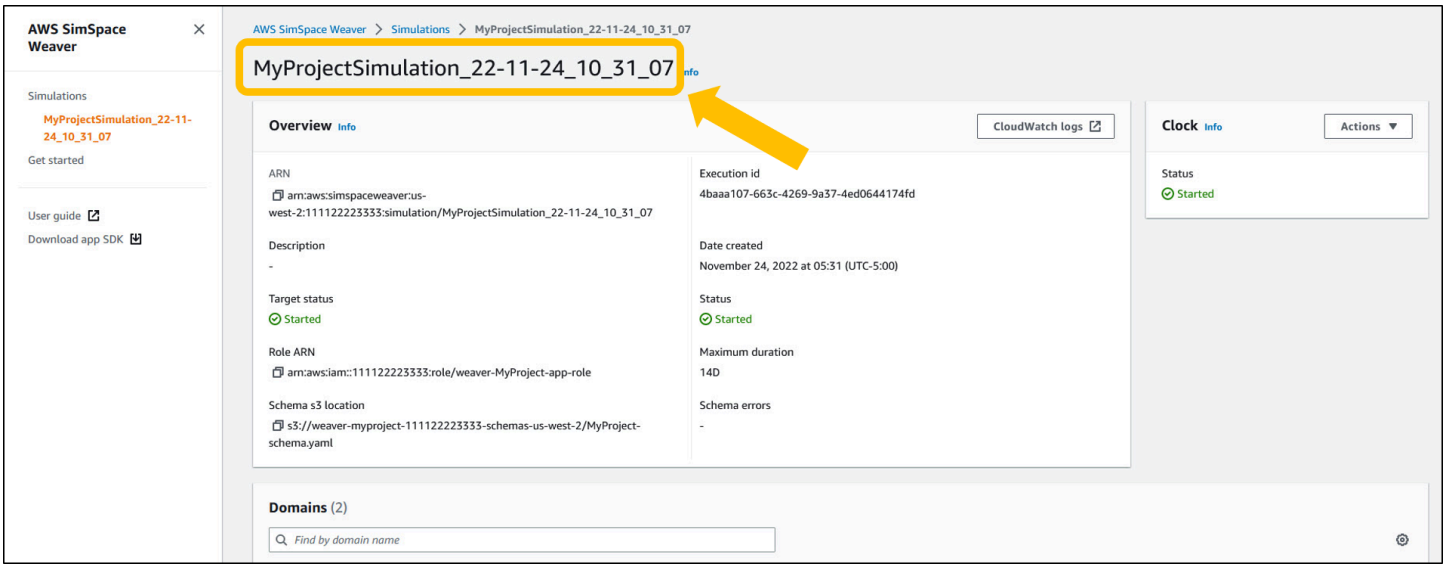
# SimSpace Weaver Amazon CloudWatch Logs 中的日誌

## 存取您的 SimSpace Weaver 日誌

從 SimSpace Weaver 模擬產生的所有日誌都存放在 Amazon CloudWatch Logs 中。若要存取您的日誌，您可以使用 SimSpace Weaver 主控台中模擬概觀窗格中的 CloudWatch 日誌按鈕，這將引導您直接前往該特定模擬的日誌。




您也可以透過 CloudWatch 主控台存取日誌。您需要模擬的名稱，才能搜尋其日誌。




# SimSpace Weaver 日誌

SimSpace Weaver 會將模擬管理訊息和主控台輸出從應用程式寫入 Amazon CloudWatch Logs。如需使用日誌的詳細資訊，請參閱《Amazon CloudWatch Logs 使用者指南》中的使用日誌群組和日誌串流。

您建立的每個模擬在 CloudWatch Logs 中都有自己的日誌群組。日誌群組的名稱是在模擬結構描述中指定。在下列結構描述程式碼片段中， 的值`log_destination_service`為 `logs`。這表示 的值`log_destination_resource_name`是日誌群組的名稱。在此情況下，日誌群組為 `MySimulationLogs`。

```
simulation_properties:
  log_destination_service: "logs"
  log_destination_resource_name: "MySimulationLogs"
  default_entity_index_key_type: "Vector3<f32>"
```

您也可以在啟動日誌群組之後，使用 DescribeSimulation API 來尋找日誌群組的名稱以進行模擬。

```
aws simspaceweaver describe-simulation --simulation simulation-name
```

下列範例顯示DescribeSimulation描述記錄組態的 輸出部分。日誌群組的名稱會顯示在 的結尾LogGroupArn。

```
    "LoggingConfiguration": {
        "Destinations": [
            {
                "CloudWatchLogsLogGroup": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:111122223333:log-
group:MySimulationLogs"
                }
            }
        ]
    },
```

每個模擬日誌群組包含數個日誌串流：

- 管理日誌串流 – SimSpace Weaver 服務產生的模擬管理訊息。

  ```
  /sim/management
  ```

- 錯誤日誌串流 – SimSpace Weaver 服務產生的錯誤訊息。只有在發生錯誤時，此日誌串流才會存在。 會將應用程式寫入的錯誤 SimSpace Weaver 存放在其自己的應用程式日誌串流中 （請參閱下列日誌串流）。

  ```
  /sim/errors
  ```

- 空間應用程式日誌串流（每個工作者上的每個空間應用程式 1 個）– 空間應用程式產生的主控台輸出。每個空間應用程式都會寫入自己的日誌串流。*spatial-app-id* 是 結尾斜線後面的所有字元*worker-id*。

```
/domain/spatial-domain-name/app/worker-worker-id/spatial-app-id
```

- 自訂應用程式日誌串流（每個自訂應用程式執行個體 1 個）– 自訂應用程式產生的主控台輸出。每個自訂應用程式執行個體都會寫入自己的日誌串流。

```
/domain/custom-domain-name/app/custom-app-name/random-id
```

- 服務應用程式日誌串流（每個服務應用程式執行個體 1 個）– 服務應用程式產生的主控台輸出。每個服務應用程式都會寫入自己的日誌串流。*service-app-id* 是 結尾斜線後面的所有字元*service-app-name*。

```
/domain/service-domain-name/app/service-app-name/service-app-id
```

# SimSpace Weaver 使用 Amazon CloudWatch 進行監控

您可以使用 SimSpace Weaver Amazon CloudWatch 監控 ，它會收集原始資料並將其處理為可讀且近乎即時的指標。這些統計資料會保留 15 個月，以便您存取歷史資訊，並更清楚 Web 應用程式或服務的執行效能。您也可以設定留意特定閾值的警示，當滿足這些閾值時傳送通知或採取動作。如需詳細資訊，請參閱 Amazon CloudWatch 使用者指南。

SimSpace Weaver 服務會在 `AWS/simspaceweaver` 命名空間中報告下列指標。

## SimSpace Weaver 帳戶層級的 指標

SimSpace Weaver 命名空間包含與 AWS 帳戶層級活動相關的下列指標。

| 指標 | 描述 |
| --- | --- |
| `SimulationCount` | 目前帳戶的模擬數目。<br><br>單位：計數<br><br>維度：無<br><br>統計資料：平均、最小值、最大值 |

# 使用 記錄 AWS SimSpace Weaver API 呼叫 AWS CloudTrail

AWS SimSpace Weaver 已與 服務整合 AWS CloudTrail，此服務提供使用者、角色或 AWS 服務在其中採取之動作的記錄 SimSpace Weaver。CloudTrail 會將 SimSpace Weaver 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 SimSpace Weaver 主控台的呼叫，以及對 SimSpace Weaver API 操作的程式碼呼叫。如果您建立線索，您可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 的事件 SimSpace Weaver。如果您未設定線索，仍然可以在 的 CloudTrail 主控台中檢視最新的事件Event history。您可以使用 CloudTrail 所收集的資訊來判斷提出的請求 SimSpace Weaver、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》。

## SimSpace Weaver CloudTrail 中的資訊

當您建立帳戶 AWS 帳戶 時，您的 上會啟用 CloudTrail。在 中發生活動時 SimSpace Weaver，該活動會與 中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中Event history。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」。

若要持續記錄 中的事件 AWS 帳戶，包括 的事件 SimSpace Weaver，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。依預設，當您在主控台中建立追蹤時，該追蹤會套用至所有的 AWS 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案交付至您指定的 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，請參閱下列內容：

- 建立追蹤的概觀
- CloudTrail 支援的服務和整合
- 設定 CloudTrail 的 Amazon SNS 通知
- 接收多個區域的 CloudTrail 日誌檔案和接收多個帳戶的 CloudTrail 日誌檔案

CloudTrail 會記錄所有 SimSpace Weaver 動作，並記錄在 AWS SimSpace Weaver API 參考中。例如，對 `ListSimulations`、`DescribeSimulation` 和 `DeleteSimulation` 動作發出的呼叫會在 CloudTrail 記錄檔案中產生專案。

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：

- 是否使用根或 AWS Identity and Access Management (IAM) 使用者登入資料提出請求。
- 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 SimSpace Weaver 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。事件代表來自任何來源的單一請求，並包含所請求動作的相關資訊，例如動作的日期和時間、請求參數和其他詳細資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

以下範例顯示的是展示 `ListSimulations` 動作的 CloudTrail 日誌項目。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:aws-console-signin-utils",
        "arn": "arn:aws:sts::111122223333:assumed-role/ConsoleSigninRole/aws-console-
signin-utils",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/ConsoleSigninRole",
                "accountId": "111122223333",
                "userName": "ConsoleSigninRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-02-14T15:57:02Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-02-14T15:57:08Z",
    "eventSource": "simspaceweaver.amazonaws.com",
    "eventName": "ListSimulations",
    "awsRegion": "us-west-2",
```

```
    "sourceIPAddress": "192.0.2.10",
    "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/86.0.4240.0 Safari/537.36",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "1234abcd-1234-5678-abcd-12345abcd123",
    "eventID": "5678abcd-5678-1234-ab12-123abc123abc",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

# SimSpace Weaver 端點和配額

下表說明 SimSpace Weaver 的服務端點和服務配額。服務配額也稱為限制，是 的服務資源或操作數量上限 AWS 帳戶。如需詳細資訊，請參閱 AWS 一般參考中的 AWS 服務配額。

