バージョン 1.16.0 のユーザーガイド

AWS SimSpace Weaver

AWS SimSpace Weaver: バージョン 1.16.0 のユーザーガイド

Table of Contents

AWS SimSpace Weaver とは

AWS SimSpace Weaver は、AWS クラウド で大規模な空間シミュレーションを構築および実行するために使用できるサービスです。例えば、群集シミュレーション、大規模な現実世界環境、没入感のあるインタラクティブな体験を作成できます。

SimSpace Weaver を使用すると、シミュレーションワークロードを複数の Amazon Elastic Compute Cloud (Amazon EC2) インスタンスに分散できます。 SimSpace Weaver は基盤となる AWS インフラストラクチャをデプロイし、シミュレーションを実行している Amazon EC2 インスタンス間のシミュレーションデータ管理とネットワーク通信を処理します。

SimSpace Weaver の主要なコンセプト

シミュレーションやゲームは、それを実行するコンピューターによって制限されます。仮想化世界の規模と複雑さが増すにつれて、処理性能は低下し始めます。計算に時間がかかり、システムのメモリが不足し、クライアントのフレームレートが低下します。リアルタイムのパフォーマンスを必要としないシミュレーションでは、これは煩わしいだけである可能性があります。あるいは、処理の遅延が増えるとコストが増加する、ビジネスクリティカルな状況となる可能性があります。シミュレーションやゲームにリアルタイムのパフォーマンスが必要な場合、パフォーマンスの低下は間違いなく問題です。

パフォーマンスの限界に達したシミュレーションの一般的なソリューションは、シミュレーションを単純化することです。多くのユーザーがいるオンラインゲームでは、仮想化世界のコピーを異なるサーバー上に作成し、ユーザーをサーバー全体に分散させることで、スケールの問題に対処することがよくあります。

SimSpace Weaver は仮想化世界を空間的に分割し、その一部を AWS クラウド で実行されるコンピュートインスタンスのクラスターに分散させることで、スケールの問題を解決します。コンピュートインスタンスは連携して、シミュレーション世界全体をパラレル処理します。シミュレーション世界は、その中のすべてのものと、それに接続するすべてのクライアントにとって、単一の統合空間のように見えます。ハードウェアのパフォーマンスの限界のためにシミュレーションを単純化する必要はもうありません。代わりに、クラウドにコンピューティング容量を追加することもできます。

トピック

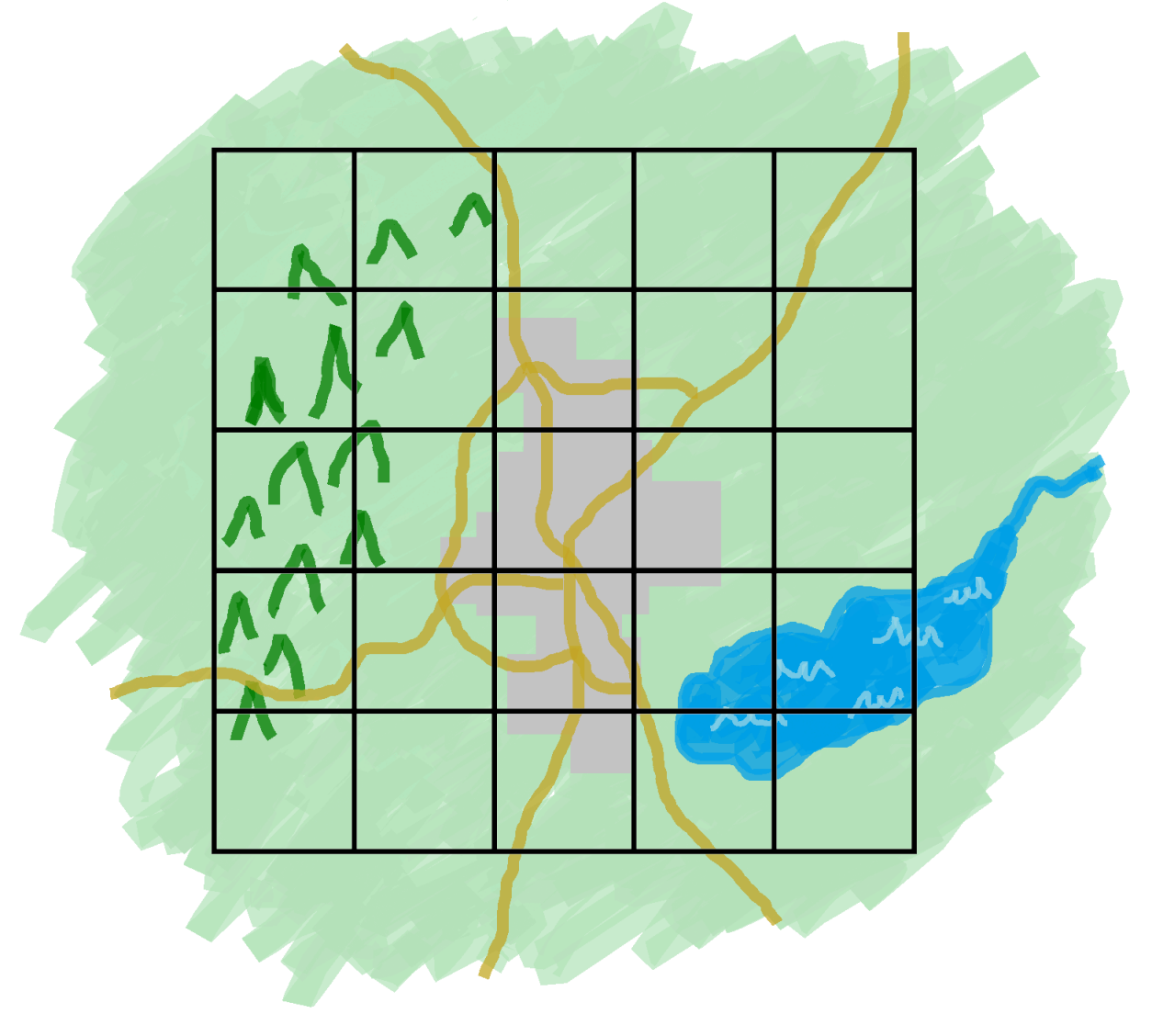
- シミュレーションスキーマ
- ワーカーおよびリソースユニット
- シミュレーションクロック
- パーティション
- ステートファブリック
- エンティティ
- アプリケーション

SimSpace Weaver の仕組み

シミュレーションは、その中にオブジェクトが存在する世界で構成されます。一部のオブジェクト (人や乗り物など) は動いて何かをします。他のオブジェクト (木や建物など) は静的です。SimSpace Weaver では、エンティティはシミュレーション世界内のオブジェクトです。

シミュレーション世界の境界を定義し、それをグリッドに分割します。グリッド全体で動作するシミュレーションロジックを作成する代わりに、グリッドの 1 つのセルで動作するシミュレーションロジックを作成します。SimSpace Weaver では、空間アプリケーションとは、グリッドのセルのシミュレーションロジックを実装するプログラムです。これには、そのセル内のすべてのエンティティのロジックが含まれます。空間アプリケーションの所有領域は、空間アプリケーションが制御するグリッドセルです。

> **Note**
>
> SimSpace Weaver では、「アプリケーション」という用語は、アプリケーションのコードまたはそのコードの実行中のインスタンスを指す場合があります。



シミュレーション世界はグリッドに分割されています

シミュレーション世界をグリッドに分割します 各空間アプリケーションは、そのグリッド内の 1 つのセルに対してシミュレーションロジックを実装します。

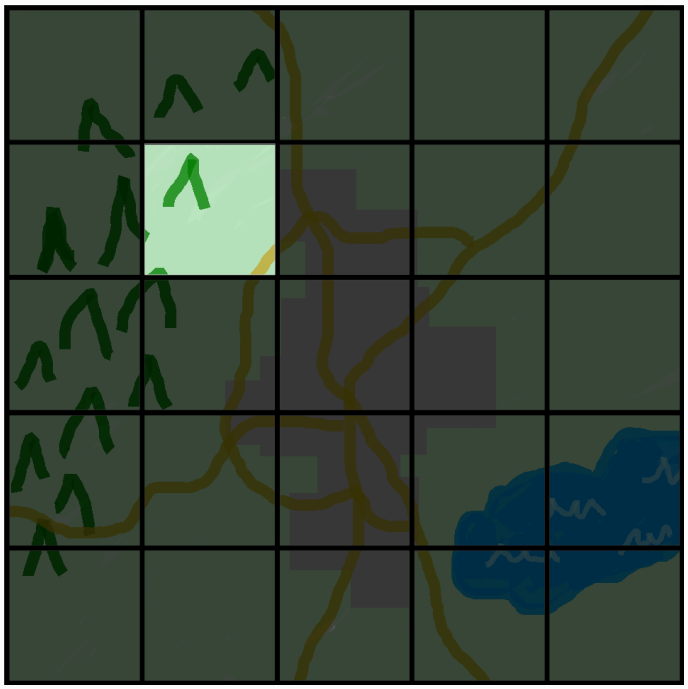
SimSpace Weaver はグリッドのセルごとに空間アプリケーションコードのインスタンスを実行します。すべての空間アプリケーションインスタンスはパラレル実行されます。基本的に、SimSpace Weaver はシミュレーション全体を複数の小規模なシミュレーションに分割します。小規模なシミュレーションはそれぞれ、シミュレーション世界全体の一部を処理します。SimSpace Weaver では、これらの小規模なシミュレーションを AWS クラウド 内の複数の Amazon Elastic Compute Cloud

(Amazon EC2) インスタンス (ワーカーと呼ばれる) に分散して実行できます。単一のワーカーで複数の空間アプリケーションを実行できます。

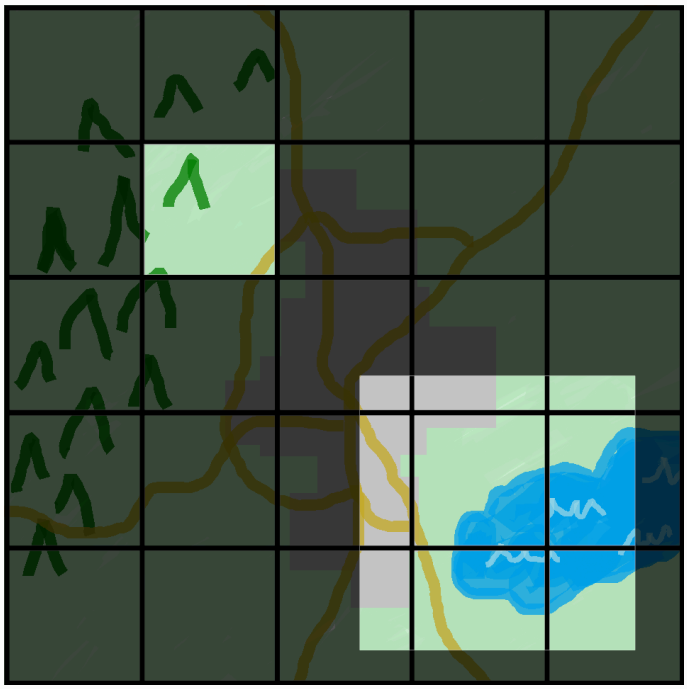
エンティティはシミュレーション世界内を移動できます。エンティティが別の空間アプリケーションの所有領域 (グリッド内の別のセル) に入ると、新しい領域の空間アプリケーションの所有者がエンティティの制御を引き継ぎます。シミュレーションが複数のワーカーで実行される場合、エンティティはあるワーカーの空間アプリケーションの制御から別のワーカーの空間アプリケーションの制御に移る可能性があります。エンティティが別のワーカーに移動すると、SimSpace Weaver は基盤となるネットワーク通信を処理します。

サブスクリプション

世界の空間アプリケーションビューは、それ自体が所有する領域です。シミュレーション世界の別の場所で何が起きているかを調べるために、空間アプリケーションはサブスクリプションを作成します。サブスクリプション領域はシミュレーション世界領域全体のサブセットです。サブスクリプション領域には、空間アプリケーション独自の所有領域など、複数の所有領域の一部を含めることができます。 SimSpace Weaver は空間アプリケーションに、サブスクリプション領域内で発生するすべてのエンティティイベント (入力、終了、作成、更新、削除など) を通知します。



世界の空間アプリケーションビュー



サブスクリプション領域が追加された空間アプリケーションビュー

世界の空間アプリケーションビューは、所有領域であり、世界グリッド内の 1 つのセルです。	空間アプリケーションは、サブスクリプションを使用してシミュレーション世界の別の場所で何が起こっているかを調べます。サブスクリプション領域には、複数のグリッドセルとセルの一部を含めることができます。

例えば、物理的に相互作用するエンティティをシミュレーションするアプリケーションでは、所有領域の空間的境界のすぐ向こう側にあるエンティティについて知る必要がある場合があります。これを実現するために、アプリケーションは所有領域の境界となる領域をサブスクライブできます。サブスクリプションを作成すると、アプリケーションはその領域のエンティティイベントに関する通知を受け取り、エンティティを読み取ることができます。もう 1 つの例として、領域を所有するアプリケーションに関係なく、200 メートル先にあるすべてのエンティティを確認する必要がある自動走行車があります。車両用アプリケーションでは、表示可能な領域をカバーする軸に沿ったバウンディングボックス (AABB) としてフィルターを使用してサブスクリプションを作成できます。

シミュレーションの空間的側面を管理する必要のないシミュレーションロジックを作成できます。カスタムアプリケーションは 単一のワーカーで実行される実行可能なプログラムです。カスタムアプリケーションのライフサイクル (開始と停止) を制御します。シミュレーションクライアントはカスタムアプリケーションに接続して、シミュレーションを表示したり操作したりできます。また、すべてのワーカーで実行されるサービスアプリケーションを作成することもできます。SimSpace Weaver はシミュレーションを実行するすべてのワーカーでサービスアプリケーションのインスタンスを起動します。

カスタムアプリケーションとサービスアプリケーションは、エンティティイベントについて学習したりエンティティを読み取ったりするためのサブスクリプションを作成します。これらのアプリケーションは空間的ではないため、所有領域はありません。サブスクリプションを使用することが、シミュレーションの世界で何が起こっているのかを知る唯一の方法です。

SimSpace Weaver の使用方法

SimSpace Weaver を使用する際の主な手順は以下のとおりです。

1. SimSpace Weaver アプリケーション SDK を統合する C++ アプリケーションを作成して構築します。
   a. アプリケーションは API コールを行ってシミュレーション状態とやりとりします。
2. 一部のアプリケーションを通じてシミュレーションを表示したり操作したりするクライアントを作成します。

3. シミュレーションをテキストファイルで構成します。
4. アプリケーションパッケージとシミュレーション構成をサービスにアップロードします。
5. シミュレーションを開始します。
6. カスタムアプリケーションを、必要に応じて起動または停止します。
7. クライアントをカスタムアプリケーションまたはサービスアプリケーションに接続して、シミュレーションを表示または操作します。
8. Amazon CloudWatch Logs でシミュレーションログを確認します。
9. シミュレーションを停止します。
10. シミュレーションをクリーンアップします。

シミュレーションスキーマ

シミュレーションスキーマ (またはスキーマ) は、シミュレーションの構成情報を含む YAML フォーマットのテキストファイルです。SimSpace Weaver はシミュレーションの開始時にスキーマを使用します。SimSpace Weaver アプリケーション SDK の配布可能パッケージには、サンプルプロジェクトのスキーマが含まれています。自身のスキーマの開始ポイントとしてこれを使用できます。シミュレーションスキーマの詳細については、「SimSpace Weaver シミュレーションスキーマリファレンス」を参照してください。

ワーカーおよびリソースユニット

ワーカーは、シミュレーションを実行する Amazon EC2 インスタンスです。シミュレーションスキーマでワーカータイプを指定します。SimSpace Weaver はワーカータイプを、サービスが使用する特定の Amazon EC2 インスタンスタイプにマッピングします。SimSpace Weaver はワーカーを自動的に起動および停止し、ワーカー間のネットワーク通信を管理します。SimSpace Weaver はシミュレーションごとにワーカーセットを起動します。シミュレーションが異なれば、使用するワーカーも異なります。

ワーカーで使用できるコンピューティング (プロセッサおよびメモリ) 容量は、コンピュートリソースユニット (またはリソースユニット) と呼ばれる論理ユニットに分割されます。リソースユニットは、一定量のプロセッサおよびメモリの容量を表します。

> **Note**
>
> 以前はコンピュートリソースユニットをスロットと呼んでいました。ドキュメントには、現在でもこの過去の用語が使用されている可能性があります。

シミュレーションクロック

各シミュレーションには独自のクロックがあります。API コールまたは [SimSpace Weaver] コンソールを使用してクロックを起動および停止します。シミュレーションはクロックが動作しているときにのみ更新されます。シミュレーションのすべての操作は、ティックと呼ばれる時間セグメント内で行われます。クロックはすべてのワーカーに各ティックの開始時間を通知します。

クロックレート (またはティックレート) は、クロックが通知する 1 秒あたりのティック数 (ヘルツ、または Hz) です。シミュレーションに必要なクロックレートは、シミュレーションスキーマの一部です。ティックのすべての操作は、次のティックが開始される前に完了する必要があります。このため、実効クロックレートは目的のクロックレートよりも低くなる可能性があります。実効クロックレートが目的のクロックレートより高くなることはありません。

パーティション

パーティションは、ワーカーの共有メモリの一部です。各パーティションにはシミュレーション状態データの一部が格納されます。

空間アプリケーションのパーティション (空間アプリケーションパーティションまたは空間パーティションとも呼ばれます) には、空間アプリケーションの所有領域内のすべてのエンティティが含まれます。SimSpace Weaver は各エンティティの空間的位置に基づいて、空間アプリケーションパーティションにエンティティを配置します。つまり、SimSpace Weaver は互いに空間的に近いエンティティを同じワーカーに配置しようとします。これにより、所有していないエンティティについてアプリケーションが必要とする知識の量を最小限に抑え、所有しているエンティティをシミュレートできます。

ステートファブリック

ステートファブリックは、すべてのワーカー上の共有メモリ (すべてのパーティションの集まり) のシステムです。シミュレーションのすべての状態データが格納されます。

ステートファブリックは、エンティティをそのエンティティの各データフィールドの初期データと更新ログのセットとして記述するカスタムバイナリフォーマットを使用します。このフォーマットを使用すると、シミュレーション時間内の前の時点におけるエンティティの状態にアクセスし、それを現実世界の特定の時点にマップし直すことができます。バッファのサイズには限りがあり、バッファ内のサイズを超えて時間を戻すことはできません。SimSpace Weaver は各フィールドの更新ログ内の現在のオフセットへのポインタを使用し、フィールド更新の一部としてポインタを更新します。SimSpace Weaver は共有メモリを使用して、これらの更新ログをアプリケーションのプロセス空間にマッピングします。

このオブジェクトフォーマットではオーバーヘッドが少なく、シリアル化のコストもかかりません。また、SimSpace Weaver はこのオブジェクトフォーマットを使用してインデックスフィールド (エンティティの位置など) を解析および識別します。

エンティティ

エンティティは、シミュレーションにおけるデータの最小の構成要素です。エンティティの例としては、アクター (人や乗り物など) や静的オブジェクト (建物や障害物など) があります。エンティティには、SimSpace Weaver に永続データとして保存できるプロパティ (位置や向きなど) があります。エンティティはパーティション内に存在します。

アプリケーション

SimSpace Weaverアプリケーションは、各シミュレーションティックを実行するカスタムロジックを含む、ユーザーが記述するソフトウェアです。ほとんどのアプリケーションの目的は、シミュレーションの実行時にエンティティを更新することです。アプリケーションは SimSpace Weaver アプリケーション SDK の API を呼び出して、シミュレーション内のエンティティに対してアクション (読み取りや更新など) を実行します。

アプリケーションと必要なリソース (ライブラリなど) を.zip ファイルとしてパッケージ化し、SimSpace Weaver にアップロードします。アプリケーションはワーカーの Docker コンテナで実行されます。SimSpace Weaver は各アプリケーションに一定数のワーカーのリソースユニットを割り当てます。

SimSpace Weaver は各アプリケーションに 1 つ (1 つのみ) のパーティションの所有権を割り当てます。アプリケーションとそのパーティションは同じワーカー上にあります。各パーティションにはアプリケーション所有者が 1 人しかいません。アプリケーションは、パーティション内のエンティティを作成、読み取り、更新、および削除できます。アプリケーションはパーティション内のすべてのエンティティを所有します。

アプリケーションには、空間アプリケーション、カスタムアプリケーション、サービスアプリケーションの 3 種類があります。ユースケースやライフサイクルによって異なります。

> **Note**
>
> SimSpace Weaver では、「アプリケーション」という用語は、アプリケーションのコードまたはそのコードの実行中のインスタンスを指す場合があります。

空間アプリケーション

空間アプリケーションは、シミュレーションに空間的に存在するエンティティの状態を更新します。例えば、ティックごとに速度、形状、サイズに基づいてエンティティを移動したり衝突させたりする Physics アプリケーションを定義できます。この場合、SimSpace Weaver は Physics アプリケーションの複数のインスタンスをパラレル実行して、ワークロードのサイズを処理します。

SimSpace Weaver は空間アプリケーションのライフサイクルを管理します。シミュレーションスキーマで空間アプリケーションパーティションの配置を指定します。シミュレーションを起動すると、SimSpace Weaver は空間アプリケーションのパーティションごとに空間アプリケーションを起動します。シミュレーションを停止すると、SimSpace Weaver は空間アプリケーションをシャットダウンします。

他の種類のアプリケーションはエンティティを作成できますが、エンティティを更新できるのは空間アプリケーションだけです。他の種類のアプリケーションでは、作成したエンティティを空間ドメインに転送する必要があります。SimSpace Weaver はエンティティの空間位置を使用して、そのエンティティを空間アプリケーションのパーティションに転送します。これにより、エンティティの所有権が空間アプリケーションに転送されます。

カスタムアプリケーション

カスタムアプリケーションを使用してシミュレーションを操作します。カスタムアプリケーションはサブスクリプションを使用してエンティティデータを読み取ります。カスタムアプリケーションはエンティティを作成できます。ただし、エンティティをシミュレーションに含めて更新するには、アプリケーションがエンティティを空間アプリケーションに転送する必要があります。SimSpace Weaver はネットワークエンドポイントをカスタムアプリケーションに割り当てることができます。シミュレーションクライアントはネットワークエンドポイントに接続してシミュレーションを操作できます。シミュレーションスキーマでカスタムアプリケーションを定義しますが、(SimSpace Weaver API コールを使用して) 起動および停止するのはユーザーの責任です。ワーカー上でカスタムアプリケーションインスタンスを起動した後は、SimSpace Weaver はそのインスタンスを別のワーカーに転送しません。

サービスアプリケーション

サービスアプリケーションは、すべてのワーカーで読み取り専用プロセスを実行する必要がある場合に使用できます。例えば、大規模なシミュレーションを行っていて、シミュレーション内を移動して表示されているエンティティだけをユーザーに表示する表示クライアントが必要な場合は、サービスアプリケーションを使用できます。この場合、1 つのカスタムアプリケーションインスタンスで

はシミュレーション内のすべてのエンティティを処理することはできません。すべてのワーカーで起動するようにサービスアプリケーションを構成できます。これらのサービスアプリケーションはそれぞれ、割り当てられたワーカーのエンティティをフィルタリングして、関連するエンティティだけを接続しているクライアントに送信できます。これにより、閲覧中のクライアントは、シミュレーション空間内を移動する際にさまざまなサービスアプリケーションに接続できます。シミュレーションスキーマでサービスアプリケーションを設定します。SimSpace Weaver はサービスアプリケーションを自動的に起動および停止します。

アプリケーションの要約

以下の表は、SimSpace Weaver アプリケーションのタイプ別の特性をまとめたものです。

	空間アプリケーション	カスタムアプリケーション	サービスアプリケーション
エンティティの読み込み	はい	はい	はい
エンティティの更新	はい	いいえ	いいえ
エンティティを作成する	はい	はい*	はい*
ライフサイクル	管理対象 (SimSpace Weaver が制御)	管理対象外 (ユーザーが制御)	管理対象 (SimSpace Weaver が制御)
起動方法	スキーマの指定に従って、SimSpace Weaver が空間パーティションごとに 1 つのアプリケーションインスタンスを起動します。	ユーザーが各アプリケーションインスタンスを起動します。	スキーマの指定に従って、SimSpace Weaver が各ワーカーで 1 つ以上のアプリケーションインスタンスを起動します。
クライアントが接続できる	いいえ	はい	はい

* カスタムアプリケーションまたはサービスアプリケーションでエンティティを作成する場合、空間アプリケーションがエンティティの状態を更新できるように、アプリケーションはそのエンティティの所有権を空間アプリケーションに転送する必要があります。

ドメイン

SimSpace Weaver ドメインは、同じ実行可能なアプリケーションコードを実行し、同じ起動オプションとコマンドを持つアプリケーションインスタンスの集まりです。ドメインは、その中に含まれるアプリケーションの種類 (空間ドメイン、カスタムドメイン、サービスドメイン) で呼ばれます。アプリケーションはドメイン内で設定します。

サブスクリプションとレプリケーション

アプリケーションは空間領域へのサブスクリプションを作成して、その領域内のエンティティイベント (入力、終了、作成、更新、削除など) を学習します。アプリケーションは、所有していないパーティション内のエンティティのデータを読み取る前に、サブスクリプションからのエンティティイベントを処理します。

パーティションはアプリケーションと同じワーカーに存在できますが (ローカルパーティションと呼ばれます)、別のアプリケーションがそのパーティションを所有できます。パーティションは別のワーカー (リモートパーティションと呼ばれます) に存在することもできます。サブスクリプションがリモートパーティションに対するものである場合、ワーカーはレプリケーションと呼ばれるプロセスを通じてリモートパーティションのローカルコピーを作成します。次に、ワーカーはローカルコピー (複製されたリモートパーティション) を読み取ります。ワーカー上の別のアプリケーションが同じティックでそのパーティションから読み取る必要がある場合、ワーカーは同じローカルコピーを読み取ります。

SimSpace Weaver のユースケースの例

エージェントベースのモデルの SimSpace Weaver や、空間コンポーネントを含む離散時間ステップシミュレーションに使用できます。

大勢の群集シミュレーションを作成する

SimSpace Weaver を使用して、実際の環境で群集をシミュレーションできます。SimSpace Weaver を使用して独自の動作を持つ何百万もの動的オブジェクトにシミュレーションをスケールできます。

都市規模の環境を作成する

SimSpace Weaver を使用して、都市全体のデジタルツインを作成します。都市計画、交通経路の設計、および環境ハザード対応の計画のためのシミュレーションを作成します。独自の地理空間データソースを環境の構成要素として使用できます。

没入感のあるインタラクティブな体験を作成します。

複数のユーザーが参加して交流できるシミュレーション体験を作成します。Unreal Engine や Unity などの一般的な開発ツールを使用して、3 次元 (3D) の仮想化世界を構築します。独自のコンテンツと動作で 3D 体験をカスタマイズします。

SimSpace Weaver の設定

SimSpace Weaver を初めて使用するための設定を行うには、AWS アカウント とローカル環境を設定する必要があります。これらのタスクを完了したら、開始方法のチュートリアルに進むことができます。

タスクの設定

1. SimSpace Weaver を使用するように AWS アカウント を設定する.
2. SimSpace Weaver のローカル環境をセットアップする.

SimSpace Weaver を使用するように AWS アカウント を設定する

SimSpace Weaver を使用するように AWS アカウント を設定するには、以下のタスクを実行します。

AWS アカウント にサインアップする

AWS アカウントがない場合は、以下のステップを実行して作成します。

AWS アカウントにサインアップするには

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話のキーパッドを使用して検証コードを入力するように求められます。

   AWS アカウントにサインアップすると、AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービスとリソースへのアクセス権があります。セキュリティのベストプラクティスとして、管理ユーザーに管理アクセスを割り当て、ルートユーザーアクセスが必要なタスクを実行する場合にのみ、ルートユーザーを使用してください。

サインアップ処理が完了すると、AWS からユーザーに確認メールが送信されます。https://aws.amazon.com/ の [アカウント] をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理ユーザーの作成

AWS アカウント にサインアップしたら、AWS アカウントのルートユーザーをセキュリティで保護し、AWS IAM Identity Centerを有効にして、管理ユーザーを作成します。これにより、日常的なタスクにルートユーザーを使用しないようにします。

AWS アカウントのルートユーザーをセキュリティで保護する

1. [ルートユーザー] を選択し、AWS アカウント のメールアドレスを入力して、アカウント所有者として AWS Management Console にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、「AWS サインイン User Guide」の「Signing in as the root user」を参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「IAM ユーザーガイド」の「AWS アカウントのルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」を参照してください。

管理ユーザーを作成する

1. IAM Identity Center を有効にする

   手順については、「AWS IAM Identity Centerユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。
2. IAM アイデンティティセンターで、管理ユーザーに管理アクセス権を付与します。

   IAM アイデンティティセンターディレクトリをアイデンティティソースとして使用するチュートリアルについては、「AWS IAM Identity Centerユーザーガイド」の「デフォルト IAM アイデンティティセンターディレクトリでのユーザーアクセスの設定」を参照してください。

管理ユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM アイデンティティセンターのユーザーを使用してサインインする方法については、「AWS サインイン User Guide」の「Signing in to the AWS access portal」を参照してください。

SimSpace Weaver を使用するアクセス許可を追加する

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「アクセス許可一式を作成」の手順を実行します。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。
- IAM ユーザー:
  - ユーザーに設定できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス許可の追加」の指示に従ってください。

Example SimSpace Weaver を使用するアクセス許可を付与する IAM ポリシー

```
{
    "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"
                }
            }
        }
    ]
}
```

SimSpace Weaver のローカル環境をセットアップする

SimSpace Weaver シミュレーションはコンテナ化 Amazon Linux 2 (AL2) 環境で実行されます。アプリケーションをコンパイルして SimSpace Weaver アプリケーション SDK にリンクするには、AL2 環境が必要です。標準のローカル開発環境は Docker の AL2 コンテナです。Docker を使用しない場合は、Windows Subsystem for Linux (WSL) で AL2 環境を実行するための代替手順が用意されています。また、独自の方法でローカル AL2 環境を作成できます。AL2 をローカルで実行するその他の方法については、「Amazon EC2 のドキュメント」を参照してください。

> **Important**
>
> Microsoft Windows の Docker は標準的な開発環境です。便宜上、ローカル開発環境を設定する他の方法も提案しますが、これらは標準ではなく、サポートされていません。

トピック

- Docker での Amazon Linux 2 (AL2) を設定する
- Windows Subsystem for Linux (WSL) での Amazon Linux 2 (AL2) を設定する

Docker での Amazon Linux 2 (AL2) を設定する

このセクションでは、Docker でローカル AL2 環境を設定する手順を説明します。Windows Subsystem for Linux (WSL) での AL2 を設定する手順については、「Windows Subsystem for Linux (WSL) での Amazon Linux 2 (AL2) を設定する」を参照してください。

要件

- Microsoft Windows 10 以上
- Microsoft Visual Studio 2019 またはそれ以降、Desktop development with C++ ワークロードをインストールした状態
- CMake3
- Git
- Docker Desktop
- AWS CLI

Docker での AL2 を設定する

1. AWS CLI の AWS 認証情報をまだ設定していない場合は、「AWS CLI の設定」の指示に従ってください。SimSpace Weaver のみを使用している場合は、デフォルトで SimSpace Weaver 認証情報を使用するように AWS CLI を設定できます。
2. SimSpace Weaver アプリケーション SDK の配布可能パッケージをダウンロードします。以下の要素が含まれます。
   - SimSpace Weaver アプリケーション開発用のバイナリおよびライブラリ
   - 開発ワークフローの一部を自動化するヘルパースクリプト
   - SimSpace Weaver コンセプトを示すサンプルアプリケーション
3. ファイルを任意の *sdk-folder* に解凍します。
4. *sdk-folder* に移動します。
5. Docker イメージを作成するため、以下のコマンドを入力します。

```
docker-create-image.bat
```

> **Note**
>
> このステップ中にエラーが発生した場合は、Docker が実行されていることを確認します。

Windows Subsystem for Linux (WSL) での Amazon Linux 2 (AL2) を設定する

このセクションでは、Windows Subsystem for Linux (WSL) でローカル AL2 環境を設定する手順を説明します。Docker での AL2 を設定する手順については、「Docker での Amazon Linux 2 (AL2) を設定する」を参照してください。

> **Important**
>
> このセクションでは、Amazon が所有、開発、サポートしていないバージョンの AL2 を使用するソリューションについて説明します。このソリューションは、Docker を使用しないことを選択した場合の利便性のみを目的として提供されています。このソリューションを使用することを選択した場合、Amazon および AWS は一切の責任を負いません。

要件

- Windows 10 での Hyper-V
- Windows Subsystem for Linux (WSL)
- WSL 用のサードパーティ製オープンソース AL2 配布 (ダウンロードバージョン 2.0.20200722.0-update.2) (手順を参照してください)

> **Important**
>
> このWSL 手順では、WSL 用の AL2 配布の 2.0.20200722.0-update.2 バージョンを使用しています。他のバージョンを使用すると、エラーが発生する可能性があります。

WSL での AL2 を設定する

1. Windows のコマンドプロンプトで、WSL の AL2 環境を起動します。

```
wsl -d Amazon2
```

> **Important**
>
> WSL での実行中は、このガイドの Windows の手順ではなく、Linux の手順を使用します (選択できる場合)。Linux の手順がない場合は、以下の代替方法を使用してください。
>
> - `tools\windows` の代わりに `tools/linux` を使用する
> - `.bat` スクリプトの代わりに `.sh` スクリプトを使用する

2. Linux シェルプロンプトで、yum パッケージマネージャーを更新します。

```
yum update -y
```

> **Important**
>
> このステップがタイムアウトになった場合は、WSL1に切り替えてこれらの手順を再試行する必要がある場合があります。WSL AL2 セッションを終了し、Windows コマンドプロンプトで以下を入力します。
>
> ```
> wsl --set-version Amazon2 1
> ```

3. 解凍ツールをインストールします。

```
yum install -y unzip
```

4. yum がインストールされている AWS CLI をすべて削除します。yum に AWS CLI がインストールされているか不明な場合は、以下のコマンドを両方試します。

```
yum remove awscli
```

```
yum remove aws-cli
```

5. 一時ディレクトリを作成して、そこに移動します。

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

6. AWS CLI をダウンロードおよびインストールします。

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

7. 一時ディレクトリは削除できます。

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

8. シェルセッションを再開して、環境内のパスを更新します。

```
exec
```

9. AL2 環境の AWS CLI の AWS 認証情報を設定します。詳細については、「AWS CLI の設定」を参照してください。AWS IAM Identity Center を使用する場合は、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」を参照してください。

```
aws configure
```

10. Git をインストールします。

```
yum install -y git
```

11. `wget` をインストールします。

```
yum install -y wget
```

12. SimSpace Weaver アプリケーション SDK 用のフォルダを作成します。

```
mkdir sdk-folder
```

13. SDK フォルダに移動します。

```
cd sdk-folder
```

14. SimSpace Weaver アプリケーション SDK の配布可能なファイルをダウンロードします。以下の要素が含まれます。

    - SimSpace Weaver アプリケーション開発用のバイナリおよびライブラリ
    - 開発ワークフローの一部を自動化するヘルパースクリプト
    - SimSpace Weaver コンセプトを示すサンプルアプリケーション

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

15. ファイルを解凍します。

```
unzip *.zip
```

16. 追加のセットアップスクリプトを実行します。

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

> **Note**
>
> これを実行する必要があるのは、WSL の AL2 環境で 1 回だけです。

AWS SimSpace Weaver でライセンス対象ソフトウェアを使用する

AWS SimSpace Weaver では、選択したシミュレーションエンジンとコンテンツでシミュレーションを構築できます。SimSpace Weaver の使用に関連して、シミュレーションで使用するソフトウェアまたはコンテンツのライセンス条項を取得、維持、遵守する責任はユーザーにあります。仮想化ホスト環境でのソフトウェアとコンテンツの展開がライセンス契約書上で許可されていることを確認してください。

SimSpace Weaver の開始方法

このセクションでは、SimSpace Weaver の使用を開始するのに役立つチュートリアルをご覧いただけます。これらのチュートリアルでは、SimSpace Weaver を使用してシミュレーションを構築するための一般的なワークフローを紹介します。どちらのチュートリアルでも、SimSpace Weaver でシミュレーションを作成、デプロイ、実行する方法を説明します。シミュレーションを数分で実行するには、クイックスタートチュートリアルから始めることをお勧めします。次に、詳細なチュートリアルをご覧いただき、この手順の各ステップの詳細を説明します。

これらのチュートリアルでは、設定中にダウンロードした SimSpace Weaver アプリケーション SDK .zip ファイルに含まれるサンプルアプリケーション (`PathfindingSample`) を使用します。このサンプルアプリケーションは、空間パーティション、パーティション間のエンティティ引き継ぎ、アプリ、サブスクリプションなど、すべての SimSpace Weaver シミュレーションに共通する概念を示しています。

チュートリアルでは、4 つの空間パーティションを含むシミュレーションを作成します。`PathfindingSample` 空間アプリケーションの個別のインスタンスが個々のパーティションを管理します。空間アプリケーションは独自のパーティションにエンティティを作成します。エンティティはシミュレーション世界の特定の位置に障害物を避けながら移動します。別のクライアントアプリケーション (SimSpace Weaver アプリケーション SDK に含まれます) を使用してシミュレーションを表示できます。

トピック

- クイックスタートチュートリアル: シミュレーションを数分で構築して実行する
- 詳細なチュートリアル: サンプルアプリケーションを構築しながら詳細を説明します

クイックスタートチュートリアル: シミュレーションを数分で構築して実行する

このチュートリアルでは、SimSpace Weaver でシミュレーションを数分で構築して実行するプロセスについて解説します。このチュートリアルから始め、その後に詳細なチュートリアルを進めることをお勧めします。

要件

開始する前に、必ず「SimSpace Weaver の設定」の手順を完了してください。

ステップ

ステップ 1: プロジェクトを作成する

配布可能な SimSpace Weaver アプリケーション SDK には、PathfindingSample プロジェクトのバンドルからプロジェクトを作成するスクリプトが含まれています。ファイルシステム内の場所から、スクリプトを実行する必要があります。このスクリプトは、コマンドラインで指定した値を使用して##内に *project-name* を作成します。

Docker

プロジェクトを作成する

1. Windows コマンドプロンプトで、プロジェクトフォルダに移動します。

```
cd sdk-folder
```

2. create-project.bat スクリプトを実行します。

```
.\create-project.bat --name project-name --path path
```

コマンドとパラメータの全リストは以下のとおりです。

```
.\create-project.bat --name project-name --path path --app-sdk-version version-
number --template template-name --overwriteproject
```

> **Important**
>
> 20 文字を超えるプロジェクト名は使用しないでください。この制限を超えると、エラーが発生する可能性があります。

> **Note**
>
> Docker でドライブを共有するように求めるメッセージが表示された場合は [Yes] を選択します。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

プロジェクトを作成する

1. Linuxシェルプロンプトで、[プロジェクト] フォルダに移動します。

```
cd sdk-folder
```

2. create-project.sh スクリプトを実行します。

```
./create-project.sh --name project-name --path path
```

コマンドとパラメータの全リストは以下のとおりです。

```
./create-project.sh --name project-name --path path --profile cli-profile-name
 --app-sdk-version version-number --template template-name --overwriteproject
```

> **Important**
>
> 20 文字を超えるプロジェクト名は使用しないでください。この制限を超えると、エラーが発生する可能性があります。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを default プロファイルにコピーするか、--profile *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

パラメータ

name

プロジェクト名。

path

ファイルシステム内のプロジェクトの場所。プロジェクト構造には、オペレーティングシステムのパス長の制限を超える長いファイルパスが含まれている場合があります。できるだけ短いパス名を使用することをお勧めします。

profile

スクリプトが認証に使用する AWS CLI プロファイル名。詳細については、「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイル設定」を参照してください。このパラメータは、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降でのみ使用できます。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。

app-sdk-version

(オプション) プロジェクトが使用する SimSpace Weaver アプリケーション SDK のバージョン。このバージョンを使用してアプリケーションをビルドしてリンクします。スクリプトが配布可能な場所にバージョンが見つからない場合や、バージョン番号を指定しない場合、スクリプトは自動的に最新バージョンをダウンロードします。

template

(オプション) スクリプトがプロジェクトの作成に使用するプロジェクトテンプレート。テンプレートを指定しない場合、スクリプトは `PathfindingSample` を使用します。有効値:

- **`PathfindingSample`** — 単一のワーカーを使用するサンプルアプリケーション。
- **`MultiWorkerPathfindingSample`** — 複数のワーカーを使用するサンプルアプリケーションのバージョン。

overwriteproject

(オプション) このオプションを使用して、同じプロジェクト名とパスを持つ既存のプロジェクトフォルダを上書きします。

ステップ 2: ログ記録を有効にする (オプション)

PathfindingSample プロジェクトのロギングはデフォルトではオフになっています。このチュートリアルでは、ロギングがオンであることを前提としています。ロギングを有効にできますが、必須ではありません。

> **Important**
>
> PathfindingSample は大量のログデータを生成します。ロギングをオンにすると、ログデータに対して料金が発生します。ログデータが存在する限り、そのログデータに対する料金は引き続き発生します。ロギングをオンにした場合、このシミュレーションを中止し、チュートリアルの最後にあるクリーンアップ手順をできるだけ早く実行することを強くお勧めします。

ロギングを有効にする

1. テキストエディターで以下のファイルを開きます。

   Docker

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

   > **Note**
   >
   *##########*はプロジェクトを作成したときに入力した値を使用する *path**project-name* です。

   WSL

   > **Important**
   >
   便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

   project-folder/tools/*project-name*-schema.yaml

> **Note**
>
> *##########*はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

2. ファイルの冒頭で、simulation_properties: セクションを検索します。

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

3. 行 simulation_properties: の後に以下の 2 行を挿入します。

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

4. simulation_properties: セクションが以下と同じであることを確認します。

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

5. ファイルを保存し、テキストエディタを終了します。

ステップ 3: クイックスタートスクリプトを実行する

サンプルアプリケーションにはクイックスタートスクリプトが含まれています。このスクリプトは、シミュレーションとそのアプリケーションを作成、ビルド、アップロード、起動します。

Docker

クイックスタートスクリプトを実行する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

```
cd project-folder\tools\windows
```

2. このプロジェクトのクイックスタートスクリプトを実行します。

```
.\quick-start-project-name-cli.bat
```

> **Important**
>
> バージョン 1.12.3 以降の場合、quick-start スクリプトは最大 1 時間でシミュレーションを開始します。--maximum-duration パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを default プロファイルにコピーするか、--profile *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

クイックスタートスクリプトを実行する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. このプロジェクトのクイックスタートスクリプトを実行します。

   ```
   ./quick-start-project-name-cli.sh
   ```

   > **Important**
   >
   > バージョン 1.12.3 以降の場合、`quick-start` スクリプトは最大 1 時間でシミュレーションを開始します。`--maximum-duration` パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0

です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

スクリプトはループを開始し、すべてのコンポーネントが `STARTED` になると自動的に停止します。以下と似た出力を検索します。

```
[2022-10-04T22:15:28] [INFO] Describe Simulation Results:
[2022-10-04T22:15:28] [INFO] {
    "Status": "STARTED",
    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "RoleArn": "arn:aws:iam::111122223333:role/weaver-MyProject-app-role",
    "CreationTime": 1664921418.09,
```

ステップ 4: IP アドレスとポート番号を取得する

シミュレーションに接続するには、ビュー (カスタム) アプリケーションの IP アドレスとポート番号を取得する必要があります。以下の手順は、シミュレーションについて (シミュレーション名など) 何も知らないことを前提としています。この手順を使用して、カスタムアプリケーションまたはサービスアプリケーションの IP アドレスとポート番号をいつでも確認できます。以下の出力例は、`MyProject` という名前のプロジェクトについてのものです。

Docker

IP アドレスとポート番号を取得する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. ListSimulations API を使用してシミュレーションの名前を取得します。

   ```
   .\weaver-project-name-cli.bat list-simulations
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

   出力例:

   ```
   {
       "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"
        }
    ]

}
```

3. DescribeSimulation API を使用して、シミュレーション内のドメインのリストを取得します。

```
.\weaver-project-name-cli.bat describe-simulation --simulation simulation-name
```

出力の LiveSimulationState セクションで Domains セクションを探します。

出力例:

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

4. ListApps API を使用して、ドメイン内のカスタムアプリケーションのリストを取得します。サンプルプロジェクトのビュー (カスタム) アプリケーションのドメイン名は MyViewDomain です。出力でアプリケーション名を探します。

```
.\weaver-project-name-cli.bat list-apps --simulation simulation-name --
domain domain-name
```

出力例:

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

5. DescribeApp API を使用して、IP アドレスとポート番号を取得します。サンプルプロジェクトでは、ドメイン名は MyViewDomain で、アプリケーション名は ViewApp です。

```
.\weaver-project-name-cli.bat describe-app --simulation simulation-name --
domain domain-name --app app-name
```

IP アドレスとポート番号は出力の EndpointInfo ブロックに含まれます。IP アドレスは Address の値で、ポート番号は Actual の値です。

出力例:

```
{
    "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**
>
> `Declared` の値はアプリケーションコードのバインド先となるポート番号です。`Actual` の値は、アプリケーションに接続する際に SimSpace Weaver がクライアントに公開するポート番号です。SimSpace Weaver は `Declared` ポートを `Actual` ポートにマップします。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

IP アドレスとポート番号を取得する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

```
cd project-folder/tools/linux
```

2. ListSimulations API を使用してシミュレーションの名前を取得します。

```
./weaver-project-name-cli.sh list-simulations
```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile cli-profile-name` パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

出力例:

```
{
    "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"
        }
    ]
```

```
}
```

3. DescribeSimulation API を使用して、シミュレーション内のドメインのリストを取得します。

```
./weaver-project-name-cli.sh describe-simulation --simulation simulation-name
```

出力の LiveSimulationState セクションで Domains セクションを探します。

出力例:

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

4. ListApps API を使用して、ドメイン内のカスタムアプリケーションのリストを取得します。サンプルプロジェクトのビュー (カスタム) アプリケーションのドメイン名は MyViewDomain です。出力でアプリケーション名を探します。

```
./weaver-project-name-cli.sh list-apps --simulation simulation-name --
domain domain-name
```

出力例:

```
{
```

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

5. DescribeApp API を使用して、IP アドレスとポート番号を取得します。サンプルプロジェクトでは、ドメイン名は MyViewDomain で、アプリケーション名は ViewApp です。

```
./weaver-project-name-cli.sh describe-app --simulation simulation-name --
domain domain-name --app app-name
```

IP アドレスとポート番号は出力の EndpointInfo ブロックに含まれます。IP アドレスは Address の値で、ポート番号は Actual の値です。

出力例:

```
{
    "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**
>
> `Declared` の値はアプリケーションコードのバインド先となるポート番号です。`Actual` の値は、アプリケーションに接続する際に SimSpace Weaver がクライアントに公開するポート番号です。SimSpace Weaver は `Declared` ポートを `Actual` ポートにマップします。

ステップ 5: シミュレーションを表示する

SimSpace Weaver アプリケーション SDK には、サンプルアプリケーションを表示するためのさまざまなオプションが用意されています。Unreal Engine 開発用のローカルサポートがない場合は、サンプルコンソールクライアントを使用できます。Unreal Engine クライアントへの説明では、Windows の使用を前提としています。

コンソールクライアントは、エンティティイベントが発生するとそのリストを表示します。クライアントは `ViewApp` からエンティティイベント情報を取得します。コンソールクライアントがイベントのリストを表示すると、シミュレーション内の `ViewApp` およびアクティビティとのネットワーク接続を確認します。

`PathfindingSample` シミュレーションでは、二次元平面上に静止しているエンティティと動いているエンティティが作成されます。移動するエンティティは静止しているエンティティの周りを移動します。Unreal Engine クライアントはエンティティイベントを視覚化します。

Windows console client

要件

- Microsoft Windows 10 以上
- Microsoft Visual Studio 2019 またはそれ以降、Desktop development with C++ ワークロードをインストールした状態
- CMake3
- Git

サンプルコンソールクライアントを使用してサンプルアプリケーションに接続する

1. コマンドプロンプトウィンドウで、コンソールクライアントのフォルダ (アプリケーション SDK フォルダ内) に移動します。

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

2. CMake3 を使用して、このフォルダに Visual Studio ソリューションを作成します。

```
cmake .
```

> **Note**
>
> 末尾に必ずスペースとピリオドを含めます。

> **Important**
>
> 以降の手順については、コマンドプロンプトウィンドウを開いたままにしておきます。

3. 前のステップで作成した Visual Studio で、`PathfindingSampleConsoleClient.sln` を開きます。
4. RelWithDebInfo ビルド設定を選択します。
5. [Build]、[Build Solution] の順に選択します。
6. 前のコマンドプロンプトウィンドウで、[コンソールクライアント] フォルダにある [出力をビルド] フォルダに移動します。

```
cd RelWithDebInfo
```

7. `ViewApp`の IP アドレスとポート番号を使用してクライアントを実行します。

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

コマンドプロンプトウィンドウには、以下の出力例のように、エンティティの更新、削除、および作成イベントの番号が表示されます。

> **Note**
>
> 以下の出力例の IP アドレスとポート番号はプレースホルダーです。コンソールクライアントに、ViewApp の IP アドレスとポート番号を指定します。AWS クラウドで動作する ViewApp に接続する場合は、Actual ポート番号を指定します。ローカルシステムで動作する ViewApp に接続するときは、IP アドレスとポート番号 127.0.0.1:7000 を指定します。詳細については、「ローカル開発」を参照してください。

```
##PathfindingSample#ViewApp Message Reader##

Added argument url:tcp://198.51.100.135:4321
Some subscription arguments are missing, restoring defaults.

