Unity 用 Genvid SDK プラグイン

概要

Unity 用 Genvid SDK プラグインは Genvid SDK を Unity プロジェクトにインテグレーションするための簡易な方法を提供します。このドキュメントでは、Genvid SDK および Unity パッケージのインストール、提供されているプレハブのセットアップ、Genvid クラスターへの接続の準備、およびストリーム用 Web サイトオーバーレイ作成について説明します。

このドキュメントでは実行に必要となる基本的な手順を説明していますが、詳細については Genvid SDK ドキュメントを確認することをお勧めします。

ライセンス

Unity と Genvid Code EULA

Unity 用 Genvid SDK プラグインを使用する際には、Genvid SDK のEULAに従う必要があります。Asset Store EULA とあわせて、Genvid SDK 同梱のライセンスファイルをよくお読みいただき、同意した上でご利用ください。

ライセンスについての詳細は、Genvid Technologies website を参照してください。

インストールと設定

Genvid SDK をインストールする

1. Genvid Technologies Web サイトのダウンロードセクション (https://www.genvidtech.com/sdk-download/) を開く。
2. サインインまたは無料アカウントの作成を行って、Developer Portal にアクセスします。
3. Genvid SDK の最新バージョンをダウンロードします。
4. インストーラを起動し、画面の指示に従って SDK をインストールします。

Prefab の設定

GenvidSessionManager プレハブをプロジェクトに追加する

1. Unity Editor でプロジェクトを開きます。
2. `/Assets/Genvid/Prefabs/Genvid` フォルダで、**GenvidSessionManager** プレハブをルートシーンにドラッグ＆ドロップします。このプレハブはデフォルト設定のままにします。

3. 同じフォルダで、**GenvidSession** を **GenvidSessionManager** に**GenvidSessionManager** の子としてドラッグ＆ドロップします。

4. **GenvidSessionManager** には、**GenvidSession** オブジェクトが必要な **Session** フィールドがあります。現在の **GenvidSession** プレハブをこのフィールドにドラッグ＆ドロップします。

GenvidSession にその他のプレハブを追加する

Video

1. **GenvidSession** に **GenvidSession** の子として **Video** プレハブを追加します。
2. **GenvidSession** には **Video Stream** セクションがあります。現在の **Video** プレハブをこのフィールドにドラッグ＆ドロップします。

3. ビデオストリームの名前を変更します。*Video* だけでも構いません。
4. **Capture Type** を選択します。
    - **Automatic** は、ゲームと UI をすべてキャプチャします。
    - **Texture** は、**Video Source** (カメラまたはテクスチャ) で設定されたオブジェクトをキャプチャします。このキャプチャタイプは UI を削除しますが、キャプチャ、テクスチャを変更する場合、**VideoSource** 公開オブジェクトを更新する必要があります。

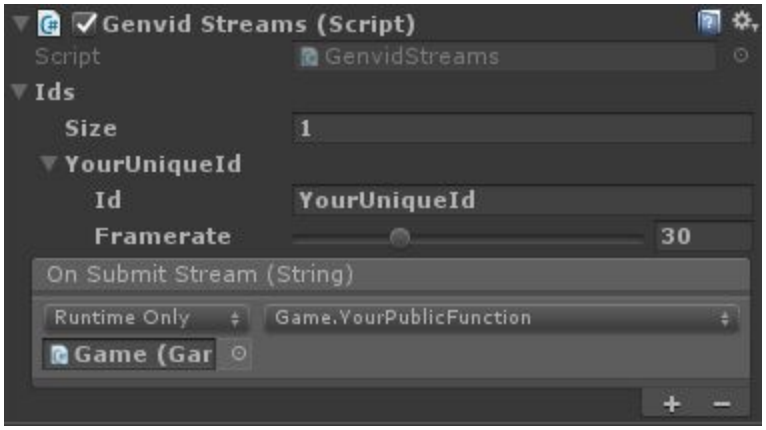
Audio

1. **Audio** プレハブを **GenvidSession** に、**GenvidSession** の子としてドラッグ＆ドロップします。
2. **GenvidSession** には **Audio Stream** セクションがあります。現在の **Audio** プレハブをこのフィールドにドラッグ＆ドロップします。