## 服務端點

| 區域名稱 | 區域 | 端點 | 通訊協定 |
| --- | --- | --- | --- |
| 美國東部 (維吉尼亞北部) | us-east-1 | simspaceweaver.us-east-1.amazonaws.com | HTTPS |
| 美國東部 (俄亥俄) | us-east-2 | simspaceweaver.us-east-2.amazonaws.com | HTTPS |
| 美國西部 (奧勒岡) | us-west-2 | simspaceweaver.us-west-2.amazonaws.com | HTTPS |
| 亞太區域 (新加坡) | ap-southeast-1 | simspaceweaver.ap-southeast-1.amazonaws.com | HTTPS |
| 亞太區域 (悉尼) | ap-southeast-2 | simspaceweaver.ap-southeast-2.amazonaws.com | HTTPS |
| 歐洲 (斯德哥爾摩) | eu-north-1 | simspaceweaver.eu-north-1.amazonaws.com | HTTPS |
| 歐洲 (法蘭克福) | eu-central-1 | simspaceweaver.eu-central-1.amazonaws.com | HTTPS |

| 區域名稱 | 區域 | 端點 | 通訊協定 |
| --- | --- | --- | --- |
| 歐洲 (愛爾蘭) | eu-west-1 | simspaceweaver.eu-west-1.amazonaws.com | HTTPS |
| AWS GovCloud （美國東部） | us-gov-east-1 | simspaceweaver.us-gov-east-1.amazonaws.com | HTTPS |
| AWS GovCloud （美國西部） | us-gov-west-1 | simspaceweaver.us-gov-west-1.amazonaws.com | HTTPS |

# Service Quotas

| 名稱 | 預設 | 可調整 | 描述 |
| --- | --- | --- | --- |
| 每個應用程式的運算資源單位 | 每個受支援的區域：4 | 否 | 您可以分配給每個應用程式的運算資源單位數量上限。 |
| 每個工作者的運算資源單位 | 每個支援的區域：17 | 否 | 每個工作者可用的運算資源單位數量。 |
| 每個實體的資料欄位 | 每個支援的區域：7 | 否 | 實體可以擁有的資料（非索引）欄位數目上限。 |
| 分割區中的實體 | 每個支援的區域：8，192 | 否 | 1 個分割區中的實體數量上限。 |
| 實體資料欄位大小 | 每個支援的區域：1，024 個位元組 | 否 | 實體的資料（非索引）欄位的大小上限。 |

| 名稱 | 預設 | 可調整 | 描述 |
| --- | --- | --- | --- |
| 工作者之間的實體轉移 | 每個受支援的區域：25 | 否 | 每個分割區和每個刻度的工作者之間實體轉移數量上限。 |
| 同一工作者上的實體轉移 | 每個受支援的區域：500 | 否 | 每個分割區和每個刻度在相同工作者上傳輸的實體數量上限。 |
| 每個實體的索引欄位 | 每個受支援的區域：1 | 否 | 實體可以擁有的索引欄位數目上限。 |
| 模擬的最大持續時間 （以天為單位） | 每個支援的區域：14 | 否 | 您可以指定為模擬持續時間上限的最大天數。即使您未指定 值，所有模擬都有最長的持續時間。模擬達到其最長持續時間時會自動停止。 |
| 每個運算資源單位的記憶體 | 每個支援的區域：1 GB | 否 | 應用程式為每個運算資源單位取得的隨機存取記憶體 (RAM) 數量。 |
| 每個工作者的遠端訂閱 | 每個支援的區域：24 | 否 | 每個工作者的遠端訂閱數量上限。 |
| 模擬計數 | 每個支援的區域：2 | 是 | 您帳戶中目標狀態為 STARTED 的模擬數目上限。您可以請求提高配額，最多 10 個。 |
| 模擬工作者 | 每個支援的區域：2 | 是 | 您可以指派給 1 個模擬的工作者數量上限。您可以請求提高配額，最多 10 個。 |

| 名稱 | 預設 | 可調整 | 描述 |
| --- | --- | --- | --- |
| 每個運算資源單位vCPUs | 每個支援的區域：2 | 否 | 應用程式為每個運算資源單位取得的虛擬中央處理單元 (vCPUs) 數量。 |

## 訊息配額

下列配額適用於 SimSpace Weaver Local和 中的應用程式傳訊 AWS 雲端。

| 名稱 | 預設 | 可調整 | 描述 |
| --- | --- | --- | --- |
| 訊息大小上限 | 每個支援的區域：256 個位元組 | 否 | 訊息承載的大小上限。 |
| 訊息傳送速率上限 | 每個支援的區域：128 | 否 | 每個應用程式每個刻度可傳送的訊息數量上限。 |

## 時鐘速率

模擬結構描述會指定模擬的時鐘速率 （ 也稱為刻度速率)。下表指定您可以使用的有效時鐘速率。

| 名稱 | 有效值 | 描述 |
| --- | --- | --- |
| 時鐘速率 | 每個支援的區域：「10」、「15」、「30」、「無限制」 | 模擬的有效時鐘速率。 |
| 時鐘速率 (1.13 和 1.12 版 ) | 每個支援的區域：10、15、30 | 模擬的有效時鐘速率。 |

## 的服務配額 SimSpace Weaver Local

下列服務配額SimSpace Weaver Local僅適用於 。所有其他配額也適用於 SimSpace Weaver Local。

| 名稱 | 預設 | 可調整 | 描述 |
| --- | --- | --- | --- |
| 分割區上限 | SimSpace Weaver Local：24 | 否 | 模擬的分割區數目上限。 |
| 應用程式上限 | SimSpace Weaver Local：24 | 否 | 模擬的應用程式（任何類型的）總數上限。 |
| 網域上限 | SimSpace Weaver Local：24 | 否 | 模擬的網域（任何類型）總數上限。 |
| 每個分割區的實體 | SimSpace Weaver Local：4，096 | 否 | 每個分割區中的實體數量上限。 |
| 每個實體的欄位數 | SimSpace Weaver Local：8 | 否 | 每個實體的欄位數量上限。 |
| 欄位大小 | SimSpace Weaver Local：1024 位元組 | 否 | 實體欄位的大小上限。 |

# 在 中進行故障診斷 SimSpace Weaver

主題



## AssumeRoleAccessDenied

如果您的模擬無法啟動，您可能會收到下列錯誤：

```
Unable to assume role arn:aws:iam::111122223333:role/weaver-project-name-app-role;
 verify the role exists and has trust policy on SimSpace Weaver
```

如果下列有關模擬的 AWS Identity and Access Management (IAM) 角色的其中一項為 true，則可能會收到此錯誤：

- Amazon Resource Name (ARN)是指不存在的 IAM 角色。
- 不允許新模擬名稱擔任角色的 IAM 角色信任政策。

檢查 以確認角色存在。如果角色存在，請檢查您的角色信任政策。下列範例信任政策`aws:SourceArn`中的 僅允許名稱開頭為 `MySimulation`的模擬（在帳戶 111122223333 中）擔任該角色。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "simspaceweaver.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:simspaceweaver:us-
west-2:111122223333:simulation/MySimulation*"
                }
            }
        }
    ]
}
```

若要允許名稱開頭為 的另一個模擬MyOtherSimulation擔任該角色，必須修改信任政策，如下列已編輯的範例所示：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "simspaceweaver.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": [
                        aws:SourceArn": "arn:aws:simspaceweaver:us-
west-2:111122223333:simulation/MySimulation*",
                        aws:SourceArn": "arn:aws:simspaceweaver:us-
west-2:111122223333:simulation/MyOtherSimulation*"
                        ]
                }
            }
        }
    ]
}
```

# InvalidBucketName

建立專案時，您可能會收到下列錯誤：

```
An error occurred (InvalidBucketName) when calling the CreateBucket operation: The
 specified bucket is not valid.
```

您收到此錯誤是因為 SimSpace Weaver 傳遞給 Amazon Simple Storage Service (Amazon S3) 的名稱違反儲存貯體命名規則（如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存貯體命名規則)。

SimSpace Weaver 應用程式 SDK 中的`create-project`指令碼會使用您提供給指令碼的專案名稱來建立儲存貯體名稱。儲存貯體名稱使用以下格式：

- 1.13.x 版或更新版本
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-`*`region`*
- 1.12.x 版
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-app-zips-`*`region`*
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-schemas-`*`region`*

例如，假設有下列專案屬性：

- 專案名稱： `MyProject`
- AWS 帳戶 號碼： `111122223333`
- AWS 區域: `us-west-2`

專案會有下列儲存貯體：

- 1.13.x 版或更新版本
  - `weaver-myproject-111122223333-us-west-2`
- 1.12.x 版
  - `weaver-myproject-111122223333-app-zips-us-west-2`
  - `weaver-myproject-111122223333-schemas-us-west-2`

您的專案名稱不得違反 Amazon S3 命名規則。您也必須使用夠短的專案名稱，以便`create-project`指令碼建立的儲存貯體名稱不會超過 Amazon S3 儲存貯體的名稱長度限制。

# ServiceQuotaExceededException

您可能會在啟動模擬時收到下列錯誤：

```
An error occurred (ServiceQuotaExceededException) when calling the StartSimulation
 operation: Failed to start simulation due to: simulation quota has already been
 reached.
```

如果您嘗試啟動新的模擬，但您的帳戶目前具有目標狀態為 STARTED 的模擬數量上限，則會收到此錯誤。這包括執行模擬、失敗的模擬，以及因為達到最長持續時間而停止的模擬。您可以刪除已停止或失敗的模擬，以允許您啟動新的模擬。如果所有模擬都在執行中，您可以停止並刪除執行中的模擬。如果您尚未達到請求限制，也可以請求提高服務配額。

# TooManyBuckets

建立專案時，您可能會收到下列錯誤：

```
An error occurred (TooManyBuckets) when calling the CreateBucket operation: You have
 attempted to create more buckets than allowed.
```

Amazon Simple Storage Service (Amazon S3) 對 AWS 帳戶中可以擁有的儲存貯體數量有限制（如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存貯體限制)。