*****
Sample usage without a MoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50
Sample usage with CircleMoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50 --subs-move-strategy circle --circle-center-x 500 --
circle-center-y 500 --circle-speed 0.001

*****
Starting NNG client. NNG version: 1.2.4
Creating socket ...done.
Connecting to View App ... done.
Initiating connection to tcp:// 198.51.100.135:4321 ... done.

Receiving messages ...
[2022-10-04 19:13:00.710] CreateEntity Count: 72
[2022-10-04 19:13:00.756] UpdateEntity Count: 42
[2022-10-04 19:13:00.794] DeleteEntity Count: 72
[2022-10-04 19:13:03.690] CreateEntity Count: 11
[2022-10-04 19:13:03.725] UpdateEntity Count: 2
[2022-10-04 19:13:03.757] UpdateEntity Count: 2
[2022-10-04 19:13:03.790] UpdateEntity Count: 2
```

```
```

> **Note**
>
> トラブルシューティングのガイダンスについては、「PathfindingSample コンソールクライアントが接続に失敗する」を参照してください。

8. CTRL+C を押してコンソールクライアントを終了します。

Linux console client

> **Important**
>
> 便宜上、これらの指示を使用します。Linux 環境の一部では動作しない可能性があります。これらの手順はサポートされていません。

この手順は、すべてを Linux 環境内で作業していることを前提としています。Windows 組み込みクライアントを使用してシミュレーションを表示することもできます。

要件

- CMake3
- C コンパイラ (Amazon Linux 2 に既に含まれています)
- Git

サンプルコンソールクライアントを使用してサンプルアプリケーションに接続する

1. Linux シェルプロンプトで、[コンソールクライアント] フォルダ (アプリケーション SDK フォルダ内) に移動します。

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

2. ビルドフォルダを作成します。

```
mkdir build
```

3. ビルドフォルダに移動します。

```
cd build
```

4. CMake3 を使用して、クライアントをビルドします。

```
cmake3 ../ && cmake3 --build .
```

> **Note**
>
> 末尾に必ずスペースとピリオドを含めます。

5. ViewAppの IP アドレスとポート番号を使用してクライアントを実行します。

```
./ConsoleClient --url tcp://ip-address:port-number
```

コマンドプロンプトウィンドウには、以下の出力例のように、エンティティの更新、削除、および作成イベントの番号が表示されます。

> **Note**
>
> 以下の出力例の IP アドレスとポート番号はプレースホルダーです。コンソールクライアントに、ViewApp の IP アドレスとポート番号を指定します。AWS クラウドで動作する ViewApp に接続する場合は、Actual ポート番号を指定します。ローカルシステムで動作する ViewApp に接続するときは、IP アドレスとポート番号 127.0.0.1:7000 を指定します。詳細については、「ローカル開発」を参照してください。

```
##PathfindingSample#ViewApp Message Reader##

Added argument url:tcp://198.51.100.135:4321
Some subscription arguments are missing, restoring defaults.

*****
Sample usage without a MoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50
```

```
Sample usage with CircleMoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50 --subs-move-strategy circle --circle-center-x 500 --
circle-center-y 500 --circle-speed 0.001

*****
Starting NNG client. NNG version: 1.2.4
Creating socket ...done.
Connecting to View App ... done.
Initiating connection to tcp:// 198.51.100.135:4321 ... done.

Receiving messages ...
[2022-10-04 19:13:00.710] CreateEntity Count: 72
[2022-10-04 19:13:00.756] UpdateEntity Count: 42
[2022-10-04 19:13:00.794] DeleteEntity Count: 72
[2022-10-04 19:13:03.690] CreateEntity Count: 11
[2022-10-04 19:13:03.725] UpdateEntity Count: 2
[2022-10-04 19:13:03.757] UpdateEntity Count: 2
[2022-10-04 19:13:03.790] UpdateEntity Count: 2
```

> **Note**
>
> トラブルシューティングのガイダンスについては、「PathfindingSample コンソールクライアントが接続に失敗する」を参照してください。

6. CTRL+C を押してコンソールクライアントを終了します。

Unreal Engine on Windows

要件

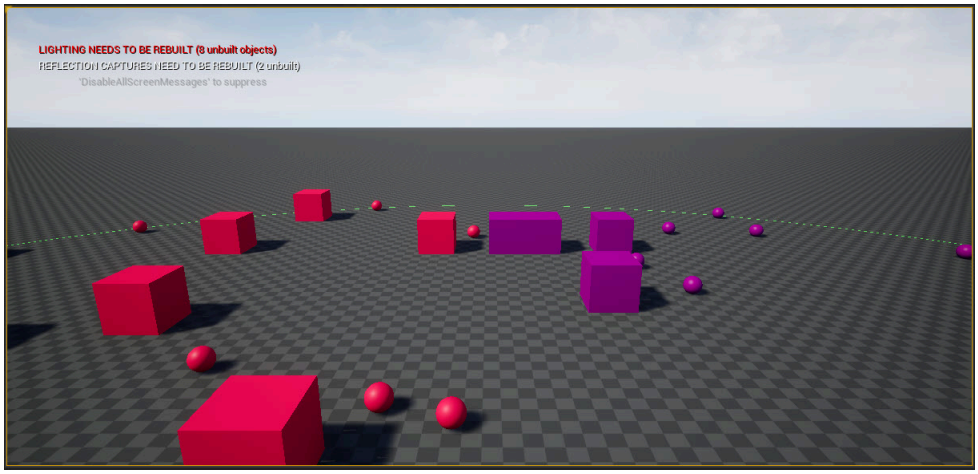
- Unreal Engine 5 開発環境
- Microsoft .NET Framework 4.8 Developer Pack
- Windows コンソールクライアント (このページの「Windows コンソールクライアント」タブを参照してください)

> **Important**
>
> 他のバージョンの Unreal Engine および .NET はサポートされていないため、問題が発生する可能性があります。

サンプル Unreal クライアントを使用してサンプルアプリケーションに接続する

1. Unreal Engine クライアントはコンソールクライアントの NNG ライブラリを使用します。Windows のコンソールクライアントをまだビルドしていない場合は、ビルドする必要があります。詳細は、このページの [Windows コンソールクライアント] タブを参照してください。
2. ファイルマネージャーウィンドウで、*sdk-folder*\packaging-tools\clients\PathfindingSampleClients\UnrealClient に進みます。
3. UnrealClient.uproject を開きます。
4. エディタから UnrealClient モジュールを再構築するかどうか尋ねられたら、[yes] を選択します。
5. テキストエディタで *sdk-folder*\packaging-tools\clients\PathfindingSampleClients\UnrealClient\view_app_url.txt を開きます。
6. ビューアプリケーションの IP アドレスとポート番号で URL を更新します: tcp://*ip-address*:*port-number* (tcp://198.51.100.135:1234 のようになります)。
7. Unreal エディタ で [play] を選択します。

Unreal エディタには、以下のスクリーンショットのようなシミュレーションのビジュアルが表示されます。



> **Note**
>
> ローカル開発システムの性能によっては、Unreal エディタがシミュレーションを表示するまでに数分かかることがあります。この間、システムがフリーズしているように見えることがあります。

W、A、S、D キーを使用して Unreal クライアント内を移動します。マウスボタンを押したままマウスをドラッグすると回転します。

[(左角かっこ) キーを押すと、サブスクリプション領域のサイズを小さくできます。] (右角かっこ) キーを押すと、サブスクリプション領域のサイズを大きくできます。サブスクリプション領域のサイズによって、クライアントに表示されるエンティティの数が決まります。

C キーを押すと、シミュレーションにエンティティを作成できます。クライアントはビューアプリケーションに `CreateEntity` コマンドを送信します。次に、ビューアプリケーションはエンティティを作成し、空間ドメインに転送します。

`project-folder``\src\PathfindingSample\ViewApp\Driver\ViewAppDriver.cpp` の `ViewAppDriver::HandleEntityCreationRequests` コードを調べて、アプリケーションがこのプロセスをどのように実装しているかを確認できます。

ステップ 6: シミュレーションを停止してクリーンアップする

不要になったシミュレーションはクリーンアップすることが重要です。SimSpace Weaver シミュレーションが停止しても、シミュレーションリソースは Service Quotas (制限) に加算されます。実行中のシミュレーションについては、引き続き料金が発生します。また、Amazon CloudWatch Logs や Amazon Simple Storage Service など、サポートする のサービスのデータストレージに対する請求が発生する場合もあります。SimSpace Weaverサービスクォータの詳細については、「」を参照してくださいSimSpace Weaver エンドポイントとクォータ。

シミュレーションをクリーンアップする準備ができたら、このセクションの手順に従います。

> **Important**
>
> 停止したシミュレーションは再開できません。

> **Important**
>
> 削除されたシミュレーションを復元することはできません。

SimSpace Weaver のシミュレーションリソースをクリーンアップする

シミュレーションは、停止してから削除する必要があります。シミュレーションを削除すると、SimSpace Weaver 内のリソースのみが削除されます。シミュレーションで作成したリソースや他のサービスで使用しているリソースを削除するには、別の手順を実行する必要があります (以下のセクションを参照してください)。

Docker

シミュレーションをクリーンアップする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. シミュレーション名を検索します。

   ```
   .\weaver-project-name-cli.bat list-simulations
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定で

きます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

3. シミュレーションを停止します。

```
.\weaver-project-name-cli.bat stop-simulation --simulation simulation-name
```

4. 停止したシミュレーションを削除します。

```
.\weaver-project-name-cli.bat delete-simulation --simulation simulation-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

シミュレーションをクリーンアップする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

```
cd project-folder/tools/linux
```

2. シミュレーション名を検索します。

```
./weaver-project-name-cli.sh list-simulations
```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

3. シミュレーションを停止します。

```
./weaver-project-name-cli.sh stop-simulation --simulation simulation-name
```

4. 停止したシミュレーションを削除します。

```
./weaver-project-name-cli.sh delete-simulation --simulation simulation-name
```

AWS Management Console

シミュレーションをクリーンアップする

1. [SimSpace Weaver] コンソールで [SimSpace Weaver] を開きます。
2. ナビゲーションペインで、[Simulations] を選択します。
3. [Simulations] リストから、削除するシミュレーション名の横にあるオプションを選択します。
4. 選択したシミュレーションの Status が STARTED の場合:
   a. [Actions] ドロップダウンメニューを選択します。
   b. [Stop] を選択します。

c. 確認するには、シミュレーション名を入力します。

d. [Stop] を選択します。

e. シミュレーションの Status が STOPPED になるまで待ちます。

5. [Actions] ドロップダウンメニューを選択します。
6. [Delete] を選択します。
7. [Delete] を選択して確定します。

サポートサービスのシミュレーションリソースをクリーンアップする

シミュレーションをサポートするために、SimSpace Weaver は他のサービスのリソースを作成します。シミュレーションを削除しても、SimSpace Weaver ではこれらのリソースは削除されません。これらの追加リソースが不要な場合は削除できます。

> **Important**
>
> これらのリソースを削除しない場合、料金が発生する可能性があります。

プロジェクトのサポートリソースを削除する

1. プロジェクトが終了したら、その AWS CloudFormation スタックを削除します。AWS CloudFormation の使用の詳細については、「AWS CloudFormation ユーザーガイド」の「[AWS CloudFormation] コンソールでスタックを削除する」を参照してください。

   - weaver-*project-name*-stack

   > **Important**
   >
   > 同じプロジェクトから開始したシミュレーションは、アプリケーションロールなどのリソースを共有します。AWS CloudFormation スタックを削除すると、アプリケーションロールを削除することになります。同じリソースを共有する他のシミュレーションがある場合は、AWS CloudFormation スタックを削除しないでください。

> **Note**
>
> 空でない Amazon S3 バケットは削除できないため、AWS CloudFormation スタックが DELETE_FAILED を報告する可能性が高くなります。次のステップで Amazon S3 バケットを削除します。

2. プロジェクトが終了したら、その Amazon S3 バケットを削除します。Amazon S3 バケットの使用の詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケットの削除」を参照してください。

   - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-`*`region`*

   以下の例では、リージョン `MyProject` 内のアカウント `111122223333` に登録されている `us-west-2` という名前のプロジェクトには以下のバケットがあります。

   - `weaver-myproject-111122223333-us-west-2`

   > **Note**
   >
   > Amazon S3 バケットを削除するには、その前にバケットの内容を削除する必要があります。

   > **Note**
   >
   > SimSpace Weaver アプリケーション SDK バージョン 1.12.x のプロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
   >
   > - weaver*`lowercase-project-name`*`-`*`account-number`* -app-zips-*`region`*
   > - weaver*`lowercase-project-name`*`-`*`account-number`* -schemas-*`region`*

3. シミュレーションのログ記録を有効にした場合は、Logs CloudWatch ロググループを削除します。 CloudWatch ログの操作の詳細については、「Amazon Logs ユーザーガイド」の「ロググループとログストリームの使用」を参照してください。 CloudWatch

シミュレーションのロググループの名前は、スキーマ (設定ファイル) *project-folder*\tools*project-name*.yaml で指定されています。

ロググループ名は log_destination_resource_name の値です。以下のスキーマスニペットは、サンプルアプリケーションのロググループが MySimulationLogs であることを示しています。

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

Warning

同じロググループを指定する複数のシミュレーションを開始すると、それらのシミュレーションのログデータはすべて同じロググループに送られます。ロググループを削除すると、そのロググループを使用するすべてのシミュレーションのログデータが削除されます。実行中のシミュレーションのロググループを削除すると、シミュレーションは失敗します。

Important

シミュレーションのスキーマで log_destination_service: "logs"と が指定されlog_destination_resource_nameていても CloudWatch 、ロググループに ログが見つからない場合は、シミュレーションが実行されAWS リージョンたのと同じ を確認してください。

詳細なチュートリアル: サンプルアプリケーションを構築しながら詳細を説明します

このチュートリアルの手順は、クイックスタートチュートリアルで説明したものと同じものですが、より詳しく説明します。クイックスタートチュートリアルでは、多くの手順が簡略化され、自動化の詳細が隠されています。このチュートリアルでは、これらの詳細を明らかにして説明します。

このチュートリアルに進む前に、クイックスタートチュートリアルをすべて確認しておくことをお勧めします。

要件

開始する前に、必ず「SimSpace Weaver の設定」の手順を完了してください。

ステップ

- ステップ 1: プロジェクトを作成する
- ステップ 2: ログ記録を有効にする (オプション)
- ステップ 3: シミュレーションスキーマをアップロードする
- ステップ 4: プロジェクトをビルドする
- ステップ 5: アプリケーションをアップロードする
- ステップ 6: シミュレーションを開始する
- ステップ 7: シミュレーションを取得する
- ステップ 8: カスタムアプリケーションを起動する
- ステップ 9: クロックを起動する
- ステップ 10: ログを確認する
- ステップ 11: シミュレーションを表示する
- ステップ 12: シミュレーションを停止してクリーンアップする

ステップ 1: プロジェクトを作成する

配布可能な SimSpace Weaver アプリケーション SDK には、`PathfindingSample` プロジェクトのバンドルからプロジェクトを作成するスクリプトが含まれています。ファイルシステム内の場所から、スクリプトを実行する必要があります。このスクリプトは、コマンドラインで指定した値を使用して`##`内に *`project-name`* を作成します。

Docker

プロジェクトを作成する

1. Windows コマンドプロンプトで、プロジェクトフォルダに移動します。

```
cd sdk-folder
```

2. `create-project.bat` スクリプトを実行します。

```
.\create-project.bat --name project-name --path path
```

コマンドとパラメータの全リストは以下のとおりです。

```
.\create-project.bat --name project-name --path path --app-sdk-version version-number --template template-name --overwriteproject
```

> **Important**
>
> 20 文字を超えるプロジェクト名は使用しないでください。この制限を超えると、エラーが発生する可能性があります。

> **Note**
>
> Docker でドライブを共有するように求めるメッセージが表示された場合は [Yes] を選択します。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--`

`profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

WSL

Important

便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

プロジェクトを作成する

1. Linuxシェルプロンプトで、[プロジェクト] フォルダに移動します。

```
cd sdk-folder
```

2. `create-project.sh` スクリプトを実行します。

```
./create-project.sh --name project-name --path path
```

コマンドとパラメータの全リストは以下のとおりです。

```
./create-project.sh --name project-name --path path --profile cli-profile-name
 --app-sdk-version version-number --template template-name --overwriteproject
```

Important

20 文字を超えるプロジェクト名は使用しないでください。この制限を超えると、エラーが発生する可能性があります。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

パラメータ

name

プロジェクト名。

path

ファイルシステム内のプロジェクトの場所。プロジェクト構造には、オペレーティングシステムのパス長の制限を超える長いファイルパスが含まれている場合があります。できるだけ短いパス名を使用することをお勧めします。

profile

スクリプトが認証に使用する AWS CLI プロファイル名。詳細については、「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイル設定」を参照してください。このパラメータは、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降でのみ使用できます。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。

app-sdk-version

(オプション) プロジェクトが使用する SimSpace Weaver アプリケーション SDK のバージョン。このバージョンを使用してアプリケーションをビルドしてリンクします。スクリプトが配布可能な場所にバージョンが見つからない場合や、バージョン番号を指定しない場合、スクリプトは自動的に最新バージョンをダウンロードします。

template

(オプション) スクリプトがプロジェクトの作成に使用するプロジェクトテンプレート。テンプレートを指定しない場合、スクリプトは `PathfindingSample` を使用します。有効値:

- **`PathfindingSample`** — 単一のワーカーを使用するサンプルアプリケーション。
- **`MultiWorkerPathfindingSample`** — 複数のワーカーを使用するサンプルアプリケーションのバージョン。

overwriteproject

(オプション) このオプションを使用して、同じプロジェクト名とパスを持つ既存のプロジェクトフォルダを上書きします。

ステップ 2: ログ記録を有効にする (オプション)

`PathfindingSample` プロジェクトのロギングはデフォルトではオフになっています。このチュートリアルでは、ロギングがオンであることを前提としています。ロギングを有効にできますが、必須ではありません。

> **Important**
>
> `PathfindingSample` は大量のログデータを生成します。ロギングをオンにすると、ログデータに対して料金が発生します。ログデータが存在する限り、そのログデータに対する料金は引き続き発生します。ロギングをオンにした場合、このシミュレーションを中止し、チュートリアルの最後にあるクリーンアップ手順をできるだけ早く実行することを強くお勧めします。

ロギングを有効にする

1. テキストエディターで以下のファイルを開きます。

Docker

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

> **Note**
>
> ##########はプロジェクトを作成したときに入力した値を使用する *path**project-name* です。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

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

> **Note**
>
> ##########はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

2. ファイルの冒頭で、`simulation_properties:` セクションを検索します。

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

3. 行 `simulation_properties:` の後に以下の 2 行を挿入します。

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

4. `simulation_properties:` セクションが以下と同じであることを確認します。

```
simulation_properties:
```

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

5. ファイルを保存し、テキストエディタを終了します。

ステップ 3: シミュレーションスキーマをアップロードする

SimSpace Weaver はスキーマを使用してシミュレーションを設定します。スキーマはYAMLフォーマットのプレーンテキストファイルです。詳細については、「シミュレーションの設定」を参照してください。

Docker

サンプルアプリケーションにはスキーマがあらかじめ設定されています。サンプルアプリケーションのスキーマファイルは、プロジェクトの [ツール] フォルダにあります。

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

スキーマをアップロードする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. ヘルパースクリプトを使用してスキーマをアップロードします。

   ```
   .\upload-schema-project-name.bat
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バー

ジョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

サンプルアプリケーションにはスキーマがあらかじめ設定されています。サンプルアプリケーションのスキーマファイルは、プロジェクトの [ツール] フォルダにあります。

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

スキーマをアップロードする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*`project-folder`* はプロジェクトを作成したときに入力した値を使用する *`path`*`/`*`project-name`* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. ヘルパースクリプトを使用してスキーマをアップロードします。

   ```
   ./upload-schema-project-name.sh
   ```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

ステップ 4: プロジェクトをビルドする

これで、サンプルプロジェクト用の空間アプリケーションとカスタムアプリケーションを構築する準備ができました。サンプルプロジェクトには、これらのアプリケーションを自動的に構築するヘルパースクリプトが含まれています。

Docker

このスクリプトは、ローカル環境を設定したときに作成した Docker イメージを使用して Docker コンテナを起動します。このスクリプトは、Amazon Linux 環境の Docker コンテナ内のビルドを実行します。ビルドアーティファクトとその依存関係を Windows の *`project-folder`*\build フォルダに書き込みます。

プロジェクトを構築する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*`project-folder`* はプロジェクトの作成時に入力した値を使用する *`path`**`project-name`* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

```
cd project-folder\tools\windows
```

2. ヘルパースクリプトを使用してプロジェクトを構築します。

```
.\build-project-name.bat
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

このスクリプトはビルドアーティファクトとその依存関係を *project-folder*/build フォルダに書き込みます。

プロジェクトを構築する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

```
cd project-folder/tools/linux
```

2. ヘルパースクリプトを使用してプロジェクトを構築します。

```
./build-project-name.sh
```

ステップ 5: アプリケーションをアップロードする

ビルドスクリプトはアプリケーションを zip ファイルとしてパッケージ化しました。クラウドで SimSpace Weaver シミュレーションを実行するには、これらの zip ファイルを Amazon Simple

Storage Service の特定のバケットにアップロードする必要があります。SimSpace Weaver アプリケーション SDK には、アップロードを処理するヘルパースクリプトが用意されています。

Docker

アプリケーションをアップロードする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. ヘルパースクリプトを使用してアプリケーションをアップロードします。

   ```
   .\upload-app-project-name.bat
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

アプリケーションをアップロードする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. ヘルパースクリプトを使用してアプリケーションをアップロードします。

   ```
   ./upload-app-project-name.sh
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Amazon S3 リソースを確認する

Amazon S3 バケットをチェックして、すべてのアップロードが成功したことを確認できます。Amazon S3 でファイルを管理する方法については、「Amazon Simple Storage Service ユーザーガイド」の「Amazon S3 バケットの作成、設定、および使用」を参照してください。

サンプルアプリケーションでは、スキーマ (前のステップでアップロードしたもの) とアプリケーションリソースは以下の名前フォーマットを使用します。

- スキーマバケット: simspaceweaver-*project-name-lowercase*-*account-number*-schemas-*region*
  - スキーマファイル: *project-name*-schema.yaml
- アプリケーションバケット: simspaceweaver-*project-name-lowercase*-*account-number*-app-zips-*region*
  - 空間アプリケーション: *project-name*Spatial.zip
  - ビュー (カスタム) アプリケーション: *project-name*View.zip

例えば、以下のプロジェクトプロパティがあるとします。

- プロジェクト名: MyProject
- AWS アカウント番号: 111122223333
- AWS リージョン: us-west-2

スキーマとアプリケーションリソースには以下のような名前が付きます。

- スキーマバケット: simspaceweaver-myproject-111122223333-schemas-us-west-2
  - スキーマファイル: MyProject-schema.yaml
- アプリケーションバケット: simspaceweaver-myproject-111122223333-apps-zips-us-west-2
  - 空間アプリケーション: MyProjectSpatial.zip
  - ビュー (カスタム) アプリケーション: MyProjectView.zip

ステップ 6: シミュレーションを開始する

SimSpace Weaver アプリケーション SDK には、シミュレーションを開始するためのヘルパースクリプトが用意されています。このスクリプトは API StartSimulationコールのラッパーです。API コールに以下のパラメータを提供します。

- シミュレーションスキーマ (前のステップでアップロードした) の名前
- シミュレーション名
- SimSpace Weaver サービスエンドポイント

Docker

シミュレーションを開始するには

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. ヘルパースクリプトを使用してシミュレーションをアップロード開始します。

   ```
   .\start-simulation-project-name.bat
   ```

   > **Important**
   >
   > バージョン 1.12.3 以降の場合、`start-simulation` スクリプトは最大 1 時間でシミュレーションを開始します。`--maximum-duration` パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

Important

AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Note

`run-`*`project-name`*`.bat` は、シミュレーションクロックも起動する代替ヘルパースクリプトです。このチュートリアルでは、後のステップでクロックを個別に起動します。

バージョン 1.12.3 以降の場合、`run` スクリプトは最大 1 時間でシミュレーションを開始します。`--maximum-duration` パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

シミュレーションを開始するには

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. ヘルパースクリプトを使用してシミュレーションをアップロード開始します。

   ```
   ./start-simulation-project-name.sh
   ```

   > **Important**
   >
   > バージョン 1.12.3 以降の場合、`start-simulation` スクリプトは最大 1 時間でシミュレーションを開始します。`--maximum-duration` パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0

です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

> **Note**
>
> `run-`*`project-name`*`.sh` は、シミュレーションクロックも起動する代替ヘルパースクリプトです。このチュートリアルでは、後のステップでクロックを個別に起動します。
>
> バージョン 1.12.3 以降の場合、`run` スクリプトは最大 1 時間でシミュレーションを開始します。`--maximum-duration` パラメータを使用して、別の最大時間を指定できます。バージョン 1.12.2 以前の場合、スクリプトに最大期間を指定することはできず、シミュレーションの最大所要時間は 14 日間です。シミュレーションの最大時間の詳細については、「シミュレーションの最大期間」を参照してください。

スクリプトは、シミュレーションのステータスが `STARTED` または `FAILED` になるまでループします。シミュレーションが開始されるまで数分かかる場合があります。シミュレーションが正常に開始されると、以下のような出力が表示されます。

```
[2022-10-04T22:15:28] [INFO] Describe Simulation Results:
[2022-10-04T22:15:28] [INFO] {
    "Status": "STARTED",
    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "RoleArn": "arn:aws:iam::111122223333:role/weaver-MyProject-app-role",
    "CreationTime": 1664921418.09,
    "SchemaS3Location": {
        "ObjectKey": "MyProject-schema.yaml",
```

```
        "BucketName": "weaver-myproject-111122223333-us-west-2"
    },
```

> **Note**
>
> SimSpace Weaver アプリケーション SDK バージョン 1.12.x のプロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
>
> - weaver*lowercase-project-name-account-number* -app-zips-*region*
> - weaver*lowercase-project-name-account-number* -schemas-*region*

ステップ 7: シミュレーションを取得する

SimSpace Weaver アプリケーション SDK には、AWS CLI を包含するヘルパースクリプトが用意されています。このスクリプトは SimSpace Weaver サービスエンドポイントを提供することで、AWS CLI への呼び出しを簡素化します。このヘルパースクリプトを使用して SimSpace Weaver API を呼び出します。DescribeSimulation API は、シミュレーションの状態など、シミュレーションに関する詳細を提供します。シミュレーションは、以下の状態のいずれかになります。

シミュレーションライフサイクルの状態

1. **STARTING** — StartSimulation 呼び出し後の初期状態
2. **STARTED** — すべての空間アプリケーションが起動し、正常に動作している
3. **STOPPING** — StopSimulation 呼び出し後の初期状態
4. **STOPPED** — すべてのコンピュートリソースが停止している
5. **DELETING** — DeleteSimulation 呼び出し後の初期状態
6. **DELETED** — シミュレーションに割り当てられたすべてのリソースが削除されている
7. **FAILED** — シミュレーションに重大なエラー/障害が発生して停止している
8. **SNAPSHOT_IN_PROGRESS** — スナップショットが進行中

Docker

シミュレーションの詳細を取得する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. CLI ヘルパースクリプトを使用して ListSimulations API を呼び出します。

   ```
   .\weaver-project-name-cli.bat list-simulations
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

   このスクリプトには、以下のような各シミュレーションの詳細が表示されます。

   ```
   {
       "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"
}
```

3. DescribeSimulation を呼び出して、シミュレーションの詳細を取得します。*simulation-name* を前のステップの出力のシミュレーションの Name に置き換えます。

```
.\weaver-project-name-cli.bat describe-simulation --simulation simulation-name
```

スクリプトには、以下のように、指定したシミュレーションに関する詳細が表示されます。

```
{

    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "ExecutionId": "1a2b3c4d-0ab1-1234-567a-12ab34cd5e6f",
    "Arn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15",
    "RoleArn": "arn:aws:iam::111122223333:role/weaver-MyProject-app-role",
    "CreationTime": 1664921418.09,
    "Status": "STARTED",
    "TargetStatus": "STARTED",
    "SchemaS3Location": {
        "ObjectKey": "MyProject-schema.yaml",
        "BucketName": "weaver-myproject-111122223333-us-west-2"
    },
    "SchemaError": "[]",
    "LoggingConfiguration": {
        "Destinations": [
            {
                "CloudWatchLogsLogGroup": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:111122223333:log-
group:MySimulationLogs"
                }
            }
        ]
    },
    "LiveSimulationState": {
        "Domains": [
            {
```

```
                "Type": "",
                "Name": "MySpatialSimulation",
                "Lifecycle": "Unknown"
            },
            {
                "Type": "",
                "Name": "MyViewDomain",
                "Lifecycle": "ByRequest"
            }
        ],
        "Clocks": [
            {
                "Status": "STARTED",
                "TargetStatus": "STARTED"
            }
        ]
    },
    "MaximumDuration": "1H",
    "StartError": "[]"
}
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

シミュレーションの詳細を取得する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. CLI ヘルパースクリプトを使用して ListSimulations API を呼び出します。

```
./weaver-project-name-cli.sh list-simulations
```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

このスクリプトには、以下のような各シミュレーションの詳細が表示されます。

```
{
    "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"
}
```

3. DescribeSimulation を呼び出して、シミュレーションの詳細を取得します。*`simulation-name`* を前のステップの出力のシミュレーションの Name に置き換えます。

```
./weaver-project-name-cli.sh describe-simulation --simulation simulation-name
```

スクリプトには、以下のように、指定したシミュレーションに関する詳細が表示されます。

```
{

    "Name": "MyProjectSimulation_22-10-04_22_10_15",
    "ExecutionId": "1a2b3c4d-0ab1-1234-567a-12ab34cd5e6f",
    "Arn": "arn:aws:simspaceweaver:us-west-2:111122223333:simulation/
MyProjectSimulation_22-10-04_22_10_15",
    "RoleArn": "arn:aws:iam::111122223333:role/weaver-MyProject-app-role",
    "CreationTime": 1664921418.09,
    "Status": "STARTED",
    "TargetStatus": "STARTED",
    "SchemaS3Location": {
        "ObjectKey": "MyProject-schema.yaml",
        "BucketName": "weaver-myproject-111122223333-us-west-2"
    },
    "SchemaError": "[]",
    "LoggingConfiguration": {
        "Destinations": [
            {
                "CloudWatchLogsLogGroup": {
                    "LogGroupArn": "arn:aws:logs:us-west-2:111122223333:log-
group:MySimulationLogs"
                }
            }
        ]
    },
    "LiveSimulationState": {
        "Domains": [
            {
                "Type": "",
                "Name": "MySpatialSimulation",
                "Lifecycle": "Unknown"
            },
            {
                "Type": "",
                "Name": "MyViewDomain",
                "Lifecycle": "ByRequest"
            }
        ],
        "Clocks": [
```

```
            {
                "Status": "STARTED",
                "TargetStatus": "STARTED"
            }
        ]
    },
    "MaximumDuration": "1H",
    "StartError": "[]"
}
```

ステップ 8: カスタムアプリケーションを起動する

SimSpace Weaver はカスタムアプリケーションのライフサイクルを管理しません。カスタムアプリケーションを起動する必要があります。シミュレーションクロックを開始する前にカスタムアプリケーションを起動するのがベストプラクティスですが、カスタムアプリケーションはシミュレーションクロックを開始した後でも起動できます。

CLI ヘルパースクリプトを使用して StartApp API を呼び出し、カスタムアプリケーションを起動できます。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Docker

```
.\weaver-project-name-cli.bat start-app --simulation simulation-name --name app-name
 --domain domain-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
./weaver-project-name-cli.sh start-app --simulation simulation-name --name app-name
 --domain domain-name
```

StartApp API コールは、指定した名前を使用してカスタムアプリケーションの新しいインスタンスを作成して起動します。既に存在するアプリケーション名を指定すると、エラーが返されます。特定のアプリケーション (インスタンス) を再起動する場合は、まずそのアプリケーションを停止して削除する必要があります。

> **Note**
>
> カスタムアプリケーションの起動前は、シミュレーションのステータスは STARTED である必要があります。シミュレーションのステータスを確認するには、「ステップ 7: シミュレーションを取得する」を参照してください。

サンプルアプリケーションには、シミュレーションを表示する ViewApp カスタムアプリケーションが用意されています。このアプリケーションは、シミュレーションクライアントを接続するための静的 IP アドレスとポート番号を提供します (これについてはこのチュートリアルの後のステップで行います)。domain は、同じ実行コードと起動オプションを持つアプリケーションのクラスと考えることができます。app name はアプリケーションのインスタンスを識別します。SimSpace Weaver の概念の詳細については、「SimSpace Weaver の主要なコンセプト」を参照してください。

DescribeApp API を使用して、起動後にカスタムアプリケーションのステータスを確認できます。

Docker

```
.\weaver-project-name-cli.bat describe-app --simulation simulation-name --app app-name --domain domain-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
./weaver-project-name-cli.sh describe-app --simulation simulation-name --app app-name --domain domain-name
```

Docker

このチュートリアルでビューアプリケーションを起動する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. CLI ヘルパースクリプトを使用して StartApp を `ViewApp` に呼び出します。

   ```
   .\weaver-project-name-cli.bat start-app --simulation simulation-name --name
    ViewApp --domain MyViewDomain
   ```

3. DescribeApp を呼び出して、カスタムアプリケーションのステータスを確認します。

   ```
   .\weaver-project-name-cli.bat describe-app --simulation simulation-name --app
    ViewApp --domain MyViewDomain
   ```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

このチュートリアルでビューアプリケーションを起動する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. CLI ヘルパースクリプトを使用して StartApp を ViewApp に呼び出します。

   ```
   ./weaver-project-name-cli.sh start-app --simulation simulation-name --name
    ViewApp --domain MyViewDomain
   ```

3. DescribeApp を呼び出して、カスタムアプリケーションのステータスを確認します。

   ```
   ./weaver-project-name-cli.sh describe-app --simulation simulation-name --app
    ViewApp --domain MyViewDomain
   ```

カスタムアプリケーション (インスタンス) のステータスが STARTED になると、DescribeApp の出力にはそのカスタムアプリケーション (インスタンス) の IP アドレスとポート番号が含まれます。以下の出力例では、IP アドレスは Address の値で、ポート番号は EndpointInfo ブロック内の Actual の値です。

```
{
    "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**
>
> `Declared` の値はアプリケーションコードのバインド先となるポート番号です。`Actual` の値は、アプリケーションに接続する際に SimSpace Weaver がクライアントに公開するポート番号です。SimSpace Weaver は `Declared` ポートを `Actual` ポートにマップします。

> **Note**
>
> クイックスタートチュートリアルの手順を使用して、このワークフローとは関係なく、起動したカスタムアプリケーションの IP アドレスとポート番号を取得できます。

ステップ 9: クロックを起動する

シミュレーションを初めて作成したとき、クロックはありますが、作動していません。クロックが作動していないときは、シミュレーションの状態は更新されません。クロックを起動すると、アプリにティックが送信され始めます。空間アプリケーションは、ティックのたびに所有するエンティティを順番に確認し、SimSpace Weaver に結果をコミットします。

> **Note**
>
> クロックの起動には 30 ~ 60 秒かかることがあります。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Docker

クロックを起動する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*`project-folder`* はプロジェクトの作成時に入力した値を使用する *`path`*`\`*`project-name`* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. CLI ヘルパースクリプトを使用して StartClock API を呼び出します。

   ```
   .\weaver-project-name-cli.bat start-clock --simulation simulation-name
   ```

> **Note**
>
> StartClock API は *simulation-name* を使用します。これは ListSimulations API を使用して検索できます。
>
> ```
> .\weaver-project-name-cli.bat list-simulations
> ```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

クロックを起動する

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

   ```
   cd project-folder/tools/linux
   ```

2. CLI ヘルパースクリプトを使用して StartClock API を呼び出します。

   ```
   ./weaver-project-name-cli.sh start-clock --simulation simulation-name
   ```

   > **Note**
   >
   > StartClock API は *simulation-name* を使用します。これは ListSimulations API を使用して検索できます。

```
./weaver-project-name-cli.sh list-simulations
```

ステップ 10: ログを確認する

SimSpace Weaver は、アプリケーションからのシミュレーション管理メッセージとコンソール出力を Amazon CloudWatch Logs に書き込みます。ログの使用の詳細については、「Amazon Logs ユーザーガイド」の「ロググループとログストリームの使用」を参照してください。 CloudWatch

作成した各シミュレーションには、 CloudWatch Logs に独自のロググループがあります。ロググループの名前は、シミュレーションスキーマで指定されます。以下のスキーマスニペットでは、`log_destination_service` の値は `logs` です。つまり、`log_destination_resource_name` の値はロググループの名前です。この場合、ロググループは `MySimulationLogs` です。

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

DescribeSimulation API を使用して、シミュレーションを開始した後でシミュレーション用のロググループの名前を検索することもできます。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユー

ザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Docker

```
project-folder\tools\windows\weaver-project-name-cli.bat describe-simulation --
simulation simulation-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
project-folder/tools/linux/weaver-project-name-cli.sh describe-simulation --
simulation simulation-name
```

以下の例は、ロギング設定を説明する、DescribeSimulation から出力の一部を示しています。ロググループの名前は LogGroupArn の末尾に表示されます。

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

各シミュレーションロググループには、いくつかのログストリームが含まれます。

- 管理ログストリーム — SimSpace Weaver サービスによって生成されるシミュレーション管理メッセージ。

```
/sim/management
```

- エラーログストリーム — SimSpace Weaver サービスによって生成されるエラーメッセージ。このログストリームはエラーがある場合にのみ存在します。SimSpace Weaver はアプリケーションによって書き込まれたエラーを独自のアプリケーションログストリームに保存します (以下の「ログストリーム」を参照してください)。

```
/sim/errors
```

- 空間アプリケーションログストリーム (各ワーカーの空間アプリケーションごとに 1 つ) — 空間アプリケーションによって生成されるコンソール出力。各空間アプリケーションは、独自のログストリームに書き込みます。*spatial-app-id* は、*worker-id* の末尾にあるスラッシュの後のすべての文字です。

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

- カスタムアプリケーションログストリーム (カスタムアプリケーションインスタンスごとに 1 つ) — カスタムアプリケーションによって生成されるコンソール出力。各カスタムアプリケーションインスタンスは、独自のログストリームに書き込みます。

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

- サービスアプリケーションログストリーム (サービスアプリケーションインスタンスごとに 1 つ) — サービスアプリケーションによって生成されるコンソール出力。各サービスアプリケーションは、独自のログストリームに書き込みます。*service-app-id* は、*service-app-name* の末尾にあるスラッシュの後のすべての文字です。

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

> **Note**
>
> サンプルアプリケーションにはサービスアプリケーションはありません。

ステップ 11: シミュレーションを表示する

SimSpace Weaver アプリケーション SDK には、サンプルアプリケーションを表示するためのさまざまなオプションが用意されています。Unreal Engine 開発用のローカルサポートがない場合は、サンプルコンソールクライアントを使用できます。Unreal Engine クライアントへの説明では、Windows の使用を前提としています。

コンソールクライアントは、エンティティイベントが発生するとそのリストを表示します。クライアントは `ViewApp` からエンティティイベント情報を取得します。コンソールクライアントがイベントのリストを表示すると、シミュレーション内の `ViewApp` およびアクティビティとのネットワーク接続を確認します。

`PathfindingSample` シミュレーションでは、二次元平面上に静止しているエンティティと動いているエンティティが作成されます。移動するエンティティは静止しているエンティティの周りを移動します。Unreal Engine クライアントはエンティティイベントを視覚化します。

Windows console client

要件

- Microsoft Windows 10 以上
- Microsoft Visual Studio 2019 またはそれ以降、Desktop development with C++ ワークロードをインストールした状態
- CMake3
- Git