3. オーディオストリームの名前を変更します。*Audio* だけでも構いません。
4. Audio Format モードを選択します。特に問題なければ、F32LE のままにしておくことを推奨します。
5. 適切な Audio Mode を選択します。
    - Unity はデフォルトで、**Unity** オーディオを使用します。これは、多くのオーディオプラグインと互換性があります。
    - **WASAPI** は、PC のオーディオを使用します。使用中のオーディオプラグインで Unity オーディオが使えない場合に便利です。

Streams

1. **Streams** プレハブを **GenvidSession** に、**GenvidSession** の子としてドラッグ＆ドロップします。
2. **GenvidSession** には **Streams** セクションがあります。現在の **Streams** プレハブをこのフィールドにドラッグ＆ドロップします。



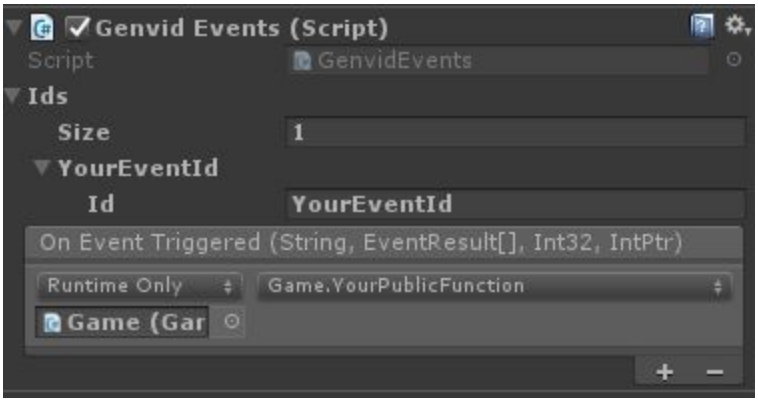
3. **Ids Size** を作成するゲームデータストリーム数に変更します。
4. エレメントを開き、**Id** フィールドにユニーク名を付けます。
5. リピート送信を行うゲームデータの **Framerate** を選択します。
    - 再送信を行わない場合は選択不要。
6. ゲームデータを送信するスクリプトで、1つの文字列パラメータを持つpublic関数を作成します。
7. **On Submit Stream (String)** ボックスにスクリプトとオブジェクトをドラッグ＆ドロップし、ドロップダウンリストから作成したpublic関数を選択します。
    - 再送信を行う場合は選択不要。
    - この処理中、**String** フィールドが表示されることがあります。エレメントの **Id** をこのフィールドに追加しておかないと、この値で呼び出しがオーバーライドされます。
8. public関数で、プレハブからデータ送信コールを使用することができます。
    - **SubmitAnnotation**: 特定時間になったときに、リピートが *don't* に設定されている場合に、Web サイトにデータを送信します。
    - **SubmitGateData**: 特定時間になったときに、リピートが *do* に設定されている場合に、Web サイトにデータを送信します。
    - **SubmitNotification**: リピートが *don't* に設定されている場合に (時間指定なく) すぐに、Web サイトにデータを送信します。

Events

イベントは、視聴者とゲームとのインタラクティブ要素を定義します。例えば、視聴者が特定のプレイヤーの「応援」機能 ボタンをクリックすることで、現在のゲームのプレイヤー人気投票に影響を与えることができるようになります。

イベントは MapReduce システムを通してデータを送ることで、Web サイト上で視聴者から送信されるイベントでゲームがオーバーフローしてしまうことを防止します。

1. **Events** プレハブを **GenvidSession** に、**GenvidSession** の子としてドラッグ＆ドロップします。
2. **GenvidSession** には **Events** セクションがあります。現在の **Events** プレハブをこのフィールドにドラッグ＆ドロップします。



3. **Ids Size** を作成するゲームデータストリーム数に変更します。
4. エレメントを開き、**Id** フィールドにユニーク名を付けます。
5. イベントを受け付けるスクリプトで、次の順にパラメータを持つpublic関数を作成します。
    - 文字列。
    - **EventResult** オブジェクトの配列です。
    - 32ビットの整数。
    - 整数ポインタ。
6. **On Event Triggered** ボックスにスクリプトとオブジェクトをドラッグ＆ドロップし、ドロップダウンリストから作成したpublic関数を選択します。
7. public関数でイベントから受信した情報を使用することができます。
    - **String:** イベント ID。