您必須執行下列其中一項操作，才能繼續：

- 刪除您不需要的 2 個或多個現有 Amazon S3 儲存貯體。
- 請求提高 Amazon S3 限制（如需詳細資訊，請參閱《Amazon Simple Storage Service 使用者指南》中的儲存貯體限制)。
- 使用不同的 AWS 帳戶。

> **Note**
>
> 中的 `DeleteSimulation` API SimSpace Weaver 不會刪除與您的模擬相關聯的 Amazon S3 資源。當您不再需要模擬時，建議您移除所有與模擬相關聯的資源。

# 模擬啟動期間許可遭拒

開始模擬時，您可能會收到錯誤訊息，指出許可遭拒，或存取應用程式成品時發生錯誤。當您為模擬指定 SimSpace Weaver 未為您建立的 Amazon S3 儲存貯體 （透過主控台或 SimSpace Weaver 應用程式 SDK 指令碼） 時，可能會發生此問題。

以下是最可能的根本原因：

- 此服務沒有存取您在模擬結構描述中指定之一或多個 Amazon S3 儲存貯體的許可 – 請檢查您的應用程式角色許可政策、Amazon S3 儲存貯體政策和 Amazon S3 儲存貯體許可，以確保 `simspaceweaver.amazonaws.com`具有存取儲存貯體的正確許可。如需應用程式角色許可政策的詳細資訊，請參閱 為您 SimSpace Weaver 建立的許可。
- 您的 Amazon S3 儲存貯體可能與您的模擬不同 AWS 區域 – 模擬成品的 Amazon S3 儲存貯體必須與模擬 AWS 區域 位於相同位置。檢查您的 Amazon S3 主控台，查看您的儲存 AWS 區域 貯體所在位置。如果您的 Amazon S3 儲存貯體位於不同的 中 AWS 區域，請選取 AWS 區域 與您的模擬相同的儲存貯體。

# 使用 時與時間相關的問題 Docker

如果您使用 Docker並從 SimSpace Weaver 應用程式 SDK 執行指令碼時收到與時間相關的錯誤，原因可能是您的Docker虛擬機器時鐘不正確。如果您的電腦正在執行，Docker然後從睡眠或休眠中恢復，就可能會發生這種情況。

要嘗試的解決方案

- 重新啟動 Docker。
- 在 中停用然後重新啟用時間同步Windows PowerShell：

```
Get-VMIntegrationService -VMName DockerDesktopVM -Name "Time Synchronization" |
 Disable-VMIntegrationService
Get-VMIntegrationService -VMName DockerDesktopVM -Name "Time Synchronization" |
 Enable-VMIntegrationService
```

# PathfindingSample 主控台用戶端無法連線

當您連線到 教學課程中所述的PathfindingSample模擬時，您可能會從主控台用戶端收到下列錯誤入門 SimSpace Weaver。發生此錯誤是因為用戶端無法在您提供的合併 IP ViewApp 地址和連接埠號碼開啟與 的網路連線。

```
Fatal error in function nng_dial. Error code: 268435577. Error message: no link
```

對於 中的模擬 AWS 雲端

- 您的網路連線是否正常運作？ 確認您可以連線到其他 IP 地址或應該運作的網站。請確定您的 Web 瀏覽器未從其快取載入網站。
- 您的模擬是否正在執行？ 您可以使用 ListSimulations API 來取得模擬的狀態。如需詳細資訊，請參閱取得自訂應用程式的 IP 地址和連接埠號碼。您也可以使用 SimSpace Weaver 主控台來檢查模擬的狀態。
- 您的應用程式是否正在執行？ 您可以使用 DescribeApp API 來取得應用程式的狀態。如需詳細資訊，請參閱取得自訂應用程式的 IP 地址和連接埠號碼。您也可以使用 SimSpace Weaver 主控台來檢查模擬的狀態。
- 您的應用程式是否正在執行？ 您可以使用 DescribeApp API 來取得應用程式的狀態。如需詳細資訊，請參閱取得自訂應用程式的 IP 地址和連接埠號碼。您也可以使用 SimSpace Weaver 主控台來檢查模擬的狀態。
- 您是否使用正確的 IP 地址和連接埠號碼？ 當您透過網際網路連線時，您必須使用 的 IP 地址和Actual連接埠號碼ViewApp。您可以在 DescribeApp API 輸出的 EndpointInfo區塊中找到 IP Address和Actual連接埠號碼。您也可以使用 SimSpace Weaver 主控台，在ViewAppMyViewDomain詳細資訊頁面尋找 的 IP 地址 (URI) 和連接埠號碼 （ 輸入連接埠)。
- 您的網路連線是否通過防火牆？ 您的防火牆可能會封鎖與 IP 地址或連接埠號碼 （ 或兩者 ） 的連線。檢查您的防火牆設定或向您的防火牆管理員確認。

對於本機模擬

- 您可以連線到您的迴路地址 (127.0.0.1) 嗎？ 如果您在 Windows 中有ping命令列工具，您可以開啟命令提示視窗，並嘗試 ping 127.0.0.1。按 Ctrl-C 結束 ping。

```
ping 127.0.0.1
```

Example ping 輸出

```
C:\>ping 127.0.0.1

Pinging 127.0.0.1 with 32 bytes of data:
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Reply from 127.0.0.1: bytes=32 time=1ms TTL=128

Ping statistics for 127.0.0.1:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms
Control-C
^C
C:\>
```

如果 ping 顯示遺失封包，您可能有其他軟體（例如本機防火牆、安全設定或反惡意軟體）封鎖您的連線。

- 您的應用程式是否正在執行？ 您的本機模擬會針對每個應用程式執行不同的視窗。確定空間應用程式 和 的視窗ViewApp已開啟。如需詳細資訊，請參閱中的本機開發 SimSpace Weaver。
- 您是否使用正確的 IP 地址和連接埠號碼？ 連線至本機模擬tcp://127.0.0.1:7000時，您必須使用 。如需詳細資訊，請參閱中的本機開發 SimSpace Weaver。
- 您的本機安全軟體可能會封鎖您的連線嗎？ 檢查您的安全設定、本機防火牆或反惡意軟體程式，以查看它們是否封鎖 TCP 連接埠 127.0.0.1上的 連線7000。

# AWS CLI 無法辨識 **simspaceweaver**

如果 AWS CLI 提供錯誤，指出其不知道 SimSpace Weaver，請執行下列命令。

```
aws simspaceweaver help
```

如果您收到以下列幾行開頭並列出所有可用選項的錯誤，則您的 AWS CLI 可能是較舊的版本。

```
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
```

```
  aws <command> help
  aws <command> <subcommand> help

aws: error: argument command: Invalid choice, valid choices are:
```

執行下列命令來檢查 的版本 AWS CLI。

```
aws --version
```

如果版本號碼早於 2.9.19則必須更新您的 AWS CLI。請注意， 的目前版本 AWS CLI 晚於 2.9.19。

若要更新您的 AWS CLI，請參閱《 第 2 版使用者指南》中的安裝或更新 的最新版本 AWS CLI。 AWS Command Line Interface

# SimSpace Weaver 模擬結構描述參考

SimSpace Weaver 使用 YAML 檔案來設定模擬的屬性。此檔案稱為模擬結構描述 （或只是結構描述)。 SimSpace Weaver 應用程式 SDK 中包含的範例模擬包含一個結構描述，您可以複製和編輯該結構描述以供您自己的模擬使用。

主題



## 完整結構描述的範例

下列範例顯示描述 SimSpace Weaver 模擬的 YAML格式化文字檔案。此範例包含屬性的虛擬值。檔案格式會根據其中sdk_version指定的 值而有所不同。結構描述格式 如需屬性及其有效值的完整說明，請參閱 。

```
sdk_version: "1.17"
simulation_properties:
  log_destination_resource_name: "MySimulationLogs"
  log_destination_service: "logs"
  default_entity_index_key_type: "Vector3<f32>"
  default_image: "111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-
repository:latest"
workers:
  MyComputeWorkers:
    type: "sim.c5.24xlarge"
    desired: 3
clock:
  tick_rate: "30"
partitioning_strategies:
  MyGridPartitioning:
    topology: "Grid"
    aabb_bounds:
      x: [-1000, 1000]
      y: [-1000, 1000]
    grid_placement_groups:
      x: 3
      y: 3
domains:
```

```
  MyCustomDomain:
    launch_apps_via_start_app_call: {}
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MyViewApp.zip"
      launch_command: ["MyViewApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports: [9000, 9001]
  MyServiceDomain:
    launch_apps_per_worker:
      count: 1
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/
MyConnectionServiceApp.zip"
      launch_command: ["MyConnectionServiceApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 9000
          - 9001
  MySpatialDomain:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 6
        y: 6
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MySpatialApp.zip"
      launch_command: ["MySpatialApp"]
      required_resource_units:
        compute: 1
  MySpatialDomainWithCustomContainer:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "MyGridPartitioning"
      grid_partition:
        x: 6
        y: 6
    app_config:
      package: "s3://weaver-myproject-111122223333-us-west-2/MySpatialApp2.zip"
      launch_command: ["MySpatialApp2"]
      required_resource_units:
        compute: 1
```

```
      image: "111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest"
placement_constraints:
  - placed_together: ["MySpatialDomain", "MySpatialDomainWithCustomContainer"]
    on_workers: ["MyComputeWorkers"]
```

# 結構描述格式

下列範例顯示結構描述的整體結構。只要父子關係相同，結構描述每個層級的屬性順序並不重要。順序對陣列中的元素很重要。

```
sdk_version: "sdk-version-number"
simulation_properties:
  simulation-properties
workers:
  worker-group-configurations
clock:
  tick_rate: tick-rate
partitioning_strategies:
  partitioning-strategy-configurations
domains:
  domain-configurations
placement_constraints:
  placement-constraints-configuration
```

章節



## SDK 版本

sdk_version 區段（必要）識別支援此結構描述 SimSpace Weaver 的應用程式 SDK 版本。有效值：1.17、1.16、1.15、1.14、1.13、1.12

> **Important**
>
> 的值sdk_version僅包含主要版本編號和第一個次要版本編號。例如，值1.12會指定所有版本 1.12.x，例如 1.12.0、 1.12.1和 1.12.2。

```
sdk_version: "1.17"
```