サンプルコンソールクライアントを使用してサンプルアプリケーションに接続する

1. コマンドプロンプトウィンドウで、コンソールクライアントのフォルダ (アプリケーション SDK フォルダ内) に移動します。

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

2. CMake3 を使用して、このフォルダに Visual Studio ソリューションを作成します。

   ```
   cmake .
   ```

> **Note**
>
> 末尾に必ずスペースとピリオドを含めます。

> **Important**
>
> 以降の手順については、コマンドプロンプトウィンドウを開いたままにしておきます。

3. 前のステップで作成した Visual Studio で、`PathfindingSampleConsoleClient.sln` を開きます。
4. RelWithDebInfo ビルド設定を選択します。
5. [Build]、[Build Solution] の順に選択します。
6. 前のコマンドプロンプトウィンドウで、[コンソールクライアント] フォルダにある [出力をビルド] フォルダに移動します。

```
cd RelWithDebInfo
```

7. `ViewApp`の IP アドレスとポート番号を使用してクライアントを実行します。

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

コマンドプロンプトウィンドウには、以下の出力例のように、エンティティの更新、削除、および作成イベントの番号が表示されます。

> **Note**
>
> 以下の出力例の IP アドレスとポート番号はプレースホルダーです。コンソールクライアントに、`ViewApp` の IP アドレスとポート番号を指定します。AWS クラウドで動作する `ViewApp` に接続する場合は、`Actual` ポート番号を指定します。ローカルシステムで動作する `ViewApp` に接続するときは、IP アドレスとポート番号 127.0.0.1:7000 を指定します。詳細については、「ローカル開発」を参照してください。

```
##PathfindingSample#ViewApp Message Reader##

Added argument url:tcp://198.51.100.135:4321
Some subscription arguments are missing, restoring defaults.

*****
Sample usage without a MoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50
Sample usage with CircleMoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50 --subs-move-strategy circle --circle-center-x 500 --
circle-center-y 500 --circle-speed 0.001

*****
Starting NNG client. NNG version: 1.2.4
Creating socket ...done.
Connecting to View App ... done.
Initiating connection to tcp:// 198.51.100.135:4321 ... done.

Receiving messages ...
[2022-10-04 19:13:00.710] CreateEntity Count: 72
[2022-10-04 19:13:00.756] UpdateEntity Count: 42
[2022-10-04 19:13:00.794] DeleteEntity Count: 72
[2022-10-04 19:13:03.690] CreateEntity Count: 11
[2022-10-04 19:13:03.725] UpdateEntity Count: 2
[2022-10-04 19:13:03.757] UpdateEntity Count: 2
[2022-10-04 19:13:03.790] UpdateEntity Count: 2
```

> **Note**
>
> トラブルシューティングのガイダンスについては、「PathfindingSample コンソールクライアントが接続に失敗する」を参照してください。

8. CTRL+C を押してコンソールクライアントを終了します。

Linux console client

> **Important**
>
> 便宜上、これらの指示を使用します。Linux 環境の一部では動作しない可能性があります。これらの手順はサポートされていません。

この手順は、すべてを Linux 環境内で作業していることを前提としています。Windows 組み込みクライアントを使用してシミュレーションを表示することもできます。

要件

- CMake3
- C コンパイラ (Amazon Linux 2 に既に含まれています)
- Git

サンプルコンソールクライアントを使用してサンプルアプリケーションに接続する

1. Linux シェルプロンプトで、[コンソールクライアント] フォルダ (アプリケーション SDK フォルダ内) に移動します。

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

2. ビルドフォルダを作成します。

   ```
   mkdir build
   ```

3. ビルドフォルダに移動します。

   ```
   cd build
   ```

4. CMake3 を使用して、クライアントをビルドします。

   ```
   cmake3 ../ && cmake3 --build .
   ```

> **Note**
>
> 末尾に必ずスペースとピリオドを含めます。

5. ViewAppの IP アドレスとポート番号を使用してクライアントを実行します。

```
./ConsoleClient --url tcp://ip-address:port-number
```

コマンドプロンプトウィンドウには、以下の出力例のように、エンティティの更新、削除、および作成イベントの番号が表示されます。

> **Note**
>
> 以下の出力例の IP アドレスとポート番号はプレースホルダーです。コンソールクライアントに、ViewApp の IP アドレスとポート番号を指定します。AWS クラウドで動作する ViewApp に接続する場合は、Actual ポート番号を指定します。ローカルシステムで動作する ViewApp に接続するときは、IP アドレスとポート番号 127.0.0.1:7000 を指定します。詳細については、「ローカル開発」を参照してください。

```
##PathfindingSample#ViewApp Message Reader##

Added argument url:tcp://198.51.100.135:4321
Some subscription arguments are missing, restoring defaults.

*****
Sample usage without a MoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50
Sample usage with CircleMoveStrategy:
ConsoleClient --url tcp://198.51.100.135:4321 --subs-center-x 600 --subs-center-
y 500 --subs-radius 50 --subs-move-strategy circle --circle-center-x 500 --
circle-center-y 500 --circle-speed 0.001

*****
Starting NNG client. NNG version: 1.2.4
```

```
Creating socket ...done.
Connecting to View App ... done.
Initiating connection to tcp:// 198.51.100.135:4321 ... done.

Receiving messages ...
[2022-10-04 19:13:00.710] CreateEntity Count: 72
[2022-10-04 19:13:00.756] UpdateEntity Count: 42
[2022-10-04 19:13:00.794] DeleteEntity Count: 72
[2022-10-04 19:13:03.690] CreateEntity Count: 11
[2022-10-04 19:13:03.725] UpdateEntity Count: 2
[2022-10-04 19:13:03.757] UpdateEntity Count: 2
[2022-10-04 19:13:03.790] UpdateEntity Count: 2
```

> **Note**
>
> トラブルシューティングのガイダンスについては、「PathfindingSample コンソールクライアントが接続に失敗する」を参照してください。

6. CTRL+C を押してコンソールクライアントを終了します。

Unreal Engine on Windows

要件

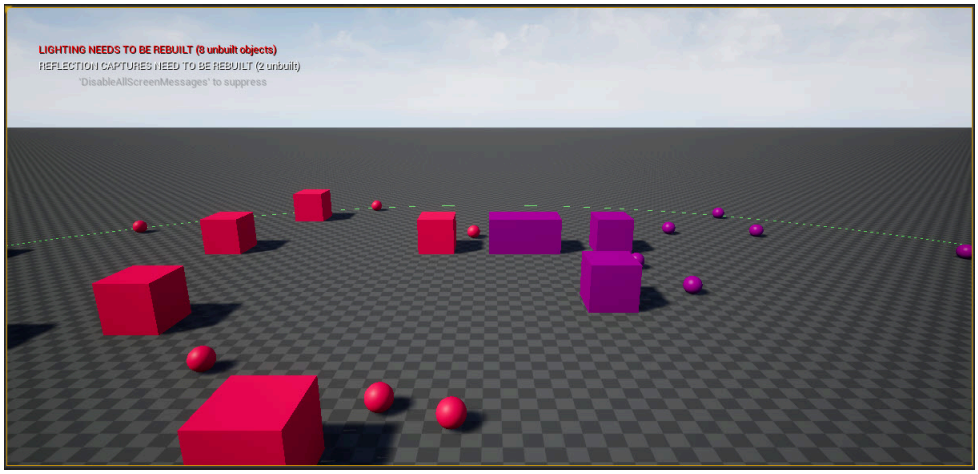
- Unreal Engine 5 開発環境
- Microsoft .NET Framework 4.8 Developer Pack
- Windows コンソールクライアント (このページの「Windows コンソールクライアント」タブを参照してください)

> **Important**
>
> 他のバージョンの Unreal Engine および .NET はサポートされていないため、問題が発生する可能性があります。

サンプル Unreal クライアントを使用してサンプルアプリケーションに接続する

1. Unreal Engine クライアントはコンソールクライアントの NNG ライブラリを使用します。Windows のコンソールクライアントをまだビルドしていない場合は、ビルドする必要があります。詳細は、このページの [Windows コンソールクライアント] タブを参照してください。
2. ファイルマネージャーウィンドウで、*sdk-folder*\packaging-tools\clients\PathfindingSampleClients\UnrealClient に進みます。
3. UnrealClient.uproject を開きます。
4. エディタから UnrealClient モジュールを再構築するかどうか尋ねられたら、[yes] を選択します。
5. テキストエディタで *sdk-folder*\packaging-tools\clients\PathfindingSampleClients\UnrealClient\view_app_url.txt を開きます。
6. ビューアプリケーションの IP アドレスとポート番号で URL を更新します: tcp://*ip-address*:*port-number* (tcp://198.51.100.135:1234 のようになります)。
7. Unreal エディタ で [play] を選択します。

Unreal エディタには、以下のスクリーンショットのようなシミュレーションのビジュアルが表示されます。



> **Note**
>
> ローカル開発システムの性能によっては、Unreal エディタがシミュレーションを表示するまでに数分かかることがあります。この間、システムがフリーズしているように見えることがあります。

W、A、S、D キーを使用して Unreal クライアント内を移動します。マウスボタンを押したままマウスをドラッグすると回転します。

[(左角かっこ) キーを押すと、サブスクリプション領域のサイズを小さくできます。] (右角かっこ) キーを押すと、サブスクリプション領域のサイズを大きくできます。サブスクリプション領域のサイズによって、クライアントに表示されるエンティティの数が決まります。

C キーを押すと、シミュレーションにエンティティを作成できます。クライアントはビューアプリケーションに `CreateEntity` コマンドを送信します。次に、ビューアプリケーションはエンティティを作成し、空間ドメインに転送します。

`project-folder``\src\PathfindingSample\ViewApp\Driver\ViewAppDriver.cpp` の `ViewAppDriver::HandleEntityCreationRequests` コードを調べて、アプリケーションがこのプロセスをどのように実装しているかを確認できます。

> **Note**
>
> ビューアプリケーションの IP アドレスとポート番号が不明な場合は、「クイックスタートチュートリアル」で手順を確認してください。

ステップ 12: シミュレーションを停止してクリーンアップする

不要になったシミュレーションはクリーンアップすることが重要です。SimSpace Weaver シミュレーションが停止しても、シミュレーションリソースは Service Quotas (制限) に加算されます。実行中のシミュレーションについては、引き続き料金が発生します。また、Amazon CloudWatch Logs や Amazon Simple Storage Service など、サポートする のサービスのデータストレージに対する請求が発生する場合もあります。SimSpace Weaverサービスクォータの詳細については、「」を参照してくださいSimSpace Weaver エンドポイントとクォータ。

シミュレーションをクリーンアップする準備ができたら、このセクションの手順に従います。

> **Important**
>
> 停止したシミュレーションは再開できません。

> **Important**
>
> 削除されたシミュレーションを復元することはできません。

SimSpace Weaver のシミュレーションリソースをクリーンアップする

シミュレーションは、停止してから削除する必要があります。シミュレーションを削除すると、SimSpace Weaver 内のリソースのみが削除されます。シミュレーションで作成したリソースや他のサービスで使用しているリソースを削除するには、別の手順を実行する必要があります (以下のセクションを参照してください)。

Docker

シミュレーションをクリーンアップする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトの作成時に入力した値を使用する *path**project-name* です。

   [Windows] コマンドプロンプトで、以下のように入力します。

   ```
   cd project-folder\tools\windows
   ```

2. シミュレーション名を検索します。

   ```
   .\weaver-project-name-cli.bat list-simulations
   ```

   > **Important**
   >
   > AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定で

きます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

3. シミュレーションを停止します。

```
.\weaver-project-name-cli.bat stop-simulation --simulation simulation-name
```

4. 停止したシミュレーションを削除します。

```
.\weaver-project-name-cli.bat delete-simulation --simulation simulation-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

シミュレーションをクリーンアップする

1. まだできていない場合は、プロジェクトとプラットフォームの [ツール] フォルダに移動します。*project-folder* はプロジェクトを作成したときに入力した値を使用する *path*/*project-name* です。

   Linux シェルプロンプトで、以下のように入力します。

```
cd project-folder/tools/linux
```

2. シミュレーション名を検索します。

```
./weaver-project-name-cli.sh list-simulations
```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

3. シミュレーションを停止します。

```
./weaver-project-name-cli.sh stop-simulation --simulation simulation-name
```

4. 停止したシミュレーションを削除します。

```
./weaver-project-name-cli.sh delete-simulation --simulation simulation-name
```

AWS Management Console

シミュレーションをクリーンアップする

1. [SimSpace Weaver] コンソールで [SimSpace Weaver] を開きます。
2. ナビゲーションペインで、[Simulations] を選択します。
3. [Simulations] リストから、削除するシミュレーション名の横にあるオプションを選択します。
4. 選択したシミュレーションの Status が STARTED の場合:
   a. [Actions] ドロップダウンメニューを選択します。
   b. [Stop] を選択します。

c. 確認するには、シミュレーション名を入力します。

d. [Stop] を選択します。

e. シミュレーションの Status が STOPPED になるまで待ちます。

5. [Actions] ドロップダウンメニューを選択します。
6. [Delete] を選択します。
7. [Delete] を選択して確定します。

サポートサービスのシミュレーションリソースをクリーンアップする

シミュレーションをサポートするために、SimSpace Weaver は他のサービスのリソースを作成します。シミュレーションを削除しても、SimSpace Weaver ではこれらのリソースは削除されません。これらの追加リソースが不要な場合は削除できます。

> **Important**
>
> これらのリソースを削除しない場合、料金が発生する可能性があります。

プロジェクトのサポートリソースを削除する

1. プロジェクトが終了したら、その AWS CloudFormation スタックを削除します。AWS CloudFormation の使用の詳細については、「AWS CloudFormation ユーザーガイド」の「[AWS CloudFormation] コンソールでスタックを削除する」を参照してください。

   - weaver-*project-name*-stack

   > **Important**
   >
   > 同じプロジェクトから開始したシミュレーションは、アプリケーションロールなどのリソースを共有します。AWS CloudFormation スタックを削除すると、アプリケーションロールを削除することになります。同じリソースを共有する他のシミュレーションがある場合は、AWS CloudFormation スタックを削除しないでください。

> **Note**
>
> 空でない Amazon S3 バケットは削除できないため、AWS CloudFormation スタックが DELETE_FAILED を報告する可能性が高くなります。次のステップで Amazon S3 バケットを削除します。

2. プロジェクトが終了したら、その Amazon S3 バケットを削除します。Amazon S3 バケットの使用の詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケットの削除」を参照してください。

   - `weaver-lowercase-project-name-account-number-region`

   以下の例では、リージョン `MyProject` 内のアカウント `111122223333` に登録されている `us-west-2` という名前のプロジェクトには以下のバケットがあります。

   - `weaver-myproject-111122223333-us-west-2`

   > **Note**
   >
   > Amazon S3 バケットを削除するには、その前にバケットの内容を削除する必要があります。

   > **Note**
   >
   > SimSpace Weaver アプリケーション SDK バージョン 1.12.x のプロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
   >
   > - weaver*lowercase-project-name-account-number* -app-zips-*region*
   > - weaver*lowercase-project-name-account-number* -schemas-*region*

3. シミュレーションのログ記録を有効にした場合は、Logs CloudWatch ロググループを削除します。 CloudWatch ログの使用の詳細については、「Amazon Logs ユーザーガイド」の「ロググループとログストリームの使用」を参照してください。 CloudWatch

シミュレーションのロググループの名前は、スキーマ (設定ファイル) *project-folder*\tools*project-name*.yaml で指定されています。

ロググループ名は log_destination_resource_name の値です。以下のスキーマスニペットは、サンプルアプリケーションのロググループが MySimulationLogs であることを示しています。

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

Warning

同じロググループを指定する複数のシミュレーションを開始すると、それらのシミュレーションのログデータはすべて同じロググループに送られます。ロググループを削除すると、そのロググループを使用するすべてのシミュレーションのログデータが削除されます。実行中のシミュレーションのロググループを削除すると、シミュレーションは失敗します。

Important

シミュレーションのスキーマで log_destination_service: "logs"と が指定されlog_destination_resource_nameていても CloudWatch 、ロググループに ログが見つからない場合は、シミュレーションが実行されAWS リージョンたのと同じ を確認してください。

SimSpace Weaver の使用

この章では、SimSpace Weaver で独自のアプリケーションを構築するのに役立つ情報とガイダンスを提供します。

トピック

シミュレーションの設定

シミュレーションスキーマ (またはスキーマ) は、シミュレーションの設定を指定する YAML フォーマットのテキストファイルです。複数のシミュレーションの開始に同じスキーマを使用できます。スキーマファイルは、シミュレーションのプロジェクトフォルダにあります。任意のテキスト工ディターを使用してファイルを編集できます。SimSpace Weaver はシミュレーションの開始時にのみスキーマを読み取ります。スキーマファイルに加えた編集は、編集後に開始する新しいシミュレーションにのみ影響します。

Docker

シミュレーションを設定するには、シミュレーションスキーマファイルを編集します。

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

シミュレーションスキーマは、新しいシミュレーションの作成時にアップロードします。プロジェクトのクイックスタートヘルパースクリプトは、シミュレーションを構築するプロセスの一環としてスキーマをアップロードします。

```
project-folder\tools\windows\quick-start-project-name-cli.bat
```

クイックスタートスクリプトを使用してシミュレーションを構築しない場合は、プロジェクト用のスキーマアップロードヘルパースクリプトを使用することもできます。

```
project-folder\tools\windows\upload-schema-project-name.bat
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

シミュレーションを設定するには、シミュレーションスキーマファイルを編集します。

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

シミュレーションスキーマは、新しいシミュレーションの作成時にアップロードします。プロジェクトのクイックスタートヘルパースクリプトは、シミュレーションを構築するプロセスの一環としてスキーマをアップロードします。

```
project-folder/tools/linux/quick-start-project-name-cli.sh
```

クイックスタートスクリプトを使用してシミュレーションを構築しない場合は、プロジェクト用のスキーマアップロードヘルパースクリプトを使用することもできます。

```
project-folder/tools/linux/upload-schema-project-name.sh
```

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザガイド」「設定と認証情報ファイルの設定」を参照してください。

シミュレーションの設定パラメータ

シミュレーションスキーマには、以下のようなブートストラップ情報が含まれています。

- シミュレーションプロパティ — SDK バージョンおよびコンピューティングの設定 (ワーカーのタイプと数)
- クロック — ティックレートと許容誤差
- 空間パーティショニング戦略 — 空間トポロジ (グリッドなど)、境界、配置グループ (ワーカーの空間パーティショングループ)
- ドメインとそのアプリケーション — アプリケーションバケット、パス、起動コマンド

SimSpace Weaver はスキーマ設定を使用して、空間パーティションの設定と配置、アプリケーションの起動、指定したティックレートでのシミュレーションの進行を行います。

> **Note**
>
> SimSpace Weaver アプリケーション SDK の create-project スクリプトは、サンプルアプリケーションに基づいてシミュレーションスキーマを自動的に生成します。

以下のトピックでは、シミュレーションスキーマのパラメータについて説明します。シミュレーションスキーマの詳細な説明については、SimSpace Weaver シミュレーションスキーマリファレンス を参照してください。

トピック

SDK のバージョン

`sdk_version` フィールドは、スキーマのフォーマット対象となる SimSpace Weaver のバージョンを指定します。有効な値:`1.16`、`1.15`、`1.14`、`1.13`、`1.12`

> **Important**
>
> `sdk_version` の値には、メジャーバージョン番号と最初のマイナーバージョン番号のみが含まれます。例えば、値 `1.12` は `1.12.0`、`1.12.1`、`1.12.2` などのすべてのバージョン `1.12.x` を指定します。

シミュレーションプロパティ

スキーマの `simulation_properties` セクションでは、エンティティのインデックスフィールド (通常は空間位置) のログ記録設定とデータタイプを指定します。

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

log_destination_service の値によって、log_destination_resource_name の値の解釈が決まります。現在、サポートされている値は logs のみです。これは、log_destination_resource_name の値が Amazon CloudWatch Logs のロググループの名前であることを意味します。

> **Note**
>
> ログ記録はオプションです。ログ送信先のプロパティを設定しない場合、シミュレーションではログは生成されません。

default_entity_index_key_type プロパティは必須です。唯一の有効な値は Vector3<f32> です。

ワーカー

workers セクションでは、シミュレーションに必要なワーカーのタイプと数を指定します。SimSpace Weaver は Amazon EC2 インスタンスタイプに対応する独自のワーカータイプを使用します。

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

マルチワーカーシミュレーションを有効にする

複数のワーカーを使用するシミュレーションを作成できます。デフォルトでは、シミュレーションは 1 人のワーカーを使用します。シミュレーションを開始する前に、シミュレーションスキーマを変更する必要があります。

> **Note**
>
> すでに開始されているシミュレーションは変更できません。実行中のシミュレーションで複数のワーカーを有効にする場合は、まずシミュレーションを停止して削除する必要があります。

複数のワーカーを使用するには、コンピュートインスタンスの desired の数を 1 より大きい値に設定します。各ワーカーにはアプリケーションの最大数があります。詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。SimSpace Weaver はワーカーのアプリケーション数がこの制限を超える場合にのみ、1 人以上のワーカーを使用します。SimSpace Weaver は利用可能な任意のワーカーにアプリケーションを配置できます。特定のワーカーへのアプリケーションの配置は保証されません。

以下のスキーマスニペットは、2 人のワーカーをリクエストするシミュレーションの設定を示しています。SimSpace Weaver はアプリケーションの数がワーカーあたりの最大アプリケーション数を超えると、2 番目のワーカーの割り当てを試みます。

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

クロック

この clock セクションではシミュレーションクロックのプロパティを指定します。現在、設定できるのはティックレート (クロックがアプリケーションに送信する 1 秒あたりのティック数) のみです。ティックレートは最大レートです。ティックに対するすべての操作 (エンティティの更新など) は次のティックの開始前に終了する必要があるため、実効ティック率が低くなる可能性があります。ティックレートはクロックレートとも呼ばれます。

tick_rateの有効値は、スキーマで特定される sdk_version によって異なります。

ティックレートの有効値

- "1.14" 以前のバージョン:
  - 10
  - 15

  - 30
- "1.14" 以降のバージョン:
  - "10"
  - "15"
  - "30"
  - "unlimited"

    詳細については、「無制限のティックレート」を参照してください。

> **Important**
>
> - "1.14" 以前の sdk_version のスキーマでは、tick_rate の値は 30 のような整数です。
> - "1.14" 以降の sdk_version のスキーマでは、tick_rate の値は "30" のような文字列です。値には二重引用符を含める必要があります。
>
>   バージョン "1.12" または "1.13" のスキーマをバージョン "1.14" 以降に変換する場合は、tick_rate の値を二重引用符で囲む必要があります。

無制限のティックレート

tick_rate を "unlimited" に設定すると、コードの実行と同じ速さでシミュレーションを実行できます。ティックレートは無制限で、SimSpace Weaver はすべてのアプリケーションが現在のティックのコミットを終了した直後に次のティックを送信します。

> **Important**
>
> 1.14.0 より前の SimSpace Weaver バージョンでは、無制限のティックレートはサポートされていません。スキーマの sdk_version の最小値は "1.14" です。

SimSpace Weaver Local で無制限のティックレート

SimSpace Weaver Local はスキーマでティックレートが 10 kHz (10000) と指定されているかのように "unlimited" を実装します。その効果は、AWS クラウド での無制限ティックレートと同じで

す。スキーマでは tick_rate: "unlimited" を引き続き指定できます。SimSpace Weaver Local の詳細については、「ローカル開発」を参照してください。

クロックに関するよくある質問

Q1. 開始したシミュレーションを別のティックレートを使用するように変更できますか?

ライフサイクルのどの段階においても、AWS クラウド にすでに存在しているシミュレーションのティックレートを変更することはできません。また、SimSpace Weaver Local で実行中のシミュレーションのティックレートは、変更できません。tick_rate をスキーマに設定して、そのスキーマから新しいシミュレーションを開始できます。

Q2. 1.14 以前のバージョンで、無制限のティックレートでシミュレーションを実行できますか？

いいえ、1.14.0 以前のバージョンでは無制限のティックレートはサポートされていません。

クロックに関するエラーのトラブルシューティング

シミュレーションが開始されない場合は、DescribeSimulation API の出力で "StartError" の値を確認できます。スキーマに無効な tick_rate の値があると、以下のエラーが発生します。

> **Note**
>
> ここに示すエラー出力は、読みやすくするために複数行で表示されています。実際のエラー出力は 1 行です。

- sdk_version は "1.14" よりも前で、tick_rate の値は無効な整数です。有効値: 10、15、30

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

- sdk_version は "1.14" よりも前で、tick_rate の値は文字列です。有効値: 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 は "1.14" よりも後で、tick_rate の値は無効な文字列です。有効値: "10"、"15"、"30"、"unlimited"

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

- sdk_version は "1.14" よりも後で、tick_rate の値は整数です。有効値: "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\"}]"
```

パーティショニング戦略

partitioning_strategies セクションでは、空間アプリケーションのパーティションの設定プロパティを指定します。パーティショニング戦略 (このセクションではプロパティのセット) に独自の名前を指定し、それを空間アプリケーションの設定に使用します。

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

topology プロパティは、シミュレーションで使用する座標システムのタイプを指定します。Grid 値は 2 次元 (2D) グリッドを指定します。

Grid トポロジーの場合、シミュレーション空間は軸に沿ったバウンディングボックス (AABB) としてモデル化されます。AABB の各軸の座標境界を aabb_bounds プロパティで指定します。シミュレーションに空間的に存在するすべてのエンティティは、AABB 内にある必要があります。

グリッド配置グループ

配置グループは、SimSpace Weaver を同じワーカーで配置したい空間アプリケーションパーティションの集まりです。配置グループの数と配置を (グリッド内の) `grid_placement_groups` プロパティで指定します。SimSpace Weaver はパーティションを配置グループ全体に均等に分散させようとします。同じ配置グループ内のパーティションを持つ空間アプリケーションの所有権エリアは、空間的に隣接します。

x * y は希望するワーカー数と同じにすることをお勧めします。等しくない場合、SimSpace Weaver は使用可能なワーカー全体で配置グループのバランスを取ろうとします。

配置グループ設定を指定しない場合、SimSpace Weaver は配置グループ設定を自動的に計算します。

ドメイン

ドメインの設定プロパティのセットの名前を指定します。ドメイン内のアプリケーションの起動設定によって、ドメインのタイプが決まります。

- **`launch_apps_via_start_app_call`** - カスタムドメイン
- **`launch_apps_by_partitioning_strategy`** — 空間ドメイン
- **`launch_apps_per_worker`** (サンプルアプリケーションには含まれていません) — サービスドメイン

> **Important**
>
> SimSpace Weaver はシミュレーションごとに最大 5 つのドメインをサポートします。これには、すべての空間ドメイン、カスタムドメイン、およびサービスドメインが含まれます。

```
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 アプリケーション SDK バージョン 1.12.x プロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
>
> - weaver-*lowercase-project-name*-*account-number*-app-zips-*region*
> - weaver-*lowercase-project-name*-*account-number*-schemas-*region*

トピック

アプリケーションの設定

アプリケーション (app_config) の設定は、そのドメインの設定の一部として指定します。すべてのタイプのドメインが同じアプリケーション設定プロパティを使用します。

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

```
    required_resource_units:
      compute: 1
```

> **Note**
>
> SimSpace Weaver アプリケーション SDK バージョン 1.12.x プロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
>
> - weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - weaver-*lowercase-project-name-account-number*-schemas-*region*

package プロパティは、S3 バケット内の zipP ファイルの S3 URI を指定します。zip ファイルには、アプリケーションの実行ファイル (バイナリとも呼ばれます) と、必要なその他のリソース (ライブラリなど) が含まれます。アプリケーション実行ファイルの各インスタンスは、ワーカーのDockerコンテナで実行されます。

launch_command プロパティは、実行ファイルの名前と、アプリケーションを実行するためのコマンドラインオプションを指定します。launch_command の値は配列です。起動コマンド文字列全体の各トークンは、配列内の要素です。

例

- 起動コマンドの場合: `MyTestApp --option1 value1`
- 指定: `launch_command: ["MyTestApp", "-option1", "value1"]`

required_resource_units プロパティは、SimSpace Weaver がこのアプリケーションに割り当てるコンピュートリソースユニットの数を指定します。コンピュートリソースユニットは、ワーカーの容量 (vCPU) とメモリ (RAM) の固定量です。この値を増やすと、ワーカー上で実行されるアプリケーションの処理能力を増やすことができます。各ワーカーが利用できるコンピュートリソースユニットの数には制限があります。詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。

空間ドメインの設定

空間ドメインの場合は、partitioning_strategy を指定する必要があります。このプロパティの値は、スキーマの別の部分で定義したパーティショニング戦略に付けた名前です。

```
  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 アプリケーション SDK バージョン 1.12.x プロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
>
> - weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - weaver-*lowercase-project-name-account-number*-schemas-*region*

Grid トポロジ (このリリースでサポートされる唯一のトポロジ) を使用したパーティショニング戦略では、SimSpace Weaver にこのドメインの空間アプリケーションパーティションをグリッドに配置するよう指示します。grid_partition プロパティは、パーティショングリッドの行数と列数を指定します。

SimSpace Weaver はパーティショングリッドのセルごとに 1 つの空間アプリケーションのインスタンスを起動します。例えば、空間ドメインに grid_partition 値 x: 2 および y: 2 があり、その空間ドメインに 2 * 2 = 4 のパーティションがあるとします。SimSpace Weaver は空間ドメインに設定されているアプリケーションの 4 つのインスタンスを起動し、各アプリケーションインスタンスに 1 つのパーティションを割り当てます。

トピック

- 空間ドメインのリソース要件
- 複数の空間ドメイン
- 空間ドメインに関するよくある質問
- 空間ドメインのトラブルシューティング

空間ドメインのリソース要件

各ワーカーには最大 17 コンピュートリソースユニットを割り当てることができます。空間ドメインの app_config セクションで、各空間アプリケーションが使用するコンピュートリソースユニットの数を指定します。

Example 空間アプリケーションのコンピュートリソースユニットを示すスキーマスニペット

```
  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
```

ドメインが必要とするコンピュートリソースユニットの数を計算するには、グリッド内のセル数 (grid_partition 内で x * y) に、空間アプリケーションに割り当てられたコンピュートリソースユニットの数を掛けます。

前の例では、ドメイン MySpatialDomain は以下を指定します。

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

MySpatialDomain のグリッドには 2 * 2 = 4 個のセルがあります。空間ドメインには 4 * 1 = 4 のコンピュートリソースユニットが必要です。

スキーマで指定する全ドメインのコンピュートリソースユニットの総数は、ワーカー数に各ワーカーのコンピュートリソースユニットの最大数 (17) を掛けた desired の数以下でなければなりません。

複数の空間ドメイン

複数の空間ドメインを使用するようにシミュレーションを設定できます。例えば、1 つの空間ドメインを使用してシミュレーションの主なアクター (人や車など) を制御し、別の空間ドメインを使用して環境を制御できます。

また、複数の空間ドメインを使用して、シミュレーションのさまざまな部分に異なるリソースを割り当てることもできます。例えば、あるタイプのエンティティが他のタイプの 10 倍のエンティティインスタンスを持つタイプのエンティティの場合、エンティティタイプごとに異なるドメインを作成して各エンティティタイプを処理し、そのエンティティの数が多いドメインにより多くのリソースを割り当てることができます。

> **Important**
>
> 1.14.0 以前の SimSpace Weaver バージョンでは、複数の空間ドメインをサポートしていません。

> **Important**
>
> 現在、AWS SimSpace Weaver Local では複数の空間ドメインはサポートされていません。SimSpace Weaver Local の詳細については、「ローカル開発」を参照してください。

> **Important**
>
> SimSpace Weaver はシミュレーションごとに最大 5 つのドメインをサポートします。これには、すべての空間ドメイン、カスタムドメイン、およびサービスドメインが含まれます。

複数の空間ドメインを設定する

複数の空間ドメインを設定するには、他の空間ドメイン定義を個別の名前付きセクションとしてスキーマに追加します。各ドメインは `launch_apps_by_partitioning_strategy` キーを指定する必要があります。次のスキーマの例を参照してください。

```
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
```

空間ドメインをまとめて配置する

シナリオによっては、空間ドメインのパーティションを別のドメインのパーティションの隣に配置する場合があります。これにより、これらのパーティションが相互にクロスドメインサブスクリプションを作成する場合に、パフォーマンス特性が向上する可能性があります。

スキーマに最上位のキー placement_constraints を追加して、SimSpace Weaver がどのドメインをまとめて配置するかを指定します。必要な on_workers キーは、スキーマ内の名前付き workers 設定を参照している必要があります。

Example 空間ドメインをまとめて配置したスキーマスニペット

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

Important

- 配置グループを使用する場合:
  - x * y がワーカー数の倍数であることを確認します。
  - 配置グループの値が、まとめて配置するドメインのグリッド寸法の共通除数であることを確認します。
- プレイスメントグループを使用しない場合:
  - 空間ドメイングリッドの 1 つの軸に、ワーカー数と等しい公約数があることを確認します。

グループ配置の詳細については、「パーティショニング戦略」を参照してください。

空間ドメインに関するよくある質問

Q1. 既存のシミュレーションに別の空間ドメインを追加する方法を教えてください。

- 実行中のシミュレーションの場合 — 実行中のシミュレーションの設定は変更できません。スキーマのドメイン設定を変更し、スキーマとアプリケーションの zip をアップロードして、新しいシミュレーションを開始します。
- 新しいシミュレーションの場合 — ドメイン設定をスキーマに追加し、スキーマとアプリケーション zip をアップロードして、新しいシミュレーションを開始します。

空間ドメインのトラブルシューティング

ドメインの設定が無効な状態でシミュレーションを開始しようとすると、以下のエラーが表示される可能性があります。

```
"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].\"}]"
```

可能性のある原因

- スキーマが、ワーカーで使用可能な量よりも多くのコンピュートリソースユニットをアプリケーションに割り当てています。
- SimSpace Weaver が複数のドメインをワーカーにまとめる方法を決定できません。これは、複数の空間ドメインを指定しても、ドメイングリッド間に共通の除数や倍数がない場合 (2x4 グリッドと 3x5 グリッドの間など) に発生します。

ネットワークエンドポイント

カスタムアプリケーションとサービスアプリケーションには、外部クライアントが接続できるネットワークエンドポイントを設定できます。endpoint_config 内の ingress_ports の値としてポート番号のリストを指定します。これらのポート番号はどちらも TCP と UDP です。カスタムアプリケーションまたはサービスアプリケーションは、ingress_ports で指定したポート番号にバインドする必要があります。SimSpace Weaver はランタイム時にポート番号を動的に割り当て、そのポートを動的ポートにマッピングします。describe-app API は、アプリケーションが動的 (実際の) ポート番号を見つけ始めた後に呼び出すことができます。詳細については、「クイックスタートチュートリアル」の「ステップ 4: IP アドレスとポート番号を取得する」を参照してください。

```
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 アプリケーション SDK バージョン 1.12.x プロジェクトでは、アプリケーションの.zip ファイルとスキーマに別々のバケットを使用します。
>
> - weaver-*lowercase-project-name-account-number*-app-zips-*region*
> - weaver-*lowercase-project-name-account-number*-schemas-*region*

> **Note**
>
> `endpoint_config` はカスタムアプリケーションとサービスアプリケーションのオプションプロパティです。`endpoint_config` を指定しない場合、アプリケーションにはネットワークエンドポイントがありません。

サービスドメインの設定

ドメイン設定に `launch_apps_per_worker:` が存在するということは、そのドメインがサービスアプリケーションを含むサービスドメインであることを示しています。SimSpace Weaver はサービスアプリケーションを自動的に起動および停止します。SimSpace Weaver がアプリケーションを起動および停止すると、そのアプリケーションではライフサイクルが管理されていると見なされます。現在、SimSpace Weaver では各ワーカーで 1 つまたは 2 つのサービスアプリケーションを起動できます。

Example 各ワーカーで 1 つのサービスアプリケーションを起動するように設定されたドメインの例

```
domains:
  MyServiceDomain:
    launch_apps_per_worker:
      count: 1
    app_config:
      package: "s3://example-bucket/PlayerConnectionServiceApp.zip"
      launch_command: ["PlayerConnectionServiceApp"]
      required_resource_units:
        compute: 1
```

```
      endpoint_config:
        ingress_ports:
          - 9000
          - 9001
```

Example 各ワーカーで 2 つのサービスアプリケーションを起動するように設定されたドメインの例

```
domains:
  MyServiceDomain:
    launch_apps_per_worker:
      count: 2
    app_config:
      package: "s3://example-bucket/PlayerConnectionServiceApp.zip"
      launch_command: ["PlayerConnectionServiceApp"]
      required_resource_units:
        compute: 1
      endpoint_config:
        ingress_ports:
          - 9000
          - 9001
```

シミュレーションの最大期間

AWS SimSpace Weaver の各シミュレーションには、シミュレーションを実行できる最大期間を指定する最大期間設定があります。シミュレーションの開始時に、最大期間をパラメータとして指定します。StartSimulation アプリケーションプログラミングインターフェイス (API) にはオプションのパラメータ MaximumDuration があります。パラメータの値は、分 (m または M)、時間 (h または H)、日 (d または D) です。たとえば、1h または 1H は 1 時間を意味します。SimSpace Weaver はこの制限に達すると、シミュレーションを停止します。

最大値

MaximumDuration の有効な最大値は 14D、またはそれと同等の時間 (336H) または分 (20160M) で表されます。

デフォルト値

MaximumDuration パラメータはオプションです。値を指定しない場合は、SimSpace Weaver は 14D の値を使用します。

最小値

MaximumDuration の有効値の最小値は、数値的には 0 と等価な値です。例えば、0M、0H、0D の値はすべて数値的には 0 と等価です。

最大期間の最小値を指定した場合、シミュレーションは STOPPING 状態に達するとすぐに STARTED 状態に移行します。

SimSpace Weaver アプリケーション SDK スクリプトを使用してシミュレーションを開始する

以下のいずれかのスクリプトを使用してシミュレーションを開始するときに、maximum-duration パラメータの値を指定できます。

- quick-start-*project-name*-cli.bat --maximum-duration *value*
- start-simulation-*project-name*.bat --maximum-duration *value*
- run-*project-name*.bat --maximum-duration *value*

各スクリプトは StartSimulation API に maximum-duration の値を渡します。

> **Important**
>
> maximum-duration の値を指定しない場合、SimSpace Weaver はデフォルト値 (14D) を使用します。

コンソールを使用したシミュレーションの開始

SimSpace Weaver コンソールでシミュレーションを開始するときに最大期間の値を指定できます。[シミュレーションの開始] を選択するときに、[シミュレーション設定] フォームの [最大期間] フィールドに値を入力します。

> **Important**
>
> [最大期間] に値を指定しない場合は、SimSpace Weaver はデフォルトの値 (14D) を使用します。

最大期間に達したシミュレーションのステータス

SimSpace Weaver が最大期間に達したシミュレーションを自動的に停止すると、シミュレーションのステータスは STOPPING (進行中の場合) または STOPPED になります。SimSpace Weaver コンソールでは、シミュレーションのターゲットステータスはまだ STARTED です。これは、ユーザーが最後に要求した状態であったためです。

アプリケーション開発

シミュレーションは AWS Cloud の Amazon Linux で実行されるため、SimSpace Weaver 開発にはアプリケーションを構築するための Amazon Linux 2 (AL2) 環境が必要です。Windows を使用している場合は、SimSpace Weaver アプリケーション SDK のスクリプトを使用して、SimSpace Weaver アプリケーションの構築に必要な依存関係で AL2 を実行する Docker コンテナを作成して起動できます。Windows Subsystem for Linux (WSL) またはネイティブ AL2 システムを使用して AL2 環境を起動することもできます。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

> **Note**
>
> ローカル開発環境の設定にかかわらず、AWS クラウド で実行するためにアプリケーションをアップロードして実行すると、アプリケーションは Docker コンテナ内で実行されます。アプリケーションはホストオペレーティングシステムに直接アクセスできません。

SimSpace Weaver アプリケーションの一般的な流れ

1. アプリケーションを作成します。
2. ループ:
   a. `Transaction` を作成して更新を開始します。
      - シミュレーションが停止する場合はループを終了します。
   b. サブスクリプションと所有エンティティイベントを処理します。
   c. シミュレーションを更新します。
   d. `Transaction` をコミットして更新を終了します。
3. アプリケーションを破棄します。

空間アプリケーション

各空間アプリケーションには、シミュレーション世界の空間領域である所有権エリアがあります。空間アプリケーションの所有権エリアにあるエンティティは、アプリケーションに割り当てられたパーティションに保存されます。1 つの空間アプリケーションが、割り当てられたパーティション内のすべてのエンティティに対して完全な所有権 (読み取り権と書き込み権限) を持ちます。他のアプリケーションはそれらのエンティティに書き込むことはできません。空間アプリケーションはエンティティの状態を進めます。各空間アプリケーションはパーティションを 1 つだけ所有します。SimSpace Weaver はエンティティの空間位置を使用してインデックスを作成し、空間アプリケーションパーティションに割り当てます。

SimSpace Weaver アプリケーション SDK はサンプルアプリケーションを提供します。サンプルアプリケーションの空間アプリケーションのソースコードは以下のフォルダで確認できます。

Docker

```
project-folder\src\PathfindingSample\SpatialApp
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
project-folder/src/PathfindingSample/SpatialApp
```

カスタムアプリケーション

シミュレーションを操作するカスタムアプリケーションを作成して使用します。

カスタムアプリケーションは以下を実行できます

- エンティティを作成する
- 他のパーティションをサブスクライブする

- 変更をコミットする

カスタムアプリケーションの一般的なフロー

1. アプリケーションを作成します。
2. シミュレーション内の特定のリージョンをサブスクライブします。
    a. `Transaction` を作成して最初の更新を開始します。
    b. 特定のリージョンのサブスクリプションを作成します。
    c. `Transaction` をコミットして最初の更新を終了します。
3. ループ:
    a. `Transaction` を作成して更新を開始します。
        - シミュレーションが停止する場合はループを終了します。
    b. 状態の変更を処理します。
    c. `Transaction` をコミットして更新を終了します。
4. アプリケーションを破棄します。

カスタムアプリケーションでエンティティを作成したら、そのエンティティを空間ドメインに転送して、そのエンティティをシミュレーション内に空間的に存在させる必要があります。SimSpace Weaver はエンティティの空間位置を使用して、そのエンティティを適切な空間アプリケーションパーティションに配置します。エンティティを作成したカスタムアプリケーションは、空間ドメインに転送した後でそのエンティティを更新または削除することはできません。

SimSpace Weaver アプリケーション SDK はサンプルアプリケーションを提供します。サンプルアプリケーションに含まれるカスタムアプリケーションを、独自のカスタムアプリケーションのモデルとして使用できます。サンプルアプリケーションのビューアプリケーション (カスタムアプリケーション) のソースコードは、以下のフォルダで確認できます。

Docker

```
project-folder\src\PathfindingSample\ViewApp
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
project-folder/src/PathfindingSample/ViewApp
```

クライアントアプリケーションの開発

クライアントをシミュレーションに接続すべき理由は、以下のとおりです。

- 都市規模のシミュレーションにリアルタイムの交通情報を注入します。
- human-in-the-loop シミュレーションを作成して、人間のオペレーターがシミュレーションの一部を制御します。
- トレーニングシミュレーションなどで、ユーザーがシミュレーションを操作できるようにします。

これらの例のカスタムアプリケーションが、シミュレーションの状態と外部とのインターフェースとして機能します。クライアントはカスタムアプリケーションに接続してシミュレーションを操作します。

SimSpace Weaver はクライアントアプリケーションの通信、およびクライアントアプリケーションとカスタムアプリケーションとの通信は処理しません。クライアントアプリケーションの設計、作成、運用、セキュリティ、およびクライアントアプリケーションとカスタムアプリケーションとの通信については、お客様の責任となります。SimSpace Weaver はクライアントが接続できるように、各カスタムアプリケーションの IP アドレスとポート番号のみを公開します。

SimSpace Weaver アプリケーション SDK はサンプルアプリケーション用のクライアントを提供します。これらのクライアントは、独自のクライアントアプリケーションのモデルとして使用できます。サンプルアプリケーションクライアントのソースコードは、以下のフォルダで確認できます。

Docker

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

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

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

サンプルアプリケーションクライアントの構築と使用について詳しくは、本ガイドの「クイックスタートチュートリアル」の「ステップ 5: シミュレーションを表示する」を参照してください。

ローカル開発

SimSpace Weaver アプリケーションをローカルにデプロイして、迅速なテストとデバッグを行うことができます。SimSpace Weaver Local は Microsoft Windows での構築のみでサポートされています。

> **Important**
>
> Unity および Unreal Engine での開発については、「他のエンジンのサポート」を参照してください。

> **Important**
>
> バージョン 1.15.3 で C++、Python、Unity、または Unreal Engine を使用している場合は、「バージョン 1.15.3 におけるローカル開発の相違点」を参照してください

要件

- Microsoft Windows 10 以上
- Microsoft Visual Studio 2019 またはそれ以降、Desktop development with C++ ワークロードをインストールした状態

トピック

SimSpace Weaver Local のシミュレーションを構築する

SimSpace Weaver Local の使用方法を知るには、SimSpace Weaver の開始方法 チュートリアル中にクラウドで実行したのと同じ Pathfinding Sample アプリケーションを使用しますが、今回はローカルのハードウェアで実行します。

SimSpace Weaver Local のサンプルアプリケーションを構築する方法

1. コマンドプロンプトで、*project-folder*\tools\local に進みます。
2. generate_visual_studio_project.bat を実行します。
3. Visual Studio で *project-folder*\buildlocal\PathfindingSampleLocal.sln を開きます。
4. ビルド設定を RelWithDebInfo に設定します。
5. [Build]、[Build Solution] の順に選択します。

   Visual Studio はビルドアーティファクトを以下の場所に配置します。

   - *project-folder*\buildlocal\out\RelWithDebInfo.

