audiokinetic

CONSERVATORY OF RECORDING ARTS & SCIENCES

Wwise-101 技能検定コース

7つのレッスン、7つのクイズ、そして1つの検定試験

Wwise-101 技能検定コース

7つのレッスン、7つのクイズ、そして1つの検定試験

リビジョン 290

7つのレッスンで、Wwise-101 の技能検定に合格するための全てを学べます。また、検定を受ける前に自分の知識を確認できるように、7つのクイズも用意しています。クイズの受験は任意ですが、十分に準備するためにも、是非受けてみることをお勧めします。Audiokineticチーム一同が、あなたを応援しています！

Legal Notice

目次

Wwiseをインストールする 1
Wwiseをインストールし、レッスンの内容を準備する 3
Wwise-101コースの教材をインストールする 4
Cubeデモをプレイする 9
Cubeをサイレンスにする 12
レッスン 14
レッスン 1：クイックスタート - サイレンスからサウンドへ 16
Wwiseの起動 17
ゲームのプロファイリング 22
イベントの作成 26
サウンドのインポート 31
Actionの適用 38
ゲームへのサウンドインテグレーション 41
ゲームをプレイする 48
レッスン 2：Soundscapeのデザイン 51
プロジェクトにおけるサウンドの追加 52
単一のサウンドを複数の用途に使用する 59
オブジェクトのパラメーター調整 61
Source Editorの使用 64
単一イベントへの複数アクションの適用 65
ランダマイゼーション（ランダム化）機能の使用 69
サウンドのグラニュラ化 79
作業の確認 97
レッスン 3：Game Syncの理解 99
スイッチの使用 100
Game Parametersの使用 114
Statesの活用 126
CubeデモへのGame Syncsインテグレーション 132
ProfilerでのGame Syncsの確認 135
レッスン 4：イベントの作成 139
3D Game Definedの使用 140
3D User Definedを理解する 160
2Dパンニングの使用 169
ゲームをプレイする 174
レッスン 5：オーディオシグナルフローを理解する 178
アクターミキサーの構築 179
マスタミキサー階層の使用 184
各種エフェクトの使用 198
スケマティックビューの使用 212
レッスン 6：ミックスの仕上げ 219
サウンドキャスターの使用 220
ミキシングデスクの構築 229
コントロールサーフェスの使用 242
レッスン 7：ゲームの最適化 252
メモリの管理 253
プロセッサーにおける最適化 273

プロファイラーを使用したリアルタイムモニタリング 277

Wwiseをインストールする

audiokinetic

CONSERVATORY OF RECORDING ARTS & SCIENCES

Wwiseをインストールし、レッスンの内容を準備する 3
Wwise-101コースの教材をインストールする 4
Cubeデモをプレイする 9
Cubeをサイレンスにする 12

Wwiseをインストールし、レッスンの内容を準備する

Wwise-101コースの教材をインストールする 4
Cubeデモをプレイする 9
Cubeをサイレンスにする 12

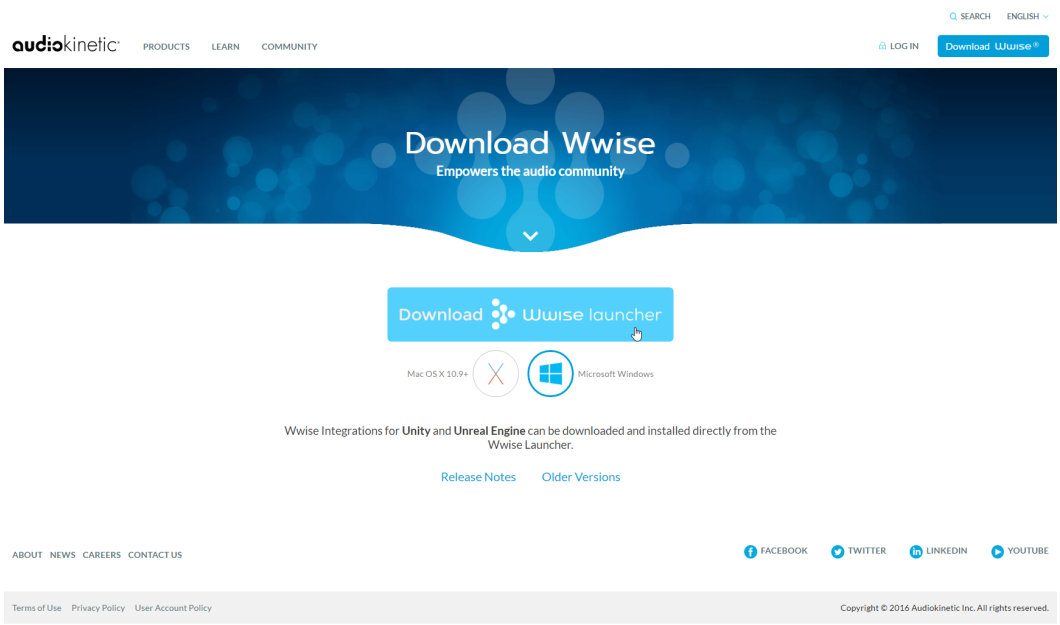
ようこそあなたはこれから、最新のパワフルなゲームオーディオのインテグレーションツールとサウンドエンジンであるWwiseを活用して、ビデオゲームに音を組み込む技能を取得する課程を始めます。Wwiseのようにパワフルで柔軟性の高いツールの技能取得は大変な労力を要するものと思わるかもしれませんが、 本レッスンを一歩一歩進めることで素早くWwiseを使い慣れることができます。ここではCube（キューブ）というゲームに音を実装することでWwiseを学んでいきます。CubeデモはFPSスタイルのゲームで、あなたはそのゲームにサウンドを組み込むことで生き生きとしたものに仕上げることを楽しみながら、悪人たちから世界を救う役を担います。

Wwise-101コースの教材をインストールする

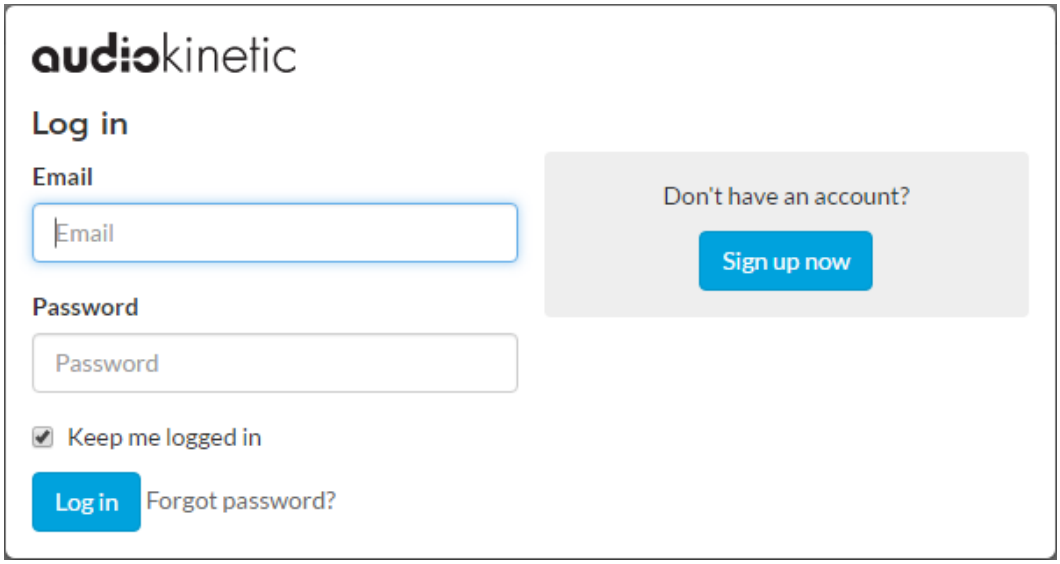
最初に、Wwise-101コースの受講に必要な各種リソースをインストールします。具体的には、Wwiseオーサリングアプリケーション、Cubeデモゲーム、そして本コース専用のWwiseプロジェクトやオーディオファイルが入ったレッスンフォルダなどです。

Wwiseをインストールする

1. まだWwise Launcherをインストールしていない場合は、Wwiseのダウンロードページ www.audiokinetic.com/download に進みます。既にWwise Launcherをインストールしてあれば、手順5に進んでください。

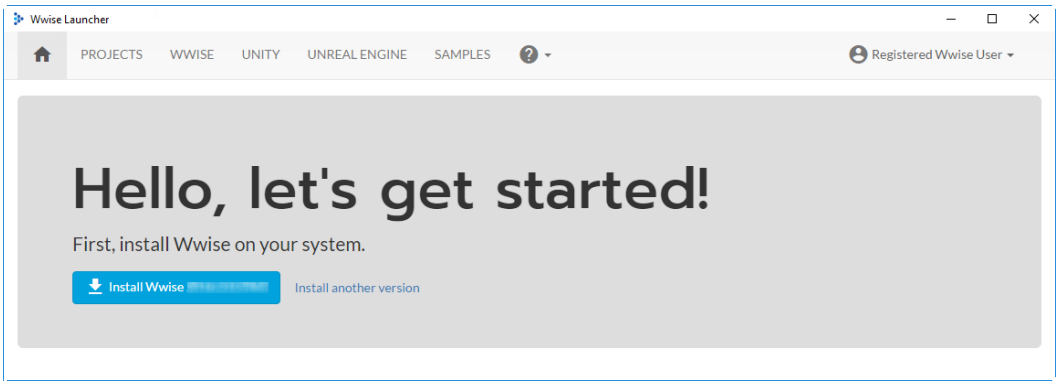


2. 使用OSによってWindowsまたはMacのオプションを選択し、**Download Wwise launcher**ボタンをクリックします。
3. コンピューターにダウンロードしたファイルを開き、**WwiseLauncher**インストーラーを実行してインストールの手順に従います。インストールが完了すると自動的にWwise Launcherが開き、ログイン画面が表示されます。



既にAudiokineticのユーザーアカウントをお持ちであれば、ログインできます。

4. Wwise Launcherが表示されます。

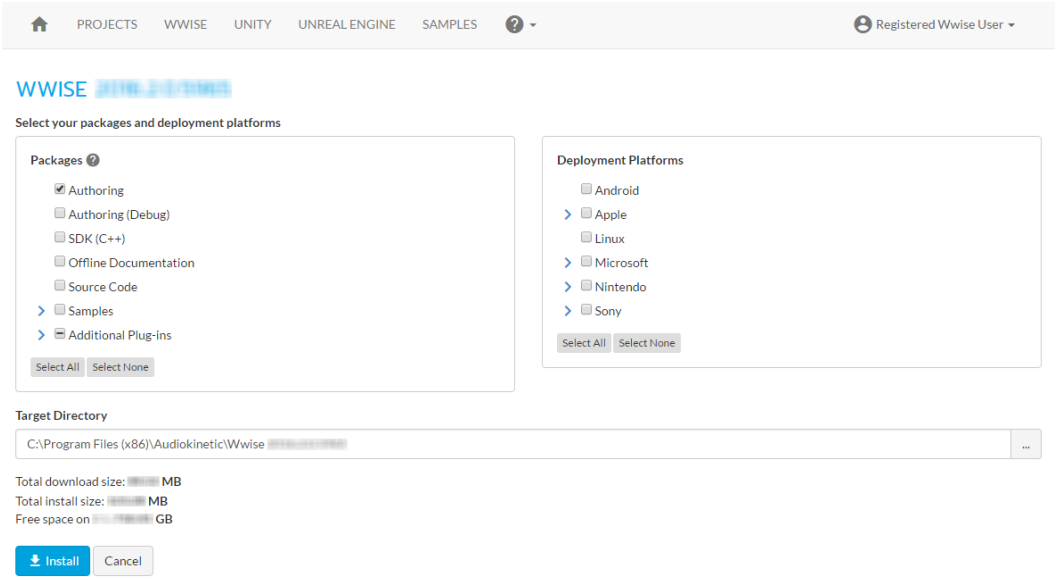


注記

Wwiseのダウンロード時期によって利用可能なバージョンが異なるので、以下インストールのスクリーンショットではバージョン番号をぼかしています。使用中のWwiseがWwise 2016.1以降のWwiseであれば、どのバージョンでもWwise-101コースの受講に影響ありません。

Wwise Launcherのホーム画面には、Wwiseコミュニティの最新情報やWwiseオーサリングアプリケーションを起動するオプションなどが表示されます。今までWwiseをインストールしたことがなければ、Wwiseのインストールを勧めるプロンプトが表示されます。