## 模擬屬性

simulation_properties 區段（必要）指定模擬的各種屬性。使用本節來設定記錄並指定預設容器映像。即使您未設定記錄或選擇指定預設容器映像，仍需要此區段。

```
simulation_properties:
  log_destination_resource_name: "log-destination-resource-name"
  log_destination_service: "log-destination-service"
  default_entity_index_key_type: "Vector3<f32>"
  default_image: "ecr-repository-uri"
```

屬性

log_destination_resource_name

指定 SimSpace Weaver 將寫入日誌的資源。

必要：否。如果未包含此屬性， SimSpace Weaver 不會撰寫模擬的日誌。

類型：字串

有效值：

- CloudWatch Logs 日誌群組的名稱（例如，MySimulationLogs)
- CloudWatch Logs 日誌群組的 Amazon Resource Name (ARN)（例如，arn:aws:logs:us-west-2:111122223333:log-group/MySimulationLogs)

> **Note**
>
> SimSpace Weaver 僅支援相同帳戶和 AWS 區域 模擬中的日誌目的地。

log_destination_service

當您指定不是 ARN 的 時logging_destination resource_name，表示記錄目的地資源的類型。

必要：如果log_destination_resource_name已指定 且 不是 ARN，則必須指定此屬性。如果log_destination_resource_name未指定 或 是 ARN，則無法指定此屬性。

類型：字串

有效值：

- logs：日誌目的地資源是日誌群組。

default_entity_index_key_type

指定模擬實體索引鍵欄位的資料類型。

必要：是

類型：字串

有效值：Vector3<f32>

default_image

指定模擬的預設容器映像（版本 1.13和 不支援）1.12。如果指定此屬性，則未指定 的網域image會使用 default_image。

必要：否

類型：字串

有效值：

- Amazon Elastic Container Registry (Amazon ECR) 中儲存庫的 URI（例如，111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

## 工作程序

workers 區段（必要）會指定工作者群組（工作者群組）的組態。 SimSpace Weaver 會搭配使用此資訊與 placement_constraints來設定模擬的基礎基礎設施。目前僅支援 1 個工作者群組。

若要指定工作者群組的屬性，*worker-group-name*請以您選擇的名稱取代 。名稱長度必須為 3-64 個字元，且可包含 A -Z、a -z、0 -9 和 _ - （連字號）。在名稱後面指定工作者群組的屬性。

```
workers:
  worker-group-name:
    type: "sim.c5.24xlarge"
    desired: number-of-workers
```

**屬性**

`type`

指定工作者類型。

必要：是

類型：字串

有效值：`sim.c5.24xlarge`

`desired`

指定此工作者群組所需的工作者數量。

必要：是

類型：整數

有效值：1-3。模擬工作者數量的服務配額 （限制） 會決定此屬性的最大值。例如，如果您的服務配額是 2，則此屬性的最大值為 2。您可以請求提高服務配額。如需詳細資訊，請參閱SimSpace Weaver 端點和配額。

## 時鐘

`clock` 區段 （必要） 指定模擬時鐘的屬性。

```
clock:
  tick_rate: tick-rate
```

**屬性**

`tick_rate`

指定時鐘發佈至應用程式的每秒刻度數。

必要：是

類型：

- 版本 *1.14*和 *1.15*：字串
- 版本 *1.13*和 *1.12*：整數

有效值：

- 版本 *1.14*和 *1.15*： `"10" | "15" | "30" | "unlimited"`
  - `"unlimited"`：當所有應用程式完成目前刻度的遞交操作時，時鐘會傳送下一個刻度。
- 版本 *1.13*和 *1.12*： `10 | 15 | 30`

## 分割策略

`partitioning_strategies` 區段（必要）指定空間網域的分割區組織。

> **Note**
>
> SimSpace Weaver 僅支援 1 個分割策略。

若要指定分割策略的屬性，請以您選擇的名稱取代 *partitioning-strategy-name*。名稱長度必須為 3-64 個字元，且可包含 A -Z、a -z、0 -9 和 _ -（連字號）。在名稱後面指定分割策略的屬性。

```
partitioning_strategies:
  partitioning-strategy-name:
    topology: "Grid"
    aabb_bounds:
      x: [aabb-min-x, aabb-max-x]
      y: [aabb-min-y, aabb-max-y]
    grid_placement_groups:
      x: number-of-placement-groups-along-x-axis
      y: number-of-placement-groups-along-y-axis
```

**屬性**

`topology`

指定此分割策略的拓撲 （分割區排列方案）。

必要：是

類型：字串

有效值："Grid"

`aabb_bounds`

指定(AABB)模擬主軸對齊週框方塊的邊界。您可以將邊界指定為 2 元素排序陣列，描述每個軸的最小值和最大值 （依該順序） (x 和 y)。

必要：有條件限制。如果拓撲設定為 ，則需要此屬性 （且只能指定）"Grid"。

類型：Float陣列 （針對每個軸）

有效值：-3.4028235e38-3.4028235e38

`grid_placement_groups`

指定網格拓撲中沿著每個軸 (x 和 y) 放置群組的數量。置放群組是空間上相鄰的分割區集合 （位於相同網域中）。

必要：有條件限制。如果拓撲設定為 ，則需要此屬性 （且只能指定）"Grid"。如果您未指定置放群組組態， SimSpace Weaver 會為您計算一個。任何使用沒有置放群組組態的分割策略的網域都必須指定 `grid_partition`（請參閱 空間網域分割策略)。

類型：整數 （適用於每個軸）

有效值：1-20。我們建議 x * y 等於所需的工作者數量。否則， SimSpace Weaver 會嘗試平衡所有可用工作者的置放群組。

## 網域

`domains` 區段 （必要） 指定每個網域的屬性。所有模擬必須至少有一個空間網域區段。您可以為其他網域建立多個區段。每種網域類型都有自己的組態格式。

> **Important**
>
> 版本 `1.13`和 `1.12` 不支援多個空間網域。

> **Important**
>
> SimSpace Weaver 每個模擬最多支援 5 個網域。這包括所有空間、自訂和服務網域。

```
domains:
  domain-name:
    domain-configuration
  domain-name:
    domain-configuration
  ...
```

網域組態

- 空間網域組態
- 自訂網域組態
- 服務網域組態

## 空間網域組態

若要指定空間網域的屬性，請以您選擇的名稱取代 *spatial-domain-name*。名稱長度必須為 3-64 個字元，且可包含 A -Z、a -z、0 -9 和 _ - （連字號）。在名稱後面指定空間網域的屬性。

```
  spatial-domain-name:
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "partitioning-strategy-name"
      grid_partition:
        x: number-of-partitions-along-x-axis
        y: number-of-partitions-along-y-axis
    app_config:
      package: "app-package-s3-uri"
      launch_command: ["app-launch-command", "parameter1", ...]
      required_resource_units:
        compute: app-resource-units
    image: "ecr-repository-uri"
```

空間網域分割策略

`launch_apps_by_partitioning_strategy` 區段（必要）指定模擬空間的分割策略和維度（分割區數量）。

```
    launch_apps_by_partitioning_strategy:
      partitioning_strategy: "partitioning-strategy-name"
      grid_partition:
        x: number-of-partitions-along-x-axis
        y: number-of-partitions-along-y-axis
```

屬性

partitioning_strategy

指定此空間網域的分割策略。

必要：是

類型：字串

有效值：此屬性的值必須符合 partitioning_strategies區段中定義的分割策略名稱。如需詳細資訊，請參閱分割策略。

grid_partition

指定網格拓撲中每個軸 (x 和 y) 的分割區數量。這些維度說明此網域的總模擬空間。

必要：有條件限制。只有在拓撲設定為 時，才能指定此屬性"Grid"。此屬性取決於此網域指定分割策略的grid_placement_groups屬性：

- 如果此網域的分割策略未指定grid_placement_groups組態，則需要此屬性。
- 如果有grid_placement_groups組態，但您未指定 grid_partition ，則 SimSpace Weaver 將使用與grid_placment_groups組態相同的維度。
- 如果您同時指定 grid_placement_groups和 grid_partition ， 的維度grid_partition必須是 維度的倍數 grid_placement_groups（例如，如果您的grid_placement_groups維度為 2x2，則 的部分有效維度grid_partition為 2x2、4x4、6x6、8x8、10x10)。

類型：整數 （每個軸）

有效值：1-20

空間應用程式組態

app_config 區段 （必要） 指定此網域中應用程式的套件、啟動組態和資源需求。

```
  app_config:
    package: "app-package-s3-uri"
    launch_command: ["app-launch-command", "parameter1", ...]
    required_resource_units:
      compute: app-resource-units
```

屬性

`package`

指定包含應用程式可執行檔/二進位檔的套件 (zip 檔案 ) 。套件必須存放在 Amazon S3 儲存貯體中。僅支援 zip 檔案格式。

必要：是

類型：字串

有效值：Amazon S3 儲存貯體中套件的 Amazon S3 URI。例如：`s3://weaver-myproject-111122223333-app-zips-us-west-2/MySpatialApp.zip`。

`launch_command`

指定可執行檔/二進位檔案名稱和命令列參數以啟動應用程式。每個命令列字串字符都是陣列中的元素。

必要：是

類型：字串陣列

`required_resource_units`

指定 SimSpace Weaver 應配置給此應用程式每個執行個體的資源單位數量。資源單位是工作者(RAM)上固定數量的虛擬中央處理單位(vCPUs)和隨機存取記憶體。如需資源單位的詳細資訊，請參閱 端點和服務配額。`compute` 屬性會指定`compute`系列工作者的資源單位配置，目前是唯一有效的配置類型。

必要：是

類型：整數

有效值：1-4

### 自訂容器映像

image 屬性（選用）會指定容器映像的位置，該映像 SimSpace Weaver 會用來在此網域中執行應用程式（版本 1.13和 不支援）1.12。將 URI 提供給 Amazon Elastic Container Registry (Amazon ECR) 中包含映像的儲存庫。如果未指定此屬性，但在頂層simulation_properties區段中指定 default_image ，則此網域中的應用程式會使用 default_image。如需詳細資訊，請參閱自訂容器。

```
    image: "ecr-repository-uri"
```