   そのフォルダ内に、以下の実行ファイルが表示されます。

   - PathfindingSampleLocalSpatial.exe
   - PathfindingSampleLocalView.exe

SimSpace Weaver Local でシミュレーションを実行する

SimSpace Weaver Local を使用すると、最大 24 個の空間アプリケーションまたはカスタムアプリケーションを任意に組み合わせてローカルコンピューター上で実行できます。シミュレーションクロックは、スキーマで定義されているすべての空間アプリケーションが起動した後に開始されます。

SimSpace Weaver Local でアプリケーションを実行する方法

1. ファイル選択ウィンドウで、*project-folder*\buildlocal\out\RelWithDebInfo に進みます。
2. SimSpace Weaver Local アプリケーションには、アプリケーションの作業ディレクトリにある schema.yaml と名付けられたスキーマファイルが必要です。スキーマから必要な情報を読み取れなかった場合、アプリケーションは終了します。

   SimSpace Weaver Local のスキーマは *project-folder*\tools*project-name*-schema.yaml と同じである必要はありませんが、これを起点として使用できます。

   以下のうちのひとつを選択します。

   - そのスキーマを *project-folder*\buildlocal\out\RelWithDebInfo\schema.yaml にコピーします。
   - 環境変数 WEAVERLOCAL_SCHEMA_PATH を、別のパスまたはファイル名のスキーマファイルの名前に設定します。

     Example 例

     ```
     set WEAVERLOCAL_SCHEMA_PATH=c:\projects\MyProject\tools\MyProject-schema.yaml
     ```

     > **Note**
     >
     > 環境変数をコマンドラインから設定した場合、その環境変数 (その値を含む) には、そのコマンドプロンプトセッション (コンソールウィンドウ) からのみアクセスできます。

3. サンプルアプリケーションのスキーマでは、4 つのパーティションを作成する 2x2 グリッドが定義されています。スキーマで指定されている空間アプリケーションの数と一致するように、空間アプリケーションの 4 つのインスタンスを起動するスクリプトを実行します。また、このスクリプトは 1 つのビューアプリケーションを起動します。すべての空間アプリケーションが起動し、パーティションが割り当てられると、シミュレーションは自動的にティックを開始します。

アプリケーションを起動する方法

a. コマンドプロンプトで、プロジェクトのローカルツールフォルダーに移動します。

```
cd project-folder\tools\local
```

b. スクリプトを実行してアプリケーションを起動します。

```
launch_simulation_locally.bat
```

> **Note**
>
> スキーマファイルの名前に WEAVERLOCAL_SCHEMA_PATH を設定した場合、環境変数を設定したのと同じセッション (ウィンドウ) のコマンドラインで空間アプリケーションを起動する必要があります。

> **Important**
>
> Windows セキュリティポップアップが表示された場合、ビューアプリケーションに接続してシミュレーションを視覚化できるように Allow Access を選択します。

> **Note**
>
> 空間アプリケーションとビューアプリケーションを手動で起動することもできます。そのためには、空間アプリケーションの 4 つのインスタンスと 1 つのビューアプリケーションを手動で起動する必要があります。
>
> - 空間アプリケーション: `start PathfindingSampleLocalSpatial.exe`
> - ビューアプリケーション: `start PathfindingSampleLocalView.exe`

ローカルシミュレーションを表示する

ローカルシミュレーションを表示するには、SimSpaceWeaverAppSDKDistributable に含まれている任意のクライアントを使用できます。サンプルクライアントの構築と使用の詳細については、「ク

イックスタートチュートリアル」の「ステップ 5: シミュレーションを表示する」を参照してください。

ローカルシミュレーションのビューアプリケーションに接続するには、クライアントの IP アドレスとポート番号を更新する必要があります。SimSpace Weaver Local では常に以下の値を使用します。

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

選択したクライアントによっては、IP アドレスとポート番号を以下のように更新できます。

- Unreal — `view_app_url.txt` の 1 行目の URL を変更します
- コンソール — IP アドレスとポート番号 URL をパラメータとしてクライアントを起動します

ローカルシミュレーションを停止する

ローカル空間アプリケーションが有効の場合、ローカルシミュレーションは引き続き実行されます。空間アプリケーションウィンドウの 1 つを閉じると、シミュレーション全体が停止します。他のウィンドウをすべて閉じて、残りのシミュレーションをクリーンアップします。

各アプリケーションウィンドウを手動で閉じることも、以下のスクリプトを使用してすべてのアプリケーションウィンドウを自動的に閉じることもできます。

- *`project-folder`*`\tools\local\terminate_local_simulation.bat`

> **Note**
>
> 1 つの空間アプリケーションウィンドウを閉じるとシミュレーションは停止しますが、他のアプリケーションウィンドウは必ず閉じてください。前のシミュレーションでまだウィンドウが開いていると、別のローカルシミュレーションを正常に起動できません。

ローカルシミュレーションのデバッグ

Microsoft Visual Studio で SimSpace Weaver Local アプリケーションをデバッグできます。Visual Studio でデバッグする方法の詳細については、「Microsoft Visual Studio documentation」を参照してください。

ローカルシミュレーションをデバッグする方法

1. `schema.yaml` が作業ディレクトリにあることを確認します。
2. Visual Studio で、デバッグする各アプリケーションのコンテキストメニュー (`PathfindingSampleLocalSpatial` または `PathfindingSampleLocalView` など) を開き、デバッグセクションで作業ディレクトリを設定します。
3. デバッグするアプリケーションのコンテキストメニューを開き、[スタートアッププロジェクトとして設定] を選択します。
4. F5 を選択して、アプリケーションのデバッグを開始します。

シミュレーションをデバッグするための要件は、シミュレーションを正常に実行するための要件と同じです。スキーマで指定された数の空間アプリケーションを起動する必要があります。例えば、スキーマで 2x2 グリッドが指定されている場合に、空間アプリケーションをデバッグモードで起動した場合、さらに 3 つの空間アプリケーションを (デバッグモードまたはデバッグモードでない状態で) 起動するまでシミュレーションは実行されません。

カスタムアプリケーションをデバッグするには、まず空間アプリケーションを起動し、次にデバッガーでカスタムアプリケーションを起動する必要があります。

シミュレーションはロックステップで実行されることに注意してください。アプリケーションがブレークポイントに達するとすぐに、他のすべてのアプリケーションは一時停止します。そのブレークポイントから続行すると、他のアプリケーションは続行されます。

バージョン 1.15.3 におけるローカル開発の相違点

このセクションでは、バージョン 1.15.3 以降の SimSpace Weaver Local を使用した開発における変更点について説明します。これらの変更は、C++、Python、Unity、および Unreal Engine で SimSpace Weaver Local プロジェクトのワークフローに影響します。

トピック

ファイルの変更点

- SimSpaceWeaverAppSdkLocal ライブラリファイル
  は、weaver_app_sdk_cxx_v1_full_local という名前になりました。
  - これらのファイルは *sdk-folder*\SimSpaceWeaverAppSdk-1.15.3\lib\weaverlocal\windows にあります。
- プロジェクトで SimSpaceWeaverAppSDK-1.15.1\include\aws\weaverruntime\local_ffi にリンクしたり、含めたりしないでください。
- SimSpace Weaver Local の Python スクリプトでは cmake は不要になりました。
- SimSpace Weaver Local の新しい Python スクリプトや、名前が変更された Python スクリプトはありません。
  - build-local – 起動用の Python アプリケーションをビルドします。Python コードと SimSpace Weaver Local ライブラリファイルはすべて buildlocal ディレクトリに配置されます。
  - local-config – ローカルスクリプトの環境変数を定義します。
  - start-python-locally – アプリケーションの PythonPath を設定し、Python を起動します。
  - 以下のスクリプトは、似た名前のクラウドスクリプトと同等です。
    - quick-start-local
    - start-simulation-local
    - stop-simulation-local

既存の C++ プロジェクトを SimSpace Weaver Local 1.15.3 に更新する

要件

- バージョン 1.15.3 SimSpace Weaver アプリケーション SDK
- SimSpace Weaver Local バージョン 1.15.2 以前用の既存の C++ プロジェクト

プロジェクトを更新するには

1. プロジェクトに必要なヘッダーが含まれていることを確認します。

```
#include <aws/weaverruntime/ffi/weaver_app_sdk_cxx_ffi_v1/src/lib.rs.h>
```

> **Note**
>
> 通常、これは通常 aws\weaverruntime\detail.h に含まれています。

2. プロジェクトに以下の廃止予定のヘッダーが含まれていないことを確認します。

```
#include <aws/weaverruntime/local_ffi/Bridge.h>
```

> **Note**
>
> 通常、これは通常 aws\weaverruntime\detail.h に含まれています。

3. CMake ファイルまたはビルドスクリプトで、SimSpaceWeaverAppSdkLocal 静的ライブラリ名を新しい weaver_app_sdk_cxx_v1_full_local の名前に置き換えます。
4. バージョン 1.15.3 から *sdk-folder*\docker-create-image.bat を実行していない場合は、今すぐ実行します。これは一度だけ行えばよいだけです。
5. 通常の手順に従って、プロジェクトをビルドし、実行します。

トラブルシューティング

リンカーエラー (未解決の外部シンボル) が発生する

コンパイラが次のようなリンカーエラーを出力します (読みやすいように改行が追加されています)。

```
Error    LNK2019    unresolved external symbol
  "class outcome_v2_92ee5284::basic_result<class
  Aws::WeaverRuntime::Application,
  enum Aws::WeaverRuntime::ffi::weaver_app_sdk_cxx_ffi_v1::ErrorCode
```

この問題を解決するには

- プロジェクトに <aws/weaverruntime/local_ffi/Bridge.h> が含まれていないことを確認します。

型の非互換性エラーが発生する

次のようなエラーが発生する場合があります (読みやすいように改行が追加されています)。

```
the object has type qualifiers that are not compatible with the member function
  object type is: const rust::cxxbridge1::String
```

この問題を解決するには

1. SimSpace Weaver ヘッダーファイルを 1.15.3 バージョンで更新します。
2. `rust::cxxbridge1::String` の使用が次のようになっていることを確認します。

   ```
   rust::cxxbridge1::String domain_name = domain.name.value;
   if (domain.type_ == Api::DomainType::Spatial && Name.Compare(domain_name.c_str())
    == 0)
   ```

SimSpace Weaver Local 1.15.3 で新しい Python プロジェクトを実行する

要件

- バージョン 1.15.3 SimSpace Weaver アプリケーション SDK
- 新しい SimSpace Weaver Local バージョン 1.15.3 の `PythonBubblesSample` プロジェクト

プロジェクトを実行するには

1. コマンドプロンプトウィンドウで、*`project-folder`*`\tools\local\windows` に移動します。
2. `quick-start-local.bat` を実行する

   4 つの空間アプリケーション、1 つの表示アプリケーション、および 1 つのクライアントがローカルに起動されます。クライアントにはバブルが表示されます。

トラブルシューティング

`GLIBCXX` が見つからないことを示すエラーが表示されます。

次のエラーは、SimSpace Weaver Python アプリケーション SDK のインポートに失敗したことを示しています。原因としては、C++ ライブラリが古くなっていることが考えられます。

```
ImportError: /lib64/libstdc++.so.6: version `GLIBCXX_3.4.29' not found
```

この問題を解決するには

1. libstdc++ ライブラリのソース (gcc や msvc など) を、指定のバージョン GLIBCXX (この例では 3.4.29) を含むバージョンに更新します。
2. LD_LIBRARY_PATH システム環境変数が、lib64 への正しいパスに設定されていることを確認します。

クイックスタートスクリプトが失敗する

クイックスタートスクリプトが失敗し、次のようなメッセージが表示されます。

```
python: can't open file 'C:\usr\project\buildlocal\bIn\bubbles_tkinter_client.py':
  [Errno 2] No such file or directory
```

この問題を解決するには

1. *project-folder*\tools\local\windows に移動し、local-config.bat を編集します。
2. 以下の環境変数がローカルシステムの正しいパスに設定されていることを確認します。
   - TOOLS_DIR が *project-folder*\tools に設定されている
   - TOOLS_DIR_WINDOWS が *project-folder*\tools\local\windows に設定されている
   - PROJECT_ROOT が *project-folder* に設定されている
   - BUILD_DIR が *project-folder*\buildlocal に設定されている
   - APP_SDK_DIR が *sdk-folder* に設定されている

既存の Unity プロジェクトを SimSpace Weaver Local 1.15.3 に更新する

要件

- バージョン 1.15.3 SimSpace Weaver アプリケーション SDK
- SimSpace Weaver Local バージョン 1.15.2 以前用の既存の Unity プロジェクト

プロジェクトを更新するには

1. Unity で、既存の AWS SimSpace Weaver パッケージを削除します。
   a. 既存の Unity プロジェクトを開きます。

b. エディターウィンドウで、[ウィンドウ] > [パッケージマネージャー] を選択します。

c. [パッケージ - Unity テクノロジー] で [AWS SimSpace Weaver] を選択し、[削除] を選択します。

2. バージョン 1.15.3 *sdk-folder* で、`download-unity-package.bat` を実行します。

3. `Unity_SDK_for_AWS_SimSpace_Weaver.pdf` の手順に従って、新しくダウンロードした `SimSpaceWeaverUnityPackage.zip` を、AWS SimSpace Weaver パッケージとして Unity エディターに追加します。

既存の Unreal Engine プロジェクトを SimSpace Weaver Local 1.15.3 に更新する

要件

- バージョン 1.15.3 SimSpace Weaver アプリケーション SDK
- SimSpace Weaver Local バージョン 1.15.2 以前用の既存の Unreal Engine プロジェクト

プロジェクトを更新するには

1. Unreal プロジェクトとコードエディタのウィンドウをすべて閉じます。
2. コマンドプロンプトウィンドウで、バージョン 1.15.3 *sdk-folder* に移動します。
3. `update-unreal-project.bat --path` *project-folder* `--name` *project-name* を実行します。

> **Note**
>
> これにより、既存のプラグインが新しいプラグインに置き換わります。変更はすべて消去されます。

4. `AWS_SimSpace_Weaver_Unreal_Guide.pdf` の手順に従って、SimSpace Weaver Local のプロジェクトをビルドします。

トラブルシューティング

既存のプラグインファイルを削除できない

次のようなエラーが表示される場合があります。

```
cannot remove '/usr/src/project/{PROJECT_NAME}/src/PathfindingSampleUnrealSpatial/
PathfindingUnrealProject/Plugins/SimSpaceWeaverAppSdkPlugin/Binaries/Win64/
UnrealEditor-WeaverAppSdk.dll': Operation not permitted
cannot remove '/usr/src/project/{PROJECT_NAME}/src/PathfindingSampleUnrealSpatial/
PathfindingUnrealProject/Plugins/SimSpaceWeaverAppSdkPlugin/Binaries/Win64/
UnrealEditor-WeaverAppSdkLocal.dll': Operation not permitted
cannot remove '/usr/src/project/{PROJECT_NAME}/src/PathfindingSampleUnrealSpatial/
PathfindingUnrealProject/Plugins/SimSpaceWeaverAppSdkPlugin/Binaries/Win64/
UnrealEditor-WeaverCppMetrics.dll': Operation not permitted
```

この問題を解決するには

1. Unreal プロジェクトエディタウィンドウとコードエディタが閉じていることを確認します。
2. `update-unreal-project.bat` を実行します。

バージョン SimSpace Weaver Local 1.15.3 に関するよくある質問

- Q1: Python プロジェクトの SimSpace Weaver Local 環境変数を変更する方法を教えてください。
  - Python プロジェクトの*`project-folder`*`\tools\windows\local-config` を編集します。

AWS SimSpace Weaver アプリケーション SDK

SimSpace Weaver アプリケーション SDK には、シミュレーション内のエンティティを制御したり、SimSpace Weaver イベントに応答したりするための API が用意されています。これには以下の名前空間が含まれます。

- API — API のコア定義とその使用方法

以下のライブラリとリンクします。

- `libweaver_app_sdk_cxx_v1_full.so`

> **Important**
>
> ライブラリは、AWS クラウド でアプリケーションを実行するとダイナミックリンクに使用できます。アプリケーションと一緒にアップロードする必要はありません。

> **Note**
>
> SimSpace Weaver アプリケーション SDK API はシミュレーション内のデータを制御します。AWS では、これらの API は SimSpace Weaver サービスリソース (シミュレーション、アプリケーション、クロックなど) を制御する SimSpace Weaver サービス API とは別のものです。詳細については、「SimSpace Weaver API リファレンス」を参照してください。

トピック

API メソッドは Result を返します。

SimSpace Weaver API 関数の大部分の戻り値タイプは `Aws::WeaverRuntime::Result<T>` です。関数が正常に実行されると、`Result` には `T` が含まれます。それ以外の場合、`Result` には Rust App SDK からのエラーコードを表す `Aws::WeaverRuntime::ErrorCode` が含まれます。

Example 例

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

この方法:

- `BeginUpdate()` が正常に実行された場合は `Transaction` を返します。
- `BeginUpdate()` に失敗した場合は `Aws::WeaverRuntime::ErrorCode` を返します。

最上位レベルでのアプリケーション SDK とのやり取り

ライフサイクル

- SimSpace Weaver アプリケーション SDK はアプリケーションのライフサイクルを管理します。アプリケーションのライフサイクル状態を読み取ったり書き込んだりする必要はありません。

パーティション

- `Result <PartitionSet> AssignedPartitions(Transaction& txn);` を使用して、所有しているパーティションを取得します。
- `Result <PartitionSet> AllPartitions(Transaction& txn);` を使用して、シミュレーション内のすべてのパーティションを取得します。

シミュレーション管理

このセクションでは、一般的なシミュレーション管理タスクのソリューションについて説明します。

トピック

- シミュレーションを開始する
- シミュレーションを更新する
- シミュレーションを終了する

シミュレーションを開始する

`CreateApplication()` を使用して、アプリケーションを作成します。

Example 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()));
```

シミュレーションを更新する

以下の BeginUpdate 関数を使用してアプリケーションを更新します。

- `Result<Transaction> BeginUpdate(Application& app)`
- `Result<bool> BeginUpdateWillBlock(Application& app)` — `BeginUpdate()` がブロックするかしないかを伝えます。

`Result<void> Commit(Transaction& txn)` を使用して、以下の変更をコミットします。

Example 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();
}
```

シミュレーションを終了する

Result<void> DestroyApplication(Application&& app) を使用して、アプリケーションとシミュレーションを終了します。

他のアプリケーションは、BeginUpdateWillBlock() または BeginUpdate() への呼び出しから ErrorCode::ShuttingDown を受信すると、シミュレーションがシャットダウン中であることを認識します。アプリケーションが ErrorCode::ShuttingDown を受信すると、Result<void> DestroyApplication(Application&& app) 呼び出しを行って自動的に終了できます。

Example 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**
>
> Api::Commit() の後にのみ Result<void> DestroyApplication(Application&& app) を呼び出しできます。更新中にアプリケーションを破棄すると、未定義の動作が発生する可能性があります。

> **Important**
>
> プログラムが終了する前に DestroyApplication() を呼び出して、アプリケーションの正常終了レポートを確認する必要があります。
> プログラムの終了時に DestroyApplication() 呼び出しに失敗すると、ステータスは FATAL とみなされます。

サブスクリプション

サブスクリプション領域とドメイン ID を使用してサブスクリプションを作成します。ドメイン ID は、そのサブスクリプション領域を所有するドメインを表します。BoundingBox2F32 はサブスクリプション領域を表します。以下の関数を使用してサブスクリプションを作成します。

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

Example 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();
}
```

CreateSubscriptionBoundingBox2F32() から返送された Api::SubscriptionHandle を使用してサブスクリプションを変更できます。以下の関数の引数として渡します。

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

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

エンティティ

StoreとLoad API を呼び出すには、CreateEntity()、またはエンティティがアプリケーションのサブスクリプション領域に入った際の所有権変更イベントから返された、Result<Api::Entity> の Api:Entity を使用します (詳細については、「エンティティイベント」を参照してください)。これらの API で使用できるように、Api::Entity オブジェクトを追跡することをお勧めします。

トピック

- エンティティを作成する
- エンティティを空間ドメインに転送します。

- エンティティフィールドデータの書き込みおよび読み取り
- エンティティの位置を保存する
- エンティティの位置をロードする

エンティティを作成する

CreateEntity() を使用して、エンティティを作成します。この関数に渡す、Api::TypeId の意味を定義します。

```
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**
>
> 0 ~ 511 の Api::BuiltinTypeId の値は予約されています。エンティティ TypeID (この例では k_entityTypeId) の値は 512 以上である必要があります。

エンティティを空間ドメインに転送します。

カスタムアプリケーションまたはサービスアプリケーションでエンティティを作成したら、そのエンティティを空間ドメインに転送して、そのエンティティをシミュレーションに空間的に存在させる必要があります。空間ドメイン内のエンティティは、他のアプリケーションで読み取ったり、空間アプリケーションで更新したりできます。ModifyEntityDomain() API を使用してエンティティを空間ドメインに転送します。

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

DomainId が呼び出し元のアプリケーションに割り当てられた Partition と一致しない場合、DomainId は DomainType::Spatial Domain のものである必要があります。新しい Domain への所有権の転送は、Commit(Transaction&&) の間に行われます。

パラメータ

txn

現在の Transaction。

entity

Domain の変更ターゲット Entity。

domainId

Entity の送信先 Domain の DomainId。

この API は、エンティティドメインが正常に変更された場合に Success を返します。

エンティティフィールドデータの書き込みおよび読み取り

エンティティデータフィールドはすべて BLOB タイプです。1 つのエンティティには最大 1,024 バイトのデータを書き込むことができます。BLOB のサイズを大きくするとパフォーマンスが低下するため、BLOB はできるだけ小さくすることをお勧めします。BLOB に書き込むときは、SimSpace Weaver にデータへのポインタと長さを渡します。BLOB から読み取ると、SimSpace Weaver は読み取るポインタと長さを提供します。すべての読み取りは、アプリケーションが Commit() を呼び出す前に完了する必要があります。読み取りの呼び出しから返されたポインタは、アプリケーションが Commit() を呼び出すと無効になります。

> **Important**
>
> - Commit() の後のキャッシュされた BLOB ポインタからの読み取りはサポートされないため、シミュレーションが失敗する可能性があります。
> - 読み取りの呼び出しから返された BLOB ポインタへの書き込みはサポートされないため、シミュレーションが失敗する可能性があります。

トピック

- エンティティのフィールドデータを保存する

- エンティティのフィールドデータをロードする
- 削除されたエンティティのフィールドデータの読み込み

エンティティのフィールドデータを保存する

以下の例は、アプリケーションが所有するエンティティのフィールドデータを保存する (ステートファブリックに書き込む) 方法を示しています。以下の例では、以下の関数を使用します。

```
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` パラメータは渡されたデータのデータタイプを表します。

`Api::TypeId keyTypeId` パラメータは `Api::BuiltinTypeId` から対応する `Api::TypeId` を受け取る必要があります。適切な変換がない場合は、`Api::BuiltinTypeId::Dynamic` を使用できます。

複雑なデータタイプの場合は、`Api::BuiltInTypeId::Dynamic` を使用します。

> **Note**
>
> `FieldIndex index` の値はゼロより大きい必要があります。値 0 はインデックスキー専用です (`StoreEntityIndexKey()` を参照してください)。

Example プリミティブデータタイプを使用する例

```
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 struct を使用してデータを保持する例

```
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));
}
```

エンティティのフィールドデータをロードする

以下の例は、エンティティのフィールドデータをロードする (ステートファブリックから読み取る) 方法を示しています。以下の例では、以下の関数を使用します。

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

`Api::TypeId keyTypeId` パラメータは `Api::BuiltinTypeId` から対応する `Api::TypeId` を受け取る必要があります。適切な変換がない場合は、`Api::BuiltinTypeId::Dynamic` を使用できます。

> **Note**
>
> `FieldIndex` インデックスの値は 0 より大きい必要があります。値 0 はインデックスキー専用です (`StoreEntityIndexKey()` を参照してください)。

Example プリミティブデータタイプを使用する例

```
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 struct を使用してデータを保持する例

```
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);
}
```

削除されたエンティティのフィールドデータの読み込み

アプリケーションの所有権とサブスクリプション領域から削除されたエンティティのエンティティフィールドデータをロードする (ステートファブリックから読み取る) ことはできま

せん。以下の例では、Api::ChangeListAction::Remove の結果としてエンティティの Api::LoadIndexKey() を呼び出しているため、エラーが発生します。2 番目の例は、エンティティデータをアプリケーションに直接保存して読み込む正しい方法を示しています。

Example 誤ったコードの例

```
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 アプリケーションにエンティティデータを保存して読み込む正しい方法の例

```
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;
            }
        }
    }

    /* ... */
}
```

エンティティの位置を保存する

整数データ構造を使用してエンティティの位置を保存 (ステートファブリックへの書き込み) できます。以下の例では、以下の関数を使用します。

```
Result<void> StoreEntityIndexKey(
    Transaction& txn,
    const Entity& entity,
```

```
    TypeId keyTypeId,
    std::int8_t* src,
    std::size_t length)
```

> **Note**
>
> 以下の例に示すように、Api::BuiltinTypeId::Vector3F32 から Api::StoreEntityIndexKey() を指定する必要があります。

Example 配列を使用して位置を表す例

```
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 struct を使用して位置を表す例

```
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));
}
```

エンティティの位置をロードする

整数データ構造を使用してエンティティの位置をロードする (ステートファブリックから読み取る) ことができます。以下の例では、以下の関数を使用します。

> **Note**
>
> 以下の例に示すように、`Api::BuiltinTypeId::Vector3F32` から `Api::LoadEntityIndexKey()` を指定する必要があります。

Example 配列を使用して位置を表す例

```
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 struct を使用して位置を表す例

```
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);
}
```

エンティティイベント

SimSpace Weaverアプリケーション SDK で次の関数を使用して、すべての所有権およびサブスクリプションイベントを取得できます。

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

コールバック駆動型のエンティティイベント処理が必要な場合は、SimSpace Weaver デモフレームワークを使用できます。詳細については、以下のヘッダーファイルを参照してください。

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

独自のエンティティイベント処理を作成することもできます。

トピック

所有エンティティのイベントを繰り返し処理します。

OwnershipChanges( ) を使用して、所有エンティティ (アプリケーションの所有領域にあるエンティティ) のイベントのリストを取得します。関数には以下のような署名があります。

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

次に、以下の例に示すように、ループを使用してエンティティを繰り返し処理します。

Example 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;
```

```
    }
}
```

イベントタイプ

- None — エンティティが領域内にあり、その位置とフィールドのデータは変更されていません。
- Remove — エンティティが領域から削除されました。
- Add — エンティティが領域に追加されました。
- Update — エンティティが領域内にあり、変更されました。
- Reject — アプリケーションは領域からエンティティを削除できませんでした。

> **Note**
>
> Reject イベントが発生した場合、アプリケーションは次のティックで転送を再試行します。

登録したエンティティのイベントを繰り返し処理します。

AllSubscriptionEvents() を使用して、登録済みエンティティ (アプリケーションのサブスクリプション領域にあるエンティティ) のイベントリストを取得します。関数には以下のような署名があります。

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

次に、以下の例に示すように、ループを使用してエンティティを繰り返し処理します。

Example 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;
    }
}
```

イベントタイプ

- None — エンティティが領域内にあり、その位置とフィールドのデータは変更されていません。
- Remove — エンティティが領域から削除されました。
- Add — エンティティが領域に追加されました。
- Update — エンティティが領域内にあり、変更されました。
- Reject — アプリケーションは領域からエンティティを削除できませんでした。

> **Note**
>
> Reject イベントが発生した場合、アプリケーションは次のティックで転送を再試行します。

エンティティの所有権変更イベントを繰り返し処理します。

エンティティが所有領域とサブスクリプション領域の間を移動するイベントを取得するには、現在と以前のエンティティの所有領域とサブスクリプションイベントの変化を比較します。

これらのイベントは、以下を読むことで処理できます。

- Api::SubscriptionChangeList
- Api::OwnershipEvents

その後、変更内容を以前に保存したデータと比較できます。

以下の例は、エンティティの所有権の変更イベントを処理する方法を示しています。この例では、サブスクライブされたエンティティと所有されたエンティティの間を (どちらの方向でも) 移行するエンティティについて、所有権の削除/追加イベントが最初に発生し、次のティックでサブスクリプションの削除/追加イベントが発生することを前提としています。

Example 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 とエラー処理

この Aws::WeaverRuntime::Result<T> クラスはサードパーティの Outcome ライブラリを使用しています。以下のパターンを使用して、Result を確認し、API コールによって返されるエラーを発見できます。

```
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 制御文マクロ

戻り値タイプ Aws::WeaverRuntime::Result<T> を持つ関数内では、前のコードパターンの代わりに WEAVERRUNTIME_TRY マクロを使用できます。マクロは渡された関数を実行します。渡された関数が失敗すると、マクロは囲んでいる関数にエラーを返送させます。渡された関数が成功すると、

実行は次の行に進みます。以下の例は、以前の DoBeginUpdate() 関数の再記述を示しています。このバージョンでは、if-else 制御構造の代わりに WEAVERRUNTIME_TRY マクロを使用しています。関数の戻り値タイプは Aws::WeaverRuntime::Result<void> です。

```
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();
}
```

BeginUpdate() に失敗した場合、マクロは障害発生時に早くDoBeginUpdate() を返送します。WEAVERRUNTIME_EXPECT_ERROR マクロを使用して BeginUpdate() から Aws::WeaverRuntime::ErrorCode を取得できます。以下の例は、障害発生時に Update() 関数が DoBeginUpdate() を呼び出してエラーコードを取得する方法を示しています。

```
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);

    }
}
```

戻り値タイプを Update() から Aws::WeaverRuntime::Result<void> に変更することで、BeginUpdate() からのエラーコードを Update() を呼び出す関数に利用できるようにすることができます。この手順を繰り返して、エラーコードを呼び出しスタックのさらに下位に送り続けることができます。

ジェネリックとドメインタイプ

SimSpace Weaver アプリケーション SDK には、単精度データタイプ Api::Vector2F32 および Api::BoundingBox2F32、倍精度データタイプ Api::Vector2F64 および Api::BoundingBox2F64 が用意されています。これらのデータタイプは受動的なデータ構造で、便利な方法はありません。API は Api::Vector2F32 および Api::BoundingBox2F32 のみを使用することに注意してください。これらのデータタイプを使用してサブスクリプションを作成および変更できます。

SimSpace Weaver デモフレームワークでは、Vector3 および Aabb を含む AzCore 数学ライブラリの最小バージョンが提供されています。詳細については、以下のヘッダーファイルを参照してください。

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

その他のアプリケーション SDK 操作

トピック

- AllSubscriptionEvents および OwnershipChanges は前回の呼び出しからのイベントを含みます
- SubscriptionChangeList の処理後に読み取りロックを解除する
- テスト用のスタンドアロンアプリケーションインスタンスを作成する

AllSubscriptionEvents および OwnershipChanges は前回の呼び出しからのイベントを含みます

Api::AllSubscriptionEvents() および Api::OwnershipChanges() への呼び出しの戻り値が含むのは、最後の呼び出しからのイベントであり、最後のティックからのイベントではありませ

ん。以下の例では、secondSubscriptionEvents および secondOwnershipChangeList は空です。なぜなら、関数は最初の呼び出しの直後に呼び出されるからです。

10 ティック待ってから Api::AllSubscriptionEvents() および Api::OwnershipChanges() を呼び出した場合、その結果として (最後のティックではなく) 最後の 10 ティックからのイベントと変更の両方が含まれます。

Example 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**
>
> 関数 AllSubscriptionEvents() は実装されていますが、関数 SubscriptionEvents() は実装されていません。

SubscriptionChangeList の処理後に読み取りロックを解除する

更新を開始すると、コミットされたデータ用の共有メモリセグメントが他のパーティションに前回のティックで残ります。これらの共有メモリセグメントはリーダーによってロックされている可能性が

あります。すべてのリーダーがロックを解除するまで、アプリケーションは完全にコミットできません。最適化のため、アプリケーションは Api::SubscriptionChangelist アイテムを処理した後にロックを解除するための Api::ReleaseReadLeases() 呼び出しを行う必要があります。これにより、コミット時の競合が減少します。デフォルトでは Api::Commit() は読み取りリースを解放しますが、サブスクリプションの更新を処理した後に手動でリリースするのがベストプラクティスです。

Example 例

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

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

    WEAVERRUNTIME_EXPECT(Api::ReleaseReadLeases(transaction));

    ...
}
```

テスト用のスタンドアロンアプリケーションインスタンスを作成する

Api::CreateStandaloneApplication()を使用して、スタンドアロンアプリケーションを作成し、実際のシミュレーションでコードを実行する前にアプリケーションのロジックをテストできます。

Example 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 デモフレームワーク

AWS SimSpace Weaver デモフレームワーク (デモフレームワーク) は、SimSpace Weaver アプリケーションの開発に使用できるユーティリティのライブラリです。

デモフレームワークには以下のものが含まれます

- 使用したり調べたりできるコードサンプルとプログラミングパターン
- シンプルなアプリケーションの開発を効率化する抽象化とユーティリティ関数
- SimSpace Weaver アプリケーション SDK の実験的機能をテストする簡単な方法

より高いパフォーマンスを実現するために、SimSpace Weaver API への低レベルのアクセスが可能な SimSpace Weaver アプリケーション SDK を設計しました。これとは対照的に、より高いレベルの抽象化と、SimSpace Weaver を使いやすくする API へのアクセスが可能なデモフレームワークを設計しました。使いやすさの対価として、SimSpace Weaver アプリケーション SDK を直接使用する場合と比べてパフォーマンスが低くなります。パフォーマンスの低下を許容できるシミュレーション (リアルタイムのパフォーマンス要件がないシミュレーションなど) は、デモフレームワークの使用に適している場合があります。デモフレームワークは完全なツールキットではないため、複雑なアプリケーションに SimSpace Weaver アプリケーション SDK のネイティブ機能を使用することをお勧めします。

デモフレームワークには以下が含まれます

- 以下をサポートおよび実証する作業用コードサンプル:
  - アプリケーションフローの管理
  - コールバック主導型のエンティティイベント処理
- 以下のサードパーティー製ユーティリティライブラリのセット:
  - spdlog (ログ記録ライブラリ)
  - AZCore (数学ライブラリ) の最小バージョンで、以下のみを含むもの:
    - Vector3
    - Aabb
  - cxxopts (コマンドラインオプションパーサーライブラリ)
- SimSpace Weaver 固有のユーティリティ関数

デモフレームワークは、ライブラリ、ソースファイル、CMakeLists で構成されています。これらのファイルは、SimSpace Weaver アプリケーション SDK の配布可能パッケージに含まれています。

Service Quotas との連携

このセクションでは、SimSpace Weaver の Service Quotas の使用方法について説明します。Quotas はリミットとも呼ばれます。Service Quotas のリストについては、「SimSpace Weaver エンドポイントとクォータ」を参照してください。このセクションの API はアプリケーション API セットの一部です。アプリケーション API はサービス API とは異なります。アプリケーション API は SimSpace Weaver アプリケーション SDK の一部です。アプリケーション API のドキュメントは、ローカルシステムのアプリケーション SDK フォルダで確認できます。

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

トピック

- アプリケーションの制限を取得する
- アプリケーションが使用しているリソース量を取得する
- メトリクスをリセットする
- 制限の超過
- メモリの不足
- ベストプラクティス

アプリケーションの制限を取得する

RuntimeLimits アプリケーション SDK を使用してアプリケーションの制限をクエリできます。

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

パラメータ

Application& アプリケーション

アプリケーションへの参照。

LimitType タイプ

以下の制限タイプを含む列挙値。

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

以下の例では、エンティティ数の制限をクエリします。

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

アプリケーションが使用しているリソース量を取得する

RuntimeMetrics アプリケーション SDK を呼び出して、アプリケーションが使用しているリソースの量を取得できます。

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

パラメータ

Application& アプリケーション

アプリケーションへの参照。

API は、メトリクスを含む struct への参照を返します。カウンターメトリクスには現在の合計値が含まれ、増加のみします。ゲージメトリクスには増減する値が含まれます。アプリケーションランタイムは、イベントによって値が増加するたびにカウンターを更新します。ランタイムは API を呼び出したときのみゲージを更新します。SimSpace Weaver はリファレンスがアプリケーションの存続期間中有効であることを保証します。API を繰り返し呼び出しても、参照は変更されません。

```
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
}
```

メトリクスをリセットする

ResetRuntimeMetrics アプリケーション SDK は AppRuntimeMetrics struct 内の値をリセットします。

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

次の例では、アプリケーション内での ResetRuntimeMetrics の呼び出し方法を示します。

```
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));
}
```

制限の超過

アプリケーションの API コールが制限を超えると、エンティティ転送を除き、ErrorCode::CapacityExceeded が返されます。SimSpace Weaver はCommit 操作および BeginUpdate アプリケーション API 操作の一部としてエンティティ転送を非同期的に処理するため、エンティティ転送の上限が原因で転送が失敗した場合にエラーを返す特定の操作はありません。転送の失敗を検出するには、 rejected_incoming_transfer_counter および

`rejected_outgoing_transfer_counter` (AppRuntimeMetrics struct 内) の現在の値を以前の値と比較できます。拒否されたエンティティはパーティションには含まれませんが、アプリケーションではそのエンティティをシミュレートできます。

メモリの不足

SimSpace Weaver はガベージコレクタープロセスを使用して、解放されたメモリをクリーンアップして解放します。ガベージコレクターがメモリを解放するよりも速くデータを書き込むことができます。この場合、書き込み操作がアプリケーションの予約メモリの制限を超える可能性があります。SimSpace Weaver は `OutOfMemory` (および追加情報) を含むメッセージを含む内部エラーを返します。詳細については、「書き込みを時系列に分散させる」を参照してください。

ベストプラクティス

以下のベストプラクティスは、制限を超えないようにアプリケーションを設計するための一般的なガイドラインです。特定のアプリケーションデザインには当てはまらない場合があります。

頻繁に監視し、速度を低下させる

メトリクスを頻繁に監視し、制限に近づいている処理は遅くする必要があります。

サブスクリプション制限や転送制限の超過を避ける

可能であれば、リモートサブスクリプションとエンティティ転送の数を減らすようにシミュレーションを設計します。配置グループを使用して同じワーカーに複数のパーティションを配置し、ワーカー間のリモートエンティティ転送の必要性を減らすことができます。

書き込みを時系列に分散させる

ティック内の更新の数とサイズは、トランザクションのコミットに必要な時間とメモリに大きな影響を与える可能性があります。メモリ要件が大きいと、アプリケーションランタイムのメモリが不足する可能性があります。書き込みを時系列に分散させて、1 ティックあたりの更新の平均合計サイズを減らすことができます。これにより、パフォーマンスを向上させ、制限を超えないようにすることができます。各ティックに平均 12 MB、または各エンティティに平均 1.5 KB を超えて書き込まないことをお勧めします。

シミュレーションのデバッグ

以下の方法を使用して、シミュレーションに関する情報を取得できます。

トピック

SimSpace Weaver Local を使用してコンソール出力を確認する

まずシミュレーションをローカルで開発してから、AWS クラウド で実行することをお勧めします。SimSpace Weaver Local を実行するとコンソールの出力を直接表示できます。詳細については、「ローカル開発」を参照してください。

Amazon CloudWatch Logs でログを確認する

コンソールでシミュレーションを実行するAWS クラウドと、アプリケーションの出力が Amazon CloudWatch Logs のログストリームに送信されます。シミュレーションでは他のログデータも書き込まれます。シミュレーションでログデータを書き込むには、シミュレーションスキーマのログ記録を有効にする必要があります。詳細については、「SimSpace Weaver Amazon CloudWatch Logs のログ」を参照してください。

> **Warning**
>
> シミュレーションでは大量のログデータが生成される可能性があります。ログデータは非常に急速に増大する可能性があります。ログを注意深く観察し、実行する必要がなくなったらシミュレーションを停止する必要があります。ログ記録には多額のコストがかかる可能性があります。

describe API コールを使用する

以下のサービス API を使用して、AWS クラウド 内のシミュレーションに関する情報を取得できます。

> **Important**
>
> AWS IAM Identity Center または AWS Command Line Interface (AWS CLI) にプロファイルを使用したり名前を付けたりする場合は、SimSpace Weaver アプリケーション SDK バー

ジョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを `default` プロファイルにコピーするか、`--profile` *`cli-profile-name`* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザガイド」「設定と認証情報ファイルの設定」を参照してください。

- ListSimulations — AWS クラウド にあるすべてのシミュレーションのリストを取得します。

  Example 例

  ```
  tools\windows\weaver-MyProject-cli.bat list-simulations
  ```

- DescribeSimulation — シミュレーションの詳細を取得します。

  Example 例

  ```
  tools\windows\weaver-MyProject-cli.bat describe-simulation --simulation MySimulation
  ```

- DescribeApp — アプリケーションの詳細を取得します。

  Example 例

  ```
  tools\windows\weaver-MyProject-cli.bat describe-app --simulation MySimulation --
  domain MyCustomDomain --app MyCustomApp
  ```

SimSpace Weaver API の詳細については、SimSpace Weaver API リファレンス を参照してください。

クライアントを接続する

シミュレーションスキーマの `endpoint_config` で定義した実行中のカスタムアプリケーションまたはサービスアプリケーションにクライアントを接続できます。SimSpace Weaver アプリケーション SDK には、サンプルアプリケーションケーションを表示するために使用できるサンプルクライア

ントが含まれています。これらのサンプルクライアントのソースコードとサンプルアプリケーションケーションを見て、独自のクライアントを作成する方法を確認できます。サンプルクライアントの構築および実行方法の詳細については、「ステップ 5: シミュレーションを表示する」を参照してください。

サンプルクライアントのソースコードは、以下のフォルダで確認できます。

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

ローカルシミュレーションのデバッグ

Microsoft Visual Studio で SimSpace Weaver Local アプリケーションをデバッグできます。Visual Studio でデバッグする方法の詳細については、「Microsoft Visual Studio documentation」を参照してください。

ローカルシミュレーションをデバッグする方法

1. schema.yaml が作業ディレクトリにあることを確認します。
2. Visual Studio で、デバッグする各アプリケーションのコンテキストメニュー (PathfindingSampleLocalSpatial または PathfindingSampleLocalView など) を開き、デバッグセクションで作業ディレクトリを設定します。
3. デバッグするアプリケーションのコンテキストメニューを開き、[スタートアッププロジェクトとして設定] を選択します。
4. F5 を選択して、アプリケーションのデバッグを開始します。

シミュレーションをデバッグするための要件は、シミュレーションを正常に実行するための要件と同じです。スキーマで指定された数の空間アプリケーションを起動する必要があります。例えば、スキーマで 2x2 グリッドが指定されている場合に、空間アプリケーションをデバッグモードで起動した場合、さらに 3 つの空間アプリケーションを (デバッグモードまたはデバッグモードでない状態で) 起動するまでシミュレーションは実行されません。

カスタムアプリケーションをデバッグするには、まず空間アプリケーションを起動し、次にデバッガーでカスタムアプリケーションを起動する必要があります。

シミュレーションはロックステップで実行されることに注意してください。アプリケーションがブレークポイントに達するとすぐに、他のすべてのアプリケーションは一時停止します。そのブレークポイントから続行すると、他のアプリケーションは続行されます。

カスタムコンテナ

AWS SimSpace Weaver アプリケーションはコンテナ化 Amazon Linux 2 (AL2) 環境で実行されます。AWS クラウド では、SimSpace Weaver は Amazon Elastic Container Registry (Amazon ECR) から提供された `amazonlinux:2` イメージから構築された Docker コンテナでシミュレーションを実行します。カスタム Docker イメージを作成して Amazon ECR に保存し、提供されているデフォルトの Docker イメージの代わりにそのイメージをシミュレーションに使用できます。

カスタムコンテナを使用してソフトウェアの依存関係を管理し、標準の Docker イメージにはないソフトウェアコンポーネントを追加できます。例えば、アプリケーションが使用する一般公開ソフトウェアライブラリをコンテナに追加し、カスタムコードをアプリケーションの zip ファイルのみに入れることができます。