5. Wwise Launcherで、**Install Wwise xxxx.x.x.xxxx** ボタンをクリックします。ウィンドウが開くので、インストールするPackagesやDeployment Platformsを選択します。

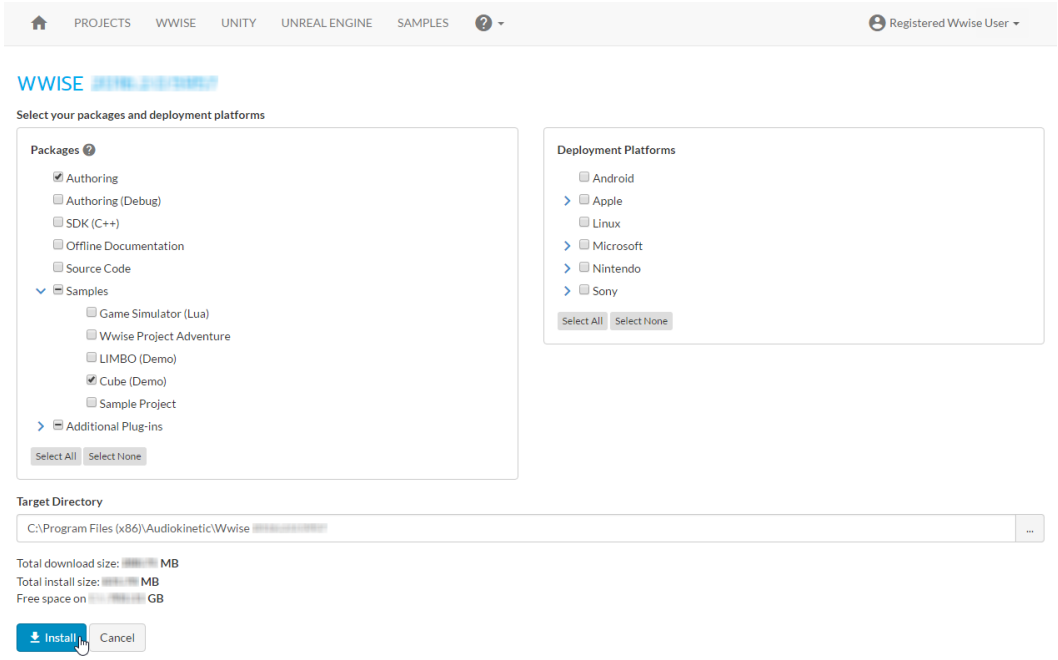


本コースの全てのレッスンがCubeゲームに基づいているので、Samples欄で選択する時に必ずCubeを含めてください。

注記

Deployment Platforms欄は、全てのオプションが選択されていない状態です。本コースにあたり、リリース予定のプラットフォームをここで選択する必要はありませんが、今後のプロジェクトのために追加プラットフォームをインストールしておくこともできます。

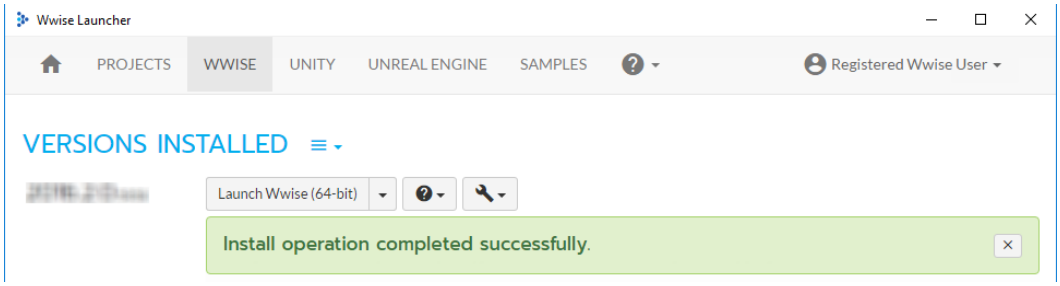
6. Samples欄を拡張して**Cube (Demo)**を選択して、**Install**をクリックします。

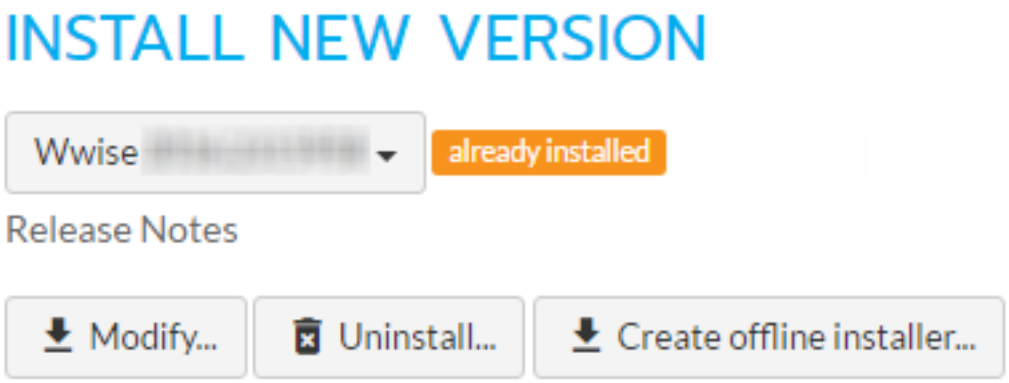


この時点でインストーラーがインストール手順を順次表示して、あなたが指定したファイルをダウンロードします。処理中に管理者ユーザーネームやパスワードを求められることがあります。

7. インストールプロセスを続けて、プロンプトに従いインストールの確認をしてください。

インストールが完了すると、インストールされたバージョンを起動するための新しいボタンが表示されます。（UnityやUnrealをインストールしてある場合はWwiseをそれらのプロジェクトに追加するオプションも表示されますが、この時点では無視しても構いません。）



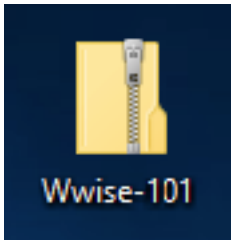


Wwiseがインストールできたので、Wwise Launcherを開いたままにして、次にWwise-101コースのプロジェクトファイルをダウンロードしてインストールします。

Wwise-101コースのプロジェクトファイルをインストールする

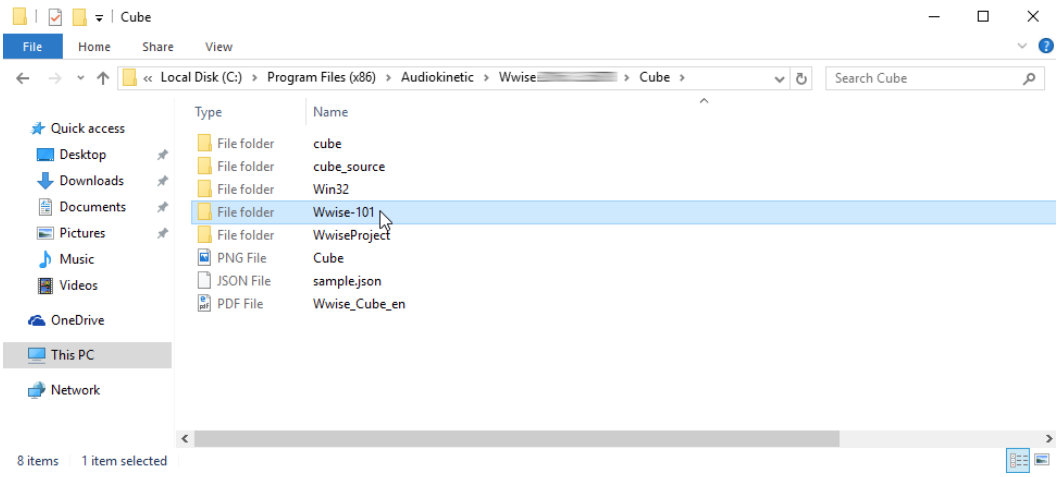
Wwiseの最新バージョンがインストールできたので、Wwise-101技能認定コースに使うプロジェクトファイルやオーディオアセットをダウンロードしてインストールする準備が整いました。

1. Wwise-101.zipファイルをここからダウンロードします： https://www.audiokinetic.com/download/lessons/Wwise-101_v2016.2.zip

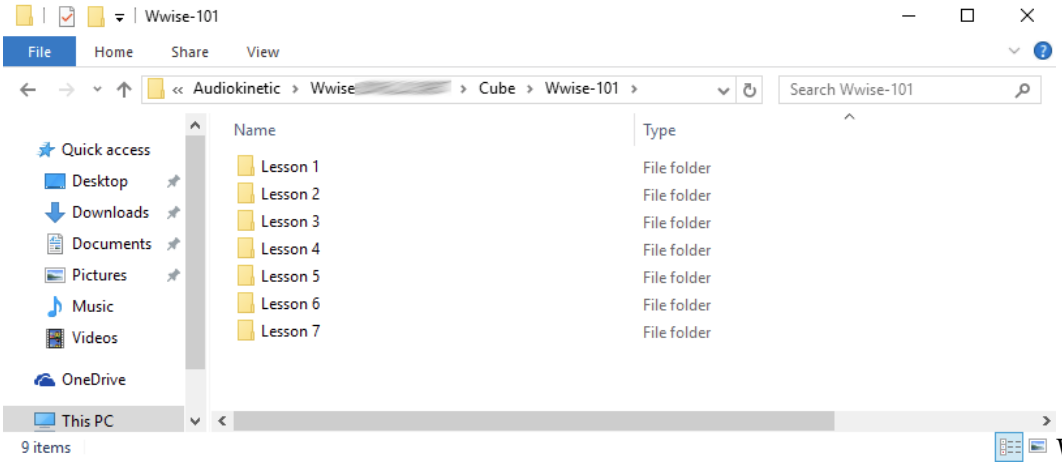


ダウンロードしたZIPファイルを解凍して、直前にインストールしたWwiseアプリケーションフォルダ内にあるCubeフォルダに入れる必要があります。Cubeフォルダのデフォルト格納場所は、**Windowsでは：** Program Files (x86)/Audiokinetic/Wwise 2016.x.x.xxxx/Cube 、**Macでは：** Applications/Wwise 2016.x.x.xxxx /Cube

2. Wwise-101.zipを解凍して、新しいWwise-101フォルダを、Wwise xxxx.x.x.xxxx フォルダ内のCubeフォルダに移動します。



3. Wwise-101フォルダを開いて、中身を確認します。



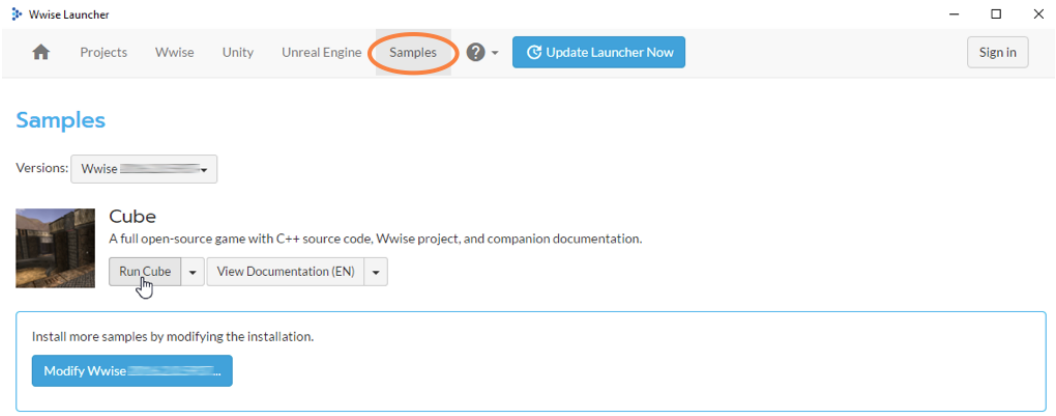
Wwise-101フォルダ内に、これから勉強するレッスン毎のリソースが入ったフォルダがあります。

Cubeデモをプレイする

本チュートリアルを通じて使用するゲームCubeデモはオープンソースの一人称シューティングゲームで、ミッションはモンスターに倒される前に、あなたがモンスターを倒すことです。あなたは次にあるバージョンのCubeデモを実行します。このバージョンではインターネットで入手できるバージョンのCubeデモとは大きな違いがあります。このバージョンのCubeデモではデモの実行中に、Wwiseソフトウエアがデモ中で何が起きているのかという詳細情報を通信することができるようになっています。これはレッスン1で学習しますが、非常に有用な情報を得ることができます。