#### 屬性

image

指定容器映像的位置，以在此網域中執行應用程式。

必要：否

類型：字串

有效值：

- Amazon Elastic Container Registry (Amazon ECR) 中儲存庫的 URI（例如，111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

## 自訂網域組態

若要指定自訂網域的屬性，*custom-domain-name*請以您選擇的名稱取代 。名稱長度必須為 3-64 個字元，且可包含字元 A -Z、a -z、0 -9 和 _ -（連字號）。在名稱後面指定自訂網域的屬性。為每個自訂網域重複此程序。

```
  custom-domain-name:
    launch_apps_via_start_app_call: {}
    app_config:
      package: "app-package-s3-uri"
      launch_command: ["app-launch-command", "parameter1", ...]
      required_resource_units:
        compute: app-resource-units
      endpoint_config:
```

```
        ingress_ports: [port1, port2, ...]
    image: "ecr-repository-uri"
```

屬性

launch_apps_via_start_app_call

需要此屬性，才能使用 StartApp API 啟動您的自訂應用程式。

必要：是

類型：N/A

有效值：{}

自訂應用程式組態

app_config section （必要） 指定此自訂網域中應用程式的套件、啟動組態、資源需求和網路連接埠。

```
    app_config:
      package: "app-package-s3-uri"
      launch_command: ["app-launch-command", "parameter1", ...]
      required_resource_units:
        compute: app-resource-units
      endpoint_config:
        ingress_ports: [port1, port2, ...]
```

屬性

package

指定包含應用程式可執行檔/二進位檔的套件 (zip 檔案）。套件必須存放在 Amazon S3 儲存貯體中。僅支援 zip 檔案格式。

必要：是

類型：字串

有效值：Amazon S3 儲存貯體中套件的 Amazon S3 URI。例如：s3://weaver-myproject-111122223333-app-zips-us-west-2/MyCustomApp.zip。

`launch_command`

指定可執行檔/二進位檔案名稱和命令列參數以啟動應用程式。每個命令列字串字符都是陣列中的元素。

必要：是

類型：字串陣列

`required_resource_units`

指定應配置給此應用程式每個執行個體的資源單位 SimSpace Weaver 數量。資源單位是工作者 (RAM)上固定數量的虛擬中央處理單元(vCPUs)和隨機存取記憶體。如需資源單位的詳細資訊，請參閱 端點和服務配額。`compute` 屬性會指定`compute`系列工作者的資源單位配置，目前是唯一有效的配置類型。

必要：是

類型：整數

有效值：1-4

`endpoint_config`

指定此網域中應用程式的網路端點。的值`ingress_ports`指定自訂應用程式繫結至傳入用戶端連線的連接埠。 SimSpace Weaver 會將動態配置的連接埠映射至您指定的輸入連接埠。輸入連接埠同時是 TCP 和 UDP。使用 DescribeApp API 尋找實際連接埠號碼來連接用戶端。

必要：否。如果您未指定端點組態，則此網域中的自訂應用程式將不會有網路端點。

類型：整數陣列

有效值：`1024-49152`。值必須是唯一的。

自訂容器映像

`image` 屬性 （選用） 會指定容器映像的位置，該映像 SimSpace Weaver 會用來在此網域中執行應用程式 （版本 `1.13`和 不支援）`1.12`。將 URI 提供給 Amazon Elastic Container Registry (Amazon ECR) 中包含映像的儲存庫。如果未指定此屬性，但在頂層`simulation_properties`區段中指定 `default_image` ，則此網域中的應用程式會使用 `default_image`。如需詳細資訊，請參閱自訂容器。

```
    image: "ecr-repository-uri"
```

屬性

image

指定容器映像的位置，以在此網域中執行應用程式。

必要：否

類型：字串

有效值：

- Amazon Elastic Container Registry (Amazon ECR) 中儲存庫的 URI （例如，111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

## 服務網域組態

若要指定服務網域的屬性，請以您選擇的名稱取代 *service-domain-name*。名稱長度必須為 3-64 個字元，且可包含 A -Z、a -z、0 -9 和 _ - （連字號）。在名稱後面指定服務網域的屬性。為每個服務網域重複此程序。

```
service-domain-name:
  launch_apps_per_worker:
    count: number-of-apps-to-launch
  app_config:
    package: "app-package-s3-uri"
    launch_command: ["app-launch-command", "parameter1", ...]
    required_resource_units:
      compute: app-resource-units
    endpoint_config:
      ingress_ports: [port1, port2, ...]
  image: "ecr-repository-uri"
```

每個工作者啟動應用程式

launch_apps_per_worker 區段 （必要） 表示這是服務網域組態，並指定每個工作者要啟動的服務應用程式數量。

```
    launch_apps_per_worker:
      count: number-of-apps-to-launch
```

屬性

count

此屬性指定每個工作者要啟動的服務應用程式數量。

必要：是

類型：整數

有效值： {} | 1 | 2。值 {}指定預設值 1。

服務應用程式組態

app_config section （必要） 指定此服務網域中應用程式的套件、啟動組態、資源需求和網路連接埠。

```
    app_config:
      package: "app-package-s3-uri"
      launch_command: ["app-launch-command", "parameter1", ...]
      required_resource_units:
        compute: app-resource-units
      endpoint_config:
        ingress_ports: [port1, port2, ...]
```

屬性

package

指定包含應用程式可執行檔/二進位檔的套件 (zip 檔案）。套件必須存放在 Amazon S3 儲存貯體中。僅支援 zip 檔案格式。

必要：是

類型：字串

有效值：Amazon S3 儲存貯體中套件的 Amazon S3 URI。例如：s3://weaver-myproject-111122223333-app-zips-us-west-2/MyServiceApp.zip。

`launch_command`

指定可執行檔/二進位檔案名稱和命令列參數以啟動應用程式。每個命令列字串字符都是陣列中的元素。

必要：是

類型：字串陣列

`required_resource_units`

指定應配置給此應用程式每個執行個體的資源單位 SimSpace Weaver 數量。資源單位是工作者 (RAM)上固定數量的虛擬中央處理單位(vCPUs)和隨機存取記憶體。如需資源單位的詳細資訊，請參閱 端點和服務配額。`compute` 屬性會指定`compute`系列工作者的資源單位配置，目前是唯一有效的配置類型。

必要：是

類型：整數

有效值：1-4

`endpoint_config`

指定此網域中應用程式的網路端點。的值`ingress_ports`指定服務應用程式為傳入用戶端連線繫結的連接埠。 會將動態配置的連接埠 SimSpace Weaver 映射到您指定的輸入連接埠。輸入連接埠同時是 TCP 和 UDP。使用 DescribeApp API 尋找實際連接埠號碼來連接用戶端。

必要：否。如果您未指定端點組態，則此網域中的服務應用程式將不會有網路端點。

類型：整數陣列

有效值：`1024-49152`。值必須是唯一的。

自訂容器映像

`image` 屬性 （選用） 會指定容器映像的位置，該映像 SimSpace Weaver 會用來在此網域中執行應用程式 （版本 `1.13`和 不支援）`1.12`。將 URI 提供給 Amazon Elastic Container Registry (Amazon ECR) 中包含映像的儲存庫。如果未指定此屬性，但在頂層`simulation_properties`區段中指定 `default_image` ，則此網域中的應用程式會使用 `default_image`。如需詳細資訊，請參閱自訂容器。

```
    image: "ecr-repository-uri"
```

**屬性**

`image`

指定容器映像的位置，以在此網域中執行應用程式。

必要：否

類型：字串

有效值：

- Amazon Elastic Container Registry (Amazon ECR) 中儲存庫的 URI（例如，`111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest`）

## 置放限制

`placement_constraints` 區段（選用）指定哪些空間網域 SimSpace Weaver 應該放在相同的工作者上。如需詳細資訊，請參閱設定空間網域。

> **Important**
>
> 版本 `1.13`和 `1.12` 不支援 `placement_constraints`。

```
placement_constraints:
  - placed_together: ["spatial-domain-name", "spatial-domain-name", ...]
    on_workers: ["worker-group-name"]
```

**屬性**

`placed_together`

指定 SimSpace Weaver 應放在一起的空間網域。

必要：是

類型：字串陣列

有效值：結構描述中指定的空間網域名稱

on_workers

指定 SimSpace Weaver 應放置網域的工作者群組。

必要：是

類型：1 元素字串陣列

有效值：結構描述中指定的工作者群組名稱

# SimSpace Weaver API 參考

SimSpace Weaver 有 2 組不同的應用程式程式設計界面 APIs)：

- 服務 APIs – 這些 APIs會控制 服務和服務資源，例如您的模擬、時鐘和應用程式。它們是主要 AWS 軟體開發套件 (SDK) 的一部分，您可以使用 AWS 命令列界面 (CLI) 來呼叫它們。如需 servcie APIs 的詳細資訊，請參閱 SimSpace Weaver API 參考。
- 應用程式開發套件 APIs – 這些 APIs會控制模擬中的資料。您可以在應用程式程式碼中使用它們來執行像是讀取和寫入實體欄位資料、使用訂閱，以及監控模擬中的事件。如需詳細資訊，請參閱您 SimSpace Weaver 應用程式 SDK 資料夾中的應用程式 SDK 文件： *sdk-folder*\SimSpaceWeaverAppSdk\documentation

> **Note**
>
> *sdk-folder* 是您解壓縮 SimSpaceWeaverAppSdkDistributable 套件的資料夾。

# AWS SimSpace Weaver 版本

我們持續改善 AWS SimSpace Weaver。如果您想要利用新功能和功能更新，則必須在我們發行新版本時下載最新的 SimSpace Weaver 應用程式 SDK。若要使用較新版本執行現有的模擬，您可能需要更新其結構描述和程式碼，然後啟動模擬的新執行個體。您不需要升級， 可以繼續執行具有先前版本的現有模擬。您可以查看此頁面，以查看版本之間的差異。目前支援所有版本。

> **Important**
>
> AWS SimSpace Weaver 使用者指南的最新版本僅涵蓋服務的最新版本。您可以在 AWS SimSpace Weaver Guide Catalog 中找到舊版的文件，可從主要文件登陸頁面取得。