> **Important**
>
> Amazon ECR Public Gallery またはお客様のプライベート Amazon ECR レジストリにある Amazon ECR リポジトリでホストされている AL2 Docker イメージのみをサポートします。Amazon ECR の外部でホストされている Docker イメージはサポートされていません。Amazon ECR の詳細については、「Amazon Elastic Container Registry のドキュメント」を参照してください。

トピック

カスタムコンテナを作成する

これらの手順は、Docker と Amazon Elastic Container Registry (Amazon ECR) の使用方法についての知識があることを前提としています。Amazon ECR の詳細については、「Amazon ECR ユーザーガイド」を参照してください。

前提条件

- これらのアクションを実行するために使用する IAM アイデンティティ (ユーザーまたはロール) には、Amazon ECR を使用するための適切なアクセス許可があります。
- Docker は、ローカルシステムにインストールされます

カスタムコンテナを作成する方法

1. `Dockerfile` を作成します。

   AWS SimSpace Weaver アプリケーションを実行する `Dockerfile` は、まず Amazon ECR の Amazon Linux 2 イメージを使用します。

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

2. `Dockerfile` を構築します。
3. コンテナイメージを Amazon ECR にアップロードします。
   - AWS Management Console を使用します。
   - AWS Command Line Interface を使用します。

   > **Note**
   >
   > コンテナイメージを Amazon ECR にアップロードしようとして `AccessDeniedException` エラーが発生した場合、お使いの IAM アイデンティティ (ユーザーまたはロール) には Amazon ECR を使用するために必要なアクセス許可がない可能性があります。IAM アイデンティティに `AmazonEC2ContainerRegistryPowerUser` AWS 管理ポリシーをアタッチして、再試行できます。ポリシーをアタッチする方法の詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM アイデンティティの許可の追加および削除」を参照してください。

カスタムコンテナを使用するようにプロジェクトを変更する

これらの手順は、AWS SimSpace Weaver の使用方法をすでに理解していて、AWS クラウド でアプリケーションストレージと開発ワークフローをより効率的にしたいと考えていることを前提としています。

前提条件

- `create-project.bat` スクリプトによって作成された既存の SimSpace Weaver プロジェクトを変更しています。
- Amazon Elastic Container Registry (Amazon ECR) にはカスタムコンテナが用意されています。カスタムコンテナの作成方法の詳細については、「カスタムコンテナを作成する」を参照してください。

カスタムコンテナを使用するようにプロジェクトを変更する方法

1. Amazon ECR を使用するアクセス許可をプロジェクトのシミュレーションアプリケーションロールに追加します。

   a. 以下のアクセス許可を持つ IAM ポリシーがない場合は、ポリシーを作成します。ポリシー名 `simspaceweaver-ecr` を付けることをお勧めします。IAM ポリシー作成方法の詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

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

b. プロジェクトのシミュレーションアプリケーションロールの名前を検索します。

i. テキストエディタで、プロジェクトの [AWS CloudFormation テンプレート] を開きます。

```
project-folder\cloudformation\weaver-project-name-stack.yaml
```

ii. WeaverAppRole の下の [RoleName] プロパティを検索します。値はプロジェクトのシミュレーションアプリケーションロール名です。

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. simspaceweaver-ecr ポリシーをプロジェクトのシミュレーションアプリケーションロールにアタッチします。ポリシーをアタッチする方法の詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM アイデンティティの許可の追加および削除」を参照してください。

2. プロジェクトのシミュレーションスキーマでコンテナイメージを指定します。

- simulation_properties の下にオプションの default_image プロパティを追加して、すべてのドメインのデフォルトのカスタムコンテナイメージを指定できます。
- カスタムコンテナイメージを使用するドメインの image プロパティを app_config に追加します。値として Amazon ECR リポジトリ URI を指定します。ドメインごとに異なるイメージを指定できます。
  - image がドメインに指定されていない状態で default_image が指定されている場合、そのドメイン内のアプリケーションはデフォルトイメージを使用します。

- image がドメインに指定されていない状態で、default_image が指定されていない場合、そのドメイン内のアプリケーションは標準 SimSpace Weaver コンテナで実行されます。

Example カスタムコンテナ設定を含むスキーマスニペット

```
sdk_version: "1.16.0"
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. 通常どおりプロジェクトを構築してアップロードします。

カスタムコンテナに関するよくある質問

Q1. コンテナの中身を変更したい場合はどうすればいいですか?

- 実行中のシミュレーションの場合 — 実行中のシミュレーションのコンテナは変更できません。新しいコンテナを構築し、そのコンテナを使用する新しいシミュレーションを開始する必要があります。
- 新しいシミュレーションの場合 — 新しいコンテナを構築して Amazon Elastic Container Registry (Amazon ECR) にアップロードし、そのコンテナを使用する新しいシミュレーションを開始します。

Q2. シミュレーション用のコンテナイメージを変更する方法を教えてください。

- 実行中のシミュレーションの場合 — 実行中のシミュレーションのコンテナは変更できません。新しいコンテナを使用する新しいシミュレーションを開始する必要があります。
- 新しいシミュレーションの場合 — プロジェクトのシミュレーションスキーマに新しいコンテナイメージを指定します。詳細については、「カスタムコンテナを使用するようにプロジェクトを変更する」を参照してください。

カスタムコンテナのトラブルシューティング

トピック

AccessDeniedException Amazon Elastic Container Registry (Amazon ECR) にイメージをアップロードする場合

コンテナイメージを Amazon ECR にアップロードしようとして `AccessDeniedException` エラーが発生した場合、お使いの IAM アイデンティティ (ユーザーまたはロール) には Amazon ECR を使用するために必要なアクセス許可がない可能性があります。IAM アイデンティティに `AmazonEC2ContainerRegistryPowerUser` AWS 管理ポリシーをアタッチして、再試行できます。ポリシーをアタッチする方法の詳細については、「AWS Identity and Access Management ユーザーガイド」の「IAM アイデンティティの許可の追加および削除」を参照してください。

カスタムコンテナを使用するシミュレーションが開始できない

トラブルシューティングのヒント

- シミュレーションでログ記録が有効になっている場合は、エラーログを確認します。詳細については、「チュートリアルの詳細」を参照してください。
- カスタムコンテナなしでシミュレーションをテストします。
- シミュレーションをローカルでテストします。詳細については、「ローカル開発」を参照してください。

Python の使用

SimSpace Weaver アプリケーションおよびクライアントに Python を使用できます。Python ソフトウェア開発キット (Python SDK) は、標準 SimSpace Weaver アプリケーション SDK 配布パッケージの一部として含まれています。Python による開発は、サポートされている他の言語での開発と同様に機能します。

> **Important**
>
> SimSpace Weaver は Python のバージョン 3.9 のみをサポートしています。

> **Important**
>
> Python の SimSpace Weaver サポートには SimSpace Weaver バージョン 1.15.0 以降が必要です。

トピック

- Python プロジェクトの作成
- Python シミュレーションを開始する
- Python のサンプルクライアント
- 独自のビルドスクリプトを作成する
- Python の使用に関するよくある質問
- Python に関連する問題のトラブルシューティング

Python プロジェクトの作成

Python 以外のプロジェクトを作成するのと同じ方法で、`create-project.bat` スクリプトを使用して Python プロジェクトを作成します。`PythonBubblesSample` テンプレートを Python プロジェクトの起点として使用できます。以下の「Python プロジェクトを作成する」を参照してください。

Python カスタムコンテナ

Python ベースの SimSpace Weaver シミュレーションを AWS クラウド で実行するには、必要な依存関係を含むカスタムコンテナを作成できます。詳細については、「カスタムコンテナ」を参照してください。

Python カスタムコンテナには以下が含まれている必要があります。

- gcc
- openssl-devel
- bzip2-devel
- libffi-devel
- wget
- tar
- gzip
- make
- Python (バージョン 3.9)

`PythonBubblesSample` テンプレートを使用してプロジェクトを作成する場合は、(プロジェクトの `tools` フォルダにある) `create-custom-container.bat` スクリプトを実行して、必要な依存関係を含む Docker イメージを作成できます。このスクリプトは、Amazon Elastic Container Registry (Amazon ECR) にイメージをアップロードします。

`create-custom-container.bat` スクリプトは以下の `Dockerfile` を使用します。

```
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
```

独自の依存関係を、Dockerfile に追加できます。

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

requirements.txt ファイルには、PythonBubblesSample サンプルシミュレーションに必要な Python パッケージのリストが含まれています。

```
Flask==2.1.1
```

独自の Python パッケージの依存関係を requirements.txt に追加できます。

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

Dockerfile および requirements.txt はプロジェクトの tools フォルダにあります。

> **Important**
>
> create-custom-container.bat または Dockerfile に変更を加えた後は、requirements.txt を実行する必要があります。

> **Important**
>
> 技術的には Python シミュレーションでカスタムコンテナを使用する必要はありませんが、カスタムコンテナを使用することを強くお勧めします。Amazon Linux 2 (AL2) の標準コンテナは、Python を提供していません。したがって、Python を含むカスタムコンテナ

(create-custom-container.bat スクリプトによって作成されたコンテナイメージなど) を使用しない場合は、SimSpace Weaver にアップロードする各アプリケーションの zip ファイルに Python と必要な依存関係を含める必要があります。

Python プロジェクトを作成する

以下の手順は Microsoft Windows 用です。Windows Subsystem for Linux (WSL) を使用している場合は、代わりに .bat スクリプトの .sh バージョンを使用します。この手順を使用するには、Amazon Elastic Container Registry (Amazon ECR) の設定を完了する必要があります。詳細については、「Amazon ECR ユーザーガイド」の「Amazon ECR でのセットアップ」を参照してください。

Python のプロジェクトを作成する方法

1. コマンドプロンプトウィンドウで SimSpace Weaver SDK フォルダに移動します。

```
cd sdk-folder
```

2. PythonBubblesSampleテンプレートを使用して create-project.bat を実行します。

```
.\create-project.bat --name project-name --path project-folder-parent-path --
template PythonBubblesSample
```

3. プロジェクトフォルダの tools フォルダに移動します。##########は *project-folder-parent-path**project-name* です。

```
cd project-folder\tools
```

4. カスタムコンテナを作成します。

```
.\create-custom-container.bat
```

Python シミュレーションを開始する

SimSpace Weaver Local 内でも、AWS クラウド 内の SimSpace Weaver でも、Python ベースのシミュレーションは、通常の SimSpace Weaver シミュレーションと同じ方法で開始できます。詳細については、次を参照してください。

SimSpace Weaver Local

- ローカル開発

AWS クラウド

- クイックスタートチュートリアルの ステップ 3: クイックスタートスクリプトを実行する
- 詳細なチュートリアル: サンプルアプリケーションを構築しながら詳細を説明します

PythonBubblesSample には独自の Python サンプルクライアントが含まれています。詳細については、「Python のサンプルクライアント」を参照してください。

Python のサンプルクライアント

PythonBubblesSample テンプレートを使用してプロジェクトを作成する場合、プロジェクトには Python サンプルクライアントが含まれます。サンプルクライアントを使用して PythonBubblesSample シミュレーションを表示できます。サンプルクライアントを起点として使用して、独自の Python クライアントを作成することもできます。

以下の手順は、PythonBubblesSample プロジェクトを作成し、シミュレーションを開始済みであることを前提としています。

Python クライアントを起動する方法

1. コマンドプロンプトウィンドウで、プロジェクトの src\PythonBubblesSample\bin フォルダに移動します。

```
cd project-folder\src\PythonBubblesSample\bin
```

2. Python クライアントを実行します。

```
python bubbles_tkinter_client.py --host ip-address --port port-number --
simsize max-entitites
```

パラメータ

host

シミュレーションの IP アドレス。AWS クラウド で開始されたシミュレーションでは、[SimSpace Weaver] コンソールでシミュレーションの IP アドレスを確認するか、「クイックスタートチュートリアル」の「ステップ 4: IP アドレスとポート番号を取得する」の手順に従ってください。ローカルシミュレーションの場合は、IP アドレスとして `127.0.0.1` を使用します。

port

シミュレーションのポート番号。AWS クラウド で開始されたシミュレーションの場合、これは `Actual` ポート番号です。シミュレーションのポート番号は [SimSpace Weaver] コンソールで確認するか、「クイックスタートチュートリアル」の「ステップ 4: IP アドレスとポート番号を取得する」手順を使用して確認できます。ローカルシミュレーションの場合は、ポート番号として `7000` を使用します。

simsize

クライアントに表示するエンティティの最大数。

独自のビルドスクリプトを作成する

Python シミュレーション用に独自のビルドスクリプトを作成できます。ビルドを成功させるには、以下のステップを実行します。

1. `src/PythonBubblesSample/` の内容を `Build/out` ディレクトリにコピーします。
2. `${WEAVER_SDK_DIRECTORY}/lib/weaver/weaver_python_app_sdk_v1` の内容を `Build/out/lib/weaver_app_sdk_v1` ディレクトリにコピーします。
3. `Build/out/lib/weaver_app_sdk_v1` ディレクトリに `${WEAVER_SDK_DIRECTORY}/lib/weaver/libweaver_app_sdk_python_v1_39.so` をコピーします。
4. `Build/out/lib/weaver_app_sdk_v1/libweaver_app_sdk_python_v1_39.so` を `libweaver_app_sdk_python_v1.so` に名前変更します。
5. `Build/out/` ディレクトリの内容を圧縮します。
6. シミュレーションのスキーマで指定されているアプリケーション zip ごとに zip 処理を繰り返します。PythonBubblesSample の場合、スキーマには *`project-name`*`Spatial.zip` および *`project-name`*`View.zip` が必要です。

これらのステップが完了すると、zip ファイルをプロジェクトの Amazon S3 バケットにアップロードできます。

Python の使用に関するよくある質問

Q1. どのバージョンの Python がサポートされていますか?

SimSpace Weaver は Python のバージョン 3.9 のみをサポートしています。

Python に関連する問題のトラブルシューティング

トピック

- カスタムコンテナ作成中の失敗
- Python シミュレーションが開始されない
- Python シミュレーションまたはビュークライアントに ModuleNotFound エラーが表示される

カスタムコンテナ作成中の失敗

`create-custom-container.bat` の実行後にエラー `no basic auth credentials` が発生した場合は、Amazon ECR の一時的な認証情報に問題がある可能性があります。AWS リージョン ID および AWS アカウント番号を指定して以下のコマンドを実行します。

```
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**
>
> 指定した AWS リージョン が、シミュレーションに使用したものと同じであることを確認します。SimSpace Weaver をサポートする AWS リージョン のひとつを使用します。詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。

aws ecr コマンドを実行したら、もう一度 create-custom-container.bat を実行してください。

確認すべきその他のトラブルシューティングリソース

- カスタムコンテナのトラブルシューティング
- 「Amazon ECR ユーザーガイド」の「Amazon ECR トラブルシューティング」
- 詳細については、「Amazon ECR ユーザーガイド」の「Amazon ECR での設定」を参照してください

Python シミュレーションが開始されない

シミュレーションの管理ログに Unable to start app エラーが表示される場合があります。これは、カスタムコンテナの作成に失敗した場合に発生する可能性があります。詳細については、「カスタムコンテナ作成中の失敗」を参照してください。ログの詳細については、SimSpace Weaver Amazon CloudWatch Logs の ログ を参照してください。

コンテナに問題がないと確信できる場合は、アプリケーションの Python ソースコードを確認します。SimSpace Weaver Local を使用してアプリケーションをテストできます。詳細については、「ローカル開発」を参照してください。

Python シミュレーションまたはビュークライアントに ModuleNotFound エラーが表示される

必要な Python パッケージが見つからない場合、Python は ModuleNotFound エラーを返します。

シミュレーションが AWS クラウド にある場合は、requirements.txt にリストされている必要な依存関係がカスタムコンテナにすべて含まれていることを確認します。requirements.txt を編集する場合は、必ずもう一度 create-custom-container.bat を実行してください。

PythonBubblesSample クライアント側でエラーが発生した場合は、pip を使用して指定のパッケージをインストールします。

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

他のエンジンのサポート

SimSpace Weaver では独自のカスタム C++ エンジンを使用できます。現在、以下のエンジンのサポートを開発中です。これらのエンジンにはそれぞれ個別のドキュメントが用意されています。

> **Important**
>
> ここに記載されているエンジンとの統合は実験段階です。これらはプレビューできます。

エンジン

- Unity (最小バージョン 2021.3.7f1)
- Unreal Engine (最小バージョン 5.0)

Unity

Unity で SimSpace Weaver シミュレーションを構築する前に、Unity 開発環境をインストールしておく必要があります。AWS SimSpace Weaver (Unity SDK) の Unity SDK を別途ダウンロードし、そのパッケージの指示に従ってください。

> **Important**
>
> 最新バージョンの SimSpace Weaver アプリケーション SDK を使用する必要があります。最新バージョンは 1.16.0 です。詳細については、「AWS SimSpace Weaver バージョン」を参照してください。

Unity SDK のダウンロードおよび使用方法

1. Windows のコマンドプロンプトで、[*sdk-folder*] に移動します。
2. ダウンロードスクリプトを実行します。AWS リージョン で*#####*をシミュレーションを開始する場所 (例:`us-west-2`) に置き換えます。

   ```
   .\download-unity-package.bat --region region
   ```

   スクリプトがダウンロードされ、`SimSpaceWeaverUnityPackage.zip` が現在のフォルダーに解凍されます。
3. 「`SimSpaceWeaverUnityPackage\Release\Documentation\Unity_SDK_for_AWS_SimSpace_Weaver.pdf`」を参照してください。

Important

Unity で `JsonProperty`または の名前空間が見つからないというエラーが表示された場合は`JsonAttribute`、以下の手順に従って NewtonsoftJson パッケージを追加します。

1. Unity エディタで、メニューバーから [ウィンドウ]、[パッケージマネージャー] の順に選択します。
2. [パッケージマネージャー] ウィンドウで、ウィンドウ上部の [+](プラス) ボタンを選択します。
3. [git URL からパッケージを追加] を選択します。
4. 次のように入力します。

```
com.unity.nuget.newtonsoft-json
```

5. [追加] を選択します。

Important

Unity SDK は AWS Command Line Interface (AWS CLI) の名前付きプロファイルをサポートしていません。AWS IAM Identity Center または AWS CLI プロファイルを使用する場合は、Unity SDK を使用する前に名前付きプロファイルを `default` プロファイルにコピーするか、名前を変更する必要があります。詳細については、「AWS Command Line Interface ユーザガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザガイド」「設定と認証情報ファイルの設定」を参照してください。

Unreal Engine

ソースコードから Unreal Engine 専有サーバーを構築する必要があります。には、PathfindingSample 用の のバージョン SimSpaceWeaverAppSdkDistributable が含まれています Unreal Engine。詳細については、以下の個別の指示を参照してください。

```
sdk-folder\AWS_SimSpace_Weaver_Unreal_Guide.pdf
```

AWS SimSpace Weaver でライセンス対象ソフトウェアを使用する

AWS SimSpace Weaver では、選択したシミュレーションエンジンとコンテンツでシミュレーションを構築できます。SimSpace Weaver の使用に関連して、シミュレーションで使用するソフトウェアまたはコンテンツのライセンス条項を取得、維持、遵守する責任はユーザーにあります。仮想化ホスト環境でのソフトウェアとコンテンツの展開がライセンス契約書上で許可されていることを確認してください。

AWS CloudFormation によるリソースの管理

AWS CloudFormation を使用して AWS SimSpace Weaver リソースを管理できます。AWS CloudFormation は、AWS インフラストラクチャをコードとして指定、プロビジョニング、管理するのに役立つ独立した AWS サービスです。AWS CloudFormation を使用して、テンプレートと呼ばれる JSON または YAML ファイルを作成します。テンプレートはインフラストラクチャの詳細を指定します。AWS CloudFormation　はテンプレートを使用して、スタックと呼ばれる単一のユニットとしてインフラストラクチャをプロビジョニングします。スタックを削除すると、AWS CloudFormation は同時にスタックのすべてのデータを削除できます。標準のソースコード管理プロセスを使用してテンプレートを管理できます (例えば、Git などのバージョン管理システムでテンプレートを追跡するなどです)。AWS CloudFormation の詳細については、「AWS CloudFormation ユーザーガイド」を参照してください。

シミュレーションリソース

AWS では、リソースはユーザーが操作できるエンティティです。例として、Amazon EC2 インスタンス、Amazon S3 バケット、IAM ロールなどがあります。SimSpace Weaver シミュレーションはリソースです。構成では、通常、フォーム `AWS::service::resource` で AWS リソースを指定します。SimSpace Weaver では、シミュレーションリソースを `AWS::SimSpaceWeaver::Simulation` として指定します。AWS CloudFormation でのシミュレーションリソースの詳細については、「AWS CloudFormation ユーザーガイド」の「SimSpace Weaver」セクションを参照してください。

SimSpace Weaver での AWS CloudFormation の使用方法

プロビジョニングする AWS リソースを指定する AWS CloudFormation テンプレートを作成できます。テンプレートでは、アーキテクチャ全体、アーキテクチャの一部、または小規模なソリューションを指定できます。例えば、Amazon S3 バケット、IAM アクセス許可、Amazon Relational Database Service または Amazon DynamoDB のサポートデータベース、および `Simulation` リ

ソースを含む SimSpace Weaver ソリューションのアーキテクチャを指定できます。その後、AWS CloudFormation を使用して、これらすべてのリソースを 1 つの単位として同時にプロビジョニングできます。

Example IAM リソースを作成してシミュレーションを開始するテンプレート

以下のテンプレート例は、SimSpace Weaver がアカウントでアクションを実行するために必要な IAM ロールとアクセス許可を作成します。SimSpace Weaver アプリケーション SDK スクリプトはプロジェクトの作成時に特定の AWS リージョン でのロールとアクセス許可を作成しますが、AWS CloudFormation テンプレートを使用して、スクリプトを再実行しなくても別の AWS リージョン にシミュレーションをデプロイできます。例えば、ディザスタリカバリを目的としたバックアップシミュレーションを設定できます。

この例では、元のシミュレーション名は `MySimulation` です。AWS CloudFormation がスタックを構築する AWS リージョン には、スキーマのバケットがすでに存在しています。バケットには、その AWS リージョン でシミュレーションを実行するように適切に設定されたバージョンのスキーマが含まれています。スキーマはアプリケーションのの場所を指定していることに留意します。この zip ファイルは、シミュレーションと同じ AWS リージョン にある Amazon S3 バケットです。AWS CloudFormation がスタックをビルドするときに、アプリケーション zip バケットとファイルがすでに AWS リージョン に存在している必要があります。そうでない場合、シミュレーションは開始されません。この例のバケット名には AWS リージョン が含まれていますが、それによってバケットが実際にどこにあるかが決まるわけではないことに注意してください。バケットが実際にその AWS リージョン にあることを確認する必要があります (バケットのプロパティは AWS CLI の Amazon S3 コンソール、Amazon S3 API、または Amazon S3 コマンドで確認できます)。

この例では、AWS CloudFormation に組み込まれている関数とパラメータを使用して変数置換を行っています。詳細については、「AWS CloudFormation ユーザーガイド」の「組み込み関数リファレンス」と「疑似パラメータリファレンス」を参照してください。

```
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'
```

AWS CloudFormation でのスナップショットの使用

スナップショットはシミュレーションのバックアップです。以下の例では、スキーマからではなくスナップショットから新しいシミュレーションを開始します。この例のスナップショットは、SimSpace Weaver アプリケーション SDK プロジェクトのシミュレーションから作成され

ました。AWS CloudFormation は新しいシミュレーションリソースを作成し、スナップショットのデータで初期化します。新しいシミュレーションは、元のシミュレーションとは異なる MaximumDuration を持つ可能性があります。

元のシミュレーションのアプリケーションロールのコピーを作成して使用することをお勧めします。元のシミュレーションのアプリケーションロールは、そのシミュレーションの AWS CloudFormation スタックを削除すると削除される可能性があります。

```
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"
```

スナップショット

スナップショットを作成して、シミュレーションエンティティデータをいつでもバックアップできます。SimSpace Weaver は Amazon S3 バケットに .zip ファイルを作成します。スナップショットを使用して新しいシミュレーションを作成できます。SimSpace Weaver はスナップショットに保存されているエンティティデータを使用して新しいシミュレーションの State Fabric を初期化し、スナップショットの作成時に実行されていた空間アプリケーションとサービスアプリケーションを起動して、時計を適切なチェックマークに設定します。SimSpace Weaver はシミュレーションの設定をスキーマファイルではなくスナップショットから取得します。アプリケーションの .zip ファイルは、Amazon S3 内の元のシミュレーションと同じ場所にある必要があります。カスタムアプリケーションはすべて個別に起動する必要があります。

トピック

- スナップショットのユースケース
- SimSpace Weaver アプリケーション SDK を使用してスナップショットを操作する

- SimSpace Weaver コンソールを使用してスナップショットを操作する
- AWS CLI を使用してスナップショットを操作する
- SimSpace Weaver API を使用してスナップショットを操作する
- AWS CloudFormation でのスナップショットの使用
- スナップショットに関するよくある質問

スナップショットのユースケース

前の状態に戻り、分岐シナリオを検討する

シミュレーションのスナップショットを作成して、特定の状態に保存できます。その後、そのスナップショットから複数の新しいシミュレーションを作成し、その状態から分岐する可能性のあるさまざまなシナリオを検討できます。

ディザスタリカバリとセキュリティのベストプラクティス

特に 1 時間以上実行されるシミュレーションや複数のワーカーを使用するシミュレーションでは、定期的にシミュレーションをバックアップすることをお勧めします。バックアップは、障害やセキュリティインシデントからの回復に役立ちます。スナップショットは、シミュレーションのバックアップを提供する方法を提供します。スナップショットでは、アプリケーションの .zip ファイルが以前と同じ Amazon S3 の場所に存在している必要があります。アプリケーションの.zip ファイルを別の場所に移動できるようにする必要がある場合は、カスタムバックアップソリューションを使用する必要があります。

その他のベストプラクティスの詳細については、「SimSpace Weaver を使用する場合のベストプラクティス」および「SimSpace Weaver のセキュリティに関するベストプラクティス」を参照してください。

シミュレーション時間を延長する

シミュレーションリソースは、SimSpace Weaver でのシミュレーションの表現です。すべてのシミュレーションリソースには `MaximumDuration` 設定があります。シミュレーションは、`MaximumDuration` に達すると自動的に停止します。`MaximumDuration` の最大値は `14D` (14 日間) です。

シミュレーションをそのシミュレーションリソースの `MaximumDuration` より長く持続させる必要がある場合は、シミュレーションリソースがその `MaximumDuration` に達する前にスナップショットを作成できます。スナップショットを使用して新しいシミュレーションを開始 (新しいシミュレー

ションリソースを作成) できます。SimSpace Weaver はスナップショットからエンティティデータを初期化し、以前に実行したのと同じ空間アプリケーションとサービスアプリケーションを起動して、クロックを復元します。カスタムアプリケーションを起動して、その他のカスタム初期化を実行できます。新しいシミュレーションリソースの `MaximumDuration` は、起動時に別の値に設定できます。

SimSpace Weaver アプリケーション SDK を使用してスナップショットを操作する

SimSpace Weaver アプリケーション SDK (最低バージョン 1.13) で提供されているスクリプトを使用して、スナップショットを作成して使用できます。

SimSpace Weaver アプリケーション SDK はシミュレーションをプロジェクトごとに整理します。1 つのプロジェクトから複数のシミュレーションを開始できます。これらのシミュレーションはそれぞれ同じスキーマとアプリケーションの .zip ファイルを使用します。SimSpace Weaver アプリケーション SDK スクリプトは、プロジェクト名、AWS アカウント 番号、および AWS リージョン に基づいて、シミュレーション用のアセットを特定の Amazon S3 バケットに配置します。このスクリプトは、そのバケットのルートにある `snapshot` フォルダにあるスナップショットファイルと連携します。`snapshot` フォルダの Amazon S3 URI のフォーマットは以下のとおりです。

```
s3://weaver-project-name-lowercase-account-number-artifacts-region/snapshot
```

例

- プロジェクト名: `MyProject`
- AWS アカウント 番号: 111122223333
- AWS リージョン: `us-west-2`
- スナップショットフォルダ Amazon S3 URI: `s3://weaver-myproject-111122223333-artifacts-us-west-2/snapshot`

別の Amazon S3 バケットを使用したい場合は、スナップショットを使用するための以下の代替方法を参照してください。

スナップショットを使用するためのその他の方法

- SimSpace Weaver コンソール
- AWS CLI

- SimSpace Weaver API

トピック

- SimSpace Weaver アプリケーション SDK を使用してスナップショットを作成する
- SimSpace Weaver アプリケーション SDK を使用してスナップショットからシミュレーションを開始する
- SimSpace Weaver アプリケーション SDK を使用して、スナップショットからシミュレーションをクイックスタートする
- SimSpace Weaver アプリケーション SDK を使用して、プロジェクトのスナップショットを一覧表示する

SimSpace Weaver アプリケーション SDK を使用してスナップショットを作成する

スナップショットを作成するには、シミュレーションが STARTED 状態である必要があります。スナップショットの作成は、現在のティックが完了した後に開始されます。SimSpace Weaver はアプリケーションへのティックの送信を停止しますが、クロックの状態は STARTED を表示したままです。シミュレーションステータスは SNAPSHOT_IN_PROGRESS に変わります。スナップショットが終了すると、シミュレーションの状態は STARTED に戻り、アプリケーションは再びティックを受け取ります。

スナップショットを作成する方法

1. Windows コマンドプロンプトで、プロジェクトの [ツール] フォルダに移動します。

   ```
   cd project-folder\tools\windows
   ```

2. シミュレーションの名前がわからない場合は、list-simulations API を呼び出してシミュレーションリソースのリストを確認します。シミュレーションのステータスが STARTED であることを確認します。

   ```
   .\weaver-project-name-cli.bat list-simulations
   ```

3. プロジェクトの create-snapshot スクリプトを実行します。

   ```
   .\create-snapshot-project-name.bat --simulation simulation-name
   ```

   例

```
.\create-snapshot-MyProject.bat --simulation MyProjectSimulation_23-04-29_12_00_00
```

SimSpace Weaver はプロジェクトのアーティファクトバケットにスナップショットファイルを作成します。

例

- プロジェクト名: `MyProject`
- AWS アカウント 番号: 111122223333
- AWS リージョン: `us-west-2`
- スナップショットフォルダ Amazon S3 URI: `s3://weaver-myproject-111122223333-artifacts-us-west-2/snapshot`
- シミュレーション名: `MyProjectSimulation_23-04-29_12_00_00`
- スナップショット時間: 2023 年 4 月 29 日 15:30:27 UTC
- スナップショットファイル名: `MyProjectSimulation_23-04-29_12_00_00-230429-1530-27.zip`
- スナップショットファイル Amazon S3 URI: `s3://weaver-myproject-111122223333-artifacts-us-west-2/snapshot/MyProjectSimulation_23-04-29_12_00_00-230429-1530-27.zip`

SimSpace Weaver アプリケーション SDK を使用してスナップショットからシミュレーションを開始する

アプリケーション SDK スクリプトを使用してスナップショットからシミュレーションを開始すると、スクリプトはスナップショットなしでシミュレーションを開始する場合と同じ方法で新しいシミュレーション名を作成します。

スナップショットファイルは、以下の Amazon S3 URI を持つ Amazon S3 のスナップショットに存在している必要があります。

```
s3://weaver-project-name-lowercase-account-number-artifacts-region/snapshot
```

アプリケーションの .zip ファイルは、スナップショットが作成されたときと同じ場所にある必要があります。

SimSpace Weaver は新しいシミュレーションリソースを作成し、スナップショットに保存されているエンティティデータで State Fabric を初期化し、スナップショットの作成時に実行されていたのと同じ空間アプリケーションとサービスアプリケーションの新しいインスタンスを起動し、クロックを適切なティックに設定します。カスタムアプリケーションは通常のプロセスとは別に起動する必要があります。

`start-from-snapshot` スクリプトは `start-simulation` スクリプトのスナップショットバージョンです。`start-simulation` スクリプトと同様、`start-from-snapshot` スクリプトは自動的にはクロックを起動しません。クロックは別に起動する必要があります。

スナップショットからシミュレーションを開始する方法

1. Windows コマンドプロンプトで、プロジェクトの [ツール] フォルダに移動します。

   ```
   cd project-folder\tools\windows
   ```

2. `start-from-snapshot` スクリプトを実行します。

   ```
   .\start-from-snapshot-project-name.bat --snapshot-s3-file snapshot-file-name
   ```

   例

   ```
   .\start-from-snapshot-MyProject.bat --snapshot-s3-file
    MyProjectSimulation_23-04-29_12_00_00-230429-1530-27.zip
   ```

SimSpace Weaver アプリケーション SDK を使用して、スナップショットからシミュレーションをクイックスタートする

スナップショットからシミュレーションをクイックスタートできます。これはスナップショットなしのクイックスタートに似ています。

スナップショットファイルは、以下の Amazon S3 URI を持つ Amazon S3 のスナップショットに存在している必要があります。

```
s3://weaver-project-name-lowercase-account-number-artifacts-region/snapshot
```

アプリケーションの .zip ファイルは、スナップショットが作成されたときと同じ場所にある必要があります。

SimSpace Weaver は新しいシミュレーションリソースを作成し、スナップショットに保存されているエンティティデータで State Fabric を初期化し、スナップショットの作成時に実行されていたのと同じ空間アプリケーションとサービスアプリケーションの新しいインスタンスを起動し、クロックを適切なティックに設定します。カスタムアプリケーションは通常のプロセスとは別に起動する必要があります。

quick-start-from-snapshot スクリプトは quick-start スクリプトのスナップショットバージョンです。quick-start スクリプトと同様に、quick-start-from-snapshot スクリプトはクロックを自動的に開始します。また、パス探索サンプルプロジェクトのビューアプリケーションも起動します。

スナップショットからシミュレーションをクイックスタートする方法

1. Windows コマンドプロンプトで、プロジェクトの [ツール] フォルダに移動します。

   ```
   cd project-folder\tools\windows
   ```

2. quick-start-from-snapshot スクリプトを実行します。

   ```
   .\quick-start-from-snapshot-project-name-cli.bat --snapshot-s3-file snapshot-file-name
   ```

   例

   ```
   .\quick-start-from-snapshot-MyProject-cli.bat --snapshot-s3-file
    MyProjectSimulation_23-04-29_12_00_00-230429-1530-27.zip
   ```

SimSpace Weaver アプリケーション SDK を使用して、プロジェクトのスナップショットを一覧表示する

list-snapshots スクリプトを使用して、プロジェクトのスナップショットを一覧表示できます。このスクリプトは、プロジェクトの snapshot フォルダ内のファイルを一覧表示します。プロジェクトは SimSpace Weaver アプリケーション SDK に固有であり、アプリケーション SDK スクリプトおよびプロジェクトでのみ実行できます。このスクリプトは、Amazon S3 の snapshot フォルダ内のすべてのファイルがスナップショットファイルであることを前提としています。フォルダからファイルを移動または削除すると、それらのファイルはリストに表示されません。

プロジェクトのスナップショットを一覧表示する方法

1. Windows コマンドプロンプトで、プロジェクトの [ツール] フォルダに移動します。

```
cd project-folder\tools\windows
```

2. `list-snapshots` スクリプトを実行します。

```
.\list-snapshots-project-name.bat
```

例

```
.\list-snapshots-MyProject.bat
```

SimSpace Weaver コンソールを使用してスナップショットを操作する

SimSpace Weaver コンソールを使用してシミュレーションのスナップショットを作成できます。

スナップショットを使用するためのその他の方法

- SimSpace Weaver アプリケーション SDK スクリプト
- AWS CLI
- SimSpace Weaver API

トピック

- コンソールでスナップショットを作成する
- コンソールを使用してスナップショットからシミュレーションを開始する

コンソールでスナップショットを作成する

スナップショットを作成する方法

1. AWS Management Console にログインし、[SimSpace Weaver] コンソールに接続します。
2. ナビゲーションペインで、[シミュレーション] を選択します。
3. シミュレーション名の横にあるラジオボタンを選択します。シミュレーションの [ステータス] は [開始] である必要があります。

4. ページの上部で、[スナップショットの作成] を選択します。
5. [スナップショットの設定] の [スナップショットの保存先] に、スナップショットを作成する SimSpace Weaver のバケットまたはバケットとフォルダの Amazon S3 URI を入力します。使用可能なバケットをブラウズして場所を選択する場合は、[S3 を参照] を選択できます。

> **Important**
>
> Amazon S3 バケットはシミュレーションと同じ AWS リージョン にある必要があります。

> **Note**
>
> SimSpace Weaver は選択したスナップショットの保存先内に `snapshot` フォルダを作成します。SimSpace Weaver はその `snapshot` フォルダにスナップショットの .zip ファイルを作成します。

6. [スナップショットを作成] を選択します。

コンソールを使用してスナップショットからシミュレーションを開始する

スナップショットからシミュレーションを開始するには、スナップショットの .zip ファイルが、シミュレーションがアクセスできる Amazon S3 バケットに存在している必要があります。シミュレーションでは、シミュレーションの開始時に選択したアプリケーションロールで定義されているアクセス許可を使用します。元のシミュレーションのアプリケーションの .zip ファイルはすべて、スナップショットが作成されたときと同じ場所に存在している必要があります。

スナップショットからシミュレーションを開始する方法

1. AWS Management Console にログインし、[SimSpace Weaver] コンソールに接続します。
2. ナビゲーションペインで、[シミュレーション] を選択します。
3. ページ上部の [シミュレーションを開始] を選択します。
4. [シミュレーション設定] で、シミュレーションの名前と説明 (オプション) を入力します。シミュレーション名は AWS アカウント 内で一意である必要があります。
5. [シミュレーション開始方法] で [Amazon S3 のスナップショットを使用する] を選択します。

6. スナップショットの Amazon S3 URI には、スナップショットファイルの Amazon S3 URI を入力するか、[S3 を参照] を選択してファイルを参照して選択します。

> **Important**
>
> Amazon S3 バケットはシミュレーションと同じ AWS リージョン にある必要があります。

7. [IAM ロール] には、シミュレーションで使用するアプリケーションロールを選択します。
8. [最大期間] には、シミュレーションリソースを実行する最大期間を入力します。最大値は 14D です。最大期間の詳細については、「https://docs.aws.amazon.com/simspaceweaver/latest/APIReference/API_StartSimulation.html」を参照してください
9. タグを追加する場合は、[タグ - オプション] で [新しいタグを追加] を選択します。
10. [シミュレーションを開始] を選択します。

AWS CLI を使用してスナップショットを操作する

AWS CLI を使用してコマンドプロンプトから SimSpace Weaver API を呼び出すことができます。正常にインストールおよび設定された AWS CLI 必要です。詳細については、「バージョン 2 用 AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンのインストールまたは更新」を参照してください。

スナップショットを使用するためのその他の方法

- SimSpace Weaver アプリケーション SDK スクリプト
- SimSpace Weaver コンソール
- SimSpace Weaver API

トピック

- AWS CLI を使用してスナップショットを作成する
- AWS CLI を使用してスナップショットからシミュレーションを開始する

AWS CLI を使用してスナップショットを作成する

スナップショットを作成する方法

- コマンドプロンプトで CreateSnapshot API を呼び出します。

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

パラメータ

シミュレーション

開始したシミュレーション名。aws simspaceweaver list-simulations を使用して、シミュレーション名およびステータスを確認できます。

宛先

スナップショットファイルの送信先の Amazon S3 バケットとオプションのオブジェクトキーのプレフィックス指定する文字列。オブジェクトキーのプレフィックスは通常、バケット内のフォルダです。SimSpace Weaver はこの宛先の snapshot フォルダ内にスナップショットを作成します。

> **Important**
>
> Amazon S3 バケットはシミュレーションと同じ AWS リージョン にある必要があります。

例

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

CreateSnapshot API の詳細については、「AWS SimSpace Weaver API リファレンス」の「CreateSnapshot」を参照してください。

AWS CLI を使用してスナップショットからシミュレーションを開始する

スナップショットからシミュレーションを開始する方法

- コマンドプロンプトで StartSimulation API を呼び出します。

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

パラメータ

名前

新しいシミュレーションの名前。シミュレーション名は AWS アカウント 内で一意である必要があります。aws simspaceweaver list-simulations を使用して、既存のシミュレーション名を確認できます。

role-arn

シミュレーションが使用するアプリケーションロールの Amazon リソースネーム (ARN)。

snapshot-s3-location

スナップショットファイルの Amazon S3 バケットとオブジェクトキーを指定する文字列。

> **Important**
>
> Amazon S3 バケットはシミュレーションと同じ AWS リージョン にある必要があります。

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
```

StartSimulation API の詳細については、「AWS SimSpace Weaver API リファレンス」の「StartSimulation」を参照してください。

SimSpace Weaver API を使用してスナップショットを操作する

SimSpace Weaver API を直接呼び出してスナップショットを操作できます。API の詳細については、「AWS SimSpace Weaver API リファレンス」を参照してください。

スナップショットを使用するためのその他の方法

- SimSpace Weaver アプリケーション SDK スクリプト
- SimSpace Weaver コンソール
- AWS CLI

スナップショットを作成する

`CreateSnapshot` API を呼び出して、シミュレーションのスナップショットを作成できます。シミュレーションの状態は `STARTED` である必要があります。SimSpace Weaver は Amazon S3 バケット内の `snapshot` フォルダと指定したオブジェクトプレフィックスにスナップショットファイルを作成します。詳細については、「AWS SimSpace Weaver API リファレンス」の「CreateSnapshot」を参照してください。

スナップショットからシミュレーションを開始する

`StartSimulation` API を呼び出して新しいシミュレーションを開始するときに、スナップショットを提供できます。`SnapshotS3Location` パラメータの引数として JSON 文字列を指定します。この文字列は、スナップショットファイルの Amazon S3 バケット名とオブジェクトキーを指定します。`SnapshotS3Location` を提供する場合、`SchemaS3Location` も提供する必要があります。詳細については、「AWS SimSpace Weaver API リファレンス」の「StartSimulation」を参照してください。

スナップショットに関するよくある質問

スナップショットの作成中もシミュレーションは実行され続けますか?

シミュレーションリソースはスナップショット中も引き続き実行され、その間も引き続き料金が発生します。この時間はシミュレーションの最大期間に加算されます。スナップショットの進行中は、アプリケーションにティックを受け取りません。スナップショットの作成開始時にクロックのステータスが `STARTED` であった場合、クロックは `STARTED` ステータスのままになります。スナップショットの終了後、アプリケーションには再びティックを受け取ります。クロックのステータスが `STOPPED` であった場合、クロックの状態は `STOPPED` のままになります。`STARTED` ステータスのシ

ミュレーションは、クロックのステータスが STOPPED であっても実行されることに注意してください。

スナップショットが進行中に、シミュレーションが最大期間に達した場合はどうなりますか?

シミュレーションはスナップショットを終了し、スナップショット処理が終了すると (成功か失敗かにかかわらず) すぐに停止します。所要時間、予測できるスナップショットファイルのサイズ、および正常に完了するかどうかを、スナップショット処理の前にテストして、確認することをお勧めします。

スナップショットが進行中のシミュレーションを停止するとどうなりますか?

進行中のスナップショットは、シミュレーションを停止するとすぐに停止します。スナップショットファイルは作成されません。

実行中のスナップショットを停止する方法を教えてください。

進行中のスナップショットを停止する唯一の方法は、シミュレーションを停止することです。停止したシミュレーションは再開できません。

スナップショットの完了にはどれくらいの時間がかかりますか?

スナップショットの作成に必要な時間は、シミュレーションによって異なります。シミュレーションにどれくらいの時間がかかるかをスナップショット処理の前にテストして、確認することをお勧めします。

スナップショットファイルのサイズはどれくらいになりますか?

スナップショットファイルのサイズは、シミュレーションによって異なります。シミュレーション用のファイルのサイズをスナップショット処理の前にテストして、確認することをお勧めします。

メッセージング

メッセージング API は、シミュレーション内のアプリケーション間通信を簡素化します。メッセージを送受信するための APIs は、SimSpace Weaverアプリケーション SDK の一部です。メッセージングは現在、ベストエフォートアプローチを使用してメッセージを送受信しています。 SimSpace Weaver は次のシミュレーションティックでメッセージを送受信しようとしますが、配信、注文、到着時間保証はありません。

トピック

- メッセージングのユースケース

メッセージングのユースケース

シミュレーションアプリケーション間の通信

メッセージング API を使用して、シミュレーション内のアプリケーション間で通信します。これを使用して、遠くのエンティティの状態を変更したり、エンティティの動作を変更したり、シミュレーション全体に情報をブロードキャストしたりできます。

メッセージの受信を確認する

送信されたメッセージには、メッセージヘッダーに送信者に関する情報が含まれます。この情報を使用して、メッセージの受信時に確認応答を返します。

カスタムアプリケーションが受信したデータをシミュレーション内の他のアプリケーションに転送する

メッセージングは、クライアントが で実行されているカスタムアプリケーションに接続する方法に代わるものではありませんSimSpace Weaver。ただし、メッセージングでは、クライアントデータを受信するカスタムアプリケーションから外部接続を持たない他のアプリケーションにデータを転送できます。メッセージフローは逆方向に動作し、外部接続のないアプリがデータをカスタムアプリに転送してからクライアントに転送することもできます。

メッセージング APIs

メッセージング APIs はSimSpace Weaverアプリケーション SDK (最小バージョン 1.16.0) に含まれています。メッセージングは C++、Python、Unreal Engine 5 および Unity との統合でサポートされています。

メッセージトランザクションを処理する 2 つの関数として、 `SendMessage`と があります`ReceiveMessages`。すべての送信済みメッセージには、送信先とペイロードが含まれます。`ReceiveMessages` API は、アプリのインバウンドメッセージキューに現在存在するメッセージのリストを返します。

C++

メッセージの送信

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

メッセージの受信

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

Python

メッセージの送信

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

メッセージの受信

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

トピック

- メッセージの送信
- メッセージの受信
- 送信者への返信

メッセージの送信

メッセージは、トランザクション (他の Weaver API コールと同様) 、ペイロード、および送信先で構成されます。

メッセージペイロード

メッセージペイロードは、最大 256 バイトの柔軟なデータ構造です。メッセージペイロードを作成するためのベストプラクティスとして、以下をお勧めします。

メッセージペイロードを作成するには

1. メッセージの内容を定義するデータ構造 (C++ `struct`の など) を作成します。
2. メッセージに送信する値を含むメッセージペイロードを作成します。
3. `MessagePayload` オブジェクトを作成します。

メッセージの送信先

メッセージの送信先は、 `MessageEndpoint` オブジェクトによって定義されます。これには、エンドポイントタイプとエンドポイント ID の両方が含まれます。現在サポートされているエンドポイントタイプは のみです。これによりPartition、シミュレーション内の他のパーティションにメッセージをアドレス指定できます。エンドポイント ID は、ターゲット送信先のパーティション ID です。

メッセージで指定できる送信先アドレスは 1 つだけです。複数のパーティションに同時にメッセージを送信する場合は、複数のメッセージを作成して送信します。

ある位置からメッセージエンドポイントを解決する方法については、「」を参照してくださいメッセージングを使用する際のヒント。

メッセージの送信

送信先オブジェクトとペイロードオブジェクトを作成した後、 `SendMessage` API を使用できます。

C++

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

Python

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

メッセージを送信する完全な例

次の例は、汎用メッセージを作成して送信する方法を示しています。この例では、16 個の個別のメッセージを送信します。各メッセージには、0 と 15 の間の値を持つペイロードと、現在のシミュレーションティックが含まれます。

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
```