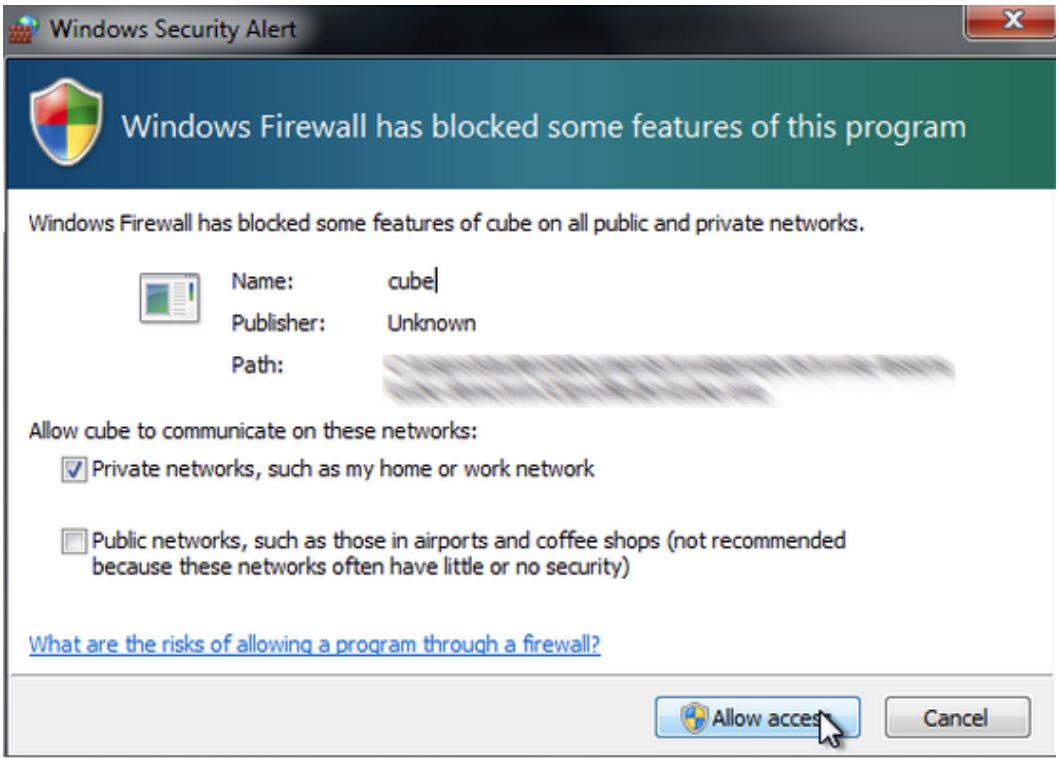
このあとのレッスンにおいてCubeデモを実行することで、あなたが実装したサウンドデザインがゲームにどのように影響をあたえるかを確認できるようになっています。Cubeデモを実行するように指示された際には、次の手順にしたがって下さい。

1. Wwise Launcherアプリケーションに戻り、**Samples**タブを選択して**Run Cube**ボタンをクリックします。

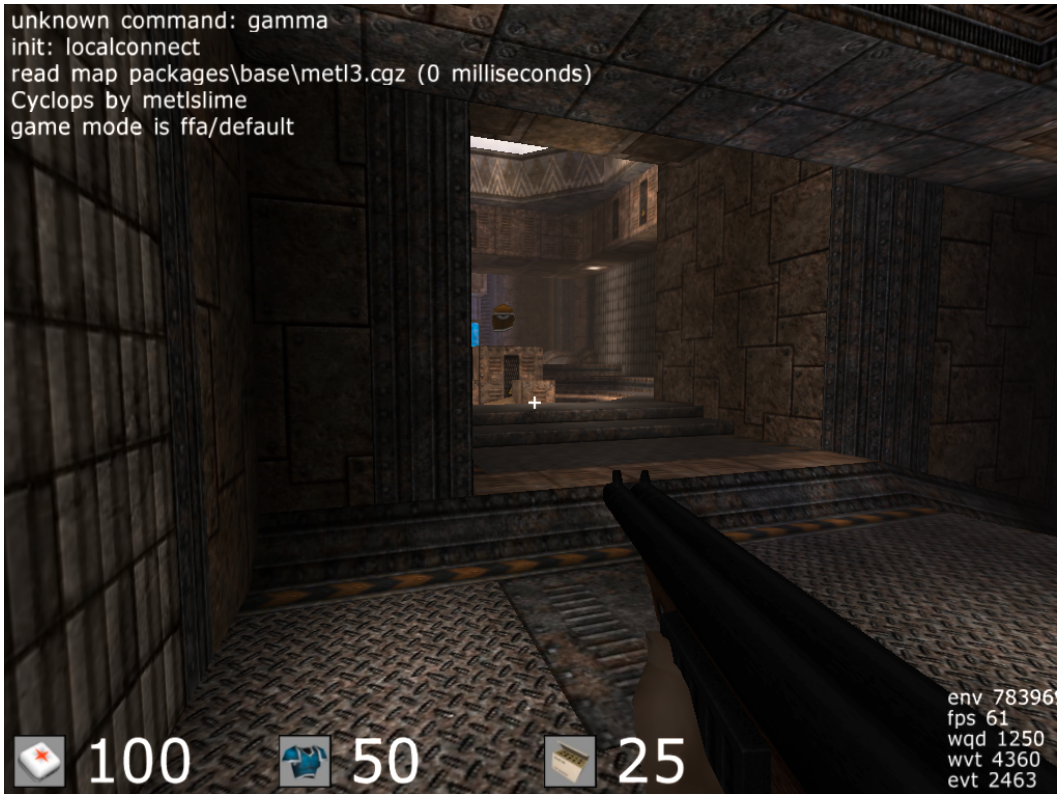


Cubeゲームが起動する時に、Windowsファイアウォールがプログラムの一部機能をブロックした旨の通知が表示されるかもしれません。これはレッスン 1：クイックスタート - サイレンスからサウンドへで学びますが、版のCubeデモがネットワーク通信を使用するからです。Windows側で本機能がブロックされないようにすることが重要です。

2. Windowsセキュリティ警告ウインドウが表示された場合には、**Allow Access**をクリックします。



ゲームが起動します。Cubeデモを最初に起動したときは 、空のレベルから始まりますので、敵に見つかることを心配すること無く、動き回って，独自のサウンドデザインを追求してみて下さい。



ゲームの操作にはFPS標準のキーボードによるコントロールを使用します。本チュートリアルにおいて、覚えて頂きたい操作は以下の通りです:

- W = 前進
- S = 後退
- A = 左へ平行移動
- D= 右へ平行移動
- スペース = ジャンプ
- 左クリック = 発砲

3. ゲームを少しプレイし、ショットガンを発砲してみて下さい。

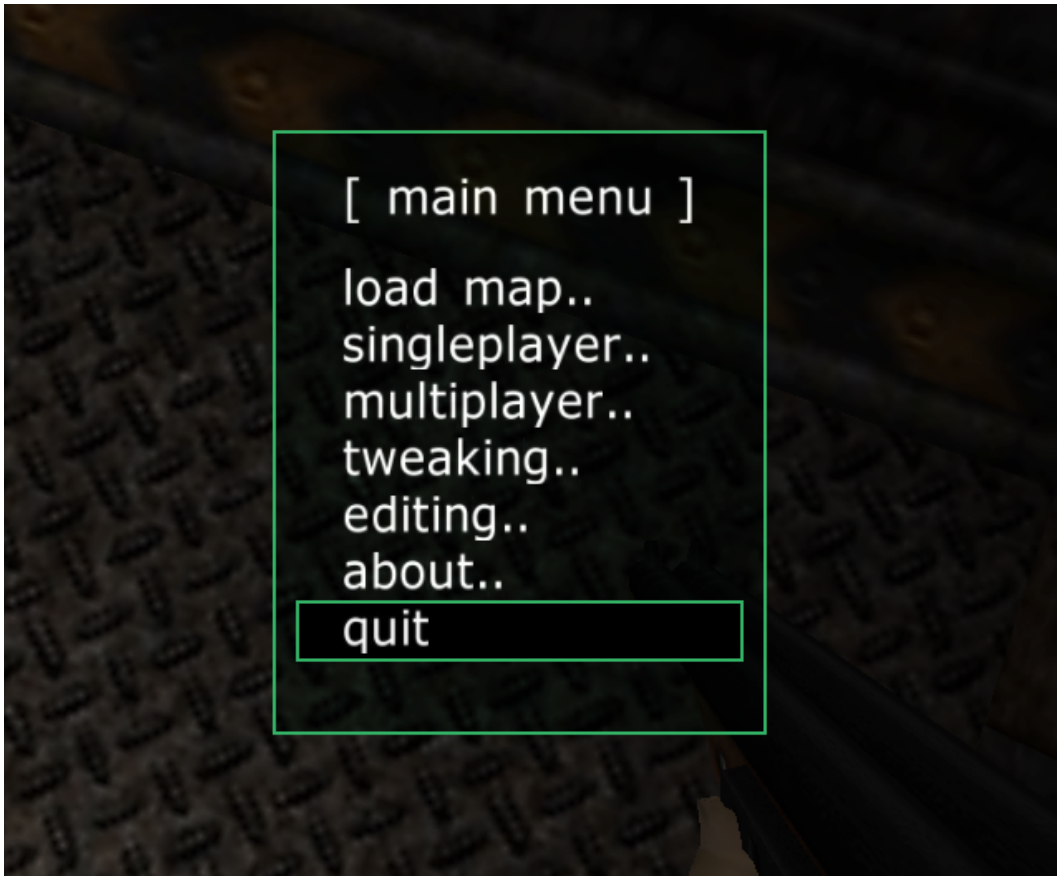
進んでいくと、歩いている地面のマテリアルによって足音が変わるのが分かります。ショットガンを発砲すると、その発砲音と共に、ショットガンのシェルが地面に落ちる音が聞こえますが、再生される度に異なるオーディオパターンとなります。音によるフィードバックのおかげで、ゲームが無音の状態よりも格段と充実します。これから続く演習では、Wwiseサウンドエンジンを活用してどのように音をCubeに実装していくかの基本原則を学んでいきます。

ティップ

Alt-TabでデスクトップやWwiseプログラムの切り替えが容易にできます。

Cubeデモの操作に慣れたら、一旦ゲームを終了させてください。

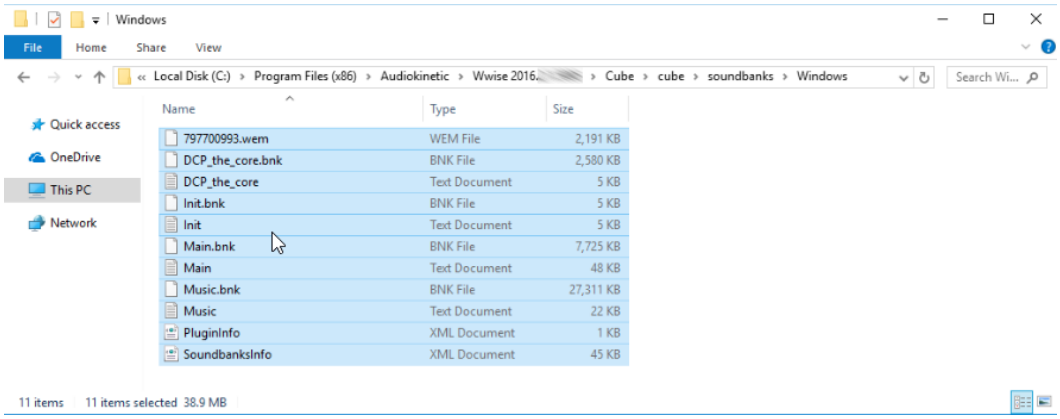
4. まず**Escape** キーを押して、 矢印キーで上下にメニューを移動できますから、**quit** を選択して**enter**を押します。



Cubeをサイレンスにする

Wwise-101コースの目的は、ゲームをサイレンス（無音）の状態から、完全にサウンドが組み込まれた状態まで作り込む一連の操作を見ることです。このプロセスを完全に理解するには、Cubeのデモインストールの一部であるサウンドの実装を、排除する必要があります。最初のレッスンで学ぶ通り、ゲームに実装されたサウンドは全て、ゲーム側が参照するsoundbanksフォルダーに入っています。プラットフォームがMacかWindowsかによって、soundbanksフォルダ内にあるフォルダが異なります。既にあるサウンドインテグレーションを一旦全て排除するには、あなたのプラットフォームのフォルダに入っている全てのコンテンツを削除する必要があります。フォルダへのパスが、WindowsとMacでやや異なります。

1. Windowsは、`Program Files (x86)/Audiokinetic/Wwise 2016.x.x.xxxx/Cube/cube` を開きます。Macは、`Applications/Audiokinetic/Wwise 2016.x.x.xxxx/Cube/cube` を開きます。