## 最新版本

最新版本為：1.17.0

## 如何尋找目前的版本

如果您使用 SimSpace Weaver 應用程式 SDK 建立模擬，`create-project`指令碼會將 SDK 程式庫的版本下載到您 中的子儲存庫*`sdk-folder`*。包含 SDK 程式庫的子目錄具有包含 SDK 版本編號的名稱：`SimSpaceWeaverAppSdk-`*`sdk-version`*。例如，1.16.0 版的程式庫位於 中。`SimSpaceWeaverAppSdk-1.16.0`

您也可以在 的文字檔案中找到 SimSpace Weaver 應用程式 SDK `app_sdk_distributable_version.txt` 可分發套件的版本*`sdk-folder`*。

## 下載最新版本

使用下列其中一個連結來下載最新版本。

- 完成應用程式開發套件可分發套件
- 只有應用程式 SDK 程式庫

您也可以從 SimSpace Weaver 主控台下載完整的 SimSpace Weaver 應用程式 SDK 可分發套件 AWS Management Console。從導覽窗格選擇下載應用程式 SDK。

> **Warning**
>
> 請勿使用 AWS CLI 下載任何看起來是 SimSpace Weaver 應用程式開發套件可分發套件的內容。僅使用此頁面上的下載連結或 主控台中的下載連結。不支援任何其他下載方法或位置，且可能包含過時、不正確或惡意程式碼。

## 對應用程式開發套件下載進行故障診斷

我們使用 Amazon CloudFront (CloudFront) 來分發應用程式 SDK .zip 檔案。您可能會遇到下列一些情況。

- 下載的套件不是最新版本
  - 如果您下載的 .zip 檔案不包含最新版本，則 CloudFront 節點的快取可能尚未更新。24 小時後再次下載。
- 您使用下載連結收到 HTTP 4xx 或 5xx 錯誤
  - 請在 24 小時後再試一次。如果您收到相同的錯誤，請使用SimSpace Weaver 主控台底部的意見回饋連結來回報問題。選取將問題報告為意見回饋類型。
- 您的瀏覽器報告無法載入頁面
  - 您可能會遇到本機網路或瀏覽器組態問題。確認您可以載入其他頁面。清除您的瀏覽器快取，然後再試一次。請確定您沒有可能封鎖下載 URL 的防火牆規則。
- 當您嘗試儲存檔案時，出現錯誤
  - 檢查您的本機檔案系統許可，以確保您擁有儲存檔案的正確許可。
- 您的瀏覽器會顯示 AccessDenied
  - 如果您在瀏覽器中手動輸入 URL，請檢查 URL 是否正確。如果您使用下載連結，請確定某件事不會干擾瀏覽器中的 URL；請再次使用連結。

## 安裝最新版本

若要安裝最新版本

1. 下載最新版本。
2. 將 SimSpaceWeaverAppSdkDistributable.zip 解壓縮至資料夾。
3. `python setup.py` 從解壓縮的最新版本 SimSpace Weaver 應用程式 SDK 資料夾執行 。

4. 使用解壓縮的最新版本 SimSpace Weaver 應用程式 SDK 資料夾，而非先前的版本。

# 服務版本

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.17.0 | SimSpace Weaver 應用程式開發套件可分發套件的主要變更<br>• 我們將 Windows 批次和 Linux Bash 指令碼取代為 Python 型指令碼。因此，即使您不使用 （或打算使用） Python SDK，現在仍需要使用 Python 3.9 指令碼和範例。<br>• 此版本增加對 Amazon Linux 2 的支援。<br>• 我們修正了 中的幾個錯誤Sim Space Weaver Local。<br>如需詳細資訊，請參閱版本備註。 | 2024 年 4 月 17 日 | 本指南 | • 完整套件<br>• 僅限程式庫<br>請參閱故障診斷。 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| | 錯誤修正<br>• 我們修正了如果實體未完成遠端工作者之間的傳輸，會導致實體成為未擁有的錯誤。 | | | |
| 1.16.0 | 新功能：<br>• 您現在可以在 SimSpace Weaver 應用程式 SDK 中使用簡訊 APIs，在應用程式之間傳送和接收訊息。此功能適用於 C++、Python 以及 Unity 和 Unreal Engine 5 整合。 | 2024 年 2 月 12 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | • 完整套件<br>• 僅限程式庫<br><br>請參閱故障診斷。 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.15.3 | SimSpace Weaver Local 更新：<br>• 我們變更SimSpace Weaver Local以更緊密地使其與 的開發保持一致 AWS 雲端。這些變更會影響的 C++、Python、Unity 和 Unreal Engine 專案和工作流程SimSpace Weaver Local。 | 2023 年 12 月 4 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |
| 1.15.2 | 應用程式開發套件可分發套件更新：<br>• 我們已更新 `Dockerfile` 以使用特定必要版本的 cmake。Docker 容器建置可能會失敗，沒有此變更。 | 2023 年 11 月 2 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.15.1 | 功能更新：<br>• Python SDK：此版本修正導致 Python 型模擬在 中失敗的問題 AWS 雲端。 | 2023 年 9 月 22 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |
| 1.15.0 | 新功能：<br>• Python SDK：您現在可以在 Python 中開發模擬。 SimSpace Weaver 應用程式開發套件可分發套件包含範例 Python 專案及其 Python 檢視用戶端的範本。 | 2023 年 8 月 31 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.14.0 | 新功能：<br>• 自訂容器：建立您自己的 Amazon Linux 2 (AL2) 型容器映像，將其存放在 Amazon Elastic Container Registry (Amazon ECR) 中，並使用它在 中執行您的 SimSpace Weaver 應用程式 AWS 雲端。<br>• 多個空間網域：在模擬中建立多個空間網域。將模擬邏輯分開，而不是全部合併到單一空間應用程式中。根據空間網域的需求，將不同的資源配置給空間網域。<br>• 無限制的刻度率：讓您的模擬能夠以程式碼執行的速度執行。設定模擬的時鐘，以便在所有 | 2023 年 7 月 26 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| | 應用程式完成目前刻度的遞交操作後，立即傳送下一個刻度。<br><br>SimSpace Weaver 應用程式開發套件：<br><br>• 的值現在`tick_rate` 是字串。值必須包含雙引號 (")。舊版的刻度率仍然是整數。 | | | |
| 1.13.1 | SimSpace Weaver 應用程式開發套件：<br><br>• 功能更新：專案建立現在可正確搭配 `PathfindingSampleUnreal` 範本使用。 | 2023 年 6 月 7 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.13.0 | SimSpace Weaver 服務 APIs：<br>• • 新CreateSnapshot 動作<br>• StartSimulation 動作的變更：<br>• 新增從快照開始的SnapshotS3Location 參數。<br>• SchemaS3Location 參數現在是選用的。<br>• DescribeSimulation 輸出的變更：<br>• SchemaError 已棄用。<br>• 新增StartError 欄位。<br>• 新增SnapshotS3Location 欄位。<br>• 新增SNAPSHOT_ | 2023 年 4 月 29 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| | IN_PROGRESS 模擬狀態。<br>• 新S3Destination 資料類型<br><br>SimSpace Weaver 主控台：<br>• 用於建立快照的新功能。<br>• 從快照啟動模擬的新功能。<br><br>SimSpace Weaver 應用程式開發套件：<br>• 支援快照的新指令碼<br>• create-snapshot- *project-name* .bat<br>• start-from-snapshot- *project-name* .bat<br>• quick-start-from-snapshot- *project-name* -cli.bat | | | |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| | • list-snapshots- *projec name* .bat<br>• 專案現在每個專案使用單一 Amazon S3 儲存貯體：weaver-*lowercase -project-name -account-number -region* | | | |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.12.3 | SimSpace Weaver 應用程式開發套件：<br>• 下列指令碼現在支援 `--maximum-duration` 參數：<br>• `quick-start-`*`project-name`*`-cli.bat`<br>• `quick-start-`*`project-name`*`-cli.sh`<br>• `start-simulation-`*`proj`*`name.bat`<br>• `start-simulation-`*`proj`*`name.sh`<br>• `run-`*`project-name`*`.bat`<br>• `run-`*`project-name`*`.sh` | 2023 年 3 月 27 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.12.2 | SimSpace Weaver 應用程式開發套件：<br>• 錯誤修正：`docker-create-image.bat` 現在可正確執行。 | 2023 年 3 月 1 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |
| 1.12.1 | SimSpace Weaver 應用程式開發套件：<br>• 指令碼現在接受用於 AWS 身分驗證的 AWS CLI 設定檔。<br>• 指令碼現在支援 AWS IAM Identity Center AWS 身分驗證。<br><br>SimSpace Weaver Local:<br>• 錯誤修正：`true`如果所有空間應用程式都尚未加入模擬，`Api::BeginUpdateWillBlock` 現在會正確傳回 。 | 2023 年 2 月 28 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

| 版本 | 備註 | 版本日期 | 文件 | 應用程式開發套件下載 |
| --- | --- | --- | --- | --- |
| 1.12.0 | 一般可用性版本 (GA) | 2022 年 11 月 29 日 | 請參閱 AWS SimSpace Weaver 指南目錄。 | 無法下載 |

# AWS SimSpace Weaver 1.17.0 版

此版本是 SimSpace Weaver 應用程式 SDK 可分發套件的全面修訂。我們以 Python 型指令碼取代過時的 Windows 批次和 Linux Bash 指令碼。

> **Important**
>
> Python 3.9 現在需要使用指令碼和範例，而不只是 Python SDK。

內容



## 1.17.0 的主要變更

- 簡化專案建立
  - 執行 之後`setup.py`，您只需複製貼上範例，即可建立自己的專案。
- 一鍵式範例
  - 分佈 zip 檔案現在包含ready-to-use的範例。
- 每個 SDK 現在都存在於自己的目錄中：`cpp`、`unreal`、 `python`和 `unity`。您可能需要更新路徑，具體取決於您使用的開發套件。
- 改善協助程式指令碼。
  - 指令碼現在包含多個 AWS CLI 選項，以最大化其彈性。
  - 快速入門的整合式主控台用戶端啟動和連線。
  - 改善主控台輸出。