メッセージの受信

SimSpace Weaver は、パーティションのインバウンドメッセージキューにメッセージを配信します。ReceiveMessages API を使用して、キューからのメッセージを含むMessageListオブジェクトを取得します。メッセージデータを取得するには、 ExtractMessage API を使用して各メッセージを処理します。

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
```

送信者への返信

すべての受信メッセージには、メッセージの元の送信者に関する情報を含むメッセージヘッダーが含まれています。message.header.source_endpoint を使用して返信を送信できます。

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
```

メッセージングを使用するタイミング

でのメッセージングSimSpace Weaverは、シミュレーションアプリケーション間で情報を交換するための別のパターンを提供します。サブスクリプションは、シミュレーションの特定のアプリケーションまたは領域からデータを読み取るためのプルメカニズムを提供します。メッセージは、シミュレーションの特定のアプリケーションまたは領域にデータを送信するプッシュメカニズムを提供します。

以下は、サブスクリプションを通じてデータを取得または読み取るのではなく、メッセージングを使用してデータをプッシュする方が役立つ 2 つのユースケースです。

Example 1: エンティティの位置を変更するコマンドを別のアプリケーションに送信する

```
// 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);
```

受信側では、アプリはエンティティの位置を更新し、ステートファブリックに書き込みます。

```
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: エンティティ作成メッセージを空間アプリケーションに送信する

```
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);
```

受信側では、アプリはステートファブリックに新しいエンティティを作成し、その位置を更新します。

```
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)));
}
```

メッセージングを使用する際のヒント

位置またはアプリ名からエンドポイントを解決する

AllPartitions 関数を使用して、メッセージパーティション ID とメッセージ送信先を決定するために必要な空間境界とドメイン IDsを取得できます。ただし、メッセージする位置はわかってい

ても、パーティション ID はわかっていない場合は、 MessageEndpointResolver 関数を使用できます。

```
/**
* 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);
};
```

メッセージペイロードのシリアル化と逆シリアル化

次の関数を使用して、メッセージペイロードを作成および読み取ることができます。詳細については、ローカルシステムのアプリケーション SDK ライブラリの MessagingUtils「.h」を参照してください。

```
/**
* 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());
}
```

メッセージングエラーとトラブルシューティング

メッセージング APIs を使用すると、次のエラーが発生することがあります。

エンドポイント解決エラー

これらのエラーは、アプリケーションがメッセージを送信する前に発生する可能性があります。

ドメイン名のチェック

無効なエンドポイントにメッセージを送信すると、次のエラーが発生します。

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

これは、カスタムアプリケーションにメッセージを送信しようとしたときに、そのカスタムアプリケーションがまだシミュレーションに参加していない場合に発生する可能性があります。DescribeSimulation API を使用して、メッセージを送信する前にカスタムアプリが起動していることを確認します。この動作は、 SimSpace Weaver Localと で同じですAWS クラウド。

位置チェック

有効なドメイン名があるが無効な位置のエンドポイントを解決しようとすると、次のエラーが発生します。

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

SimSpace Weaver アプリケーション SDK に含まれている`MessageEndpointResolverMessageUtils`ライブラリで を使用することをお勧めします。

メッセージ送信エラー

アプリがメッセージを送信すると、次のエラーが発生する可能性があります。

アプリあたりのメッセージ送信制限、ティックあたり、超過

シミュレーションティックごとにアプリごとに送信できるメッセージ数の現在の制限は 128 です。同じティックでのその後の呼び出しは、次のエラーで失敗します。

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

SimSpace Weaver は、次のティックで未送信メッセージを送信しようとします。この問題を解決するには、送信頻度を下げます。256 バイトの制限より小さいメッセージペイロードを組み合わせて、アウトバウンドメッセージの数を減らします。

この動作は、 SimSpace Weaver Localと で同じですAWS クラウド。

メッセージペイロードサイズの制限を超えました

メッセージペイロードサイズの現在の制限は、 SimSpace Weaver Localと の両方で 256 バイトですAWS クラウド。256 バイトを超えるペイロードでメッセージを送信すると、次のエラーが発生します。

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

SimSpace Weaver は各メッセージをチェックし、制限を超えたメッセージのみを拒否します。例えば、アプリが 10 個のメッセージを送信しようとして、1 個のチェックに失敗すると、その 1 つのメッセージだけが拒否されます。 SimSpace Weaverは他の 9 個のメッセージを送信します。

この動作は、 SimSpace Weaver Localと で同じですAWS クラウド。

送信先が送信元と同じである

アプリケーションは、自分が所有するパーティションにメッセージを送信することはできません。アプリケーションが所有しているパーティションにメッセージを送信すると、次のエラーが表示されます。

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

この動作は、 SimSpace Weaver Localと で同じですAWS クラウド。

ベストエフォート型メッセージング

SimSpace Weaver はメッセージの配信を保証するものではありません。サービスは、後続のシミュレーションティックでメッセージの配信を完了しようとしますが、メッセージが失われたり遅延したりする可能性があります。

SimSpace Weaver を使用する場合のベストプラクティス

SimSpace Weaver を使用する場合、以下のベストプラクティスをお勧めします。

トピック

請求アラームの設定

AWS では、簡単にリソースをプロビジョニングすることができ、たとえそれが必要でなくなっても、常に稼動させておけます。その結果、コストが跳ね上がり、請求書を受け取って驚くことがあるかもしれません。Amazon では、コストが設定したしきい値を超えたときにトリガーおよび通知 CloudWatch するアラームを設定できます。コスト管理ツールを使用してコストを調べることができます。詳細については、以下を参照してください。

- AWS の推測料金をモニタリングする請求アラームの作成
- AWS Cost Management とは

SimSpace Weaver Local を使用する

AWS クラウド の SimSpace Weaver サービスにアップロードする前に、SimSpace Weaver Local を使用してシミュレーションを開発し、テストすることをお勧めします。SimSpace Weaver Local を使用して開発することには以下のようなメリットがあります。

- 大量のアップロードを待つ必要はありません
- 作成できるローカルシミュレーションの数に制限はありません
- ローカルコンピューターの計算時間には料金は発生しません
- アプリケーションからコンソール出力に直接アクセスできます
- AWS クラウド で再作成しなくても、ローカルシミュレーションを変更、再構築、再起動できます

不要なシミュレーションを停止します

シミュレーションの実行中は、料金が発生します。料金が発生しないようにするには、シミュレーションを停止する必要があります。実行中のシミュレーションもシミュレーションの最大数のクォータにカウントされます。ロギングが設定されたシミュレーションを実行すると大量のログが生成される場合があり、それに対しても料金が発生します。追加料金が発生しないようにするには、必要のないシミュレーションをすべて中止する必要があります。

> **Important**
>
> シミュレーションクロックを停止してもシミュレーションは停止しません。クロックはアプリケーションへのティックの公開を停止するだけです。シミュレーションを停止すると、再開することはできません。

不要なリソースの削除

SimSpace Weaver でシミュレーションを作成するたびに、他の AWS サービスのリソースも作成されます。これらの他のサービスのリソースやデータに対して料金が発生することがあります。実行中のシミュレーションと失敗したシミュレーションは、最大シミュレーション回数のクォータにカウントされます。新しいシミュレーションを開始するために、失敗した不要なシミュレーションを削除する必要があります。シミュレーションを削除しても、他の AWS サービスに存在するシミュレーションのリソースは削除されない場合があります。例えば、Amazon CloudWatch Logs のシミュレーションログデータは、削除するまでそこに残ります。その口グデータには料金が発生します。不要になったシミュレーションに関連するリソースをすべて削除する必要があります。詳細については ステップ 6: シミュレーションを停止してクリーンアップする の「クイックスタートチュートリアル」を参照してください。

バックアップの取得

すべてをバックアップし、バックアップする計画を立てることをお勧めします。AWS にデータが保存されているからといって、バックアップする必要がないと考えるべきではありません。シミュレーションの状態をバックアップする必要がある場合は、独自のシステムを作成する必要があります。複数の AWS リージョン を使用し、必要に応じて本番環境のワークロードを迅速に AWS リージョンに切り替えることができるように、計画を立てることを検討してください。SimSpace Weaver をサポートする AWS リージョン の詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。

AWS SimSpace Weaver でのセキュリティ

AWS でのクラウドセキュリティは最優先事項です。AWS のユーザーは、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャを利用できます。

セキュリティは、AWS とユーザーの間の責任共有です。責任共有モデル では、これをクラウドのセキュリティおよびクラウド内のセキュリティとして説明しています。

- クラウドのセキュリティ - AWS は、AWS クラウド で AWS のサービスを実行するインフラストラクチャを保護する責任を担います。また、AWS は、ユーザーが安全に使用できるサービスも提供します。AWSコンプライアンスプログラムg11AWSコンプライアンスプログラム/g11g10AWSコンプライアンスプログラム/g10の一環として、サードパーティーの監査が定期的にセキュリティの有効性をテストおよび検証しています。AWS SimSpace Weaver に適用するコンプライアンスプログラムの詳細については、「コンプライアンスプログラムによる対象範囲内の AWS のサービス」「」を参照してください。
- クラウド内のセキュリティ - ユーザーの責任は、使用する AWS のサービスに応じて異なります。また、お客様は、データの機密性、お客様の会社の要件、および適用される法律および規制など、その他の要因についても責任を負います。

このドキュメントは、SimSpace Weaver を使用する際に責任共有モデルを適用する方法を理解するのに役立ちます。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するために SimSpace Weaver を設定する方法を示します。また、SimSpace Weaver リソースのモニタリングや保護に役立つ、その他 AWS サービスの使用方法についても説明します。

トピック

AWS SimSpace Weaver でのデータ保護

AWS責任共有モデルは、AWS SimSpace Weaver でのデータ保護に適用されます。このモデルで説明されているように、AWS には、AWS クラウド のすべてを実行するグローバルインフラストラクチャを保護する責任があります。ユーザーには、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービスのセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、「AWS セキュリティブログ」に投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事を参照してください。

データを保護するため、AWS アカウント の認証情報を保護し、AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーをセットアップすることをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみを各ユーザーに付与できます。また、次の方法でデータを保護することをおすすめします。

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 が必須です。TLS 1.3 が推奨されます。
- AWS CloudTrail で API とユーザーアクティビティロギングをセットアップします。
- AWS のサービス内でデフォルトである、すべてのセキュリティ管理に加え、AWS の暗号化ソリューションを使用します。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API により AWS にアクセスするときに FIPS 140-2 検証済み暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、API、AWS CLI、または AWS SDK で SimSpace Weaver または他の AWS のサービス を使用する場合も同様です。タグ、または名前に使用される自由形式のテキストフィールドに入力されるデータは、請求または診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

保管中の暗号化

データがディスクなどの不揮発性 (永続的) データストレージに格納されている場合、データは保管中と見なされます。メモリやレジスタなどの揮発性データストレージにあるデータは、保管中とは見なされません。

SimSpace Weaver を使用する場合、保管中のデータとは以下のデータのみです。

- Amazon Simple Storage Service (Amazon S3) にアップロードするアプリケーションとスキーマ
- Amazon に保存されているシミュレーションログデータ CloudWatch

SimSpace Weaver が内部的に使用するその他のデータは、シミュレーションを停止した後は保持されません。

保管中のデータの暗号化の方法については、以下を参照してください。

- Amazon S3 でのデータの暗号化
- ログデータの暗号化

転送中の暗号化

AWS Command Line Interface (AWS CLI)、AWS SDK、SimSpace Weaver アプリケーション SDK を介した SimSpace Weaver API への接続では、署名バージョン 4 の署名プロセスで TLS 暗号化を使用します。AWS は接続に使用するセキュリティ認証情報の IAM 定義のアクセスポリシーを使用して認証を管理します。

内部的には、SimSpace Weaver は TLS を使用して、使用する他の AWS サービスに接続します。

> **Important**
>
> アプリケーションとクライアント間の通信には SimSpace Weaver は関係ありません。必要に応じてシミュレーションクライアントとの通信を暗号化する責任は、お客様にあります。クライアント接続間でのすべての転送中のデータの暗号化を作成することをお勧めします。

暗号化ソリューションをサポートできる AWS サービスの詳細については、「AWS セキュリティブログ」を参照してください。

ネットワーク間トラフィックのプライバシー

SimSpace Weaver コンピュートリソースは、すべての SimSpace Weaver のお客様が共有する 1 つの Amazon VPC 内にあります。内部 SimSpace Weaver サービストラフィックはすべて AWS ネットワーク内にとどまり、インターネットを経由することはありません。シミュレーションクライアントとアプリケーション間の通信はインターネットを経由します。

AWS SimSpace Weaver の ID とアクセス管理

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御するために役立つ AWS のサービスです。IAM 管理者は、誰をわ認証 (サインイン) し、誰に SimSpace Weaver リソースの使用を許可する (権限を持たせる) かを制御します。IAM は、無料で使用できる AWS のサービスです。

トピック

対象者

AWS Identity and Access Management (IAM) の用途は、SimSpace Weaver で行う作業によって異なります。

サービスユーザー - SimSpace Weaver サービスを使用してジョブを実行する場合は、必要な権限と認証情報を管理者が用意します。作業を実行するためにさらに多くの SimSpace Weaver 機能を使用するとき、追加の権限が必要になる場合があります。アクセスの管理方法を理解すると、管理者から適切な権限をリクエストするのに役に立ちます。SimSpace Weaver 機能にアクセスできない場合は、「AWS SimSpace Weaver ID とアクセスのトラブルシューティング」を参照してください。

サービス管理者 - 社内の SimSpace Weaver リソースを担当している場合は、通常、SimSpace Weaver への完全なアクセスがあります。サービスのユーザーがどの SimSpace Weaver 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を確認して、IAM の基本概念を理解してください。お客様の会社で SimSpace Weaver で IAM を利用する方法の詳細については、「AWS SimSpace Weaver と IAM の連携方法」を参照してください。

IAM 管理者 - 管理者は、SimSpace Weaver へのアクセスを管理するポリシーの書き込み方法の詳細について確認する場合があります。IAM で使用できる SimSpace Weaver アイデンティティベースのポリシーの例を表示するには、「AWS SimSpace Weaver のアイデンティティベースのポリシーの例」を参照してください。

アイデンティティを使用した認証

認証とは、アイデンティティ認証情報を使用して AWSにサインインする方法です。ユーザーは、AWS アカウントのルートユーザー として、または IAM ロールを引き受けることによって、認証済み (AWS にサインイン済み) である必要があります。

ID ソースから提供された認証情報を使用して、フェデレーティッドアイデンティティとして AWS にサインインできます。 AWS IAM Identity Centerフェデレーティッドアイデンティティの例としては、IAM アイデンティティセンターユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報などがあります。フェデレーティッドアイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーションを使用して AWSにアクセスする場合、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、AWS Management Console または AWS アクセスポータルにサインインできます。AWS へのサインインの詳細については、『AWS サインイン ユーザーガイド』の「AWS アカウント にサインインする方法」を参照してください。

プログラムで AWS にアクセスする場合、AWS は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) を提供し、認証情報でリクエストに暗号で署名します。AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。リクエストに署名する推奨方法の使用については、『IAM ユーザーガイド』の「AWS API リクエストの署名」を参照してください。

使用する認証方法を問わず、追加のセキュリティ情報の提供が求められる場合もあります。例えば、AWS では多要素認証 (MFA) を使用してアカウントのセキュリティを高めることを推奨しています。詳細については、「AWS IAM Identity Center ユーザーガイド」の「多要素認証」 および「IAM ユーザーガイド」の「AWS での多要素認証 (MFA) の使用」を参照してください。

AWS アカウントのルートユーザー

AWS アカウント を作成する場合、このアカウントのすべての AWS のサービス とリソースに対して完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。このアイデンティティは AWS アカウントのルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることによってアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報を保護し、それらを使用してルートユーザーのみが実行できるタスクを実行してください。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーティッド ID

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに対し、ID プロバイダーとのフェデレーションを使用して、一時的な認証情報の使用により、AWS のサービスへのアクセスを要求します。

フェデレーテッドアイデンティティ は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、アイデンティティセンターディレクトリのユーザーか、または ID ソースから提供された認証情報を使用して AWS のサービス にアクセスするユーザーです。フェデレーティッド ID が AWS アカウント にアクセスすると、ロールが継承され、そのロールが一時的な認証情報を提供します。

アクセスを一元管理する場合は、AWS IAM Identity Center を使用することをお勧めします。IAM アイデンティティセンターでユーザーとグループを作成するか、すべての AWS アカウント とアプリケーションで使用するために、独自の ID ソースで一連のユーザーとグループに接続して同期することもできます。IAM アイデンティティセンター の詳細については、「AWS IAM Identity Center ユーザーガイド」の「What is IAM アイデンティティセンター?」(IAM アイデンティティセンター とは) を参照してください。

IAM ユーザーとグループ

IAM ユーザー は、1 人のユーザーまたは 1 つのアプリケーションに対して特定の許可を持つ AWS アカウント 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、「IAM ユーザーガイド」の「IAM ユーザー (ロールではなく) の作成が適している場合」を参照してください。

IAM ロール

IAM ロール は、特定の許可を持つ、AWS アカウント 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロールを切り替えることにより、AWS Management Console で一時的に IAM ロールを引き受けることができます。ロールを引き受けるには、AWS CLI または AWS API オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次のような状況で役立ちます。

- フェデレーティッドユーザーアクセス - フェデレーティッドアイデンティティに許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、「IAM ユーザーガイド」の「サードパーティアイデンティティプロバイダー用のロールの作成」を参照してください。IAM Identity Center を使用する場合、許可セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM アイデンティティセンターは、アクセス許可セットを IAM のロールに関連付けます。権限セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「権限セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースへのアクセスを別のアカウントの人物 (信頼できるプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の AWS のサービスでは、(ロールをプロキシとして使用する代わりに) リソースにポリシーを直接アタッチできます。クロス

アカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

- クロスサービスアクセス - 一部の AWS のサービスでは、他の AWS のサービスの機能を使用します。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して AWS でアクションを実行するユーザーは、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、AWS のサービスを呼び出すプリンシパルの権限を、AWS のサービスのリクエストと合わせて使用し、ダウンストリームのサービスに対してリクエストを行います。FAS リクエストは、サービスが、完了するために他の AWS のサービス またはリソースとのやりとりを必要とするリクエストを受け取ったときにのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
  - サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービス に権限を委任するロールの作成」を参照してください。
  - サービスにリンクされたロール - サービスにリンクされたロールは、AWS のサービス にリンクされたサービスロールの一種です。サービスがロールを引き受け、ユーザーに代わってアクションを実行できるようになります。サービスにリンクされたロールは、AWS アカウント に表示され、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの許可を表示できますが、編集はできません。
- Amazon EC2 で実行されているアプリケーション - EC2 インスタンスで実行され、AWS CLI または AWS API 要求を行っているアプリケーションの一時的な認証情報を管理するには、IAM ロールを使用できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスに添付されたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用してアクセス許可を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、「IAM ユーザーガイド」の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」を参照してください。

ポリシーを使用したアクセス権の管理

AWS でアクセスを制御するには、ポリシーを作成して AWS アイデンティティまたはリソースにアタッチします。ポリシーは AWS のオブジェクトであり、アイデンティティやリソースに関連付けて、これらのアクセス許可を定義します。AWS は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うと、これらのポリシーを評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。大半のポリシーは JSON ドキュメントとして AWSに保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWSJSON ポリシーを使用して、だれが何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するためのアクセス許可をユーザーに付与するため、IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。このポリシーがあるユーザーは、AWS Management Console、AWS CLI、または AWS API からロール情報を取得できます。

アイデンティティベースポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件を制御します。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

アイデンティティベースのポリシーは、さらに インラインポリシー または マネージドポリシー に分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれます。管理ポリシーは、AWS アカウント 内の複数のユーザー、グループ、およびロールにアタッチできるスタンドアロンポリシーです。マネージドポリシーには、AWS マネージドポリシーとカスタマー管理ポリシーがあります。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、「IAM ユーザーガイド」の「マネージドポリシーとインラインポリシーの比較」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーの例には、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーションユーザー、または AWS のサービス を含めることができます。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは IAM の AWSマネージドポリシーは使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための許可を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、AWS WAF、および Amazon VPC は、ACL をサポートするサービスの例です。ACL の詳細については、「Amazon Simple Storage Service デベロッパーガイド」の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS では、他の一般的ではないポリシータイプをサポートしています。これらのポリシータイプでは、より一般的なポリシータイプで付与される最大の許可を設定できます。

- 権限の境界 - 権限の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる許可の上限を設定する高度な機能です。エンティティに権限の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとその権限の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、権限の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。権限の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの権限の境界」を参照してください。
- サービスコントロールポリシー (SCP) - SCP は、AWS Organizations で組織や組織単位 (OU) の最大許可を指定する JSON ポリシーです。AWS Organizations は、ユーザーのビジネスが所有する複数の AWS アカウント をグループ化し、一元的に管理するサービスです。組織内のすべての

機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP はメンバーアカウントのエンティティに対する権限を制限します (各 AWS アカウントのルートユーザー など)。Organizations と SCP の詳細については、AWS Organizations ユーザーガイド の「SCP の仕組み」を参照してください。

- セッションポリシー - セッションポリシーは、ロールまたはフェデレーティッドユーザーの一時的なセッションをプログラムで作成する際に、パラメータとして渡す高度なポリシーです。結果としてセッションの権限の範囲は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」をご参照ください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関連するとき、リクエストを許可するかどうかを AWS が決定する方法の詳細については、「IAM ユーザーガイド」の「ポリシーの評価論理」を参照してください。

AWS SimSpace Weaver と IAM の連携方法

IAM を使用して SimSpace Weaver へのアクセスを管理する前に、SimSpace Weaver で利用できる IAM の機能について学びます。

AWS SimSpace Weaver で使用できる IAM の機能

IAM の機能	SimSpace Weaver サポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	いいえ
ポリシーアクション	あり
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	はい
ACL	なし

IAM の機能	SimSpace Weaver サポート
ABAC (ポリシー内のタグ)	はい
一時的な認証情報	あり
プリンシパル権限	あり
サービスロール	あり
サービスリンクロール	いいえ

SimSpace Weaver およびその他の AWS のサービスがほとんどの IAM 機能と連携する方法の概要を把握するには、「IAM ユーザーガイド」の「IAM と連携する AWS のサービス」を参照してください。

SimSpace Weaver のアイデンティティベースのポリシー

アイデンティティベースポリシーをサポートする	あり

アイデンティティベースのポリシーは、IAM ユーザー、ユーザーグループ、ロールなどのアイデンティティにアタッチできる JSON アクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件を制御します。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。プリンシパルは、それがアタッチされているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素について学ぶには、IAM ユーザーガイドのIAM JSON ポリシーの要素のリファレンスを参照してください。

SimSpace Weaver のアイデンティティベースのポリシーの例

SimSpace Weaver アイデンティティベースのポリシーの例を表示するには、「AWS SimSpace Weaver のアイデンティティベースのポリシーの例」を参照してください。

SimSpace Weaver 内のリソースベースのポリシー

リソースベースのポリシーのサポート	なし

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーの例には、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーションユーザー、または AWS のサービス を含めることができます。

クロスアカウントアクセスを有効にするには、全体のアカウント、または別のアカウントの IAM エンティティを、リソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる AWS アカウント にある場合、信頼できるアカウントの IAM 管理者は、リソースへのアクセス許可をプリンシパルエンティティ (ユーザーまたはロール) に付与する必要もあります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーを追加する必要はありません。詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

SimSpace Weaver のポリシーアクション

ポリシーアクションに対するサポート	あり

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

JSON ポリシーの`Action`要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連する AWS API オペレーションと同じです。一致する API オペレーションのない権限のみのアクションなど、いくつかの例外があります。また、ポリシーに複数アクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

SimSpace Weaver アクションのリストを確認するには、サービス認可リファレンスの「AWS SimSpace Weaver で定義されるアクション」を参照してください。

SimSpace Weaver のポリシーアクションは、アクションの前に以下のプレフィックス を使用します。

```
simspaceweaver
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "simspaceweaver:action1",
      "simspaceweaver:action2"
         ]
```

SimSpace Weaver アイデンティティベースのポリシーの例を表示するには、「AWS SimSpace Weaver のアイデンティティベースのポリシーの例」を参照してください。

SimSpace Weaver のポリシーリソース

ポリシーリソースに対するサポート	あり

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

`Resource` JSON ポリシーの要素は、オブジェクトあるいはアクションが適用されるオブジェクトを指定します。ステートメントには、`Resource` または `NotResource` 要素を含める必要があります。ベストプラクティスとしては、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの権限と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

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

SimSpace Weaver リソースのタイプとその ARN のリストを確認するには、「サービス認可リファレンス」の「AWS SimSpace Weaver で定義されるリソース」を参照してください。どのアクションで各リソースの ARN を指定できるかについては、「AWS SimSpace Weaver で定義されるアクション」を参照してください。

SimSpace Weaver アイデンティティベースのポリシーの例を表示するには、「AWS SimSpace Weaver のアイデンティティベースのポリシーの例」を参照してください。

SimSpace Weaver 向けのポリシー条件キー

サービス固有のポリシー条件キーのサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

`Condition` 要素 (または `Condition` ブロック) を使用すると、ステートメントが有効な条件を指定できます。`Condition` 要素はオプションです。equal や less than などの条件演算子を使用して条件式を作成することによって、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の `Condition` 要素が指定されている場合、または 1 つの `Condition` 要素に複数のキーが指定されている場合、AWS では `AND` 論理演算子を使用してそれらを評価します。単一の条件キーに複数の値が指定されている場合、AWS では `OR` 論理演算子を使用して条件を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる許可を付与できます。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS はグローバル条件キーとサービス固有の条件キーをサポートしています。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」の「AWS グローバル条件コンテキストキー」を参照してください。

SimSpace Weaver の条件キーのリストを確認するには、サービス認可リファレンスの「AWS SimSpace Weaver の条件キー」を参照してください。どのアクションおよびリソースと条件キーを

使用できるかについては、「AWS SimSpace Weaver で定義されるアクション」を参照してください。

SimSpace Weaver アイデンティティベースのポリシーの例を表示するには、AWS SimSpace Weaver のアイデンティティベースのポリシーの例を参照してください。

SimSpace Weaver のアクセスコントロールリスト (ACL)

ACL のサポート	なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

SimSpace Weaver での属性ベースのアクセス制御 (ABAC)

ABAC のサポート (ポリシー内のタグ)	はい

属性ベースのアクセス制御 (ABAC) は、属性に基づいてアクセス許可を定義するアクセス許可戦略です。AWS では、これらの属性はタグと呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール)、および多数の AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。次に、プリンシパルのタグがアクセスを試行するリソースのタグと一致したときにオペレーションを許可するよう、ABAC ポリシーを設計します。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを制御するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素 でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値は Yes です。サービスが一部のリソースタイプに対してのみ 3 つの条件キーすべてをサポートする場合、値は Partial です。

ABAC の詳細については、『IAM ユーザーガイド』の「ABAC とは?」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM ユーザーガイド」の「属性ベースのアクセス制御 (ABAC) を使用する」を参照してください。

SimSpace Weaver での一時的な認証情報の使用

一時的な認証情報のサポート	あり

AWS のサービス には、一時的な認証情報を使用してサインインしても機能しないものがあります。一時的な認証情報で機能する AWS のサービスなどの詳細については、「IAM ユーザーガイド」の「IAM と連携する AWS のサービス」を参照してください。

ユーザー名とパスワード以外の方法で AWS Management Console にサインインする場合は、一時的な認証情報を使用していることになります。例えば、会社のシングルサインオン (SSO) リンクを使用して AWS にアクセスすると、そのプロセスは自動的に一時的な認証情報を作成します。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「ロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、AWS CLI または AWS API を使用して手動で作成できます。作成後、一時的な認証情報を使用して AWS にアクセスできるようになります。AWS は、長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することをお勧めします。詳細については、「IAM の一時的セキュリティ認証情報」を参照してください。

SimSpace Weaver のクロスサービスプリンシパル権限

フォワードアクセスセッション (FAS) をサポート	はい

IAM ユーザーまたはロールを使用して AWSでアクションを実行するユーザーは、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行してから、別のサービスの別のアクションを開始することがあります。FAS は、AWS のサービスを呼び出すプリンシパルの権限を、AWS のサービスのリクエストと合わせて使用し、ダウンストリームのサービスに対してリクエストを行います。FAS リクエストは、サービスが、完了するために他の AWS のサービス またはリソースとのやりとりを必要とするリクエストを受け取ったときにのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FASリクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

SimSpace Weaver のサービスロール

サービスロールに対するサポート	あり

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービス にアクセス許可を委任するロールの作成」を参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、SimSpace Weaver の機能が破損する可能性があります。SimSpace Weaver が指示する場合以外は、サービスロールを編集しないでください。

SimSpace Weaver アプリケーション SDK スクリプトは、AWS CloudFormation テンプレートを使用してシミュレーションをサポートするリソースを他の AWS サービス内に作成します。これらのリソースの 1 つはシミュレーションのアプリケーションロールです。 は、ログデータを CloudWatch Logs に書き込むなど、AWS アカウントユーザーに代わって でアクションを実行するためのアプリケーションロールをSimSpace Weaver引き受けます。アプリケーションロールの詳細については、「SimSpace Weaver が作成するアクセス許可」を参照してください。

SimSpace Weaver のサービスにリンクされたロール

サービスにリンクされたロールのサポート	いいえ

サービスにリンクされたロールは、AWS のサービスにリンクされているサービスロールの一種です。サービスがロールを引き受け、ユーザーに代わってアクションを実行できるようになります。サービスにリンクされたロールは、AWS アカウント に表示され、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの許可を表示できますが、編集はできません。

サービスにリンクされたロールの作成または管理の詳細については、「IAM と提携する AWS のサービス」を参照してください。表の中から、[サービスにリンクされたロール] (サービスにリンクされたロール) 列に Yes と記載されたサービスを見つけます。サービスリンクロールに関するドキュメントをサービスで表示するには、「はい」 リンクを選択します。

AWS SimSpace Weaver のアイデンティティベースのポリシーの例

デフォルトでは、 ユーザーおよびロールには、SimSpace Weaver リソースを作成または変更する権限はありません。また、AWS Management Console、AWS Command Line InterfaceAWS CLI、または AWS API を使用してタスクを実行することもできません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者がロールに IAM ポリシーを追加すると、ユーザーはロールを引き受けることができます。

これらサンプルの JSON ポリシードキュメントを使用して、IAM アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

SimSpace Weaver が定義するアクションとリソースタイプ (リソースタイプごとの ARN のフォーマットを含む) の詳細については、サービス認証リファレンスの「AWS SimSpace Weaver のアクション、リソース、および条件キー」を参照してください。

トピック

- ポリシーのベストプラクティス
- SimSpace Weaver コンソールを使用する
- 自分の権限の表示をユーザーに許可する
- ユーザーにシミュレーションの作成と実行を許可する

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが SimSpace Weaver リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、AWS アカウントに料金が発生する可能性があります。アイデンティティベースのポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください。

- AWS マネージドポリシーを使用して開始し、最小特権の権限に移行する – ユーザーとワークロードへの権限の付与を開始するには、多くの一般的なユースケースのために権限を付与する AWS マネージドポリシーを使用します。これらは AWS アカウントで使用できます。ユースケースに応じた AWS カスタマー管理ポリシーを定義することで、許可をさらに減らすことをお勧めします。詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する - IAM ポリシーで許可を設定するときは、タスクの実行に必要な許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義

します。これは、最小特権権限とも呼ばれています。IAM を使用して許可を適用する方法の詳細については、「IAM ユーザーガイド」の「IAM でのポリシーと権限」を参照してください。

- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。また、AWS のサービス など特定の AWS CloudFormation を介して使用する場合、条件を使用してサービスアクションへのアクセスを許可することもできます。詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素: 条件」を参照してください。
- IAM アクセスアナライザーを使用して IAM ポリシーを検証し、安全で機能的な許可を確保する - IAM アクセスアナライザーは、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、「IAM ユーザーガイド」の「IAM アクセスアナライザーによるポリシーの検証」を参照してください。
- 多要素認証 (MFA) を要求する - AWS アカウント で IAM ユーザーまたはルートユーザーを要求するシナリオがある場合は、セキュリティを強化するために MFA をオンにします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「MFA 保護 API アクセスの設定」を参照してください。

IAM でのベストプラクティスの詳細については、「IAM ユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

SimSpace Weaver コンソールを使用する

AWS SimSpace Weaver コンソールにアクセスするには、一連の最小限のアクセス許可が必要です。これらのアクセス許可により、AWS アカウント の SimSpace Weaver リソースの詳細をリストおよび表示できます。最小限の必要なアクセス許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しなくなります。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソール権限を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションへのアクセスのみを許可します。

ユーザーとロールが引き続き SimSpace Weaver コンソールを使用できるようにするには、エンティティに SimSpace Weaver `ConsoleAccess` または `ReadOnly` AWS 管理ポリシーもアタッチしま

す。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を、IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、AWS CLI または AWS API を使用してプログラム的に、このアクションを完了するアクセス許可が含まれます。

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

ユーザーにシミュレーションの作成と実行を許可する

この IAM ポリシーの例には、SimSpace Weaver でのシミュレーションの作成と実行に必要な基本のアクセス許可が含まれます。

```
{
    "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"
                }
            }
        }
    ]
}
```

SimSpace Weaver が作成するアクセス許可

SimSpace Weaver プロジェクトを作成すると、サービスは `weaver-`*`project-name`*`-app-role` の名前と IAM 信頼ポリシーを使用して AWS Identity and Access Management (IAM) ロールを作成します。信頼ポリシーでは、SimSpace Weaver はロールを引き受けてユーザーに代わって操作を実行することができます。

アプリケーションロールのアクセス許可ポリシー

シミュレーションアプリケーションのロールは以下のアクセス許可ポリシーを使用します。

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

アプリケーションロール信頼ポリシー

SimSpace Weaver は信頼関係を信頼ポリシーとしてシミュレーションアプリケーションのロールに追加します。SimSpace Weaver は以下の例のように、シミュレーションごとに信頼ポリシーを作成します。

```
{
  "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**
>
> この例では、アカウント番号は 111122223333 で、シミュレーション名は MySimName です。これらの値は信頼ポリシーによって異なります。

サービス間の混乱した代理の防止

「混乱した代理」問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より権限のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。AWS では、サービス間でのなりすましが、混乱した代理問題を生じさせる可能性があります。サービス間でのなりすましは、あるサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスが操作され、それ自身のアクセス許可を使用して、本来アクセス許可が付与されるべきではない方法で別の顧客のリソースに対して働きかけることがあります。これを防ぐため、AWS では、アカウント内のリソースへのアクセス許可が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールを提供しています。

リソースポリシーで aws:SourceArn および aws:SourceAccount のグローバル条件コンテキストキーを使用して、AWS SimSpace Weaver が別のサービスに付与するアクセス許可をそのリソースに制限することをお勧めします。aws:SourceArn 値が、Amazon S3 バケットの Amazon リソースネーム (ARN) などのアカウント ID が含まれていない場合、アクセス許可を制限するため、両方のグローバル条件コンテキストキーを使用する必要があります。同じポリシーステートメントでこれらのグローバル条件コンテキストキーの両方を使用し、アカウント ID にaws:SourceArn の値が含まれていない場合、aws:SourceAccount 値と aws:SourceArn 値の中のアカウントには、同じアカウント ID を使用する必要があります。クロスサービスのアクセスにリソースを 1 つだけ関連付けたい場合は、aws:SourceArn を使用します。クロスサービスが使用できるように、アカウント内の任意のリソースを関連づける場合は、aws:SourceAccount を使用します。

aws:SourceArn の価は、拡張の ARN を使用する必要があります。

混乱した代理問題から保護するための最も効果的な方法は、リソースの完全な ARN を指定して aws:SourceArn グローバル条件コンテキストキーを使用することです。拡張の完全な ARN が不明な場合や、複数の拡張を指定する場合には、ARN の不明な部分にワイルドカード (*) を含む aws:SourceArn グローバルコンテキスト条件キーを使用します。例えば、「arn:aws:*simspaceweaver*:*:*111122223333*:*」と入力します。

以下の例では、SimSpace Weaver で aws:SourceArn および aws:SourceAccount グローバル条件コンテキストキーを使用して、混乱した代理問題を回避する方法を示します。このポリシーは、リクエストが指定されたソースアカウントから送信され、指定された ARN が提供されている場合にのみ SimSpace Weaver にロールの引き受けを許可します。この場合、SimSpace Weaver がシミュレーションからのリクエストロールを引き受けることができるのはリクエスト元のアカウント (*111122223333*) と指定されたリージョン (*us-west-2*) 内のみです。

```
{
```

```
  "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/*"
        }
      }
    }
  ]
}
```

このポリシーの記述のより安全な方法は、以下の例に示すように、aws:SourceArn にシミュレーション名を含めることです。これにより、ポリシーが MyProjectSimulation_22-10-04_22_10_15 という名前のシミュレーションに制限されます。

```
{
  "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"
        }
      }
    }
  ]
}
```

aws:SourceArn がアカウント番号を明示的に含める場合は、以下の簡略化されたポリシーのように、aws:SourceAccount の Condition 要素テストを省略できます (詳細については 「IAM ユーザーガイド」を参照してください)。

```
{
  "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"
        }
      }
    }
  ]
}
```

AWS SimSpace Weaver ID とアクセスのトラブルシューティング

以下の情報は、SimSpace Weaver と IAM の使用に伴って発生する可能性がある一般的な問題の診断や修復に役立ちます。

トピック

- SimSpace Weaver でアクションを実行する権限がない
- iam を実行する権限がありません。PassRole

- アクセスキーを表示したい
- 管理者として SimSpace Weaver へのアクセスを他のユーザーに許可したい
- 自分の AWS アカウント 以外のユーザーに SimSpace Weaver リソースへのアクセスを許可したい

SimSpace Weaver でアクションを実行する権限がない

AWS Management Console から、アクションを実行する認可がないと通知された場合、管理者に問い合わせ、サポートを依頼する必要があります。管理者とは、ユーザーにユーザー名とパスワードを提供した人です。