      注意: Events を設定したり Web サイトを呼び出す際に、この Id を使用します。

    - **EventResults[]:** イベントの結果です。
    - **Int32**: 受信した結果数です。
    - **IntPtr**: コールバックに使用するポインタ。

public関数は、外部 Web サイトで視聴者から送信されたイベントをどのように扱うかを指定します。どんなイベントを利用可能にするか、どんな処理を行うかはあなた次第です。

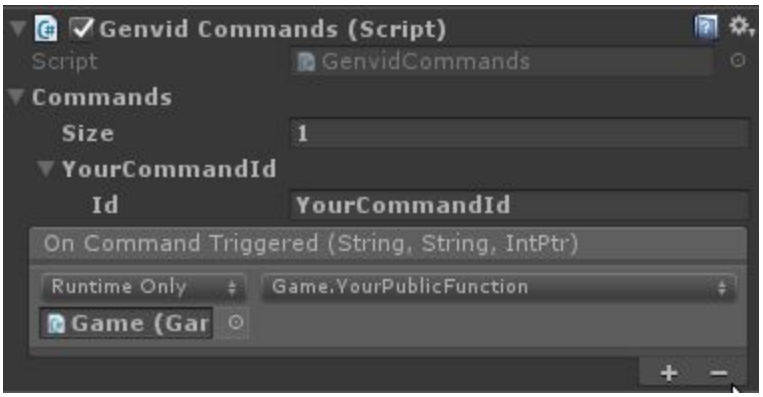
Commands

コマンドを使用して、管理サイトなど、特殊なサイトからゲームを動かすことができます。

イベントとは違い、コマンドは MapReduce システムを使わずに、Web サイトからゲームに直接指示することができます。これにより、最低限の遅延でゲームに個別に指示を出すことができます。

コマンドは、管理者またはアクセスが管理されたページに限定することをお勧めします。

1. **Commands** プレハブを **GenvidSession** に、**GenvidSession** の子としてドラッグ＆ドロップします。
2. **GenvidSession** には **Commands** セクションがあります。現在の **Commands** プレハブをこのフィールドにドラッグ＆ドロップします。



3. **Ids Size** を作成するゲームデータストリーム数に変更します。
4. エレメントを開き、**Id** フィールドにユニーク名を付けます。
5. コマンドを受け付けるスクリプトで、次の順にパラメータを持つpublic関数を作成します。
    - 文字列。
    - 2番目の文字列。
    - 整数ポインタ。
6. **On Command Triggered** ボックスにスクリプトとオブジェクトをドラッグ＆ドロップし、ドロップダウンリストから公開変数を選択します。
7. 作成したpublic関数でコマンドから受信した情報を使用することができます。
    - **1つ目の文字列:** コマンド ID。

      注意: Commands を設定したり Web サイトを呼び出す際に、この Id を使用します。
    - **2番目の文字列**: コマンドの結果。
    - **IntPtr**: コールバックに使用するポインタ。

public関数は、外部 Web サイトから送信されたコマンドをどのように扱うかを指定します。コマンドは、通常、管理機能に使用し、どんな処理を行うかはあなた次第です。管理者サイトのサンプルで、コマンドの例を見ることができます。

ローカルクラスタ設定

Genvid SDK は、クラウドベースのサービスセットに依存し、ストリームのオーディオ、ビデオ、およびデータを処理および同期します。また、テスト目的で使用できるローカルセットアップもあります。

このセクションでは、ローカルの Genvid Cluster への接続準備について説明します。Genvid Cluster および、クラウドでの設定方法については Genvid SDK ドキュメントを参照してください。

bastion サーバーの設定

1. コマンドプロンプトで、Genvid SDK インストールフォルダに移動します。
2. Genvid ツールボックスをインストールします。

   ```
   py install-toolbox.py --user
   ```

   終了後、ツールボックスが適切にインストールされているか、システムの PATH 環境変数に含まれているかをインストールスクリプトがチェックします。PATH に含まれていない場合、スクリプトを使って変更したり、手動で変更したりできます。

   詳細は、Python のドキュメントの Using Python on Windows および Installing Packages のページを参照してください。