   あなたのプラットフォームのフォルダに、ゲームが参照する様々なオーディオリソースが入っています。これらのファイルを削除すると、サウンドインテグレーションは排除されますが、ゲームのその他の部分は今まで通り、正常に機能します。

2. **Windows**または**Mac**のこのフォルダに入っているコンテンツを、全て削除してください。

   あなたのプラットフォームのサウンドバンクフォルダが削除されると、ゲームをオーディオ無しでプレイできます。

3. Cubeを再度Wwise Launcherから実行して、サウンドが聞こえなくなったことを確認します。

   以上です。Wwiseの習得を進めるにあたって、このプロファイル可能なバージョンのCubeデモを実行させたままにしてください。本レッスンでは、CubeデモとWwiseアプリケーションの間を行ったり来たりしながら進めることになります。レッスン 1：クイックスタート - サイレンスからサウンドへへ進む準備ができました。レッスン2では無音のゲームに音を追加していきます。

レッスン

audiokinetic

CONSERVATORY OF RECORDING ARTS & SCIENCES

レッスン 1：クイックスタート - サイレンスからサウンドへ 16
Wwiseの起動 17
ゲームのプロファイリング 22
イベントの作成 26
サウンドのインポート 31
Actionの適用 38
ゲームへのサウンドインテグレーション 41
ゲームをプレイする 48
レッスン 2：Soundscapeのデザイン 51
プロジェクトにおけるサウンドの追加 52
単一のサウンドを複数の用途に使用する 59
オブジェクトのパラメーター調整 61
Source Editorの使用 64
単一イベントへの複数アクションの適用 65
ランダマイゼーション（ランダム化）機能の使用 69
サウンドのグラニュラ化 79
作業の確認 97
レッスン 3：Game Syncの理解 99
スイッチの使用 100
Game Parametersの使用 114
Statesの活用 126
CubeデモへのGame Syncsインテグレーション 132
ProfilerでのGame Syncsの確認 135
レッスン 4：イベントの作成 139
3D Game Definedの使用 140
3D User Definedを理解する 160
2Dパンニングの使用 169
ゲームをプレイする 174
レッスン 5：オーディオシグナルフローを理解する 178
アクターミキサーの構築 179
マスタミキサー階層の使用 184
各種エフェクトの使用 198
スケマティックビューの使用 212
レッスン 6：ミックスの仕上げ 219
サウンドキャスターの使用 220
ミキシングデスクの構築 229
コントロールサーフェスの使用 242
レッスン 7：ゲームの最適化 252
メモリの管理 253
プロセッサーにおける最適化 273
プロファイラーを使用したリアルタイムモニタリング 277

レッスン 1：クイックスタート - サイレンスからサウンドへ

Wwiseの起動 17
ゲームのプロファイリング 22
イベントの作成 26
サウンドのインポート 31
Actionの適用 38
ゲームへのサウンドインテグレーション 41
ゲームをプレイする 48

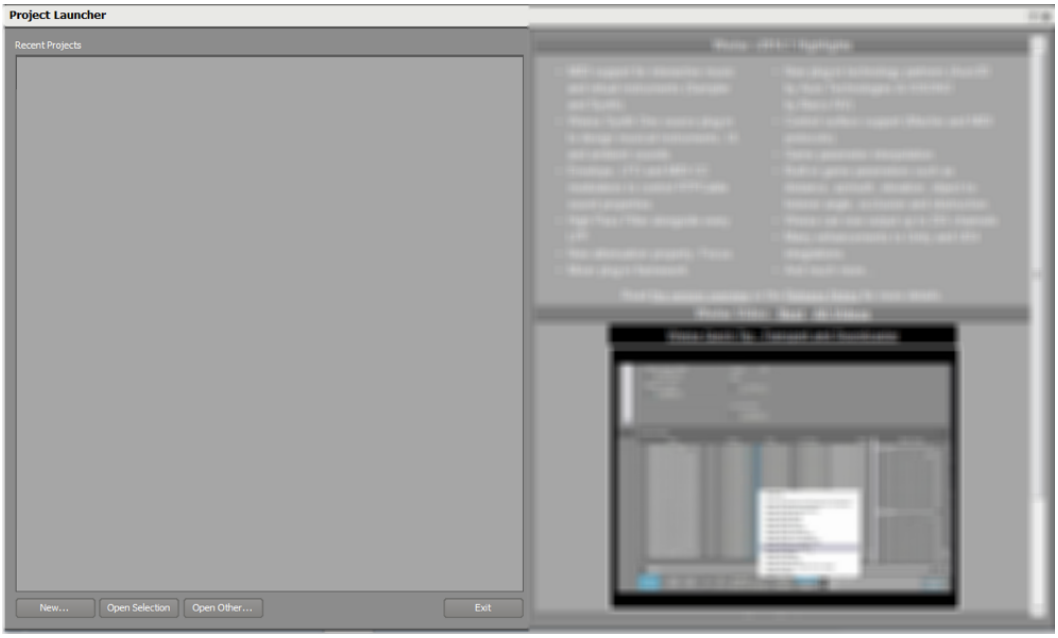
Wwiseをインストールするでは、本チュートリアル全般で使用するCubeデモをインストールし、少しプレイしました。現段階では、このCubeデモにはサウンドが組み込まれておらず、魅力的ではありません。このゲームにサウンドを加えていきます。このレッスンでは、一番わかりやすいビジュアルアニメーション、ショットガンの発射にサウンドを組み込みます。

Wwiseの起動

新しいWwiseプロジェクトを作成します。Wwiseプロジェクトは、ゲームにサウンドを組み込む際に設定し、実行するための様々なソースファイルを含んだサブフォルダー群から成り立ちます。

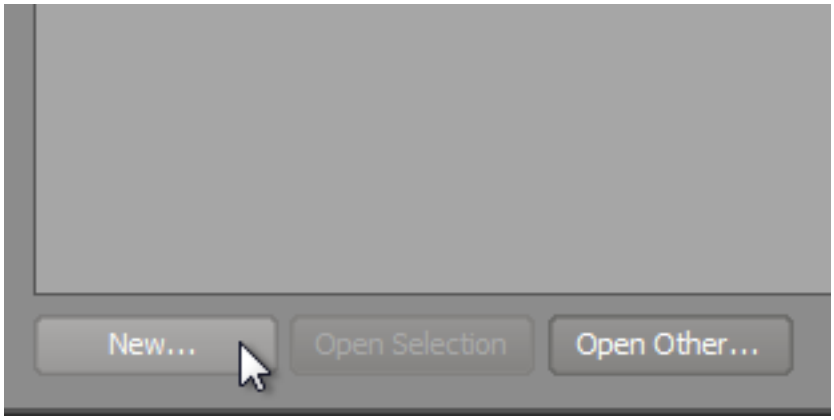
1. Wwise Launcherアプリケーションで、Wwiseタブを選択してWwiseアプリケーションを起動します。

   Wwise Project Launcherが起動し、利用可能なWwiseプロジェクトが表示されます。



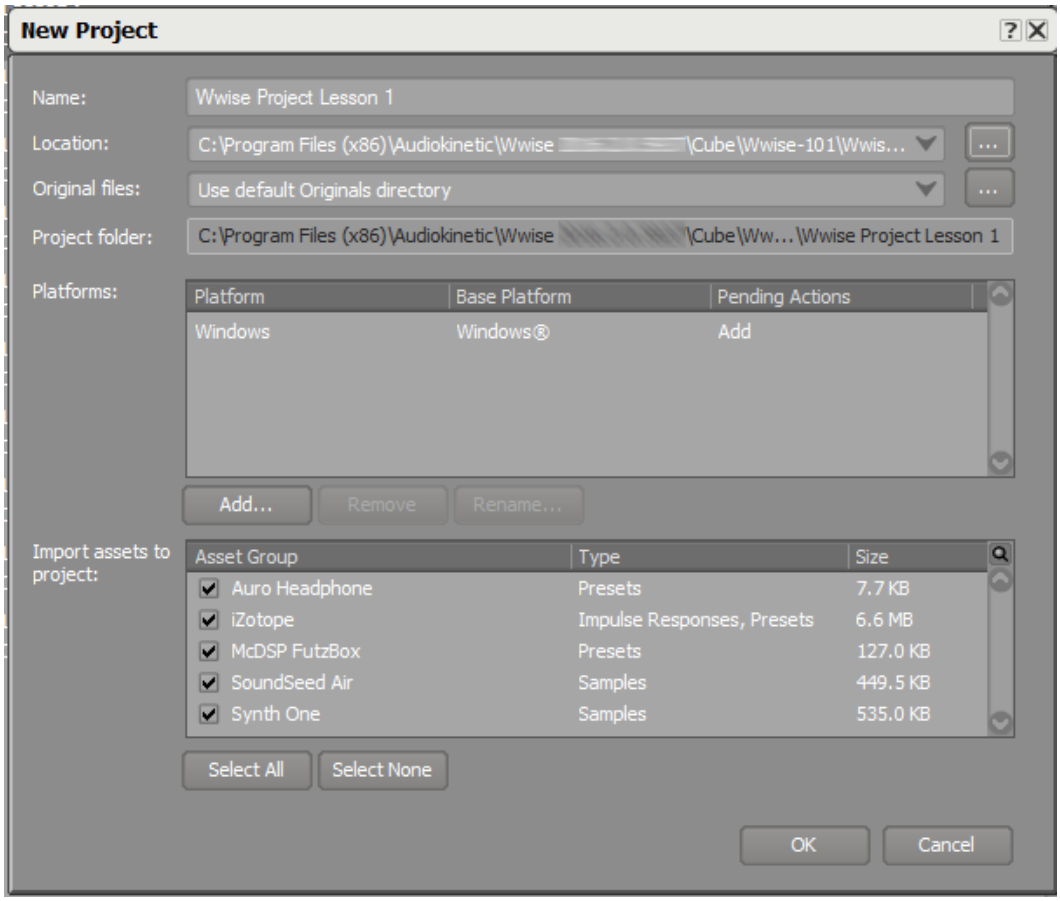
   ここでみられるプロジェクトの数は、担当者がそのコンピューターで行っている他プロジェクト数により変動します。現時点では、ここに何も表示されません。今回のケースでは、このレッスン用に新しいプロジェクトを最初から作成するため気にする必要はありません。

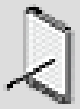
2. Project Launcherの左下にある **New**をクリックします。



ダイアログウィンドウが開き、プロジェクトの名前とプロジェクトの場所を指定します。Original Filesのフィールドは、後のレッスンでインポートするオーディオアセットを保存する場所を指定します。Wwiseに存在する他のオプションプリセット例は、チェックマークがある下のボックスに示されるサポートファイルを使用します。これらは選択された状態で問題ありません。