以下のエラー例は、mateojackson IAM ユーザーがコンソールを使用して架空の *my-example-widget* リソースに関する詳細情報を表示しようとしているが、架空の simspaceweaver:*GetWidget* アクセス許可がないという場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 simspaceweaver:GetWidget on resource: my-example-widget
```

この場合、Mateo は、simspaceweaver:*GetWidget* アクションを使用して *my-example-widget* リソースにアクセスできるように、ポリシーの更新を管理者に依頼します。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して SimSpace Weaver にロールを渡すことができるようにする必要があります。

一部の AWS のサービスでは、新しいサービスロールやサービスリンクロールを作成せずに、既存のロールをサービスに渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して SimSpace Weaver でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。Mary には、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新して Mary に iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、AWS 管理者に問い合わせてください。サインイン認証情報を提供した担当者が管理者です。

アクセスキーを表示したい

IAM ユーザーアクセスキーを作成した後は、いつでもアクセスキー ID を表示できます。ただし、シークレットアクセスキーを再表示することはできません。シークレットアクセスキーを紛失した場合は、新しいアクセスキーペアを作成する必要があります。

アクセスキーは、アクセスキー ID (例: `AKIAIOSFODNN7EXAMPLE`) とシークレットアクセスキー (例: `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`) の 2 つの部分で構成されています。ユーザー名とパスワードと同様に、リクエストを認証するために、アクセスキー ID とシークレットアクセスキーの両方を使用する必要があります。ユーザー名とパスワードと同様に、アクセスキーは安全に管理してください。

> **Important**
>
> 正規のユーザー ID を検索するためであっても、アクセスキーを第三者に提供しないでください。これを行うと、AWS アカウント への永続的なアクセス権が第三者に付与される可能性があります。

アクセスキーペアを作成する場合、アクセスキー ID とシークレットアクセスキーを安全な場所に保存するように求めるプロンプトが表示されます。このシークレットアクセスキーは、作成時にのみ使用できます。シークレットアクセスキーを紛失した場合、IAM ユーザーに新規アクセスキーを追加する必要があります。アクセスキーは最大 2 つまで持つことができます。既に 2 つある場合は、新規キーペアを作成する前に、いずれかを削除する必要があります。手順を表示するには、「IAM ユーザーガイド」の「アクセスキーの管理」を参照してください。

管理者として SimSpace Weaver へのアクセスを他のユーザーに許可したい

SimSpace Weaver へのアクセスを他のユーザーに許可するには、アクセスを必要とする人またはアプリケーションの IAM エンティティ (ユーザーまたはロール) を作成する必要があります。ユーザーは、このエンティティの認証情報を使用して AWS にアクセスします。次に、SimSpace Weaver の適切なアクセス許可を付与するポリシーを、そのエンティティにアタッチする必要があります。

すぐに開始するには、「IAM ユーザーガイド」の「IAM が委任した最初のユーザーおよびグループの作成」を参照してください。

自分の AWS アカウント 以外のユーザーに SimSpace Weaver リソースへのアクセスを許可したい

他のアカウントのユーザーや組織外のユーザーが、リソースへのアクセスに使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセス制御リスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください。

- SimSpace Weaver がこれらの機能をサポートしているかどうかを確認するには、「AWS SimSpace Weaver と IAM の連携方法」をご参照ください。
- 所有している AWS アカウント 全体のリソースへのアクセス権を付与する方法については、「IAM ユーザーガイド」の「所有している別の AWS アカウント アカウントへのアクセス権を IAM ユーザーに付与する」を参照してください。
- サードパーティーの AWS アカウント にリソースへのアクセス権を提供する方法については、「IAM ユーザーガイド」の「サードパーティーが所有する AWS アカウント にアクセス権を提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、「IAM ユーザーガイド」の「外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

AWS SimSpace Weaver でのセキュリティイベントのロギングとモニタリング

モニタリングは、SimSpace Weaver と AWSソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、AWS ソリューションのすべての部分からモニタリングデータを収集する必要があります。

AWS および SimSpace Weaver には、シミュレーションリソースをモニタリングして起こり得るインシデントに対応するためのツールがいくつか用意されています。

Amazon の ログ CloudWatch

SimSpace Weaver はログを に保存します CloudWatch。これらのログは、シミュレーション内のイベント (アプリケーションの起動や停止など) のモニタリングやデバッグに使用できます。詳細については、「SimSpace Weaver Amazon CloudWatch Logs の ログ」を参照してください。

Amazon CloudWatch アラーム

Amazon CloudWatch アラームを使用すると、指定した期間にわたって 1 つのメトリクスを監視できます。メトリクスが特定のしきい値を超えると、Amazon SNS トピックまたは AWS Auto Scaling ポリシーに通知が送信されます。 CloudWatch アラームは、状態が変化したときにトリガーされ、特定の状態ではなく、指定された期間だけ維持されます。詳細については、「Amazon SimSpace Weaverによるモニタリング CloudWatch」を参照してください。

AWS CloudTrail ログ

CloudTrail は、 のユーザー、ロール、または AWSのサービスによって実行されたアクションの記録を提供しますSimSpace Weaver。で収集された情報を使用して CloudTrail、 に対するリクエスト SimSpace Weaver、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。詳細については、「AWS CloudTrail を使用した AWS SimSpace Weaver API コールのロギング」を参照してください。

AWS SimSpace Weaver のコンプライアンス検証

SimSpace Weaver は AWS コンプライアンスプログラムの対象範囲外です。

サードパーティーの監査者は、複数の AWS コンプライアンスプログラムの一部として他の AWS のサービスのセキュリティとコンプライアンスを評価します。これらのプログラムには、SOC、PCI、FedRAMP、HIPAA などがあります。

AWS のサービス が特定のコンプライアンスプログラムの対象であるかどうかを確認するには、「コンプライアンスプログラムによる対象範囲内の AWS のサービス のサービス」を参照し、関心のあるコンプライアンスプログラムを選択してください。一般的な情報については、「AWS コンプライアンスプログラム」を参照してください。

AWS Artifact を使用して、サードパーティーの監査レポートをダウンロードできます。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

AWS のサービスを使用する際のユーザーのコンプライアンス責任は、ユーザーのデータの機密性や貴社のコンプライアンス目的、適用される法律および規制によって決まります。AWS では、コンプライアンスに役立つ次のリソースを提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド - これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境を AWS にデプロイするための手順を示します。
- 「Amazon Web Services での HIPAA のセキュリティとコンプライアンスのためのアーキテクチャ」 - このホワイトペーパーは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法を説明しています。

> **Note**
>
> すべての AWS のサービスが HIPAA 適格であるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスのリソース - このワークブックおよびガイドのコレクションは、顧客の業界と拠点に適用されるものである場合があります。
- AWS Customer Compliance Guide — コンプライアンスの観点から見た責任共有モデルを理解できます。このガイドは、AWS のサービスを保護するためのベストプラクティスを要約したものであり、複数のフレームワーク (米国標準技術研究所 (NIST)、ペイメントカード業界セキュリティ標準評議会 (PCI)、国際標準化機構 (ISO) など) にわたるセキュリティ統制へのガイダンスがまとめられています。
- 「AWS Config デベロッパーガイド」の「ルールでのリソースの評価」 - AWS Config サービスは、自社のプラクティス、業界ガイドライン、および規制に対するリソースの設定の準拠状態を評価します。
- AWS Security Hub - この AWS のサービスは、AWS 内のセキュリティ状態の包括的なビューを提供します。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。
- AWS Audit Manager - この AWS のサービス は AWS の使用状況を継続的に監査し、リスクの管理方法やコンプライアンスを業界スタンダードへの準拠を簡素化するのに役立ちます。

AWS SimSpace Weaver の耐障害性

AWS グローバルインフラストラクチャは AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。AWS リージョン は、低レイテンシー、高スループット、そして高度な冗長ネットワークで接続される物理的に独立、隔離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンを使用すると、中断することなくゾーン間で自動的にフェイルオー

バーするアプリケーションとデータベースを設計および運用できます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョン とアベイラビリティーゾーンの詳細については、「AWS グローバルインフラストラクチャ」を参照してください。

AWS SimSpace Weaver のインフラストラクチャセキュリティ

マネージドサービスである AWS SimSpace Weaver は AWS グローバルネットワークセキュリティで保護されています。AWSセキュリティサービスと AWS によるインフラストラクチャの保護方法については、「AWS クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「セキュリティの柱 - AWS Well-Architected フレームワーク」の「インフラストラクチャ保護」を参照してください。

AWS の発行済み API コールを使用して、ネットワーク経由で SimSpace Weaver にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2、できれば TLS 1.3 が必要です。
- DHE (Ephemeral Diffie-Hellman) や ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) などの Perfect Forward Secrecy (PFS) を使用した暗号スイート。これらのモードは Java 7 以降など、最近のほとんどのシステムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

ネットワーク接続セキュリティモデル

シミュレーションは、選択した AWS リージョン内の Amazon VPC 内のコンピュートインスタンスで実行されます。Amazon VPC は AWS クラウド内の仮想化ネットワークであり、ワークロードまたは組織エンティティごとにインフラストラクチャを分離します。Amazon VPC 内のコンピュートインスタンス間の通信は AWS ネットワーク内にとどまり、インターネットを経由しません。一部の内部サービス通信はインターネットを経由し、暗号化されます。同じ AWS リージョンで実行しているすべてのお客様のシミュレーションは、同じ Amazon VPC を共有します。異なるお客様のシミュレーションは、同じ Amazon VPC 内の別々のコンピューティングインスタンスを使用します。

シミュレーションクライアントと SimSpace Weaver でのシミュレーション間の通信は、インターネットを経由します。SimSpace Weaver は これらの接続を処理しません。クライアント接続を保護するのはお客様の責任です。

SimSpace Weaver サービスへの接続はインターネットを経由し、暗号化されます。これには AWS Management Console、AWS Command Line Interface、(AWS CLI)、AWS Software Development Kit (SDK)、SimSpace Weaver アプリケーション SDK を使用した接続が含まれます。

AWS SimSpace Weaver での設定と脆弱性の分析

設定および IT 制御は、AWS とお客様の間で共有される責任です。詳細については、「AWS責任分担モデル」「」「」を参照してください。AWS はコンピュートインスタンス上のオペレーティングシステムへのパッチ適用、ファイアウォールの構成、AWS インフラストラクチャのディザスタリカバリなど、基盤となるインフラストラクチャの基本的なセキュリティタスクを処理します。これらの手順は適切なサードパーティーによって確認され、認証されています。詳細については、「セキュリティ、アイデンティティ、コンプライアンスのベストプラクティス」を参照してください。

以下のシミュレーションソフトウェアのセキュリティに関する責任はユーザーにあります。

- アップデートやセキュリティパッチを含むアプリケーションコードを管理します。
- シミュレーションクライアントと接続先のアプリケーション間の通信を認証して暗号化します。
- AWS SDK SimSpace Weaver アプリケーション SDK を含む最新の SDK バージョンを使用するようにシミュレーションを更新します。

> **Note**
>
> SimSpace Weaver は実行中のシミュレーション内のアプリケーションの更新をサポートしていません。アプリケーションを更新する必要がある場合、シミュレーションを停止して削除し、更新されたアプリケーションコードを使用して新しいシミュレーションを作成する必要があります。シミュレーションを再作成する必要がある場合に復元できるように、シミュレーションの状態を外部データストアに保存することをお勧めします。

SimSpace Weaver のセキュリティに関するベストプラクティス

このセクションでは、SimSpace Weaver に特有のセキュリティに関するベストプラクティスについて説明します。AWS でのセキュリティに関するベストプラクティスの詳細については、「セキュリティ、アイデンティティ、コンプライアンスのベストプラクティス」を参照してください。

トピック

- アプリケーションとクライアント間の通信を暗号化します。
- シミュレーション状態の定期的なバックアップ
- アプリケーションと SDK の維持

アプリケーションとクライアント間の通信を暗号化します。

SimSpace Weaver はアプリケーションとクライアント間の通信を管理しません。クライアントセッションには何らかの認証と暗号化を実装する必要があります。

シミュレーション状態の定期的なバックアップ

SimSpace Weaver はシミュレーション状態を保存しません。(API コール、コンソールオプション、またはシステムクラッシュの結果として) 停止したシミュレーション状態は保存されず、元の状態に戻す方法もありません。停止したシミュレーションは再開できません。再起動と同等の処理を実行する唯一の方法は、同じ設定とデータを使用してシミュレーションを再作成することです。シミュレーション状態のバックアップを使用して、新しいシミュレーションを初期化できます。AWS には、シミュレーション状態を保存できる、信頼性が高く利用可能なクラウドストレージとデータベースサービスがあります。

アプリケーションと SDK の維持

アプリケーション、AWS Software Development Kit (SDK) のローカルインストール、および SimSpace Weaver アプリケーション SDK を維持します。AWS SDK の新しいバージョンをダウンロードしてインストールできます。新しいバージョンの SimSpace Weaver アプリケーション SDK を非本番稼働のアプリケーションビルドでテストして、アプリケーションが期待どおりに動作し続けることを確認します。実行中のシミュレーションではアプリケーションを更新できません。アプリケーションの更新方法:

1. アプリケーションコードをローカル (またはテスト環境) で更新してテストします。
2. シミュレーション状態の変更を停止して保存します (必要な場合)。

3. シミュレーションを停止します (一度停止すると、再開することはできません)。
4. シミュレーションを削除します (停止したシミュレーションのうち、削除されていないものはサービス制限にカウントされます)。
5. 同じ設定と更新したアプリケーションコードでシミュレーションを再作成します。
6. 保存した状態データを使用してシミュレーションを初期化します (使用可能な場合)。
7. 新しいシミュレーションを開始します。

> **Note**
>
> 同じ設定で作成された新しいシミュレーションは、古いシミュレーションとは別のものです。新しいシミュレーション ID があり、Amazon の新しいログストリームにログを送信します CloudWatch。

SimSpace Weaver でのログ記録とモニタリング

モニタリングは、SimSpace Weaver とその他 AWS ソリューションの信頼性、可用性、およびパフォーマンスの維持における重要な要素です。AWS は、SimSpace Weaver をモニタリングし、問題が発生した場合には報告を行い、必要に応じて自動アクションを実行するために以下のモニタリングツールを提供しています。

- Amazon CloudWatch は、 AWS リソースと AWSで実行しているアプリケーションをリアルタイムでモニタリングします。メトリクスを収集および追跡し、カスタマイズされたダッシュボードを作成し、指定されたメトリックが指定したしきい値に達したときに通知またはアクションを実行するアラームを設定できます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。
- Amazon CloudWatch Logs を使用すると、SimSpace Weaverワーカー、、およびその他のソースからのログデータをモニタリング、保存 CloudTrail、およびアクセスできます。 CloudWatch Logs はログデータ内の情報をモニタリングし、特定のしきい値に達したときに通知できます。高い耐久性を備えたストレージにログデータをアーカイブすることもできます。詳細については、「Amazon CloudWatch Logs ユーザーガイド」を参照してください。
- AWS CloudTrail は、AWS アカウント により、またはそのアカウントに代わって行われた API コールや関連イベントを取得し、指定した Amazon S3 バケットにログファイルを送信します。AWS を呼び出したユーザーとアカウント、呼び出し元の IP アドレス、および呼び出しの発生日時を特定できます。詳細については、「AWS CloudTrailユーザーガイド」を参照してください。

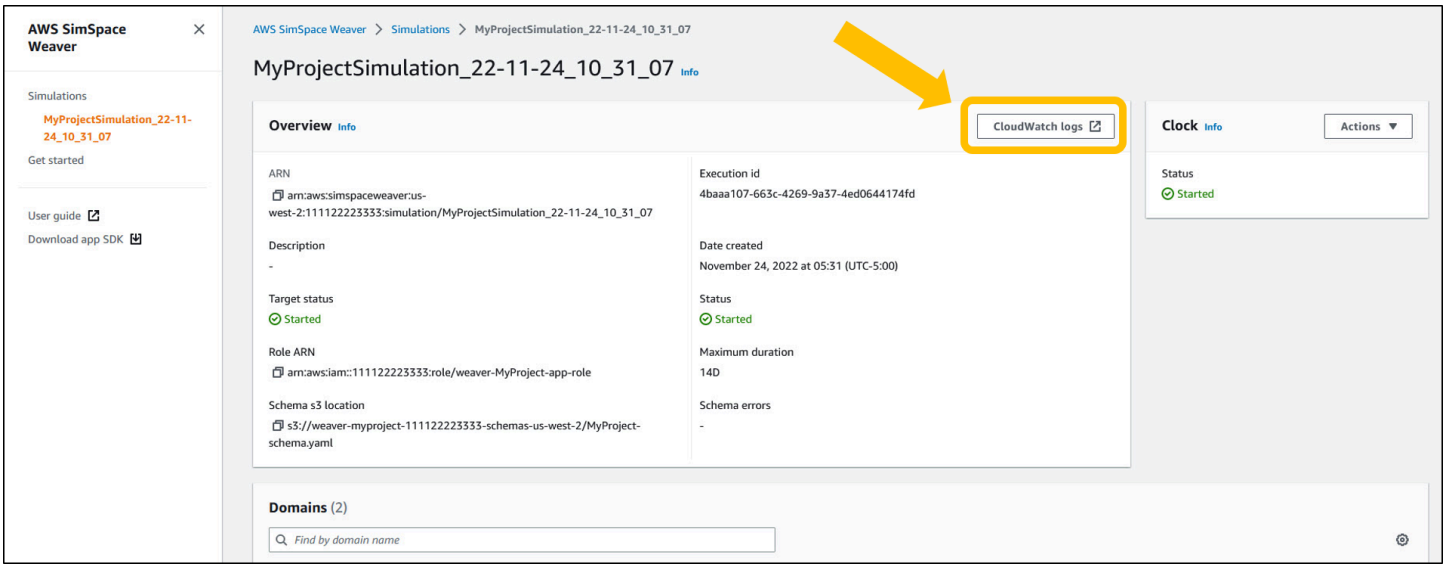
トピック

SimSpace Weaver Amazon CloudWatch Logs の ログ

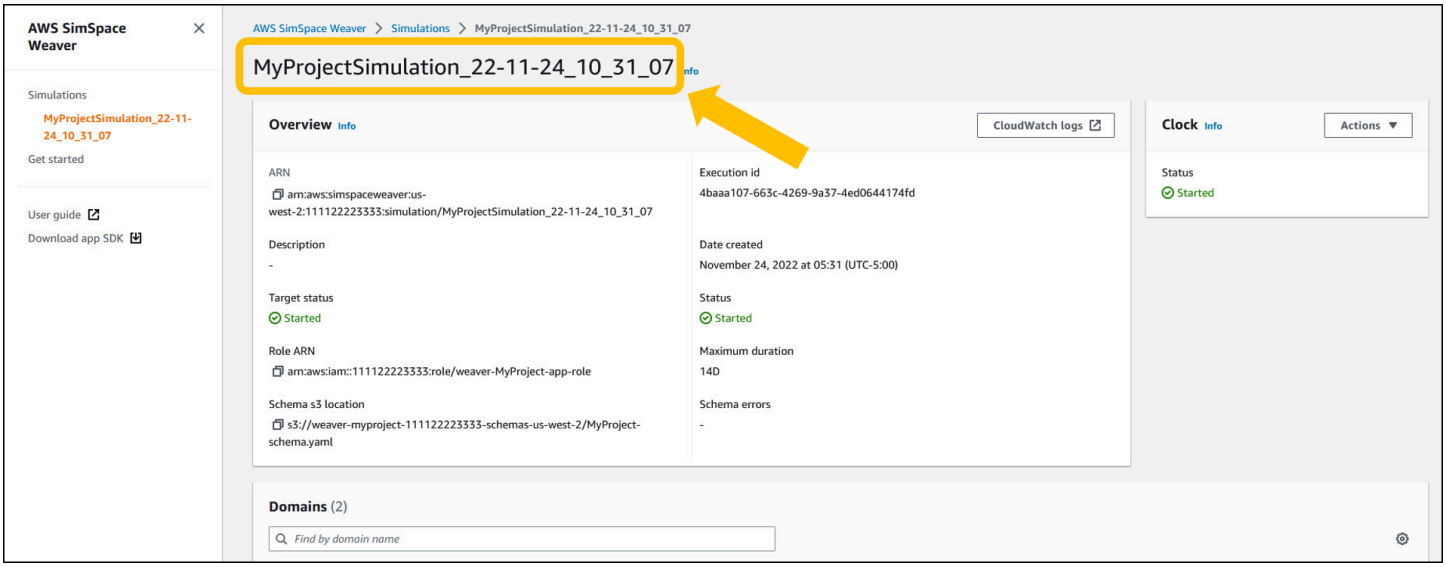
SimSpace Weaver ログのアクセス

SimSpace Weaver シミュレーションから生成されたすべてのログは Amazon CloudWatch Logs に保存されます。ログにアクセスするには、 SimSpace Weaverコンソールのシミュレーションの概要ペ

インのCloudWatch ログボタンを使用できます。これにより、その特定のシミュレーションのログに直接移動できます。



CloudWatch コンソールからログにアクセスすることもできます。ログを検索するには、シミュレーションの名前が必要です。



SimSpace Weaver ログ

SimSpace Weaver は、アプリケーションからのシミュレーション管理メッセージとコンソール出力を Amazon CloudWatch Logs に書き込みます。ログの使用の詳細については、「Amazon Logs ユーザーガイド」の「ロググループとログストリームの使用」を参照してください。 CloudWatch

作成した各シミュレーションには、 CloudWatch Logs に独自のロググループがあります。ロググループの名前は、シミュレーションスキーマで指定されます。以下の

スキーマスニペットでは、log_destination_service の値は logs です。つまり、log_destination_resource_name の値はロググループの名前です。この場合、ロググループは MySimulationLogs です。

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

DescribeSimulation API を使用して、シミュレーションを開始した後でシミュレーション用のロググループの名前を検索することもできます。

> **Important**
>
> AWS Command Line Interface (AWS CLI) で、AWS IAM Identity Center または名前の付いたプロファイルを使用する場合は、SimSpace Weaver アプリケーション SDK バージョン 1.12.1 以降を使用する必要があります。最新バージョンは 1.16.0 です。SimSpace Weaver バージョンの詳細については、「SimSpace Weaver バージョン」を参照してください。SimSpace Weaver アプリケーション SDK スクリプトは AWS CLI を使用します。IAM Identity Center を使用する場合は、AWS CLI の IAM Identity Center プロファイルを default プロファイルにコピーするか、--profile *cli-profile-name* パラメータを使用して SimSpace Weaver アプリケーション SDK スクリプトに IAM Identity Center プロファイルの名前を指定できます。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS IAM Identity Center を使用するように AWS CLI を設定する」および「AWS Command Line Interface ユーザーガイド」の「設定と認証情報ファイルの設定」を参照してください。

Docker

```
project-folder\tools\windows\weaver-project-name-cli.bat describe-simulation --
simulation simulation-name
```

WSL

> **Important**
>
> 便宜上、これらの指示を使用します。これらは Windows Subsystem for Linux (WSL) で使用するためのもので、サポートされていません。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。

```
project-folder/tools/linux/weaver-project-name-cli.sh describe-simulation --
simulation simulation-name
```

以下の例は、ロギング設定を説明する、DescribeSimulation から出力の一部を示しています。ロググループの名前は LogGroupArn の末尾に表示されます。

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

各シミュレーションロググループには、いくつかのログストリームが含まれます。

- 管理ログストリーム — SimSpace Weaver サービスによって生成されるシミュレーション管理メッセージ。

  ```
  /sim/management
  ```

- エラーログストリーム — SimSpace Weaver サービスによって生成されるエラーメッセージ。このログストリームはエラーがある場合にのみ存在します。SimSpace Weaver はアプリケーション

によって書き込まれたエラーを独自のアプリケーションログストリームに保存します (以下の「ログストリーム」を参照してください)。

```
/sim/errors
```

- 空間アプリケーションログストリーム (各ワーカーの空間アプリケーションごとに 1 つ) — 空間アプリケーションによって生成されるコンソール出力。各空間アプリケーションは、独自のログストリームに書き込みます。*spatial-app-id* は、*worker-id* の末尾にあるスラッシュの後のすべての文字です。

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

- カスタムアプリケーションログストリーム (カスタムアプリケーションインスタンスごとに 1 つ) — カスタムアプリケーションによって生成されるコンソール出力。各カスタムアプリケーションインスタンスは、独自のログストリームに書き込みます。

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

- サービスアプリケーションログストリーム (サービスアプリケーションインスタンスごとに 1 つ) — サービスアプリケーションによって生成されるコンソール出力。各サービスアプリケーションは、独自のログストリームに書き込みます。*service-app-id* は、*service-app-name* の末尾にあるスラッシュの後のすべての文字です。

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

Amazon SimSpace Weaverによるモニタリング CloudWatch

Amazon SimSpace Weaverを使用して をモニタリングすることで CloudWatch、raw データを収集し、リアルタイムに近い読み取り可能なメトリクスに加工することができます。これらの統計は 15 か月間保持されるため、履歴情報にアクセスし、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値を監視するアラームを設定し、これらのしきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon CloudWatch ユーザーガイド」を参照してください。

SimSpace Weaver サービスは AWS/simspaceweaver 名前空間の以下のメトリクスをレポートします。

アカウントレベルの SimSpace Weaver メトリクス

SimSpace Weaver 名前空間には、AWS アカウントレベルでのアクティビティに関する以下のメトリクスが含まれます。

メトリクス	説明
`SimulationCount`	現在のアカウントのシミュレーションの数。 単位: カウント 次元: なし 統計: Average、Minimum、Maximum

AWS CloudTrail を使用した AWS SimSpace Weaver API コールのロギング

AWS SimSpace Weaver は、 のユーザーAWS CloudTrail、ロール、または AWSのサービスによって実行されたアクションを記録するサービスである と統合されていますSimSpace Weaver。は、 のすべての API コールをイベントSimSpace Weaverとして CloudTrail キャプチャします。キャプチャされたコールには、SimSpace Weaver コンソールのコールと、SimSpace Weaver API オペレーションへのコードのコールが含まれます。証跡を作成する場合は、 の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができますSimSpace Weaver。証跡を設定しない場合でも、 の CloudTrail コンソールで最新のイベントを表示できますEvent history。で収集された情報を使用して CloudTrail、 に対するリクエストSimSpace Weaver、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「AWS CloudTrailユーザーガイド」を参照してください。

SimSpace Weaver 内の情報 CloudTrail

CloudTrail アカウントを作成するAWS アカウントと、 は で有効になります。でアクティビティが発生するとSimSpace Weaver、そのアクティビティは の他のAWSサービス CloudTrail イベントとともに イベントに記録されますEvent history。最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、「イベント履歴 を使用した CloudTrail イベントの表示」を参照してください。

SimSpace Weaver のイベントなど、AWS アカウント のイベントの継続的な記録に対して、追跡を作成します。証跡により、 はログファイル CloudTrail を Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョン に適用されます。証跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたデータをより詳細に分析し、それに基づく対応を行うように他の AWSサービスを設定できます。詳細については、次を参照してください:

- 「証跡作成の概要」
- CloudTrail でサポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからの CloudTrail ログファイルの受信

すべての SimSpace Weaver アクションは によってログに記録 CloudTrail され、 AWS SimSpace Weaver API リファレンス に記載されています。例えば、 および `DeleteSimulation`アクションを呼び出す`DescribeSimulation`と`ListSimulations`、 CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するために役立ちます。

- リクエストが、ルート認証情報と AWS Identity and Access Management (IAM) ユーザー認証情報のどちらを使用して送信されたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが、別の AWS サービスによって送信されたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

SimSpace Weaver ログファイルエントリについて

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには、1 つ以上のログエントリが含まれます。イベントは任意の送信元からの単一のリクエストを表し、アクションの日時、リクエストパラメータ、その他の details. CloudTrail log ファイルなど、リクエストされたアクションに関する情報が含まれます。 は、パブ

リック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次の例は、 ListSimulationsアクションを示す CloudTrail ログエントリを示しています。

```
{
    "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 エンドポイントとクォータ

次の表は、 SimSpace Weaver のサービスエンドポイントとサービスクォータを示しています。Service Quotas (制限とも呼ばれます) は、AWS アカウント のサービスリソースまたはオペレーションの最大数です。詳細については、「AWS 全般のリファレンス」の「AWS Service Quotas」を参照してください。

サービスエンドポイント

リージョン名	リージョン	エンドポイント	プロトコル
米国東部 (バージニア北部)	us-east-1	simspaceweaver.us-east-1.amazonaws.com	HTTPS
米国東部 (オハイオ)	us-east-2	simspaceweaver.us-east-2.amazonaws.com	HTTPS
米国西部 (オレゴン)	us-west-2	simspaceweaver.us-west-2.amazonaws.com	HTTPS
アジアパシフィック (シンガポール)	ap-southeast-1	simspaceweaver.ap-southeast-1.amazonaws.com	HTTPS
アジアパシフィック (シドニー)	ap-southeast-2	simspaceweaver.ap-southeast-2.amazonaws.com	HTTPS
欧州 (ストックホルム)	eu-north-1	simspaceweaver.eu-north-1.amazonaws.com	HTTPS

リージョン名	リージョン	エンドポイント	プロトコル
欧州 (フランクフルト)	eu-central-1	simspaceweaver.eu-central-1.amazonaws.com	HTTPS
欧州 (アイルランド)	eu-west-1	simspaceweaver.eu-west-1.amazonaws.com	HTTPS
AWS GovCloud （米国東部）	us-gov-east-1	simspaceweaver.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud （米国西部）	us-gov-west-1	simspaceweaver.us-gov-west-1.amazonaws.com	HTTPS

Service Quotas

名前	デフォルト	引き上げ可能	説明
各アプリケーションのコンピュートリソースユニット	サポートされている各リージョン: 4	いいえ	各アプリケーションに割り当てることができるコンピュートリソースユニットの最大数。
各ワーカーのコンピュートリソースユニット	サポートされている各リージョン: 17	いいえ	各ワーカーが利用できるコンピュートリソースユニットの数。

名前	デフォルト	引き上げ可能	説明
各エンティティのデータフィールド	サポートされている各リージョン: 7	いいえ	エンティティが保持できるデータ (非インデックス) フィールドの最大数。
パーティション内のエンティティ	サポートされている各リージョン: 8,192	いいえ	1 パーティションのエンティティの最大数。
エンティティのデータフィールドサイズ	サポートされている各リージョン: 1,024 バイト	いいえ	エンティティのデータ (非インデックス) フィールドの最大サイズ。
ワーカー間のエンティティ転送	サポートされている各リージョン: 25	いいえ	各パーティションおよび各ティックにおける、ワーカー間のエンティティ転送の最大数。
同じワーカーでのエンティティ転送	サポートされている各リージョン: 500	いいえ	同じワーカーでのパーティションごと、およびティックごとのエンティティ転送の最大数。
各エンティティのインデックスフィールド	サポートされている各リージョン: 1	いいえ	エンティティが保持できるインデックスフィールドの最大数。

名前	デフォルト	引き上げ可能	説明
シミュレーションの最大実行期間の最大日数	サポートされている各リージョン: 14	いいえ	シミュレーションの最大実行期間として指定できる最大日数。値を指定しなくても、すべてのシミュレーションには最大実行期間が設定されます。シミュレーションは、最大実行期間に達すると自動的に停止します。
各コンピュートリソースユニットのメモリ	サポートされている各リージョン: 1 GB	いいえ	コンピュートリソースユニットごとにアプリケーションが取得するランダムアクセスメモリ (RAM) の量。
各ワーカーのリモートサブスクリプション	サポートされている各リージョン: 24	いいえ	各ワーカーのリモートサブスクリプションの最大数。
シミュレーションカウント	サポートされている各リージョン: 2	はい	アカウントでターゲットステータスが STARTED のシミュレーションの最大数。最大 10 までクォータ引き上げをリクエストできます。

名前	デフォルト	引き上げ可能	説明
シミュレーション用のワーカー	サポートされている各リージョン: 2	はい	1 シミュレーションに割り当てることができるワーカーの最大数。最大 10 までクォータ引き上げをリクエストできます。
各コンピュートリソースユニットの vCPU	サポートされている各リージョン: 2	いいえ	アプリケーションが各コンピュートリソースユニットで取得する仮想化中央演算装置 (vCPU) の数。

メッセージングクォータ

以下のクォータは、 SimSpace Weaver Localおよび のアプリケーションメッセージングに適用されますAWS クラウド。

名前	デフォルト	引き上げ可能	説明
最大メッセージサイズ (MB)	サポートされている各リージョン: 256 バイト	いいえ	メッセージペイロードの最大サイズ。
最大メッセージ送信レート	サポートされている各リージョン: 128	いいえ	各アプリがティックごとに送信できるメッセージの最大数。

クロックレート

シミュレーションスキーマは、シミュレーションのクロックレート (ティックレートとも呼ばれます) を指定します。以下の表は、使用できる有効なクロックレートを指定しています。

名前	有効値	説明
クロックレート	サポートされている各リージョン：「10」、「15」、「30」、「無制限」	シミュレーションの有効なクロックレート。
クロックレート (バージョン 1.13 および 1.12)	サポートされている各リージョン: 10、15、30	シミュレーションの有効なクロックレート。

SimSpace Weaver Local の Service Quotas

以下の Service Quotas は SimSpace Weaver Local にのみ適用されます。他のすべてのクォータも SimSpace Weaver Local に適用されます。

名前	デフォルト	引き上げ可能	説明
最大パーティション数	SimSpace Weaver Local: 24	いいえ	シミュレーションのパーティションの最大数。
最大アプリケーション数	SimSpace Weaver Local: 24	いいえ	シミュレーションの (種類を問わず) アプリケーションの最大合計数。
最大ドメイン数	SimSpace Weaver Local: 24	いいえ	シミュレーションの (種類を問わず) ドメインの最大合計数。
パーティション内のエンティティ数	SimSpace Weaver Local: 4,096	いいえ	各パーティションのエンティティの最大数。

名前	デフォルト	引き上げ可能	説明
エンティティあたりのフィールド数	SimSpace Weaver Local: 8	いいえ	各エンティティのフィールドの最大数。
フィールドサイズ	SimSpace Weaver Local: 1024 バイト	いいえ	エンティティフィールドの最大サイズ。

SimSpace Weaver のトラブルシューティング

トピック

AssumeRoleAccessDenied

シミュレーションが開始されない場合、以下のエラーが表示される場合があります。

```
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
```

このエラーは、シミュレーションの AWS Identity and Access Management (IAM) ロールが以下のいずれかに当てはまる場合に表示されます。

- Amazon リソースネーム (ARN) が存在しない IAM ロールを指している。
- IAM ロールの信頼ポリシーが、新しいシミュレーション名がロールを引き受けることを許可しない。

ロールが存在することを確認します。ロールが存在する場合は、ロールの信頼ポリシーを確認します。以下の信頼ポリシーの例では、`aws:SourceArn` は名前が `MySimulation` で始まる (アカウント 111122223333 の) シミュレーションのみがロールを引き継ぐことを許可しています。

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

名前が MyOtherSimulation で始まる別のシミュレーションがロールを引き継ぐことを許可するには、以下の編集例のように信頼ポリシーを変更する必要があります。

```
{
    "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

プロジェクトの作成中に、以下のエラーが表示される場合があります。

```
An error occurred (InvalidBucketName) when calling the CreateBucket operation: The
 specified bucket is not valid.
```

このエラーを受け取ったのは、SimSpace Weaver が Amazon Simple Storage Service (Amazon S3) に渡した名前がバケット命名規則に違反していたためです (詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケット命名規則」を参照してください)。

SimSpace Weaver アプリケーション SDK の `create-project` スクリプトは、スクリプトに指定したプロジェクト名を使用してバケット名を作成します。バケット名は、以下のフォーマットを使用します。

- バージョン 1.13.x 以降
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-`*`region`*
- バージョン 1.12.x
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-app-zips-`*`region`*
  - `weaver-`*`lowercase-project-name`*`-`*`account-number`*`-schemas-`*`region`*

例えば、以下のプロジェクトプロパティがあるとします。

- プロジェクト名: `MyProject`
- AWS アカウント の数: `111122223333`
- AWS リージョン: `us-west-2`

プロジェクトには以下のバケットがあります。

- バージョン 1.13.x 以降
  - `weaver-myproject-111122223333-us-west-2`
- バージョン 1.12.x
  - `weaver-myproject-111122223333-app-zips-us-west-2`
  - `weaver-myproject-111122223333-schemas-us-west-2`

プロジェクト名は Amazon S3 の命名規則に違反していない必要があります。また、`create-project` スクリプトによって作成されたバケット名は Amazon S3 バケットの名前の長さ制限を超えない長さのプロジェクト名を使用する必要があります。

ServiceQuotaExceededException

シミュレーションを開始すると、以下のエラーが表示されることがあります。

```
An error occurred (ServiceQuotaExceededException) when calling the StartSimulation
 operation: Failed to start simulation due to: simulation quota has already been
 reached.
```

このエラーは、新しいシミュレーションを開始しようとしたときに、アカウントのターゲットのステータスが STARTED のシミュレーションの数が最大数に達している場合に表示されます。これには、実行中のシミュレーション、失敗したシミュレーション、および最大時間に達したために停止したシミュレーションが含まれます。停止または失敗したシミュレーションを削除して、新しいシミュレーションを開始できます。すべてのシミュレーションが実行中の場合は、実行中のシミュレーションを停止して削除できます。まだリクエストの上限に達していない場合は、Service Quotas の引き上げをリクエストすることもできます。詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。不要なシミュレーションをクリーンアップする方法については、「クイックスタートチュートリアル」の「ステップ 6: シミュレーションを停止してクリーンアップする」を参照してください。

TooManyBuckets

プロジェクトの作成中に、以下のエラーが表示される場合があります。

```
An error occurred (TooManyBuckets) when calling the CreateBucket operation: You have
 attempted to create more buckets than allowed.
```

Amazon Simple Storage Service (Amazon S3) では、AWS アカウントに含めることができるバケットの数に制限があります (詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケットの制約および制限」を参照してください)。

続ける前に、以下のいずれかを実行する必要があります。

- 不要な 2 つ以上の既存の Amazon S3 バケットを削除します。

- Amazon S3 の制限の引き上げをリクエストします (詳細については、「Amazon Simple Storage Service ユーザーガイド」の「バケットの制約および制限」を参照してください)。
- 別の AWS アカウントを使用します。

> **Note**
>
> SimSpace Weaver の `DeleteSimulation` API は、シミュレーションに関連付けられた Amazon S3 リソースを削除しません。不要になった場合は、シミュレーションに関連するすべてのリソースを削除することをお勧めします。シミュレーションをクリーンアップする方法については、「クイックスタートチュートリアル」の「ステップ 6: シミュレーションを停止してクリーンアップする」を参照してください。

シミュレーション開始中にアクセスが拒否される

シミュレーションを開始すると、アクセスが拒否されたこと、またはアプリケーションのアーティファクトへのアクセス中にエラーが発生したことを示すエラーメッセージが表示される場合があります。この問題は、(コンソールまたは SimSpace Weaver アプリケーション SDK スクリプトを使用して) SimSpace Weaver が作成していない Amazon S3 バケットをシミュレーションに指定した場合に発生する可能性があります。

根本原因として最も考えられるのは以下の状況です。

- このサービスには、シミュレーションスキーマで指定した 1 つ以上の Amazon S3 バケットにアクセスする権限がない - アプリケーションロールのアクセス許可ポリシー、Amazon S3 バケットポリシー、および Amazon S3 バケットアクセス許可を確認して、`simspaceweaver.amazonaws.com` がバケットにアクセスするための正しい権限があることを確認します。アプリケーションロールアクセス許可ポリシーの詳細については、「SimSpace Weaver が作成するアクセス許可」を参照してください。
- Amazon S3 バケットがシミュレーションとは異なる AWS リージョン にある場合がある - シミュレーションアーティファクト用の Amazon S3 バケットは、シミュレーションと同じ AWS リージョン にある必要があります。[Amazon S3] コンソールで、バケットが何の AWS リージョン にあるかを確認します。Amazon S3 バケットが別の AWS リージョン にある場合は、シミュレーションと同じ AWS リージョン にあるバケットを選択します。

Docker を使用時の時間に関する問題

Docker を使用していて、SimSpace Weaver アプリケーション SDK からスクリプトを実行しているときに時間に関するエラーが表示される場合は、Docker 仮想化マシンのクロックが正しくないことが原因である可能性があります。これは、コンピューターが Docker を実行していて、その後にスリープ状態または休止状態から再開した場合に発生することがあります。

試すソリューション

- Docker を再起動します。
- Windows PowerShell で時刻同期を無効にしてから再度有効にします。

```
Get-VMIntegrationService -VMName DockerDesktopVM -Name "Time Synchronization" |
 Disable-VMIntegrationService
Get-VMIntegrationService -VMName DockerDesktopVM -Name "Time Synchronization" |
 Enable-VMIntegrationService
```

PathfindingSample コンソールクライアントが接続に失敗する

クイックスタートチュートリアルおよび詳細チュートリアルで説明されている PathfindingSample シミュレーションに接続すると、コンソールクライアントから以下のエラーが表示される場合があります。このエラーは、指定した IP アドレスとポート番号を組み合わせた ViewApp へのネットワーク接続を、クライアントが開くことができないために発生します。

```
Fatal error in function nng_dial. Error code: 268435577. Error message: no link
```

AWS クラウド でのシミュレーション用

- ネットワーク接続は正しく機能していますか? 動作するはずの他の IP アドレスまたはウェブサイトに接続できることを確認します。ウェブブラウザがキャッシュからウェブサイトを読み込んでいないことを確認します。
- シミュレーションは実行中ですか? ListSimulations API を使用してシミュレーションのステータスを取得できます。詳細については、「ステップ 4: IP アドレスとポート番号を取得する」を参照してください。[SimSpace Weaver] コンソールを使用してシミュレーションのステータスを確認することもできます。
- アプリケーションは実行中ですか? DescribeApp API を使用してアプリケーションのステータスを取得できます。詳細については、「ステップ 4: IP アドレスとポート番号を取得する」を参照して

ください。[SimSpace Weaver] コンソールを使用してシミュレーションのステータスを確認することもできます。

- アプリケーションは実行中ですか? DescribeApp API を使用してアプリケーションのステータスを取得できます。詳細については、「ステップ 4: IP アドレスとポート番号を取得する」を参照してください。[SimSpace Weaver] コンソールを使用してシミュレーションのステータスを確認することもできます。
- 正しい IP アドレスとポート番号を使用しましたか? インターネット経由で接続する場合は、ViewApp の IP アドレスおよび Actual ポート番号を使用する必要があります。IP Address および Actual ポート番号は DescribeApp API 出力の EndpointInfo ブロックで確認できます。[SimSpace Weaver] コンソールを使用して、MyViewDomain 詳細ページの ViewApp で IP アドレス (URI) とポート番号 (入力ポート) を確認することもできます。
- ネットワーク接続はファイアウォールを経由していますか? ファイアウォールによって IP アドレスまたはポート番号 (あるいはその両方) への接続がブロックされている可能性があります。ファイアウォールの設定を確認するか、ファイアウォール管理者に確認してください。

ローカルシミュレーション用

- ループバックアドレス (127.0.0.1) に接続できますか? Windows に ping コマンドラインツールがある場合は、コマンドプロンプトウィンドウを開いて 127.0.0.1 に ping を実行します。Ctrl-C を押して ping を終了します。

```
ping 127.0.0.1
```

Example ping 出力

```
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:\>
```

ping でパケットが失われたと表示される場合は、他のソフトウェア (ローカルファイアウォール、セキュリティ設定、マルウェア対策プログラムなど) が接続をブロックしている可能性があります。

- アプリケーションは実行中ですか? ローカルシミュレーションはアプリケーションごとに別々のウィンドウとして実行されます。空間アプリケーションおよび ViewApp のウィンドウが開いていることを確認します。詳細については、「ローカル開発」を参照してください。
- 正しい IP アドレスとポート番号を使用しましたか? ローカルシミュレーションに接続するときには `tcp://127.0.0.1:7000` を使用する必要があります。詳細については、「ローカル開発」を参照してください。
- 接続を遮断する可能性のあるローカルセキュリティソフトウェアはありますか? セキュリティ設定、ローカルファイアウォール、またはマルウェア対策プログラムを確認して、TCP ポート7000での `127.0.0.1` への接続がブロックされていないか確認します。

AWS CLI が **`simspaceweaver`** を認識しない

AWS CLI が SimSpace Weaver を認識していないことを示すエラーが表示された場合は、以下のコマンドを実行します。

```
aws simspaceweaver help
```

以下の行で始まり、使用可能なすべての選択肢が一覧表示された場合は、AWS CLI が古いバージョンである可能性があります。

```
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:
```

AWS CLI のバージョンを確認するには、以下のコマンドを実行します。

```
aws --version
```

バージョンが 2.9.19 以前の場合は、AWS CLI を更新する必要があります。AWS CLI の現在のバージョンは 2.9.19 以降であることに注意してください。

AWS CLI を更新するには、「バージョン 2 用 AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンのインストールまたはアップデート」を参照してください。

SimSpace Weaver シミュレーションスキーマリファレンス

SimSpace Weaver は YAML ファイルを使用してシミュレーションのプロパティを設定します。このファイルはシミュレーションスキーマ (または単にスキーマ) と呼ばれます。SimSpace Weaver アプリケーション SDK に含まれるサンプルシミュレーションには、独自のシミュレーション用にコピーして編集できるスキーマが含まれています。

トピック

完全なスキーマの例

次の例は、SimSpace Weaverシミュレーションを説明する YAML形式のテキストファイルを示しています。これらの例には、プロパティのダミーの値が含まれます。ファイルのフォーマットは、ファイルに指定されている sdk_version の値によって異なります。プロパティとその有効値の詳細については、「スキーマフォーマット」を参照してください。

```
sdk_version: "1.16"
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"]
```

スキーマフォーマット

以下の例は、スキーマ構築全体を示しています。親子関係が同じであれば、スキーマの各レベルでのプロパティの順序は関係ありません。配列内の要素では順序が重要です。

```
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
```

セクション

- SDK のバージョン
- シミュレーションのプロパティ
- ワーカー
- クロック
- パーティショニング戦略
- ドメイン
- 配置の制約事項

SDK のバージョン

sdk_version セクション (必須) は、このスキーマをサポートする SimSpace Weaver アプリケーション SDK のバージョンを示します。有効な値:1.16、1.15、1.14、1.13、1.12

> **Important**
>
> sdk_version の値には、メジャーバージョン番号と最初のマイナーバージョン番号のみが含まれます。例えば、値 1.12 は、1.12.0、1.12.1、1.12.2 などのすべてのバージョン 1.12.x を指定します。

```
sdk_version: "1.16"
```

シミュレーションのプロパティ

simulation_properties セクション (必須) は、シミュレーションのさまざまなプロパティを指定します。このセクションを使用してロギングを設定し、既定のコンテナイメージを指定します。このセクションは、ロギングを設定していない場合や、デフォルトのコンテナイメージを指定する場合にも必須です。

```
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"
```

プロパティ

log_destination_resource_name

SimSpace Weaver がログを書き込むリソースを指定します。

必須: いいえ。このプロパティが含まれていない場合、SimSpace Weaver にはシミュレーションのログは書き込まれません。

タイプ: 文字列

有効値:

- CloudWatch Logs ロググループの名前 (例: `MySimulationLogs`)
- CloudWatch Logs ロググループの Amazon リソースネーム (ARN) (例: `arn:aws:logs:us-west-2:111122223333:log-group/MySimulationLogs`)

> **Note**
>
> SimSpace Weaver はシミュレーションと同じアカウントおよび AWS リージョン 内のログ送信先のみをサポートします。

`log_destination_service`

ARN ではない `logging_destination resource_name` を指定した場合のロギング先リソースのタイプを示します。

必須: `log_destination_resource_name` が指定されていて ARN ではない場合は、このプロパティを指定する必要があります。`log_destination_resource_name` が指定されていない場合、または ARN の場合、このプロパティは指定できません。

タイプ: 文字列

有効値:

- `logs`: ログ送信先リソースはロググループです。

`default_entity_index_key_type`

シミュレーションエンティティのインデックスキーフィールドのデータタイプを指定します。

必須: はい

タイプ: 文字列

有効値: `Vector3<f32>`

`default_image`

シミュレーションのデフォルトコンテナイメージを指定します (バージョン `1.13` および `1.12` ではサポートされていません)。このプロパティを指定すると、`image` を指定していないドメインは `default_image` を使用します。

必須: いいえ

タイプ: 文字列

有効値:

- Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリの URI (例: `111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest`)

ワーカー

`workers` セクション (必須) は、ワーカーグループ (ワーカーのグループ) の設定を指定します。SimSpace Weaver はこの情報を `placement_constraints` と一緒に使用して、シミュレーションの基盤となるインフラストラクチャを構成します。現在、1 ワーカーグループのみがサポートされています。

ワーカーグループのプロパティを指定するには、*`worker-group-name`* を任意の名前に置き換えます。名前は 3～64 文字で、A～Z、a～z、0～9、_ - (ハイフン) を含むことができます。名前の後にワーカーグループのプロパティを指定します。

```
workers:
  worker-group-name:
    type: "sim.c5.24xlarge"
    desired: number-of-workers
```

プロパティ

`type`

ワーカータイプを指定します。

必須: はい

タイプ: 文字列

有効値: `sim.c5.24xlarge`

`desired`

このワーカーグループに必要なワーカー数を指定します。

必須: はい

タイプ: 整数

有効値: 1-3。シミュレーションのワーカー数の Service Quotas (制限) によって、このプロパティの最大値が決まります。例えば、Service Quotas が 2 の場合、このプロパティの最大値は 2 です。Service Quotas の引き上げをリクエストすることもできます。詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。

クロック

clock セクション (必須) は、シミュレーションクロックのプロパティを指定します。