3. bastion サーバーをインストールして設定します。

   ```
   genvid-bastion install -uml --bastionid mybastion
   ```

   ```
   genvid-sdk setup -b local
   ```

4. bastion に古い設定データが残っていない状態にします。

   ```
   genvid-sdk clean-config
   ```

5. bastion にGenvid SDK サンプル設定を読み込む。

   ```
   genvid-sdk load-config-sdk
   ```

6. /config、/templates、、/web フォルダと Python スクリプト unity.py を /samples/unity フォルダからプロジェクトにコピーします。
   - プロジェクトフォルダと同じ階層に配置する必要があります (Assets フォルダの一つ上の階層になります)。

テンプレートを変更します。

Samples.hcl

1. /config フォルダの sample.hcl を開きます。
2. アプリケーションに合わせて、**description**、**game**、**name** フィールドを変更します。
3. **encode input** と **output** 設定を、実行する解像度に設定します。

Game.hcl

1. /config フォルダの game.hcl を開きます。
2. job "unity" をアプリケーション名に変更します。
3. log "game"で:
   - job をアプリケーション名に変更します。
   - fileName をアプリケーション名に変更します。

4. log "gameerr" で、job をアプリケーション名に変更します。
5. configで:
    - binary の後のunity をアプリケーション名に変更します。
    - パスを Python スクリプトの格納場所からのアプリケーションパスに変更します。

Events.json

Events.json ファイルは、Web サイトから送信されたイベントを処理する方法を Genvid SDK に指示する場所です。このサンプルでは分かりやすくするために、以前 **Events** プレハブに追加したイベントで使用する既存の定義を変更します (通常は、必要に応じてイベント定義を作成します)。

1. /config フォルダで events.json を開きます。
2. "maps" で:
    - cheer をアプリケーションのプレハブで使用するイベント ID にリネームします。
    - "where" および "key" の cheer を同じイベント ID に変更します。
3. "reductions" で:
    - id および where の cheer をイベント ID に変更します。

Unity.nomad.tmpl

1. /templates/local フォルダで unity.nomad.tmpl を game.hcl (*jobname*.nomad.tmpl) で使用する job 名に変更します。
2. jobname.nomad.tmpl を開きます。
3. job "unity" を game.hcl で使用する job 名に変更します。
4. task "unity" を game.hcl で使用する job 名に変更します。
5. コマンド の後の unity を game.hcl で使用する job 名に変更します。
6. logs/unity.log を game.hcl で使用するファイル名に変更します。

Web サイトの設定

Web サイトの構築

1. Python スクリプト unity.py を使用して Web サイトを構築します。

```
py unity.py build web
```

2. /web/public/unity.ts ファイルを開き、設定を変更します。

ゲームデータとアノテーションの設定

1. unity.tsで、**on_streams_received** 関数を探します。
2. 2番目のループは、ゲームデータとアノテーションを受け取ったときにデコードできる場所です。データは frame.data で利用可能です。通常は JSON parse() を実行しますが、適切な方法で自由に処理して構いません。

   genvidClient.d.ts 内のデータ構成については Genvid SDK ドキュメント の **IDataStreams** を参照してください。

通知の設定

1. unity.tsで **on_notifications_received()** 関数を探します。
2. for ループで通知を受信することができます。通常は、**notification.id** で処理方法をチェックしますが、自由にこのデータを使用して構いません。

`genvidClient.d.ts` 内のデータ構成については Genvid SDK ドキュメント の **IDataStreams** を参照してください。

イベントの設定

すでにテンプレートでイベントを定義済ですが、以下の処理を行う必要があります。

1. `unity.ts` で **onCheer()** 関数を探します。
2. `this.client.sendEventObject` の後の `cheer` を `events.json` で使用するイベント ID に変更します。
    a. **this.client.sendEventObject** 関数で、イベントをゲームに送信可能です。
3. 送信するデータを ":" の後に追加します。
4. このイベントを呼び出すボタンを Web ページに追加します。

コマンドの設定

1. `adminUnity.ts` で **changeScene** 関数を探します。
2. ID をアプリケーションの **Commands** プレハブのエレメントの ID に変更します。
3. アプリケーションでエフェクトをトリガーするために送信するデータを値に設定します。
4. Web リクエストが適切に行われたかを確認するために、`promise` および `promise.fail` のメッセージを変更ます。
5. このコマンドを呼び出すボタンを Web ページに追加します。