3. NameフィールドではWwise Project Lesson 1と入力します。ロケーションフィールドの右にある[...]ボタンをクリックし、あなたのWwise-101 フォルダを開き、 Lesson 1 フォルダを選択します。

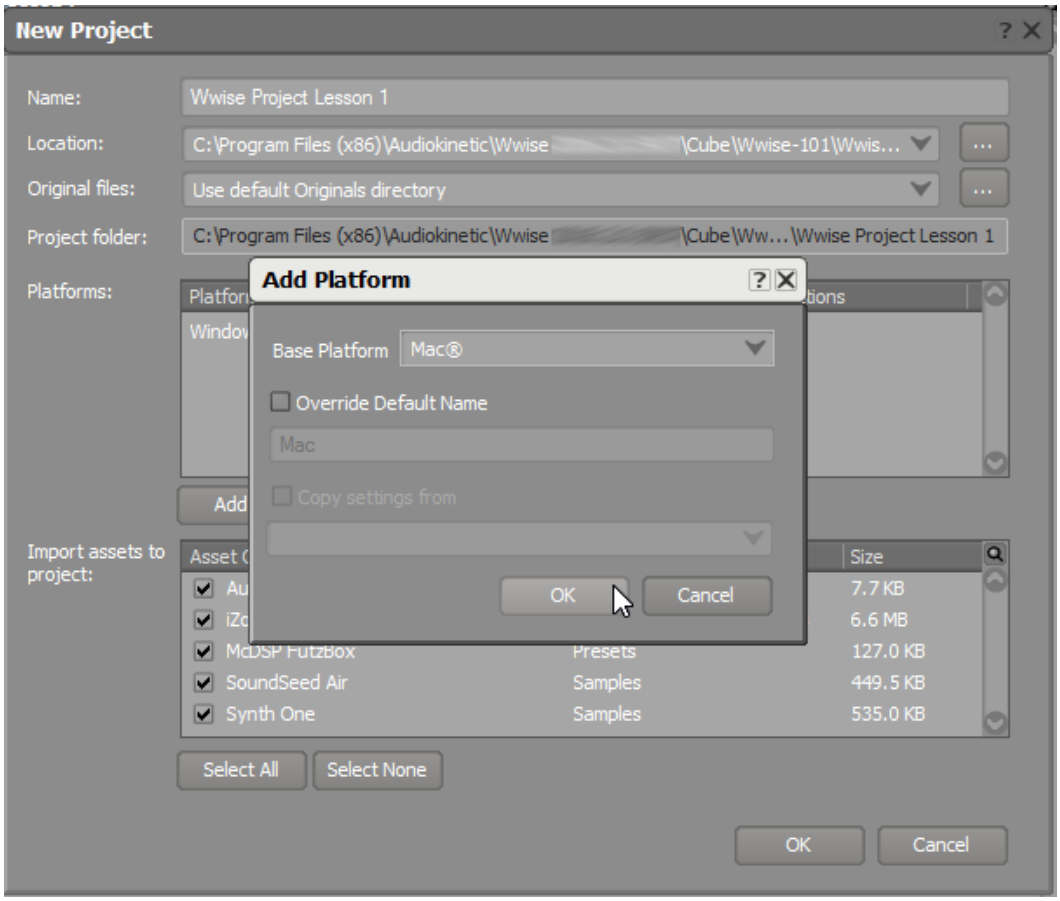




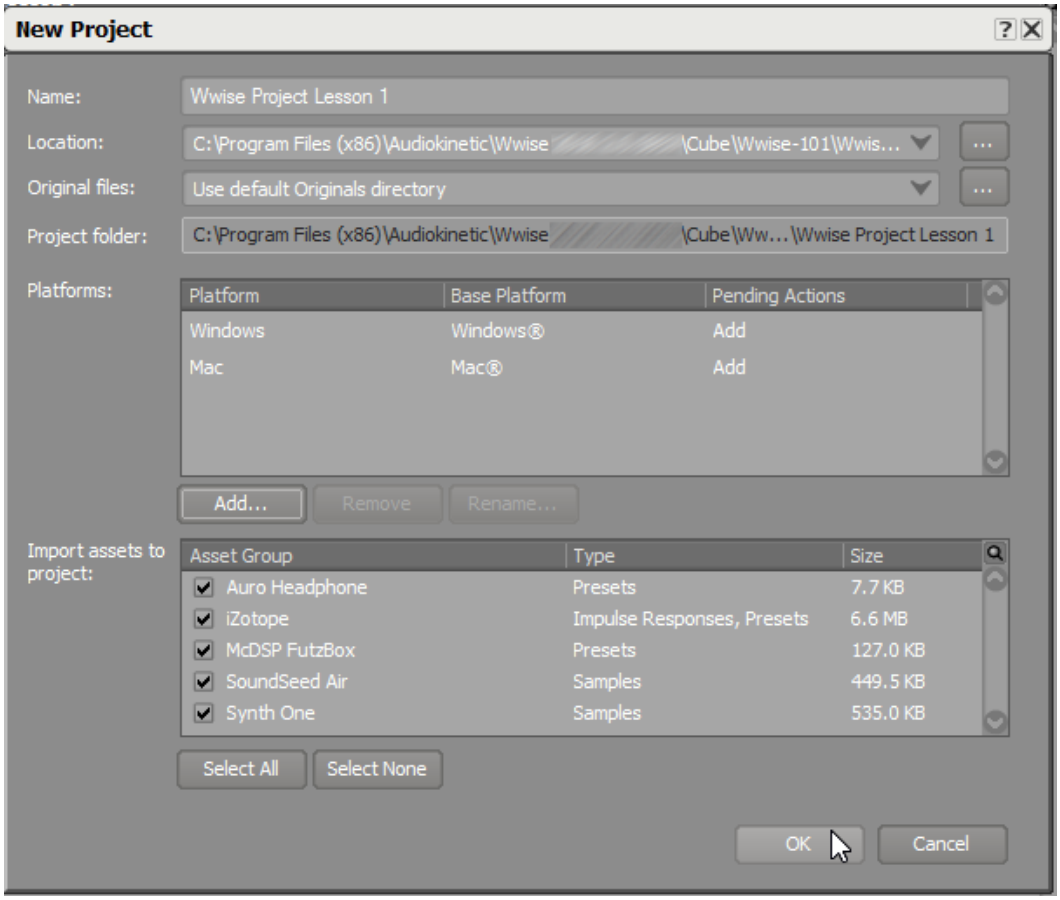
注記

Mac ユーザー: 選択する際に、ロケーションフィールドにパスが`Y:\Documents\WwiseProject\Wwise Lessons\Lesson 1\`と表示されます。この Y: パス指定が発生するのは、Mac版のWwiseがPCエミュレーター環境で動作しており、このアルファベットをドライブ指定に使用しているからです。

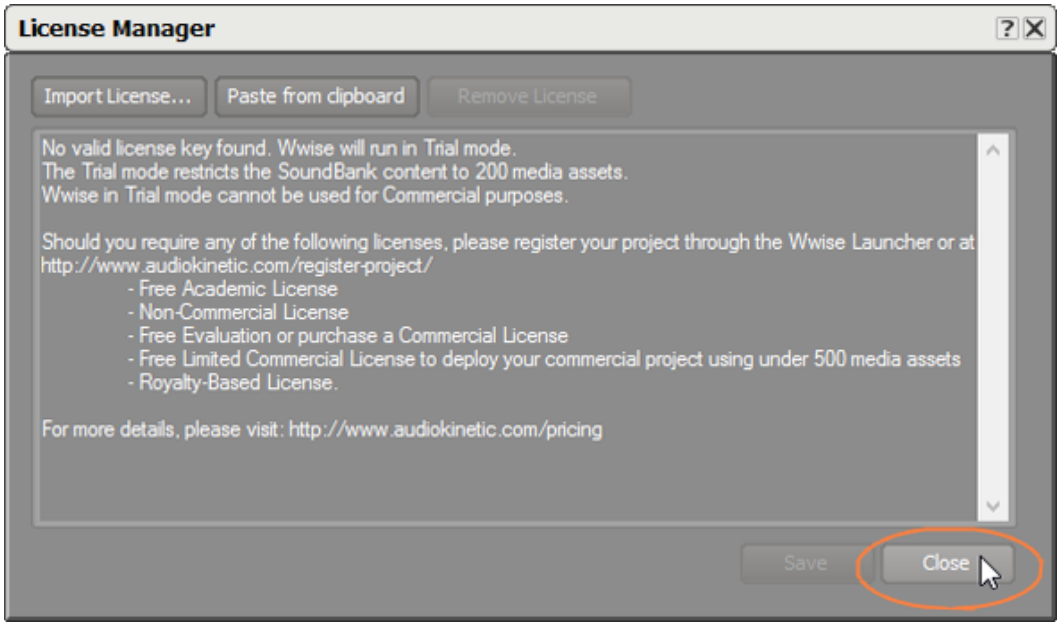
4. PlatfrmsパネルでAdd...をクリックします。Add Platformダイアログが開きます。**Mac®**を基本プラットフォームとして選択します。



Add Platformダイアログで**OK**をクリックし、New Projectダイアログで再びクリックします。

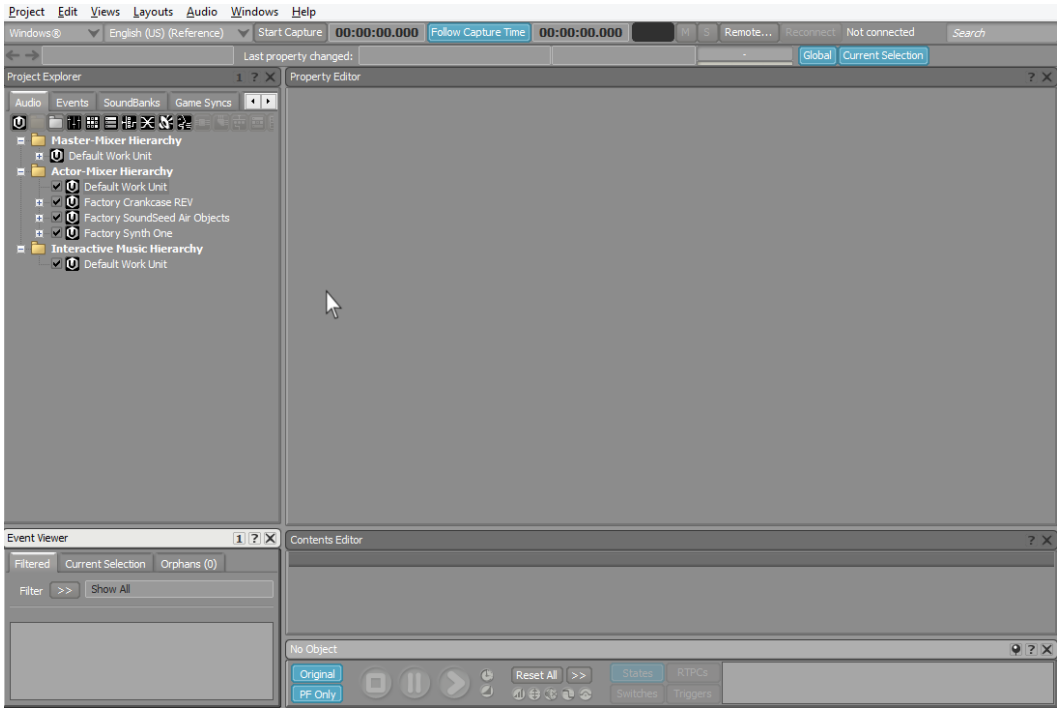


5. Wwiseが新しいプロジェクトフォルダーをビルドする際、多少の時間を要します。これが完了すると、Wwise License Managerウィンドウが開き、Wwiseがトライアル版で起動していることを示す説明が表示されます。このトライアル版のWwiseでは、全てのSoundBankで扱うことの出来るメディアアイテムの数が最大200個に限定されている以外には、正式ライセンス版と違いはありません。正式ライセンス版については、ライセンシングに関してよくある質問をまとめた、このページをご確認ください： https://www.audiokinetic.com/licensing/faq/. このレッスンの後半でSoundBankについての詳細を学ぶことができますが、Wwiseの学習目的には本トライアル版で提供するマテリアルで十分です。



6. License Manager ウインドウで、**Close**をクリックします。

   Wwiseのソフトウェアインターフェースを見ることができます。



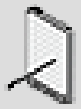
お使いの環境の画面サイズや、解像度で、表示が多少異なるかもしれませんが、メインメニューバーがあり、その下にツールバーが確認できます。これらは常に同じ場所で表示されます。その下に、ビューと呼ばれる複数のエリアが確認できます。各ビューは、サウンドがどのようにゲームに組み込まれるかを可視化したり操作するための機能を提供します。Wwiseでは、約40種

類のビューが存在します。これは多すぎるかも知れませんが、これらすべてのビューがWwiseで何の役割を担うかを把握することは必須ではありません。実際、Wwiseはチームの複数メンバーで使用されますが、一人でこれらすべてのビューを使うことはまれです。メンバーはそれぞれの担当箇所があり、Wwiseの特定のセクションのみを必要とします。

ゲームのプロファイリング

Cubeデモにサウンドを組み込む際、最初にどのような情報がゲームからWwiseに送られるかを理解しなければなりません。これを行うためには、インストールに関する章でテストした Profiler バージョンの Cube デモを起動させなくてはなりません。