```
clock:
  tick_rate: tick-rate
```

プロパティ

tick_rate

クロックがアプリケーションに公開する 1 秒あたりのティック数を指定します。

必須: はい

タイプ:

- バージョン *1.14* および *1.15*: 文字列
- バージョン *1.13* および *1.12*: 整数

有効値:

- バージョン *1.14* および *1.15*: "10" | "15" | "30" | "unlimited"
  - "unlimited": クロックは、すべてのアプリケーションが現在のティックのコミット操作を完了するとすぐに次のティックを送信します。
- バージョン *1.13* および *1.12*: 10 | 15 | 30

パーティショニング戦略

partitioning_strategies セクション (必須) は、空間ドメインのパーティションの構成を指定します。

> **Note**
>
> SimSpace Weaver は 1 つのパーティショニング戦略のみをサポートします。

パーティショニング戦略のプロパティを指定するには、 を任意の名前*partitioning-strategy-name*に置き換えます。名前は 3～64 文字で、A～Z、a～z、0～9、_ - (ハイフン) を含むことができます。名前の後にパーティショニング戦略のプロパティを指定します。

```
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
```

プロパティ

`topology`

このパーティショニング戦略のトポロジ (パーティション配置スキーマ) を指定します。

必須: はい

タイプ: 文字列

有効値: `"Grid"`

`aabb_bounds`

シミュレーションの主軸に沿ったバウンディングボックス (AABB) の境界を指定します。範囲を、各軸 (x および y) の最小値と最大値を (この順序で) 記述する 2 要素の配列として指定します。

必須: 条件的。トポロジが `"Grid"` に設定されている場合、このプロパティは必須 (指定のみ可能) です。

タイプ: `Float` 配列 (各軸)

有効値: `-3.4028235e38-3.4028235e38`

`grid_placement_groups`

グリッドトポロジの各軸 (X および Y) に沿った配置グループの数を指定します。配置グループは、空間的に隣接する (同じドメイン内の) パーティションの集まりです。

必須: 条件的。トポロジが `"Grid"` に設定されている場合、このプロパティは必須 (指定のみ可能) です。配置グループ設定を指定しない場合、SimSpace Weaver が自動的に計算します。配置グループ設定なしでパーティショニング戦略を使用するドメインでは、`grid_partition` を指定する必要があります (「空間ドメインのパーティショニング戦略」を参照してください)。

タイプ: 整数 (各軸)

有効値: 1-20。x * y は必要なワーカー数と等しくすることをお勧めします。それ以外の場合は、SimSpace Weaver は利用可能なワーカー全体で配置グループのバランスを取ろうとします。

ドメイン

`domains` セクション (必須) は、各ドメインのプロパティを指定します。すべてのシミュレーションには、空間領域用のセクションが少なくとも 1 つ必要です。追加ドメイン用に複数のセクションを作成できます。各タイプのドメインには、独自の構成フォーマットがあります。

> **Important**
>
> バージョン `1.13` および `1.12` は複数の空間ドメインをサポートしていません。

> **Important**
>
> SimSpace Weaver はシミュレーションごとに最大 5 つのドメインをサポートします。これには、すべての空間ドメイン、カスタムドメイン、およびサービスドメインが含まれます。

```
domains:
  domain-name:
    domain-configuration
  domain-name:
```

```
    domain-configuration
  ...
```

ドメイン設定

- 空間ドメイン設定
- カスタムドメイン設定
- サービスドメイン設定

空間ドメイン設定

空間ドメインのプロパティを指定するには、 を任意の名前*spatial-domain-name*に置き換えます。名前は 3～64 文字で、A～Z、a～z、0～9、_ - (ハイフン) を含むことができます。名前の後に空間ドメインのプロパティを指定します。

```
  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"
```

空間ドメインのパーティショニング戦略

launch_apps_by_partitioning_strategy セクション (必須) は、シミュレーション空間のパーティショニング戦略と次元 (パーティション数) を指定します。

```
    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
```

プロパティ

`partitioning_strategy`

この空間ドメインのパーティショニング戦略を指定します。

必須: はい

タイプ: 文字列

有効値: このプロパティの値は、`partitioning_strategies` セクションで定義されているパーティショニング戦略の名前と一致する必要があります。詳細については、「パーティショニング戦略」を参照してください。

`grid_partition`

グリッドトポロジーの各軸 (x および y) に沿ったパーティションの数を指定します。これらの次元は、このドメインのシミュレーションスペースの合計を表します。

必須: 条件的。このプロパティは、トポロジーが `"Grid"` に設定されている場合にのみ指定できます。このプロパティは、このドメインに指定されているパーティショニング戦略の `grid_placement_groups` プロパティによって異なります。

- このドメインのパーティショニング戦略で `grid_placement_groups` 設定が指定されていない場合、このプロパティは必須です。
- `grid_placement_groups` 設定があっても `grid_partition` を指定しない場合、SimSpace Weaver は `grid_placment_groups` 設定と同じ次元を使用します。
- `grid_placement_groups` と `grid_partition` の両方を指定する場合、`grid_partition` の次元は`grid_placement_groups`の次元の倍数である必要があります (例えば、`grid_placement_groups` の次元が 2x2 の場合、`grid_partition` で有効な次元は 2x2、4x4、6x6、8x8、10x10 です)。

タイプ: 整数 (各軸)

有効値: 1-20

空間アプリケーションの設定

`app_config` セクション (必須) は、このドメイン内のアプリケーションのパッケージ、起動設定、およびリソース要件を指定します。

```
    app_config:
      package: "app-package-s3-uri"
      launch_command: ["app-launch-command", "parameter1", ...]
      required_resource_units:
        compute: app-resource-units
```

プロパティ

`package`

アプリケーションの実行ファイル/バイナリを含むパッケージ (zip ファイル) を指定します。パッケージは Amazon S3 バケットに保存されている必要があります。zip ファイルフォーマットのみがサポートされています。

必須: はい

タイプ: 文字列

有効値: Amazon S3 バケット内のパッケージの Amazon S3 URI。例えば、「`s3://example-bucket/MySpatialApp.zip`」と入力します。

`launch_command`

アプリケーションを起動するための実行ファイル/バイナリファイル名とコマンドラインパラメータを指定します。各コマンドライン文字列トークンは配列内の要素です。

必須: はい

タイプ: 文字列配列

`required_resource_units`

SimSpace Weaver がこのアプリケーションの各インスタンスに割り当てるリソースユニットの数を指定します。リソースユニットは、ワーカー上の固定量の仮想化中央演算装置 (vCPUs) とランダムアクセスメモリ (RAM) です。リソースユニットの詳細については、「エンドポイントと Service Quotas」を参照してください。この `compute` プロパティは、`compute` ワーカーファミリーのリソースユニット割り当てを指定するもので、現状での唯一の有効な割り当てタイプです。

必須: はい

タイプ: 整数

有効値: 1-4

カスタムコンテナイメージ

image プロパティ (オプション) は、SimSpace Weaver がこのドメインでアプリケーションを実行するために使用するコンテナイメージの場所を指定します (バージョン 1.13 および 1.12 ではサポートされていません)。イメージを含む Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリに URI を指定します。このプロパティは指定されていないが、最上位の simulation_properties セクションで default_image が指定されている場合、このドメインのアプリケーションは default_image を使用します。詳細については、「カスタムコンテナ」を参照してください。

```
    image: "ecr-repository-uri"
```

プロパティ

image

このドメイン内のアプリケーションを実行するコンテナイメージの場所を指定します。

必須: いいえ

タイプ: 文字列

有効値:

- Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリの URI (例: 111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

カスタムドメイン設定

カスタムドメインのプロパティを指定するには、*custom-domain-name* を任意の名前に置き換えます。名前は 3～64 文字で、A～Z、a～z、0～9、_ - (ハイフン) を含むことができます。名前の後にカスタムドメインのプロパティを指定します。カスタムドメインごとにこの手順を繰り返します。

```
  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"
```

プロパティ

`launch_apps_via_start_app_call`

このプロパティは StartApp API を使用してカスタムアプリケーションを起動するために必要です。

必須: はい

タイプ: なし

有効値: {}

カスタムアプリケーションの設定

`app_config section` (必須) は、このカスタムドメイン内のアプリケーションのパッケージ、起動設定、リソース要件、ネットワークポートを指定します。

```
    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, ...]
```

プロパティ

`package`

アプリケーションの実行ファイル/バイナリを含むパッケージ (zip ファイル) を指定します。パッケージは Amazon S3 バケットに保存されている必要があります。zip ファイルフォーマットのみがサポートされています。

必須: はい

タイプ: 文字列

有効値: Amazon S3 バケット内のパッケージの Amazon S3 URI。例えば、「`s3://example-bucket/MyCustomApp.zip`」と入力します。

`launch_command`

アプリケーションを起動するための実行ファイル/バイナリファイル名とコマンドラインパラメータを指定します。各コマンドライン文字列トークンは配列内の要素です。

必須: はい

タイプ: 文字列配列

`required_resource_units`

SimSpace Weaver がこのアプリケーションの各インスタンスに割り当てるリソースユニットの数を指定します。リソースユニットは、ワーカー上の固定量の仮想化中央演算装置 (vCPUs) とランダムアクセスメモリ (RAM) です。リソースユニットの詳細については、「エンドポイントと Service Quotas」を参照してください。この `compute` プロパティは、`compute` ワーカーファミリーのリソースユニット割り当てを指定するもので、現状での唯一の有効な割り当てタイプです。

必須: はい

タイプ: 整数

有効値: 1-4

`endpoint_config`

このドメイン内のアプリケーションのネットワークエンドポイントを指定します。`ingress_ports` の値は、カスタムアプリケーションが受信クライアント接続にバインドするポートを指定します。SimSpace Weaver は動的に割り当てられたポートを、指定した入力ポートにマッピングします。入力ポートは TCP と UDP の両方です。DescribeApp API を使用して、クライアントに接続するための実際のポート番号を検索します。

必須: いいえ。 エンドポイント設定を指定しない場合、このドメインのカスタムアプリケーションにはネットワークエンドポイントがありません。

タイプ: 整数配列

有効値: `1024-49152`。値は一意である必要があります。

カスタムコンテナイメージ

image プロパティ (オプション) は、SimSpace Weaver がこのドメインでアプリケーションを実行するために使用するコンテナイメージの場所を指定します (バージョン 1.13 および 1.12 ではサポートされていません)。イメージを含む Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリに URI を指定します。このプロパティは指定されていないが、最上位の simulation_properties セクションで default_image が指定されている場合、このドメインのアプリケーションは default_image を使用します。詳細については、「カスタムコンテナ」を参照してください。

```
    image: "ecr-repository-uri"
```

プロパティ

image

このドメイン内のアプリケーションを実行するコンテナイメージの場所を指定します。

必須: いいえ

タイプ: 文字列

有効値:

- Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリの URI (例: 111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

サービスドメイン設定

サービスドメインのプロパティを指定するには、 を任意の名前*service-domain-name*に置き換えます。名前は 3～64 文字で、A～Z、a～z、0～9、_ - (ハイフン) を含むことができます。名前の後にサービスドメインのプロパティを指定します。サービスドメインごとにこの手順を繰り返します。

```
  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"
```

ワーカーごとにアプリケーションを起動する

launch_apps_per_worker セクション (必須) は、これがサービスドメイン設定であることを示し、ワーカーごとに起動するサービスアプリケーションの数を指定します。

```
    launch_apps_per_worker:
      count: number-of-apps-to-launch
```

プロパティ

count

このプロパティは、ワーカーごとに起動するサービスアプリケーションの数を指定します。

必須: はい

タイプ: 整数

有効値: {} | 1 | 2。{} の値はデフォルトの 1 の値を指定します。

サービスアプリケーションの設定

app_config section (必須) は、このサービスドメイン内のアプリケーションのパッケージ、起動設定、リソース要件、ネットワークポートを指定します。

```
    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, ...]
```

プロパティ

`package`

アプリケーションの実行ファイル/バイナリを含むパッケージ (zip ファイル) を指定します。パッケージは Amazon S3 バケットに保存されている必要があります。zip ファイルフォーマットのみがサポートされています。

必須: はい

タイプ: 文字列

有効値: Amazon S3 バケット内のパッケージの Amazon S3 URI。例えば、「`s3://example-bucket/MyServiceApp.zip`」と入力します。

`launch_command`

アプリケーションを起動するための実行ファイル/バイナリファイル名とコマンドラインパラメータを指定します。各コマンドライン文字列トークンは配列内の要素です。

必須: はい

タイプ: 文字列配列

`required_resource_units`

SimSpace Weaver がこのアプリケーションの各インスタンスに割り当てるリソースユニットの数を指定します。リソースユニットは、ワーカー上の固定量の仮想化中央演算装置 (vCPUs) とランダムアクセスメモリ (RAM) です。リソースユニットの詳細については、「エンドポイントと Service Quotas」を参照してください。この `compute` プロパティは、`compute` ワーカーファミリーのリソースユニット割り当てを指定するもので、現状での唯一の有効な割り当てタイプです。

必須: はい

タイプ: 整数

有効値: 1-4

`endpoint_config`

このドメイン内のアプリケーションのネットワークエンドポイントを指定します。`ingress_ports` の値は、サービスアプリケーションが受信クライアント接続にバインドするポートを指定します。SimSpace Weaver は動的に割り当てられたポートを、指定した入力ポートにマッピングします。入力ポートは TCP と UDP の両方です。DescribeApp API を使用して、クライアントに接続するための実際のポート番号を検索します。

必須: いいえ。 エンドポイント設定を指定しない場合、このドメインのサービスアプリケーションにはネットワークエンドポイントがありません。

タイプ: 整数配列

有効値: 1024-49152。値は一意である必要があります。

カスタムコンテナイメージ

image プロパティ (オプション) は、SimSpace Weaver がこのドメインでアプリケーションを実行するために使用するコンテナイメージの場所を指定します (バージョン 1.13 および 1.12 ではサポートされていません)。イメージを含む Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリに URI を指定します。このプロパティは指定されていないが、最上位の simulation_properties セクションで default_image が指定されている場合、このドメインのアプリケーションは default_image を使用します。詳細については、「カスタムコンテナ」を参照してください。

```
    image: "ecr-repository-uri"
```

プロパティ

image

このドメイン内のアプリケーションを実行するコンテナイメージの場所を指定します。

必須: いいえ

タイプ: 文字列

有効値:

- Amazon Elastic Container Registry (Amazon ECR) 内のリポジトリの URI (例: 111122223333.dkr.ecr.us-west-2.amazonaws.com/my-ecr-repository:latest)

配置の制約事項

placement_constraints セクション (オプション) では、SimSpace Weaver がどの空間ドメインを同じワーカーにまとめて配置するかを指定します。詳細については、「空間ドメインの設定」を参照してください。

> **Important**
>
> バージョン 1.13 および 1.12 は placement_constraints をサポートしていません。

```
placement_constraints:
  - placed_together: ["spatial-domain-name", "spatial-domain-name", ...]
    on_workers: ["worker-group-name"]
```

プロパティ

`placed_together`

SimSpace Weaver が一緒に配置する空間ドメインを指定します。

必須: はい

タイプ: 文字列配列

有効値: スキーマで指定されている空間ドメインの名前

`on_workers`

SimSpace Weaver がドメインを配置するワーカーグループを指定します。

必須: はい

タイプ: 1 要素の文字列配列

有効値: スキーマで指定されたワーカーグループの名前

SimSpace Weaver API リファレンス

SimSpace Weaver には 2 種類のアプリケーションプログラミングインターフェイス (API) セットがあります

- サービス API — これらの API は、シミュレーション、クロック、アプリケーションなどのサービスとサービスリソースを制御します。これらはメインの AWS Software Development Kit (SDK) の一部であり、AWS コマンドラインインターフェイス (CLI) を使用して呼び出すことができます。また、プロジェクトとプラットフォームの [ツール] フォルダにある便利なスクリプト (例えば、*project-folder*\tools\windows\weaver-*project-name*-cli.bat) を使用して、これらの API を呼び出すこともできます。サービス API の詳細については、「SimSpace Weaver API リファレンス」を参照してください。
- アプリケーション SDK API — これらの API はシミュレーション内のデータを制御します。これらをアプリケーションコードで使用して、エンティティフィールドデータの読み取りと書き込み、サブスクリプションの操作、シミュレーション内のイベントの監視などを行います。詳細については、「解凍したアプリ SDK フォルダにある SimSpace Weaver アプリケーション SDK ドキュメント」を参照してください: *sdk-folder*\SimSpaceWeaverAppSdk-1.16.0\documentation

> **Note**
>
> *sdk-folder* は、 SimSpaceWeaverAppSdkDistributable パッケージを解凍したフォルダです。*sdk-folder*\SimSpaceWeaverAppSdk-1.16.0 が存在しない場合は、SimSpace Weaverアプリケーション SDK のバージョン 1.16.0 を使用していることを確認してください。 SimSpaceWeaverAppSdkDistributable にはSimSpace Weaverアプリケーション SDK 全体は含まれません。SimSpace Weaver アプリケーション SDK スクリプトを初めて使用してプロジェクトを作成する場合、スクリプトは *sdk-folder* 内に SimSpaceWeaverAppSdk-1.16.0 フォルダを作成し、残りのSimSpace Weaverアプリケーション SDK をそこにダウンロードします。

AWS SimSpace Weaver バージョン

私たちは AWS SimSpace Weaver を継続的に改善しています。新機能や機能の更新を利用するには、新しいバージョンがリリースされた際に、最新の SimSpace Weaver アプリケーション SDK をダウンロードする必要があります。既存のシミュレーションを新しいバージョンで実行するには、スキーマとコードを更新してから、シミュレーションの新しいインスタンスを起動する必要がある場合があります。アップグレードする必要はなく、以前のバージョンで既存のシミュレーションを引き続き実行できます。このページでバージョン間の違いを確認できます。現在、すべてのバージョンがサポートされています。

> **Important**
>
> AWS SimSpace Weaver ユーザーガイドの最新バージョンは、最新バージョンのサービスのみを対象としています。以前のバージョンのドキュメントは、メインドキュメントのランディングページ から入手できるAWS SimSpace Weaverガイドカタログ にあります。

最新バージョン

最新バージョン: 1.16.0

現在のバージョンを検索する方法

SimSpace Weaver アプリケーション SDK を使用してシミュレーションを作成した場合、`create-project` スクリプトは SDK ライブラリのバージョンを *`sdk-folder`* のサブディレクトリにダウンロードします。SDK ライブラリを含むサブディレクトリには、SDK バージョン番号 `SimSpaceWeaverAppSdk-`*`sdk-version`* を含む名前が付いています。例えば、バージョン 1.15.3 のライブラリは にありますSimSpaceWeaverAppSdk-1.15.3。

SimSpace Weaver アプリケーション SDK の配布可能パッケージのバージョンは、*`sdk-folder`* のテキストファイル `app_sdk_distributable_version.txt` にも記載されています。

最新バージョンをダウンロードする

最新バージョンをダウンロードするには、以下のリンクを使用します。

- アプリケーション SDK の配布可能パッケージ一式

- アプリケーション SDK ライブラリのみ

SimSpace Weaver アプリケーション SDK の配布可能パッケージ一式は、AWS Management Console の [SimSpace Weaver] コンソールからダウンロードすることもできます。ナビゲーションペインで [アプリ SDK をダウンロード] を選択します。

> **Warning**
>
> SimSpace Weaver アプリケーション SDK の配布可能パッケージと思われるものを、AWS CLI を使用してダウンロードしないでください。このページのダウンロードリンクまたはコンソールのダウンロードリンクのみを使用してください。その他のダウンロード方法や場所はサポートされていないため、古いコード、正しくないコード、または悪意のあるコードが含まれている可能性があります。

アプリケーション SDK のダウンロードに関するトラブルシューティング

Amazon CloudFront （ CloudFront) を使用して、アプリケーション SDK の .zip ファイルを配布します。以下のような状況が発生する可能性があります。

- ダウンロードしたパッケージが最新バージョンではない
  - ダウンロードした .zip ファイルに最新バージョンが含まれていない場合は、 CloudFront エッジロケーションのキャッシュがまだ更新されていない可能性があります。24 時間後にもう一度ダウンロードしてください。
- ダウンロードリンクを使用すると、HTTP 4xx または 5xx エラーが表示される
  - 24 時間後にもう一度試してください。同じエラーが表示される場合は、[SimSpace Weaver] コンソールの下部にある [フィードバック] リンクを使用して問題を報告してください。フィードバックのタイプは、[問題を報告] を選択します。
- ブラウザがページを読み込めないと報告する
  - ローカルネットワークまたはブラウザの設定に問題がある可能性があります。他のページを読み込めるか確認してください。ブラウザのキャッシュをクリアして、もう一度試してください。ダウンロード URL をブロックする可能性のあるファイアウォールルールがないことを確認してください。
- ファイルを保存しようとするとエラーが発生する
  - ローカルファイルシステムのアクセス許可を確認して、ファイルを保存するための正しいアクセス許可を持っていることを確認してください。

- ブラウザが表示される AccessDenied
  - URL をブラウザに手動で入力した場合は、URL が正しいことを確認してください。ダウンロードリンクを使用した場合は、ブラウザの URL に干渉するものがないことを確認し、リンクをもう一度使用してください。

最新バージョンをインストールする

最新バージョンをインストールするには

1. 最新バージョンをダウンロードします。
2. SimSpaceWeaverAppSdkDistributable.zip をフォルダに解凍します。
3. 解凍した最新バージョンの SimSpace Weaver アプリケーション SDK フォルダから `docker-create-image.bat` (WSL の場合は `docker-create-image.sh`) を実行します。
4. 以前のバージョンの代わりに、解凍した最新バージョンの SimSpace Weaver アプリケーション SDK フォルダを使用します。

サービスバージョン

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.16.0	新機能: • SimSpace Weaver アプリケーション SDK でメッセージング APIs を使用して、アプリケーション間でメッセージを送受信できるようになりました。この機能は、C++、Python、	2024 年 2 月 12 日	このガイド	• パッケージ一式 • ライブラリのみ トラブルシューティングを参照してください。

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	Unity、Unreal Engine 5 の統合で使用できます。詳細については、「メッセージング」を参照してください。			

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.15.3	SimSpace Weaver Local の更新: • SimSpace Weaver Local を変更し、AWS クラウド の開発とより密接に連携するようにしました。これらの変更は、SimSpace Weaver Local の C++、Python、Unity、および Unreal Engine のプロジェクトとワークフローに影響します。詳細については、「バージョン 1.15.3 における ローカル開発の相違点」を参照してください。	2023 年 12 月 4 日	AWS SimSpace Weaver ガイドカタログを参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.15.2	App SDK 配布可能パッケージの更新: • cmake の特定の必要なバージョンを使用するように Dockerfile を更新しました。この変更を行わない場合、Docker コンテナのビルドに失敗する可能性があります。	2023 年 11 月 2 日	AWS SimSpace Weaver ガイドカタログを参照してください。	ダウンロードできません
1.15.1	機能更新: • Python SDK: このリリースでは、AWS クラウド で Python ベースのシミュレーションが失敗する原因となっていた問題が修正されています。詳細については、「バージョンメモ」を参照してください。	2023 年 9 月 22 日	AWS SimSpace Weaver ガイドカタログを参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.15.0	新機能: • Python SDK: Python を使用してシミュレーション開発ができるようになりました。SimSpace Weaver アプリケーション SDK の配布可能パッケージには、サンプル Python プロジェクトとその Python ビュークライアントのテンプレートが含まれています。詳細については、「Python の使用」を参照してください。	2023 年 8 月 31 日	AWS SimSpace Weaver 「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.14.0	新機能: • カスタムコンテナ: 独自の Amazon Linux 2 (AL2) ベースのコンテナイメージを作成し、Amazon Elastic Container Registry (Amazon ECR) に保存し、それを使用して AWS クラウドで SimSpace Weaver アプリケーションを実行します。詳細については、「カスタムコンテナ」を参照してください。 • 複数の空間ドメイン: シミュレーションで複数の空間ドメインを作成します。シミュレーションロジックを、すべて単一の空間アプリケーションにまとめるのではなく、分離	2023 年 7 月 26 日	AWS SimSpace Weaver「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	します。要件に基づいてさまざまなリソースを空間ドメインに割り当てます。詳細については、「空間ドメインの設定」を参照してください。 • 無制限のティックレート: コード実行と同じ速さでシミュレーションを実行できます。すべてのアプリケーションが現在のティックのコミット操作を完了するとすぐに次のティックが送信されるように、シミュレーションのクロックを設定します。詳細については、「クロック」を参照してください。			

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	SimSpace Weaver アプリケーション SDK: • `tick_rate` の値が文字列になりました。値には二重引用符 (") を含める必要があります。以前のバージョンのティックレートは整数のままです。詳細については、「クロック」を参照してください。			
1.13.1	SimSpace Weaver アプリケーション SDK: • 機能更新: プロジェクト作成が `PathfindingSampleUnreal` テンプレートで正しく機能するようになりました。	2023 年 6 月 7 日	AWS SimSpace Weaver 「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.13.0	SimSpace Weaver サービス API: • • 新しい CreateSnapshot アクション • StartSimulation アクションへの変更: • スナップショットから開始する SnapshotS3Location パラメータを追加しました。 • SchemaS3Location パラメータはオプションです。 • DescribeSimulation 出力への変更: • SchemaError を廃止しました。 • StartError フィール	2023 年 4 月 29 日	AWS SimSpace Weaver「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	ドを追加しました。 • SnapshotS3Location フィールドを追加しました。 • SNAPSHOT_IN_PROGRESS シミュレーションステータスを追加しました。 • 新しい S3Destination データタイプ SimSpace Weaver コンソール: • スナップショットを作成する新機能。 • スナップショットからシミュレーションを開始する新機能。			

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード

	SimSpace Weaver アプリケーション SDK:
	• スナップショットをサポートする新しいスクリプト
	  • create-snapshot- *project-name* .bat
	  • start-from-snapshot- *project-name* .bat
	  • quick-start-from-snapshot- *project-name* -cli.bat
	  • list-snapshots- *project-name* .bat
	• プロジェクトはプロジェクトごとに 1 つの Amazon S3 バケットを使用するようになりました。Weaver-*lowercase-project-name* -*account-*			

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	number *-region* スナップショットの詳細については、「スナップショット」を参照してください。			

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.12.3	SimSpace Weaver アプリケーション SDK: • 以下のスクリプトが `--maximum-duration` パラメータをサポートするようになりました。 • `quick-start-`*`project-name`*`-cli.bat` • `quick-start-`*`project-name`*`-cli.sh` • `start-simulation-`*`pro`* *`name`*`.bat` • `start-simulation-`*`pro`* *`name`*`.sh` • `run-`*`project-name`*`.bat` • `run-`*`project-name`*`.sh` 詳細については、「シミュレーションの最	2023 年 3 月 27 日	AWS SimSpace Weaver 「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
	大期間」を参照してください。			
1.12.2	SimSpace Weaver アプリケーション SDK: • バグ修正: `docker-create-image.bat` が正しく動作するようになりました。	2023 年 3 月 1 日	AWS SimSpace Weaver 「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.12.1	SimSpace Weaver アプリケーション SDK: • スクリプトが AWS 認証に使用する AWS CLI プロファイルを受け入れるようになりました。 • スクリプトが AWS 認証の AWS IAM Identity Center をサポートするようになりました。 SimSpace Weaver Local: • バグ修正: すべての空間アプリケーションがシミュレーションに参加していない場合でも `Api::BeginUpdateWillBlock` が `true` を正しく返すようになりました。	2023 年 2 月 28 日	AWS SimSpace Weaver 「ガイドカタログ」を参照してください。	ダウンロードできません

バージョン	メモ	リリース日	ドキュメント	アプリケーション SDK ダウンロード
1.12.0	一般提供 (GA) のリリース	2022 年 11 月 29 日	AWS SimSpace Weaver ガイドカタログを参照してください。	ダウンロードできません

AWS SimSpace Weaver バージョン 1.15.1

このリリースは、もともと SimSpace Weaver バージョン 1.15.0 でリリースされた Python SDK の必須アップデートです。Python ベースのシミュレーションが AWS クラウド で失敗する原因となっていたバージョン不一致の問題を修正しています。1.15.0 ではなく、このバージョンを使用してください。

既存の Python プロジェクトを 1.15.1 に更新します

バージョン 1.15.0 Python SDK で作成した既存の Python プロジェクトがある場合は、以下の手順を実行して 1.15.1 に更新し、AWS クラウド で実行できるようにする必要があります。

この手順に従う代わりに、1.15.1 Python SDK を使用して新しい Python プロジェクトを作成し、カスタムコードを新しいプロジェクトに移動することもできます。

1.15.0 Python プロジェクトを 1.15.1 にアップデートする

1. Python プロジェクトのフォルダに移動します。
2. `src/PythonBubblesSample/bin/run-python` で以下の行を変更します。

   ```
   export PYTHONPATH=$PYTHONPATH:/roapp/lib
   ```

   項目の変更後:

   ```
   export PYTHONPATH=$PYTHONPATH:$LD_LIBRARY_PATH:/roapp/lib
   ```

3. `CMakeLists.txt` で以下の行を削除します。

   - ```
     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)
  ```

バージョン 1.15.1 のトラブルシューティング

1.15.0 の Python シミュレーションを更新した後、AWS クラウド を起動できない

症状: シミュレーションを開始してから約 5 ~ 10 分後に、シミュレーション管理ログに `internal error` が報告され、シミュレーションステータスは `FAILED` になります。

これは、1.15.0 Python SDK のライブラリファイルがアプリケーションの zip ファイルに含まれている場合に発生する可能性があります。プロジェクトを更新する手順を完了していることを確認し、`libweaver_app_sdk_python_v1.so` が zip ファイルに含まれていないこと、または何らかの方法で参照されていないことを確認してください。

バージョン 1.15.1 に関するよくある質問

このリリースは Python SDK 以外にも影響しますか？

いいえ。

バージョン 1.15.1 に更新する必要がありますか?

空間アプリケーションに Python を使用する予定がない場合は、1.15.1 に更新する必要はありません。1.15.0 にアップデートすると、Python ベースのシミュレーションは AWS クラウド で実行されなくなります。1.15.0 を使用している場合は 1.15.1 に更新することをお勧めします。

`$LD_LIBRARY_PATH` とは

これは、AWS クラウド でシミュレーションを実行するときの Python SDK の場所です。1.15.1 では新機能です。この変更は、将来発生する Python のバージョンに関する問題を避けるためのものです。このディレクトリへのリンクは、1.15.0 での `libweaver_app_sdk_python_v1.so` へのリンクと機能的には同じです。
  ```

AWS SimSpace Weaver のドキュメント履歴

以下の表は SimSpace Weaver のドキュメントの重要な変更点をまとめたものです。

日付	変更	ドキュメントの更新	API バージョンの更新
2024 年 2 月 12 日	更新されたコンテンツ	バージョン 1.16.0 リリースのAWS SimSpace Weaver バージョン章を更新しました。	該当なし
2024 年 2 月 12 日	新しいコンテンツ	バージョン 1.16.0 リリースの一部として メッセージングセクションを追加しました。このセクションでは、SimSpace Weaverアプリケーション SDK に追加されたメッセージング APIs について説明します。これらの APIs、アプリケーション間でメッセージを送受信できます。	該当なし
2024 年 2 月 12 日	更新されたコンテンツ	バージョン 1.16.0 のSimSpace Weaver シミュレーションスキーマリファレンス章を更新しました。	該当なし
2024 年 2 月 12 日	更新されたコンテンツ	SimSpace Weaver エンドポイントとクォータ 「」章にメッセージングのサービスクォータを追加しました。	該当なし

日付	変更	ドキュメントの更新	API バージョンの更新
2024 年 2 月 12 日	新しいガイド	1.16.0 より前のバージョンのコンテンツを別のガイドに分割します。以前のバージョンのAWS SimSpace Weaverガイドにアクセスするためのガイドカタログ (メインドキュメントのランディングページ から入手可能) を追加しました。	該当なし
2023 年 12 月 4 日	更新されたコンテンツ	バージョン 1.15.3 リリースの「AWS SimSpace Weaver バージョン」という章を更新しました。	該当なし
2023 年 12 月 4 日	更新されたコンテンツ	最新バージョンのインストール手順を含むように「AWS SimSpace Weaver バージョン」という章を更新しました。	該当なし
2023 年 12 月 4 日	更新されたコンテンツ	SimSpace Weaver Local の Service Quotas を更新しました。	該当なし
2023 年 12 月 4 日	新しいコンテンツと更新されたコンテンツ	「ローカル開発」セクションを再構成し、バージョン 1.15.3 で導入された SimSpace Weaver Local との違いを説明する新しいページを追加しました。	該当なし

日付	変更	ドキュメントの更新	API バージョンの更新
2023 年 11 月 7 日	更新されたコンテンツ	Docker と WSL がアプリケーション SDK の直接ダウンロードリンク/URL を使用するように設定する手順を更新しました。詳細については、「SimSpace Weaver のローカル環境をセットアップする」を参照してください。	該当なし
2023 年 11 月 2 日	更新されたコンテンツ	1.15.2 リリースのサービスバージョンページを更新しました。詳細については、「サービスバージョン」を参照してください。	該当なし
2023 年 10 月 23 日	更新されたコンテンツ	サービスバージョンページを更新し、アプリケーション SDK 配布可能パッケージをダウンロードするための新しい手順を追加しました。承認された直接ダウンロードリンクのうちの 1 つだけを使用し、アプリケーション SDK 配布可能パッケージのダウンロードには AWS CLI を使用しないようにしてください。詳細については、「最新バージョンをダウンロードする」を参照してください。	該当なし

日付	変更	ドキュメントの更新	API バージョンの更新
2023 年 9 月 22 日	新しいコンテンツ	1.15.1 リリースの更新手順が記載されたバージョンノートページを追加しました。詳細については、「AWS SimSpace Weaver バージョン 1.15.1」を参照してください。	該当なし
2023 年 9 月 10 日	新しいコンテンツ	AWS CLI が SimSpace Weaver を認識しない場合のトラブルシューティングセクションを追加しました。詳細については、「AWS CLI が simspaceweaver を認識しない」を参照してください。	該当なし
2023 年 9 月 10 日	更新されたコンテンツ	WSL で AWS CLI の手順を更新しました。詳細については、「Windows Subsystem for Linux (WSL) での Amazon Linux 2 (AL2) を設定する」を参照してください。	該当なし
2023 年 9 月 7 日	API 更新	BucketName および ObjectKey が S3Location データ型に必要になりました。 BucketName SS3Destination データ型に が必要になりました。	AWS SDK: 2023-09-07

日付	変更	ドキュメントの更新	API バージョンの更新
2023 年 8 月 31 日	新しいコンテンツ	リリース 1.15.0 の新しいセクションを追加しました: Python の使用。	該当なし
2023 年 8 月 15 日	更新されたコンテンツ	公式の SimSpace Weaver Amazon S3 バケットのみを一覧表示するように AWS SimSpace Weaver バージョン のダウンロード手順を更新しました。他のダウンロード場所は AWS によって制御されていないため、悪意のあるコードが含まれている可能性があります。	該当なし
2023 年 7 月 26 日	更新されたコンテンツ	更新済み クロック。	該当なし
2023 年 7 月 26 日	更新されたコンテンツ	更新済み 空間ドメインの設定。	該当なし
2023 年 7 月 26 日	新しいコンテンツ	新しいセクション「カスタムコンテナ」を追加しました。	該当なし
2023 年 7 月 26 日	更新されたコンテンツ	AWS SimSpace Weaver バージョン をリリース 1.14.0 に更新しました。	該当なし
2023 年 7 月 6 日	新しいコンテンツ	新しいセクション「PathfindingSample コンソールクライアントが接続に失敗する」を追加しました。	該当なし

日付	変更	ドキュメントの更新	API バージョンの更新
2023 年 6 月 7 日	更新されたコンテンツ	AWS SimSpace Weaver バージョン をリリース 1.13.1 に更新しました。	該当なし
2023 年 5 月 15 日	新しいコンテンツ	新しいセクション「AWS CloudFormation でのスナップショットの使用」を追加しました。	該当なし
2023 年 4 月 29 日	新しいコンテンツ	リリース 1.13.0 のコンテンツを追加しました。詳細については、「AWS SimSpace Weaver バージョン」を参照してください。	AWS SDK: 2023-04-28
2023 年 3 月 27 日	新しいコンテンツ	シミュレーションの最大時間について説明するセクションを追加しました。リリース 1.12.3 のチュートリアルに、SimSpace Weaver アプリケーション SDK スクリプトへの `--maximum-duration` パラメータのサポートが追加された注記を追加しました。	該当なし

日付	変更	ドキュメントの更新	API バージョンの更新
2023 年 3 月 9 日	変更されたコンテンツ	Windows の Docker および Windows Subsystem for Linux (WSL) のみを説明対象としており、WSL (およびその他の Linux 環境) はサポートしていないことを明確にしました。	該当なし
2023 年 2 月 28 日	新しいコンテンツ	SimSpace Weaver バージョンを説明するチャプターを追加しました。	該当なし
2023 年 2 月 28 日	変更されたコンテンツ	AWS Command Line Interface (AWS CLI) での AWS IAM Identity Center および名前の付いたプロファイルの使用を含む認証についてのコンテンツを変更しました。	該当なし
2023 年 2 月 17 日	新しいコンテンツ	AWS CloudFormation でのリソース管理についてのセクションを追加しました。	該当なし
2023 年 1 月 23 日	新しいコンテンツ	ローカルシミュレーションをデバッグする手順を追加しました。	該当なし
2022 年 11 月 29 日	サービスの起動	SimSpace Weaver のユーザーガイドと API リファレンスをリリースしました。	AWS SDK: 2022-11-29

用語集

この用語集では、AWS SimSpace Weaver 特有の用語を定義します。

最新の AWS の用語については、「AWS 全般のリファレンス」の「AWS 用語集」を参照してください。

A

アプリケーション
自分で作成する実行コード (バイナリとも呼ばれます)。アプリケーションという用語は、コードまたはそのコードの実行中のインスタンスを指す場合があります。アプリケーションはシミュレーションの動作をカプセル化したものです。アプリケーションはエンティティを作成、削除、読み取り、更新します。

アプリケーション SDK
アプリケーションを SimSpace Weaver と統合するために使用する Software Development Kit (SDK)。SDK には、エンティティデータの読み取りと書き込み、およびシミュレーション時間の追跡のための API が用意されています。詳細については、「SimSpace Weaver アプリケーション SDK」を参照してください。

C

クライアント
SimSpace Weaver の外部に存在し、カスタムアプリケーションまたはサービスアプリケーションを介してシミュレーションと相互作用するプロセス (またはその定義)。クライアントを使用してシミュレーションの状態を表示または変更できます。

クロック
SimSpace Weaver の内部スケジューリングプロセスを抽象化したもの。クロックはアプリケーションにティックを表示して時間の同期を維持します。各シミュレーションには独自のクロックがあります。

クロックレート
クロックがアプリケーションに公開する 1 秒あたりのティック数。サポートされているクロックレートの詳細については、「SimSpace Weaver エンドポイントとクォータ」を参照してください。

クロックのティックレート
「クロックレート」を参照してください。

コンピュートリソースユニット	ワーカー上のコンピュートリソース (プロセッサとメモリ) のユニット。通常、アプリケーションの 単一のインスタンスには 1 つのコンピュートリソースユニットが割り当てられます。各アプリケーションには複数のコンピュートリソースユニットを割り当てることができます。
カスタムアプリケーション	シミュレーションの状態を読み取ったり操作したりするために使用するアプリケーションタイプ。カスタムアプリケーションはシミュレーションでエンティティを作成できますが、所有しているエンティティは作成できません。カスタムアプリケーションがエンティティを作成する場合、そのエンティティを空間ドメインに転送する必要があります。アプリケーション API を使用してカスタムアプリケーションのライフサイクルを制御します。SimSpace Weaver API の詳細については、「SimSpace Weaver API リファレンス」を参照してください。
カスタムドメイン	カスタムアプリケーションを含むドメイン。
カスタムパーティション	カスタムアプリケーションのパーティション。

D

デッドライン	操作 (ティックの処理など) が完了するまでの実際の時間。
ドメイン	同じ実行コード (アプリケーションバイナリ) を実行し、同じ起動オプションを持つアプリケーションインスタンスのグループ。

E

エンドポイント (サービス)	プログラム (AWS Command Line Interface など) が SimSpace Weaver サービスへの接続に使用する完全修飾ドメイン名 (FQDN)。
エンドポイント (シミュレーション)	クライアントがシミュレーションへの接続に使用する IP アドレスとポート番号。カスタムアプリケーションとサービスアプリケーションのエンドポイントを構成できます。
エンティティ	カスタマーデータオブジェクト (またはその定義)。エンティティは静的 (1 つの場所に留まる) でも動的 (シミュレーション空間内を移動する) でもかまいません。例えば、シミュレーション内の人や建物などです。

I

インデックス (シミュレーション)
空間境界や座標システムなど、シミュレーションの空間プロパティの説明。

L

(アプリケーションの) ライフサイクル
シミュレーション中にアプリケーションが行うと予想される論理的ステップの説明。ライフサイクルには、管理対象 (SimSpace Weaverアプリケーションを起動および停止) または管理対象外 (ユーザーがアプリケーションを起動および停止する) があります。

ロード (エンティティフィールドデータ)
State Fabric からのエンティティフィールドデータの読み取り。

P

パーティション
ワーカーの共有メモリのセグメント。各パーティションには、ドメイン内のエンティティの個別のサブセットが含まれます。各アプリケーションにはパーティションが割り当てられています。アプリケーションは、そのパーティション内のすべてのエンティティを所有します。アプリケーションはエンティティを作成すると、そのパーティションにエンティティを作成します。エンティティがあるパーティションから別のパーティションに移動すると、所有権はソースパーティションのアプリケーションから転送先パーティションのアプリケーションに移ります。

R

リソースユニット
「???」を参照してください。

S

スキーマ
シミュレーションの設定を記述する YAML または JSON ドキュメント。SimSpace Weaver はスキーマを使用してシミュレーションリソースを作成します。

サービスアプリケーション
シミュレーションの状態を読み取ったり操作したりするために使用するアプリケーションタイプ。サービスアプリケーションはシミュレーションで

	エンティティを作成できますが、それらを空間ドメインに転送する必要があります。SimSpace Weaver はサービスアプリケーションのライフサイクルを管理し、シミュレーションの各ワーカーで 1 つ (またはシミュレーションスキーマで指定されている場合はそれ以上) を起動します。
サービスドメイン	サービスアプリケーションを含むドメイン。
サービスパーティション	サービスアプリケーションのパーティション。
シミュレーション (リソース)	シミュレートされた仮想化空間を実行する計算クラスターを抽象化したもの。複数のシミュレーションを使用できます。シミュレーションはスキーマを使用して設定します。
空間アプリケーション	コアシミュレーションロジックをカプセル化するアプリケーションタイプ。各空間アプリケーションは 1 つ (1 つのみ) のパーティションを所有します。
空間ドメイン	空間アプリケーションを含むドメイン。
空間パーティション	空間アプリケーションのパーティション。
State Fabric	SimSpace Weaverのインメモリデータベース。State Fabric には、エンティティや内部 SimSpace Weaver データを含むシミュレーションの状態が格納されます。
ストア (エンティティフィールドデータ)	State Fabric へのエンティティフィールドデータの書き込み。
サブスクリプション	特定のアプリケーションインスタンスがサブスクリプション領域からデータを受信するように要求する、長期にわたるリクエスト。サブスクライブしているアプリは、サブスクリプションを使用して、サブスクリプション領域内のエンティティへの変更を検出します。
サブスクリプション領域	シミュレーション空間の二次元領域。サブスクリプションはサブスクリプション領域を指します。サブスクリプション領域は複数のパーティションにまたがることができ、パーティションの一部を含むこともできます。サブスクリプション領域は定義された範囲内で連続しています。

T

ティック	時間の離散値 (ウォールクロック時間またはシミュレーション時間)。アプリケーションはティック時間よりも速く繰り返し処理できますが、特定のデッドライン内に指定されたティックを書き込むことが期待されます。特定のティックに対するすべてのアプリケーションのすべての操作は、次のティックが開始される前に完了する必要があります。
ティックレート	「クロックレート」を参照してください。
時間 (実際)	現実の観点から見た現在の時刻。SimSpace Weaver は Unix エポック (January 1, 1970, 00:00:00 UTC). からのナノ秒数である 64 ビット POSIX タイムスタンプを使用します。
時間 (シミュレーション)	シミュレーションの観点から見た現在の時間。SimSpace Weaver は 64 ビット整数の論理ティックカウンターを使用しますが、実際の時間とは直接一致しない場合があります。

W

ワーカー	シミュレーションコードを実行する Amazon Elastic Compute Cloud (Amazon EC2) インスタンス。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。