  - Unreal 和 Unity 範例建置現在可搭配 使用`quick-start`，無需再手動執行任何步驟。
  - SimSpace Weaver Local 現在只需呼叫 即可運作`quick-start`，無需再手動建置和啟動。
  - SimSpace Weaver Local `quick-start` 已整合支援記錄應用程式輸出。
  - SimSpace Weaver Local 現在可以在非 GUI 環境中啟動，例如在 ssh 工作階段中。
  - 「自訂容器」功能現在已整合至`quick-start`指令碼。
  - 增加 Amazon Linux 2 (AL2) 支援：Windows 和 AL2 的指令碼工作流程現在相當。先前，AL2 專案需要更多手動步驟SimSpace Weaver Local，且不支援 AL2。
- Unreal Engine 和 Unity 外掛程式現在包含在 SimSpace Weaver 應用程式開發套件可分發套件中。
- 的錯誤修正 SimSpace Weaver Local
  - 修正實體可以指派相同實體 ID 的錯誤。
  - 已修正兩個分割區可指派相同分割區 ID 的錯誤。
  - 已修正應用程式嘗試寫入非其擁有的實體的相關錯誤。
  - 已解決記憶體流失問題。

## 將專案更新至 1.17.0

1. 設定 1.17.0 分佈：再次完成設定程序，因為我們已將它們變更為 1.17.0。如需詳細資訊，請參閱設定 SimSpace Weaver。
2. 每個 Weaver 應用程式開發套件現在都存在於其自己的目錄中。更新您的建置路徑以反映這一點。

   a. C++ 目錄： `SimSpaceWeaverAppSdk/cpp`

      - SimSpace Weaver C++ 應用程式開發套件現在使用 `FindSimSpaceWeaverAppSdk.cmake` 檔案。此檔案會設定連結至 `weaver`的目標，並在 中包含為 Weaver 建置 時的重要錯誤修正 AWS 雲端。您應該使用此功能，而不是直接連結到二進位檔。

   b. Python 目錄： `SimSpaceWeaverAppSdk/python`

   c. Unity 外掛程式： `SimSpaceWeaverAppSdk/unity`

   d. Unreal Engine 外掛程式： `SimSpaceWeaverAppSdk/unreal`

3. 先前的`tools`指令碼不適用於新的 SimSpace Weaver 分佈。若要將新的`tools`指令碼與專案搭配使用：

   a. 刪除舊的 `tools/linux`、 `tools/windows`和 `tools/local`目錄。

b. 複製範例專案的`tools`目錄，該專案使用與您的專案相同的 SimSpace Weaver 應用程式 SDK。在複製此目錄`setup.py`之前，請確定您已執行 。

> **Important**
>
> 工具指令碼僅保證可與範例專案搭配使用。您可能需要編輯這些指令碼，特別是`build.py`指令碼，才能使用您的專案。任何編輯對於您的專案都是唯一的，因此我們無法提供任何指導。

## 關於 1.17.0 版的常見問題

我是否必須更新至 1.17.0 版？

這不是必要的更新，因為 SimSpace Weaver API 或 SimSpace Weaver 應用程式 SDK 沒有任何變更。如果您想要使用 1.17.0，您必須更新至 1.17.0SimSpace Weaver Local，其中包含數個錯誤修正。

所需的最低 Python 版本為何？

Python 3.9 是最低版本。

所需的最低 CMake 版本為何？

CMake 3.13 版是最低版本。

Unreal Engine 所需的最低版本是多少？

Unreal 引擎 5.0 是最小值。

所需的 Unity 最低版本為何？

Unity 的 版本為 2022.3.19.F1 下限。

# AWS SimSpace Weaver 1.15.1 版

此版本是 Python SDK 的必要更新，最初在 1.15.0 版中 SimSpace Weaver 發行。它修正了導致 Python 型模擬在 中失敗的版本不相符問題 AWS 雲端。使用此版本而非 1.15.0。

# 將現有的 Python 專案更新至 1.15.1

如果您有使用 1.15.0 版 Python SDK 建立的現有 Python 專案，您必須執行下列步驟，將其更新為 1.15.1，以便在 中執行 AWS 雲端。

您也可以使用 1.15.1 Python SDK 建立新的 Python 專案，並將自訂程式碼移至新專案，而不是遵循此程序。

將 1.15.0 Python 專案更新至 1.15.1

1. 移至 Python 專案的 資料夾。
2. 在`src/PythonBubblesSample/bin/run-python`變更下列行：

   ```
   export PYTHONPATH=$PYTHONPATH:/roapp/lib
   ```

   變更為：

   ```
   export PYTHONPATH=$PYTHONPATH:$LD_LIBRARY_PATH:/roapp/lib
   ```
3. 在 `CMakeLists.txt` 中刪除以下行：

   - ```
     file(COPY "${SDK_PATH}/libweaver_app_sdk_python_v1_$ENV{PYTHON_VERSION}.so"
      DESTINATION "${ZIP_FILES_DIR}/lib/weaver_app_sdk_v1")
     ```
   - ```
     file(RENAME "${ZIP_FILES_DIR}/lib/weaver_app_sdk_v1/libweaver_app_sdk_python_v1_
     $ENV{PYTHON_VERSION}.so" "${ZIP_FILES_DIR}/lib/weaver_app_sdk_v1/
     libweaver_app_sdk_python_v1.so")
     ```
   - ```
     message("    * COPYING WEAVER PYTHON SDK TO BUILD DIR ${ZIP_FILES_DIR}....")
     ```
   - ```
     file(COPY ${SDK_DIR} DESTINATION ${ZIP_FILES_DIR}/lib/weaver_app_sdk_v1)
     ```

# 1.15.1 版的故障診斷

## 更新 1.15.0 Python 模擬後，無法在 中啟動 AWS 雲端

徵狀：在您開始模擬約 5-10 分鐘後，模擬管理日誌會報告 ，`internal error`而模擬狀態為 `FAILED`。

如果應用程式 zip 檔案中包含 1.15.0 Python SDK 的程式庫檔案，就可能會發生這種情況。請確定您已完成更新專案的步驟，並確定 `libweaver_app_sdk_python_v1.so` 不在您的 zip 檔案中或以任何方式參考。

## 關於 1.15.1 版的常見問答集

此版本是否會影響 Python SDK 以外的任何內容？

否.

我是否必須更新至 1.15.1 版？

如果您不打算將 Python 用於空間應用程式，則不需要更新至 1.15.1。如果您更新至 1.15.0，您的 Python 型模擬將不會在 中執行 AWS 雲端。如果您使用 1.15.0，建議您更新至 1.15.1。

什麼是 **`$LD_LIBRARY_PATH`**？

當您的模擬在 中執行時，這是 Python SDK 的位置 AWS 雲端。這是 1.15.1 的新功能。我們進行這項變更，以避免未來發生 Python 版本問題。連結至該目錄的功能與 1.15.0 `libweaver_app_sdk_python_v1.so`中的連結相同。

# 的文件歷史記錄 AWS SimSpace Weaver

下表說明 SimSpace Weaver 文件的重要變更。

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2025 年 5 月 20 日 | 終止支援通知 | 終止支援通知： 將於 2026 AWS 年 5 月 20 日結束對 的支援 AWS SimSpace Weaver。2026 年 5 月 20 日之後，您將無法再存取 SimSpace Weaver 主控台或 SimSpace Weaver 資源。如需詳細資訊，請參閱AWS SimSpace Weaver 終止支援。 | N/A |
| 2024 年 4 月 17 日 | 已更新內容 | 在 1.17.0 版的使用者指南中更新。設定 章節和開始使用教學課程的主要變更。如需詳細資訊，請參閱版本備註。 | N/A |
| 2024 年 2 月 12 日 | 已更新內容 | 更新 1.16.0 版的AWS SimSpace Weaver 版本章節。 | N/A |
| 2024 年 2 月 12 日 | 新內容 | 新增 簡訊章節做為 1.16.0 版的一部分。本節說明新增至 SimSpace Weaver 應用程式 SDK APIs。您可以使用這些 APIs應用程式之間傳送和接收訊息。 | N/A |

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2024 年 2 月 12 日 | 已更新內容 | 更新 1.16.0 版的SimSpace Weaver 模擬結構描述參考章節。 | N/A |
| 2024 年 2 月 12 日 | 已更新內容 | 新增了 SimSpace Weaver 端點和配額章節的簡訊服務配額。 | N/A |
| 2024 年 2 月 12 日 | 新指南 | 將 1.16.0 之前的版本內容分割為單獨的指南。新增 AWS SimSpace Weaver Guide Catalog （可從主要文件登陸頁面取得）以存取舊版的指南。 | N/A |
| 2023 年 12 月 4 日 | 已更新內容 | 更新 1.15.3 版的AWS SimSpace Weaver 版本章節。 | N/A |
| 2023 年 12 月 4 日 | 已更新內容 | 更新AWS SimSpace Weaver 版本章節以包含最新版本的安裝指示。 | N/A |
| 2023 年 12 月 4 日 | 已更新內容 | 已更新 的服務配額 SimSpace Weaver Local。 | N/A |
| 2023 年 12 月 4 日 | 新的與更新的內容 | 重組 中的本機開發 SimSpace Weaver章節並新增頁面，說明 1.15.3 版中SimSpace Weaver Local介紹的 差異。 | N/A |

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2023 年 11 月 7 日 | 已更新內容 | 更新設定 Docker 和 WSL 的指示，以使用應用程式 SDK 的直接下載連結/URL。如需詳細資訊，請參閱設定 的本機環境 SimSpace Weaver。 | N/A |
| 2023 年 11 月 2 日 | 已更新內容 | 已更新 1.15.2 版的服務版本頁面。如需詳細資訊，請參閱服務版本。 | N/A |
| 2023 年 10 月 23 日 | 已更新內容 | 使用新指示更新服務版本頁面，以下載應用程式 SDK 可分發套件。客戶現在應該只使用我們核准的直接下載連結之一，而不是使用 AWS CLI 下載應用程式開發套件可分發套件。如需詳細資訊，請參閱下載最新版本。 | N/A |
| 2023 年 9 月 22 日 | 新內容 | 新增版本備註頁面，其中包含 1.15.1 版的更新指示。如需詳細資訊，請參閱AWS SimSpace Weaver 1.15.1 版。 | N/A |
| 2023 年 9 月 10 日 | 新內容 | 針對 AWS CLI 無法辨識的情況新增故障診斷區段 SimSpace Weaver。如需詳細資訊，請參閱AWS CLI 無法辨識 `simspaceweaver` 。 | N/A |