1. Cubeデモを起動します。



注記

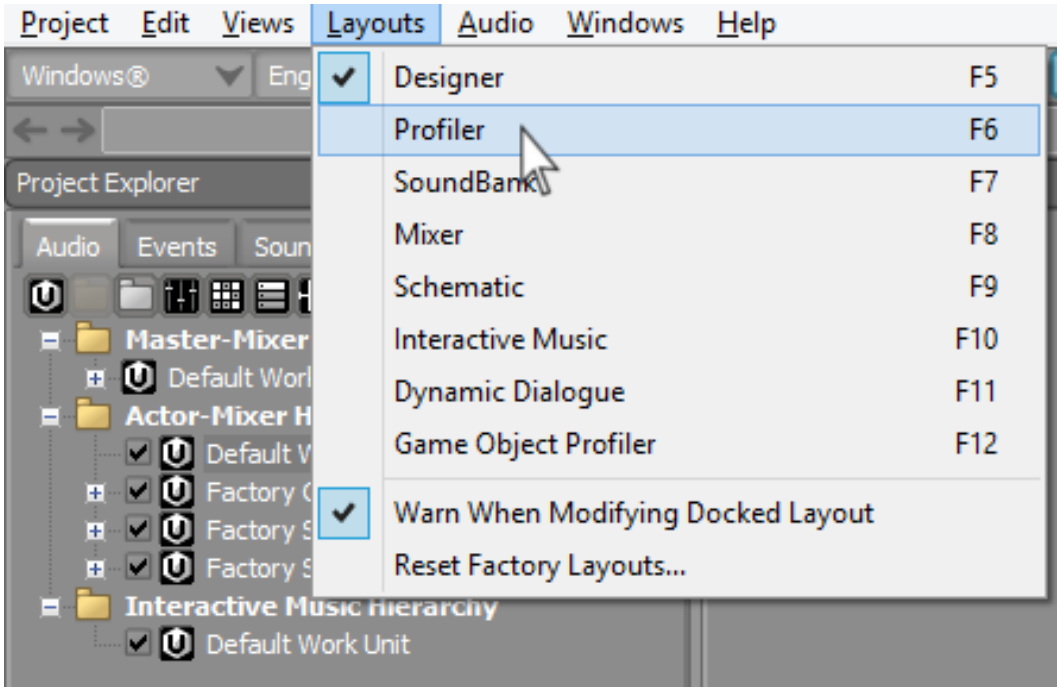
Cubeデモを起動する方法が分からなければ、インストールに関する章の最後にある、お使いのコンピュータのOS用に記述された Playing Cube 演習を参照してください。

ここでWwiseに戻ります。すでにお分かりの通り、ゲームとWwiseを常にスイッチする必要があります。WindowsではAlt+Tabを押し、MacではCommand+Tabでプログラムを切り替えることができます。

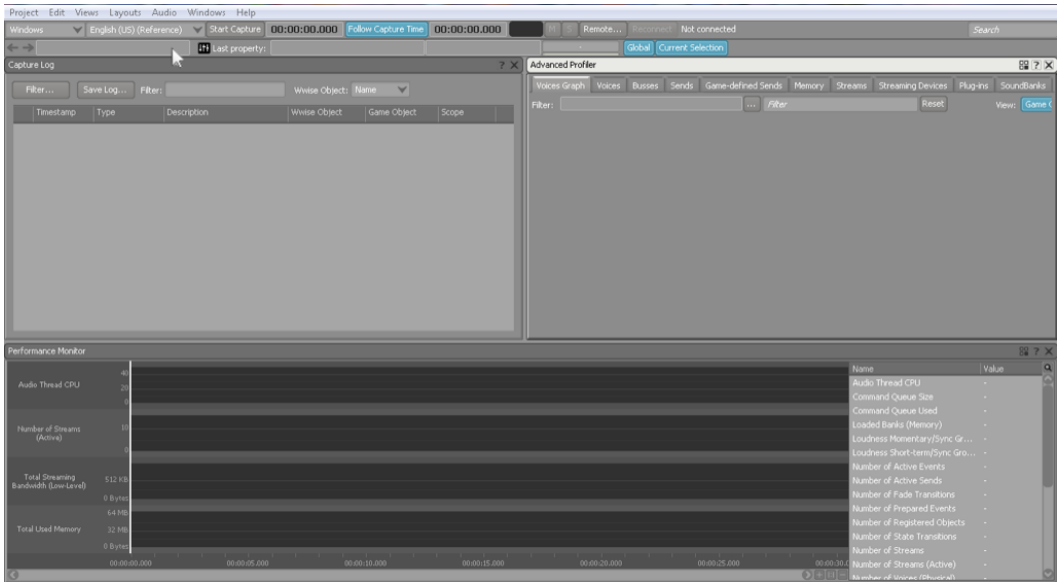
2. Windows環境では**Alt+Tab**を、 Macでは**Command+Tab**でWwiseに戻ります。

ゲームから受け取った情報をレビューするためのビューが多数存在しますが、1つ1つ開くのではなく、レイアウトを使用して一度にこれらを開くことができます。レイアウトは、共通して使われることが多いビューの事前定義された集合になります。

3. メインメニューで**Layouts**をクリックし、 **Profiler**を選択、もしくは F6を押します。

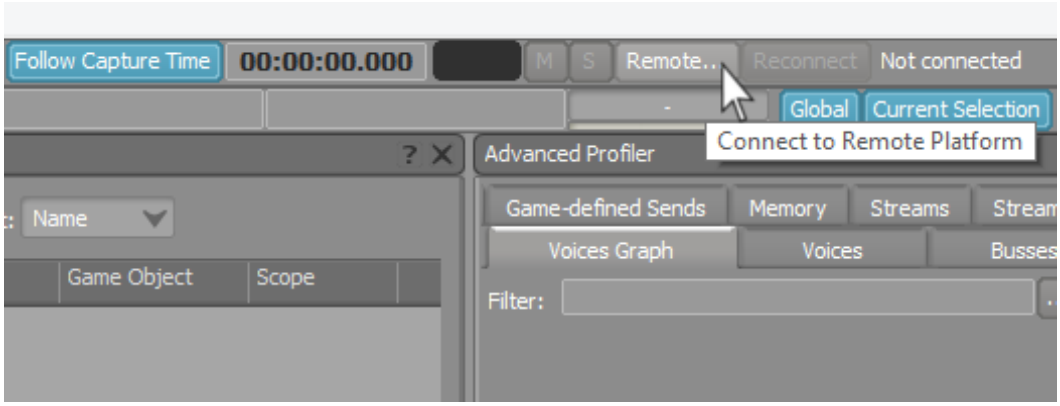


Profilerレイアウトが表示されます。このレイアウトのビューを使用して、ゲーム中で生成されるメッセージについて情報を確認し、サウンドエンジンのパフォーマンスについての詳細をモニターすることができます。

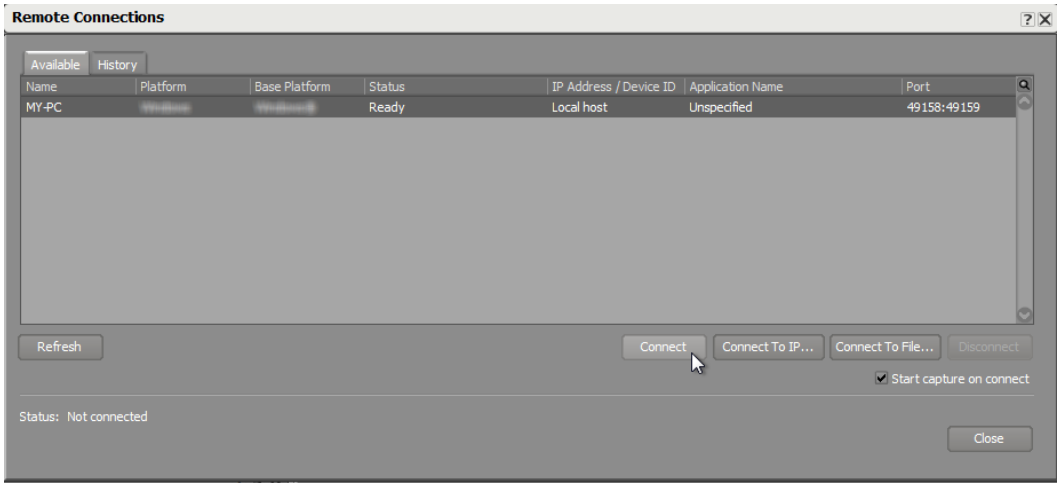


ゲームでから送られてくる情報を表示するために、左上のCapture Log ビューを使用します。この情報にアクセスするためには、バックグラウンドで起動しているCubeデモにWwiseを接続する必要があります。

4. ツールバーの**Remote**をクリックします。



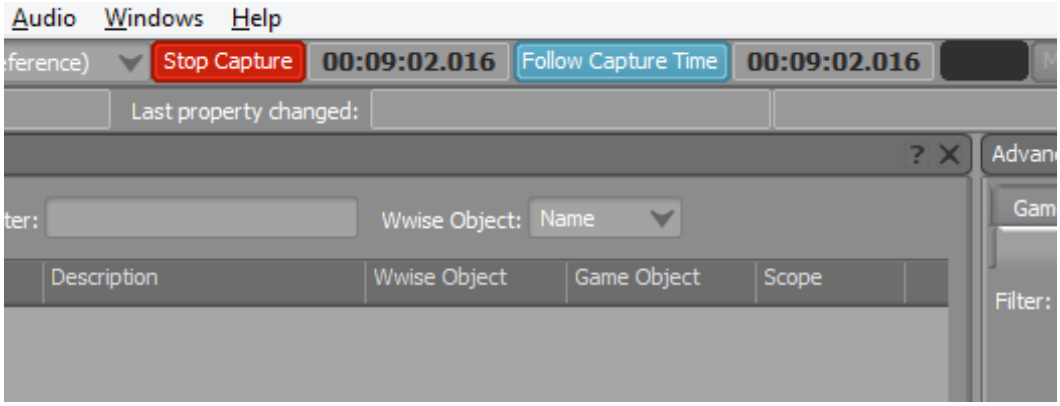
Remote Connections ウインドウが開き、ネットワーク上のWwiseサウンドエンジンを使用し、Wwiseとの通信可能なゲームを実行しているコンピュータ (ローカルのコンピュータを含む) のリストが表示されます。



5. リストからお使いのコンピュータを選択し**Connect.**をクリックします。

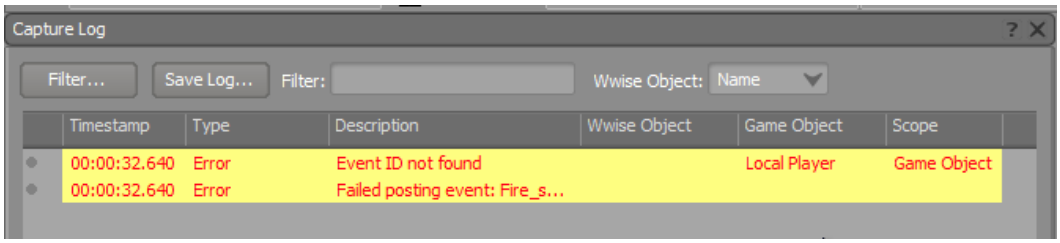
   ウィンドウが閉じ、 **Start capture on connect** のチェックボックスが選択されているため、ツールバーのCaptureボタンが赤くなり、カウンターが動きます。

   キャプチャーとは、レコーディングプロセス(リアルタイム)を意味しており、ゲームプレイに関連したWwiseサウンドエンジンからの情報のログになり、サウンド開発における大変貴重なアセットになります。

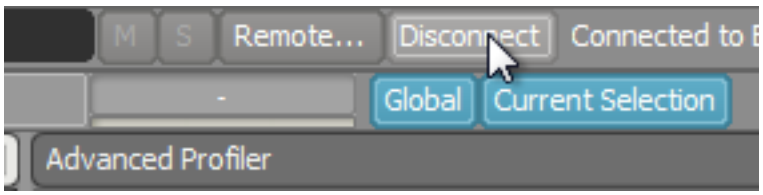


6. Cubeデモに戻り、一度ショットガンを撃ちます。
7. Wwiseに戻り、Capture Logを見ます。

   Captrue Logビューで、CubeからWwiseにショットガンを撃った事を伝えるいくつかの情報が届いていることを確認することができます。

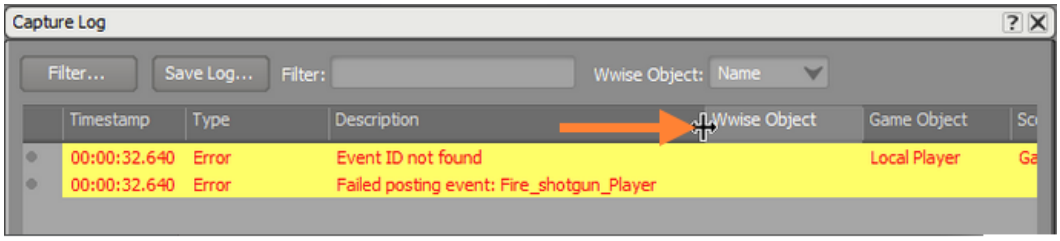


   ゲームに接続している状態では、一部のWwiseパラメーターを調整できません。目的の Capture Log 情報を確認できたら、ゲームとの接続を切るようにしてください。
8. **Disconnect**をクリックします。



   キャプチャーがストップし、Wwiseとゲームの接続が切断されます。そして何が起こったかを確認することができます。

   表示されている情報は、エラーを示しており、Wwiseは受け取ったメッセージをどのように処理するかを理解していません。Game Calls これらのメッセージは一般的にゲームコールと呼ばれており、様々な種類のコールがトリガーされることがあります。コールの詳細に関しては後で学ぶことができるため、今はエラーの詳細をより詳しく見る必要があります。
9. Description欄のヘッダーを右側にドラッグすることにより、エラーメッセージ全体を見ることができます。