注意: Commands ページには、**admin** でアクセスすることができます (デフォルトのログイン名、パスワードはどちらも "admin" です)。

最初の実行時

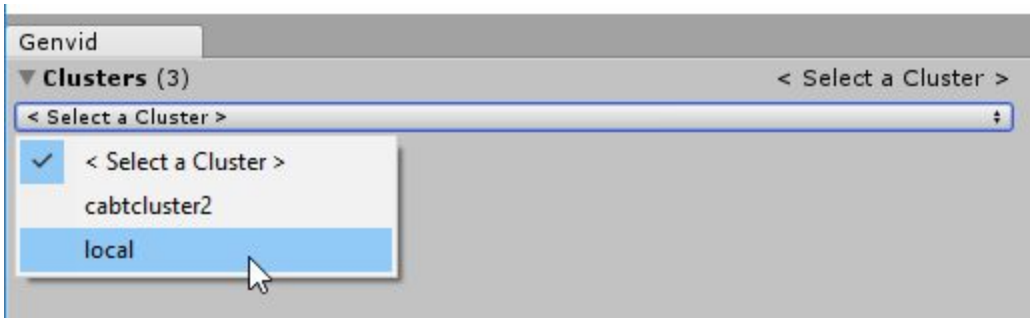
1. 作成したすべてのテンプレートファイルを Genvid スタックに読み込みます。

```
py unity.py load
```

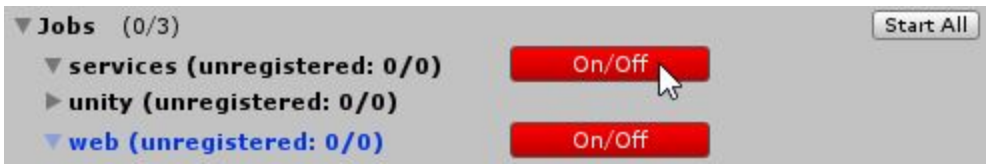
2. 変更した Web サイトを再構築します。

```
py unity.py build web
```

3. Unity editor 内で **Window > Genvid** で、Genvid ウィンドウを開きます。
4. クラスタリストが更新されたら **local** をクリックします。



5. **Jobs** で、**On/Off** ボタンをクリックして **services** と **web** ジョブを開始します。



6. **Links** で、アプリケーション名の横の **Open Link** をクリックします。
7. エディタで Start をクリックします。

ゲームが Web サイトに表示されます。Web ページを更新して、ストリームを更新する必要があるかもしれません。(すでに設定が終わっている場合には) Web ページに追加したすべてのデータ、イベント、およびコマンドは、この時点でアクセス可能になります。

Unity での Genvid のトラブルシューティング

Genvid プラグインを使用するときに発生する可能性のある一般的な問題の回避策を紹介します。

いくつかのマシンでビデオストリームが正しくキャプチャされません。

Genvid SDK は、Windows D3D11 ドライバーでのみ動作します。それ以外に設定しようとすると、エラーを出力します。-force-d3d11 オプションを付加してゲームの実行を試してください。

ビデオがストリーム内で上下逆に表示されます。

Unity は DirectX11 を使用してレンダリングしますが、テクスチャの座標には OpenGL の規約を使用します。

OpenGL は以下の規約を使用します。

[0、0] は、左下です。

[1、1] は、右上です。

DirectX は以下の規約を使用します。

[0、0] は、左上です。

[1、1] は、右下です。

Genvid プレハブでこの問題を修正しました。ただし、ストリームビデオが上下逆に表示される場合は、**video.useopenglconvention** パラメータとともに **SetParameter(Object,String,Int32)** Genvid SDK 関数を使用して反転させてください：

```
GenvidSDK.SetParameter(m_StreamName, "video.useopenglconvention", 1)
```

("1" は、ビデオの反転を切り替えます)

CheckForEvents を実行中に “Disconnected” エラーが出続ける。

“Disconnected” エラーは、一般的には Genvid スタックが実行されていないことを意味しています。アプリケーションを実行する前に **Services** および **Web** ジョブを開始する必要があります。