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2023 年 9 月 10 日 | 已更新內容 | 更新 WSL AWS CLI 中的安裝說明。如需詳細資訊，請參閱在 中設定 Amazon Linux 2(AL2) 的 SimSpace Weaver 分佈套件 Windows Subsystem for Linux (WSL)。 | N/A |
| 2023 年 9 月 7 日 | API 更新 | S3Location 資料類型現在需要 BucketName 和 ObjectKey。S3Destination 資料類型現在需要 BucketName。 | AWS SDK：2023-09-07 |
| 2023 年 8 月 31 日 | 新內容 | 新增了 1.15.0 版的新章節：使用 Python。 | N/A |
| 2023 年 8 月 15 日 | 已更新內容 | 更新 中的下載說明AWS SimSpace Weaver 版本，僅列出官方 SimSpace Weaver Amazon S3 儲存貯體。其他下載位置不受控制 AWS ，且可能包含惡意程式碼。 | N/A |
| 2023 年 7 月 26 日 | 已更新內容 | 已更新時鐘。 | N/A |
| 2023 年 7 月 26 日 | 已更新內容 | 已更新設定空間網域。 | N/A |
| 2023 年 7 月 26 日 | 新內容 | 新增章節：「自訂容器」。 | N/A |

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2023 年 7 月 26 日 | 已更新內容 | 針對 1.14.0 版AWS SimSpace Weaver 版本進行更新。 | N/A |
| 2023 年 7 月 6 日 | 新內容 | 新增章節：「PathfindingSample 主控台用戶端無法連線」。 | N/A |
| 2023 年 6 月 7 日 | 已更新內容 | 針對 1.13.1 版AWS SimSpace Weaver 版本進行更新。 | N/A |
| 2023 年 5 月 15 日 | 新內容 | 新增章節：「搭配 使用快照 AWS CloudFormation」。 | N/A |
| 2023 年 4 月 29 日 | 新內容 | 新增 1.13.0 版的內容。如需詳細資訊，請參閱AWS SimSpace Weaver 版本。 | AWS SDK：2023-04-28 |
| 2023 年 3 月 27 日 | 新內容 | 新增章節，說明模擬的最長持續時間。在 1.12.3 版的教學課程中新增了備註，將 `--maximum-duration` 參數的支援新增至 SimSpace Weaver 應用程式 SDK 指令碼。 | N/A |
| 2023 年 3 月 9 日 | 變更的內容 | 釐清我們只在 Docker 上提供 Windows和 的指示 Windows Subsystem for Linux (WSL)，且不支援 WSL （和任何其他 Linux 環境）。 | N/A |

| 日期 | 變更 | 說明文件更新 | API 版本已更新 |
| --- | --- | --- | --- |
| 2023 年 2 月 28 日 | 新內容 | 新增描述 SimSpace Weaver 版本的章節。 | N/A |
| 2023 年 2 月 28 日 | 變更的內容 | 變更身分驗證的相關內容，以包含對 AWS Command Line Interface (AWS CLI) 使用 AWS IAM Identity Center 和具名設定檔。 | N/A |
| 2023 年 2 月 17 日 | 新內容 | 新增有關使用 管理 資源的章節 AWS CloudFormation。 | N/A |
| 2023 年 1 月 23 日 | 新內容 | 新增偵錯本機模擬的指示。 | N/A |
| 2022 年 11 月 29 日 | 服務啟動 | 已發佈 的 使用者指南 和 API 參考 SimSpace Weaver。 | AWS SDK：2022-11-29 |

# 詞彙表

此詞彙表會定義特定的詞彙 AWS SimSpace Weaver。

如需最新的 AWS 術語，請參閱 AWS 一般參考中的AWS 詞彙表。

## A

**應用程式** 您建立的可執行程式碼 （ 也稱為二進位檔)。應用程式一詞可以參考該程式碼的程式碼或執行中的執行個體。應用程式會封裝模擬行為。應用程式會建立、刪除、讀取和更新實體。

**應用程式開發套件** 用來整合應用程式與 的軟體開發套件 (SDK) SimSpace Weaver。開發套件提供 APIs用於讀取和寫入實體資料，以及追蹤模擬時間。如需詳細資訊，請參閱SimSpace Weaver 應用程式開發套件。

## C

**用戶端** 存在於 外部 SimSpace Weaver 並透過自訂應用程式或服務應用程式與模擬互動的程序 （ 或其定義 ）。您可以使用用戶端來檢視或變更模擬狀態。

**時鐘** 內部排程程序 SimSpace Weaver的抽象。時鐘會將刻度發佈至應用程式，以維持時間同步。每個模擬都有自己的時鐘。

**時鐘速率** 時鐘發佈至應用程式的每秒刻度數。如需支援的時鐘速率的詳細資訊，請參閱 SimSpace Weaver 端點和配額。

**時鐘刻度率** 請參閱時鐘速率。

**運算資源單位** 工作者上的運算資源 （ 處理器和記憶體 ） 單位。應用程式的單一執行個體通常會配置 1 個運算資源單位。您可以為每個應用程式配置多個運算資源單位。

**自訂應用程式** 您用來讀取模擬狀態並與之互動的應用程式類型。自訂應用程式可以在模擬中建立實體，但不會擁有它們。當自訂應用程式建立實體時，必須將實體轉移到空間網域。您可以使用應用程式 APIs 控制自訂應用程式的生命週期。如需 SimSpace Weaver APIs的詳細資訊，請參閱 SimSpace Weaver API 參考。

| | |
|---|---|
| 自訂網域 | 包含自訂應用程式的網域。 |
| 自訂分割區 | 自訂應用程式的分割區。 |

## D

| | |
|---|---|
| 截止日期 | 操作（例如刻度處理）應完成的實際時間。 |
| 網域 | 執行相同可執行程式碼（應用程式二進位檔）並具有相同啟動選項的應用程式執行個體群組。 |

## E

| | |
|---|---|
| 端點（服務） | 程式（例如）用來連線至 SimSpace Weaver 服務的完整網域名稱 (FQDN AWS Command Line Interface)。 |
| 端點（模擬） | 用戶端用來連線至模擬的 IP 地址和連接埠號碼。您可以在自訂應用程式和服務應用程式上設定端點。 |
| 實體 | 客戶資料物件（或其定義）。實體可以是靜態（保留在一個位置）或動態（在模擬空間中移動）。例如，模擬中的人物和建築物。 |

## I

| | |
|---|---|
| 索引（模擬） | 模擬的空間屬性描述，包括其空間界限和座標系統。 |

## L

| | |
|---|---|
| 生命週期（應用程式） | 應用程式在模擬期間???經歷的預期邏輯步驟的描述。生命週期是受管 (SimSpace Weaver 啟動和停止應用程式）或未受管（啟動和停止應用程式）。 |
| load（實體欄位資料） | 從 讀取實體欄位資料State Fabric。 |

## P

| | |
|---|---|
| 分割區 | 工作者上共用記憶體的區段。每個分割區都包含網域內實體的分散子集。每個應用程式都有指派的分割區。應用程式擁有其分割區中的所有實體。當應用程式建立實體時，它會在其分割區中建立實體。當實體從一個分割區移至 |

另一個分割區時，擁有權會從來源分割區的應用程式傳輸到目的地分割區的應用程式。

## R

資源單位 請參閱 ???。

## S

結構描述 描述模擬組態的 YAML 或 JSON 文件。 SimSpace Weaver 使用結構描述來建立模擬資源。

服務應用程式 您用來讀取模擬狀態並與之互動的應用程式類型。服務應用程式可以在模擬中建立實體，但必須將它們轉移到空間網域。 會 SimSpace Weaver 管理服務應用程式的生命週期，並在模擬中的每個工作者上啟動 1 個（或更多，如模擬結構描述中所指定）。

服務網域 包含服務應用程式的網域。

服務分割區 服務應用程式的分割區。

模擬（資源） 執行模擬虛擬空間的運算叢集抽象。您可以有多個模擬。您可以使用結構描述設定模擬。

空間應用程式 封裝核心模擬邏輯的應用程式類型。每個空間應用程式擁有 1 個（且只有 1 個）分割區。

空間網域 包含空間應用程式的網域。

空間分割區 空間應用程式的分割區。

State Fabric SimSpace Weaver的記憶體內資料庫。State Fabric 存放模擬的狀態，包括實體和內部 SimSpace Weaver 資料。

store（實體欄位資料） 將實體欄位資料寫入 State Fabric。

訂閱 特定應用程式執行個體從訂閱區域接收資料的長時間執行請求。訂閱應用程式使用訂閱來探索訂閱區域內實體的變更。

訂閱區域 模擬空間的二維區域。訂閱是指訂閱區域。訂閱區域可以跨越多個分割區，也可以包含分割區的一部分。訂閱區域在其定義的範圍內是連續的。

# T

| | |
|---|---|
| 刻度 | 時間的離散值（時鐘時間或模擬時間）。應用程式可以比刻度持續時間更快的反覆運算，但預期在特定截止日期內寫入指定的刻度。特定刻度的所有應用程式的所有操作都必須完成，才能開始下一個刻度。 |
| 刻度率 | 請參閱時鐘速率。 |
| 時間（實際） | 從 reality. 的角度來看，目前的時間 SimSpace Weaver 會使用 64 位元 POSIX 時間戳記，這是自 epoch Unix 以來的奈秒數 (January 1, 1970, 00:00:00 UTC). |
| time（模擬） | 從模擬的角度來看，目前的時間。 SimSpace Weaver 會使用 64 位元整數邏輯刻度計數器，這可能不會直接對應到實際的時間。 |

# W

| | |
|---|---|
| 工作者 | 執行模擬程式碼的 Amazon Elastic Compute Cloud (Amazon EC2) 執行個體。 |