Descriptionエリアのメッセージは、イベントと呼ばれます。イベントはWwiseゲームコールの一種であり、ゲームエンジンがWwiseオーディオエンジンに送り、ゲームで何かが起こったことを示します。多くの場合、イベントがWwiseに送られる場合、サウンドをトリガーするために使われたり、プロパティーを変更したり、もしくはサウンドの停止をしたりします。イベントにはその用途を識別するために名称が指定されます。このケースでは、プレイヤーがショットガンを打つ動作に関連付けが明確にわかるイベント名Fire_shotgun_Playerが与えられています。

Local Playerを明記するGame Object欄があることに気が付きます。これは、ゲームのどの物に対しメッセージが関連付けられているかを明確にします。例えば、ゲーム中にショットガンを所有しているキャラクターが複数存在するとします。ゲームエンジンは、発射サウンドをどのキャラクターから行うべきかを適切に知る必要があります。例えば、50m離れたモンスターがショットガンを撃った場合、そのサウンドは、主人公が撃った音量と同じであってはなりません。これに関しては、 レッスン 4：イベントの作成 で詳細を学ぶことができます。

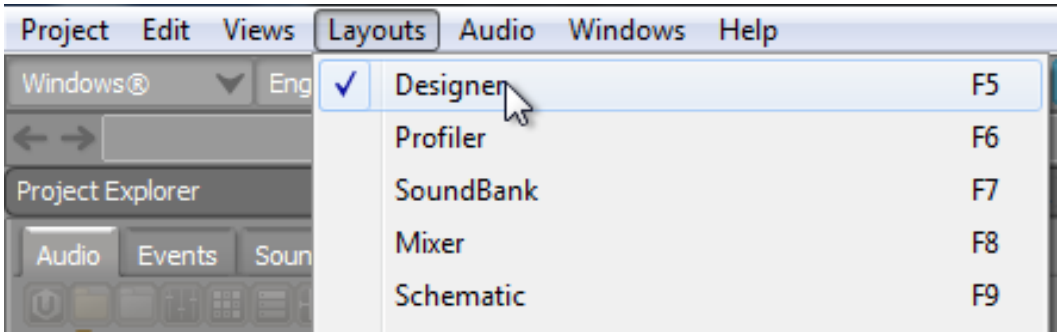
今は、Wwiseがこのイベントを使って何を行うべきかがわからない状態です。今からこの問題を解決します。

イベントの作成

ゲームで起こった事柄全てにオーディオの反応が必要になるかも知れません。プログラマーは予めWwiseに何が起こったかのメッセージを伝えるプログラムコードを追加しています。このメッセージはゲームコールと呼ばれます。ゲームコールはシンプルなメッセージで、「ショットガンが撃たれました」と伝えています。しかしながら、実際にはテキストやナンバーの文字列として送られます。このメッセージがオーディオエンジンに伝わると、ゲームオーディオデザイナーが設定した事柄が起こります。

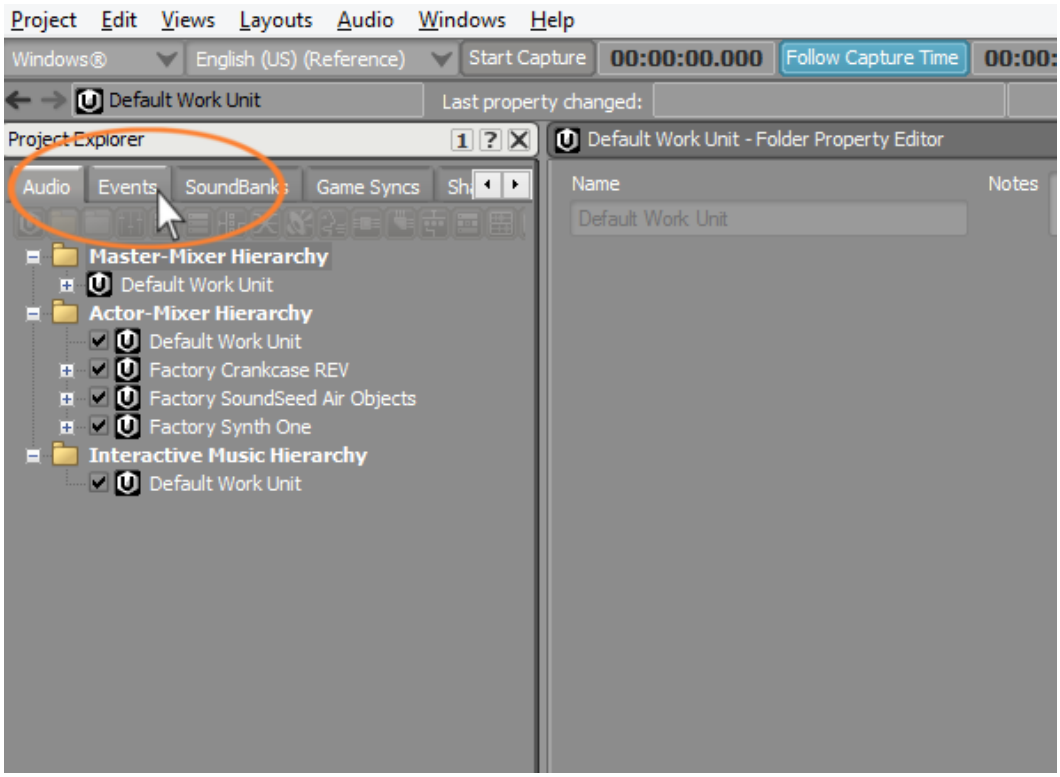
まず最初にEventを作成して、入ってくるゲームコールをWwiseが受け取れるようにする必要があります。ゲームエンジンとオーディオエンジンのキャッチボールのようなものです。ゲームエンジン側から投げられたボールはゲームコールと呼ばれ、イベントはWwiseでデザインされたゲームコールを受け取るためのオブジェクトです。重要なことは、それぞれのゲームコールはWwiseで受け取るために同じ名前のイベントが必要になるということです。

1. メインメニューから**Layout** > **Designer**を選択、もしくは**F5**を押します。



2. プロジェクトエクスプローラービューで、Eventタブをクリックします。

   ゲームエンジンから伝えられるWwiseが期待するイベントゲームコールはEventタブになります。



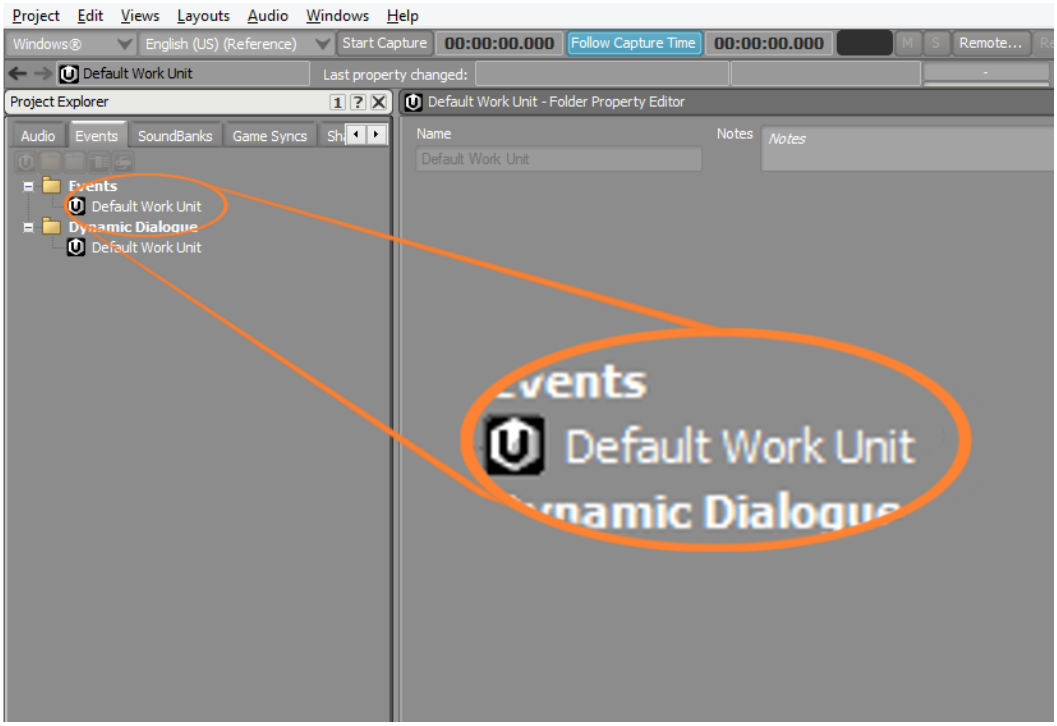
   プロジェクトエクスプローラーのEventsタブでは、Eventsと名前付けられたフォルダーと、その中にDefault Work Unitと呼ばれるオブジェクトがあります。Work unitsはWwiseの基礎となります。Work unitsはプロジェクトの特別なセクションやエレメントに関連した情報を内包しており、プロジェクトを管理するのに役に立ちます。

   多くのゲームでは、多くの人や会社がゲームのそれぞれのパーツを分業します。例えば、1つのチームはすべての武器のサウンドを担当し、違うチームはアンビエントサウンドを担当します。このシナリオでは、それぞれのチームはWwiseのコピーを所有することになり、割り当てられたイベントがあるwork unitを作成することができます。プロダクションの後期で、ゲームで必要なすべ

ての要素を持ち込むため、複数のwork unitsを1つのプロジェクトに統合することができます。

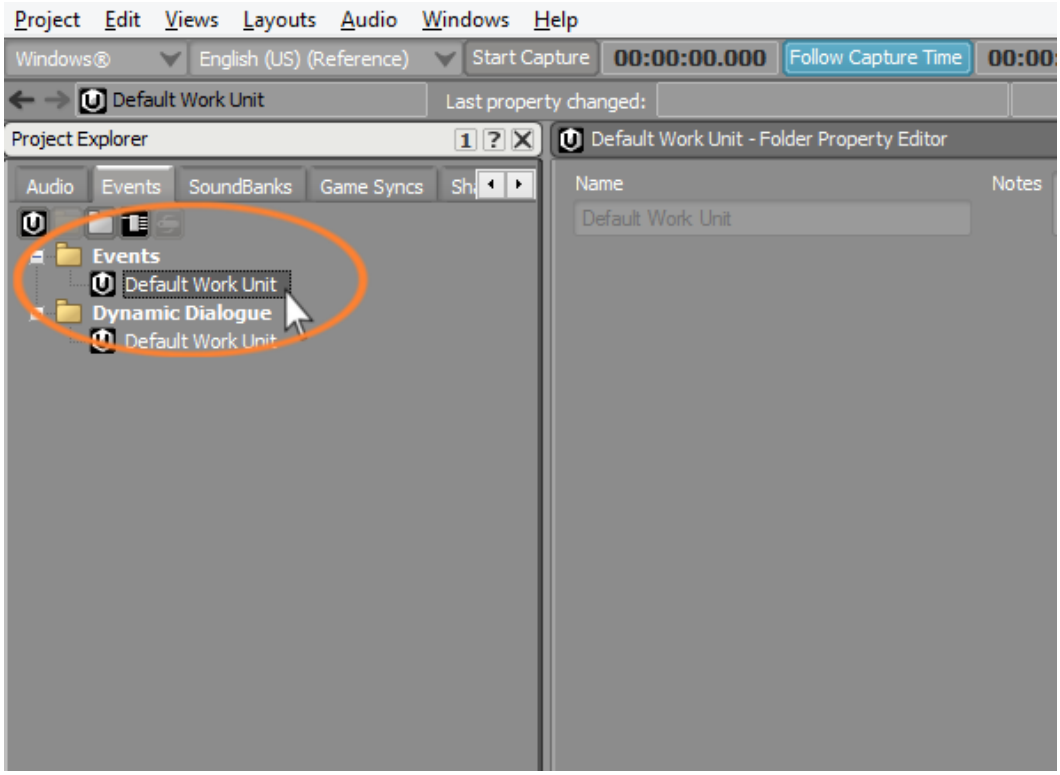
注記

Work unitsはWwiseのプロジェクト構造体内に作成されたXMLファイルです。



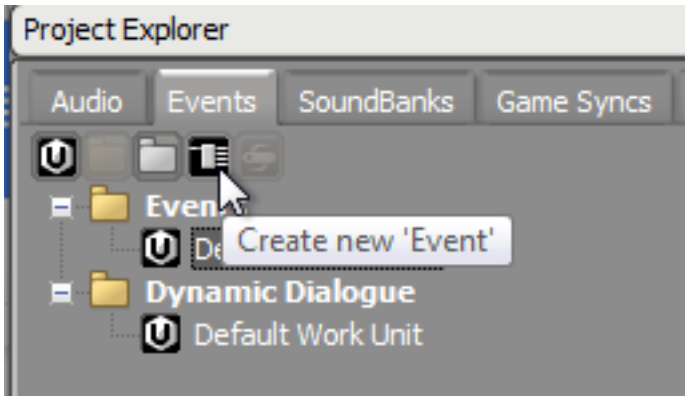
Wwiseで作成する全てのものは、オブジェクトと呼ばれます。注意いただきたいのですが、このレッスンの最初で説明したゲームのアイテムに関連したゲームオブジェクト用語と混合しないようにしてください。Wwiseのインターフェース上のオブジェクトは、小さい四角のアイコンで表示されます。20種類以上ものオブジェクトが存在し、それぞれはWwiseでどのようにサウンドを作成しコントロールするかを決定するユニークな機能を提供します。オブジェクトは階層内に存在し、通常work unitsは階層の最上位に位置します。ビルを構成するレンガのように、これらのオブジェクトは実用的なアプリケーションに使用することのできる構成単位で、複雑に、そしてクリエイティブにアレンジすることができます。ここでは、Default Work Unit内に1つのイベントオブジェクトを作成します。それは、Cubeデモのゲームエンジンから送られてくるショットガンのゲームコールを受け取るためのグローブのような役目をします。

3. Eventsフォルダー内のDefault Work Unitをクリックします。



オブジェクトが選択されると、プロジェクトエクスプローラーのアイコンバーが表示され、選択されたオブジェクトに内包することができるオブジェクトの種類を示します。アイコンにあわせると、ツールティップがアイコンによってどのようなオブジェクトなのかを示します。

4. 4つ目のアイコンにあわせ、Eventオブジェクトであることを確認します。



5. Eventアイコンをクリックし、Default Work Unit内に新しいEventオブジェクトを作成します。

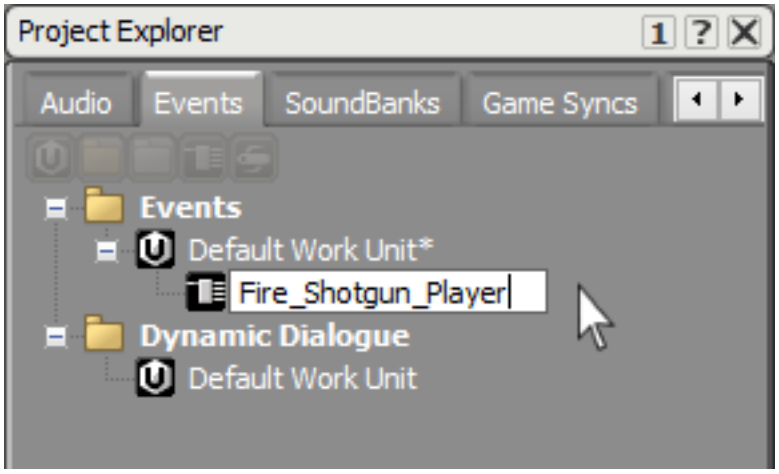
   新しいEventオブジェクトが作成され、その名前を記入します。

   Eventオブジェクトの命名には細心の注意を払う必要があります。Eventオブジェクトの名前は、ゲームエンジンから受け取るゲームコールの名前と一致す

る必要があります。電話番号と同じであり、番号が間違っている場合はつながりません。

うまく機能するためには、通常サウンドデザイナーが、サウンドエンジンがどのような事柄に配慮すべきかを決め、それぞれの事柄に向けてWwise上でイベントを作成します。サウンドデザイナーはプログラマーにEvent名を伝えます。そうすることにより、プログラマーはゲームエンジンで同じ名前を使ってゲームコールを送るようプログラムすることができます。このシナリオでは、Cubeデモのゲームエンジンにプログラムが行われており、コール名はすでに決まっています。そのため、ショットガンが撃たれるときには、Fire_Shotgun_Playerが使用されます。

6. Fire_Shotgun_Player と入力し**Enter**を押します。



注記

Wwiseは後にあなたのプロジェクトを元にソフトウエアプログラムを生成します。 コードでは全てのオブジェクト名の参照は小文字で行なわれますので、オブジェクト名における大文字、小文字の区別はありません。

ティップ

イベントエディター内でオブジェクトを選択することで、関連するプロパティを確認でき、右側にあるイベントエディターでビューのタイトルバーに Fire_Shotgun_Player と表示されています。異なるオブジェクトには、それぞれ複雑さの異なったエディターが用意されています。多くがNotesフィールドを持っており、これは非常に重要です。なぜならイベント名の混乱を防ぐための情報を追記することができるからです。Fire_Shotgun_Playerのような名前において、イベントはショットガンの発射に関連付けられますが、全てのイベントがこのように直感的に命名されているとは限りません。Notesはあなただけでなく、他の作業者があなたの作業を確認する際に、あな

たが作成したものを理解するのに役立つ情報を提供する手段を提供します。

サウンドのインポート

イベントをセットアップし、ショットガンのゲームコールを受け止める準備が整いました。それではイベントを受け取った際に鳴らしたいサウンドを持ち込みます。Wwiseでは様々な方法でサウンドを作ることができます。サウンドのシンセサイズも可能です。しかしながら、一番一般的な方法は録音されたオーディオファイルを使うことです。ショットガンを録音する必要はなく、そのブラストサウンドのオーディオファイルは提供されます。これをプロジェクトに持ち込みます。

サウンドは、通常Designer layoutのプロジェクトエクスプローラーのAudioタブで管理されます。

1. プロジェクトエクスプローラービューで、Audioタブを選択します。

   ゲームのサウンドスケープをデザインする場合、多くの時間をこのDesignerレイアウトのAudioタブで費やすことになります。Audioタブには3種類の階層があり、それぞれは特有のオブジェクトを提供し、トリガーやWwiseでどのようにオーディオが扱われるかなどの様々なタスクを構成します。

   ミュージックは例外とし、サウンドActor-Mixer HierarchyのWork Unitで管理します。

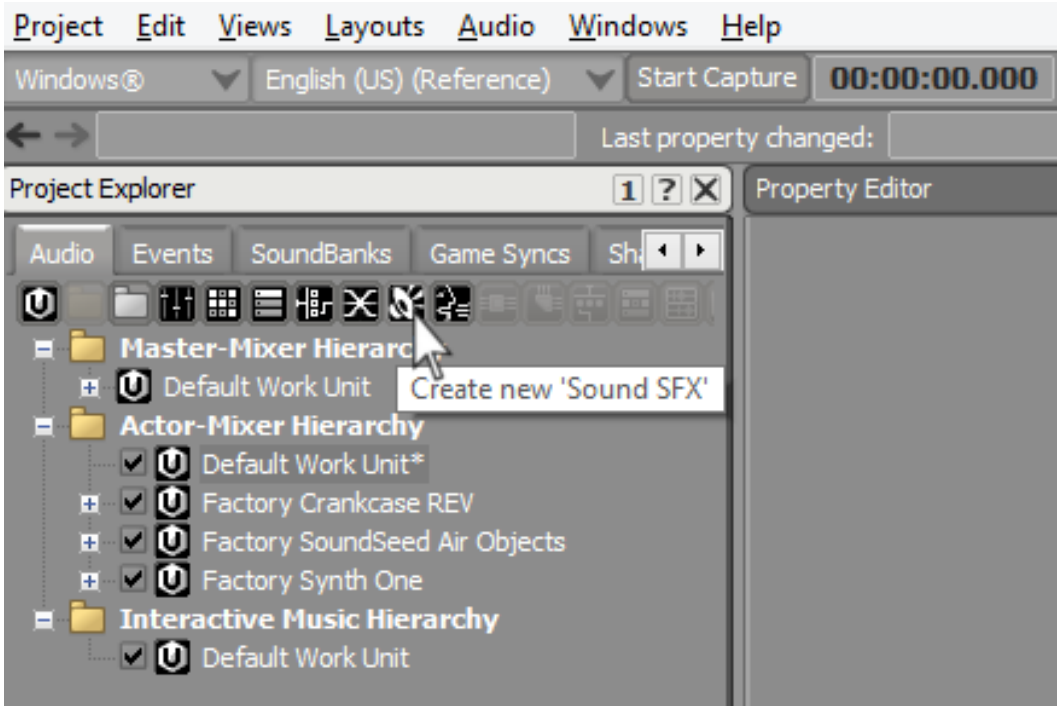
   次のいくつかのセッションで学びますが、Actor-Mixer Hierarchy内では、様々なオブジェクトが作成できます。しかしながら、単純に任意のオーディオファイルを再生したい場合は、Sound SFX オブジェクトを使うことによって可能です。

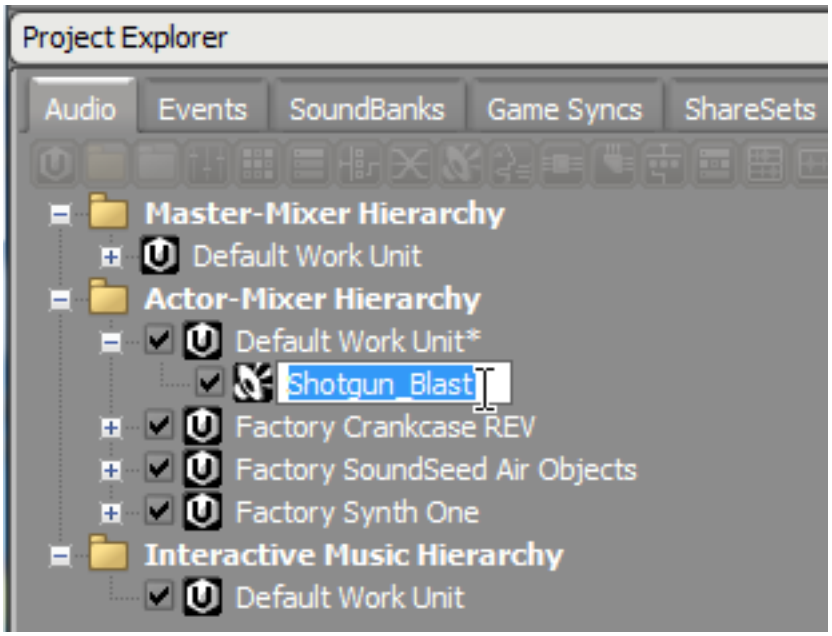
   注記

   これはまた、Sound Voice オブジェクトを使うことによっても実現可能ですが、これには多言語対応のゲームで特定のローカライゼーション機能が用意されているので通常ダイアログに使用されます。

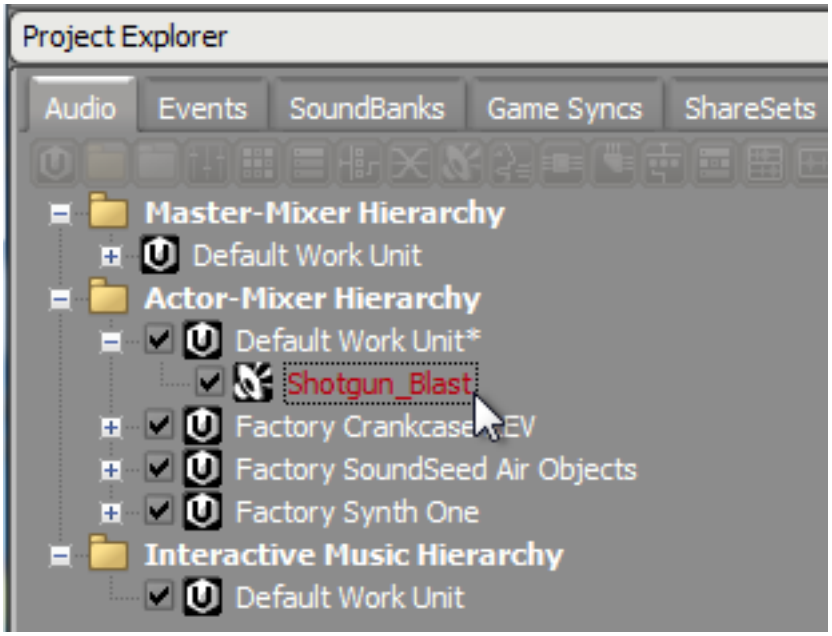
2. 提供されているオブジェクト列で、Sound SFXアイコンをクリックします。



3. Sound SFX オブジェクトをShotgun_Blastと命名します。



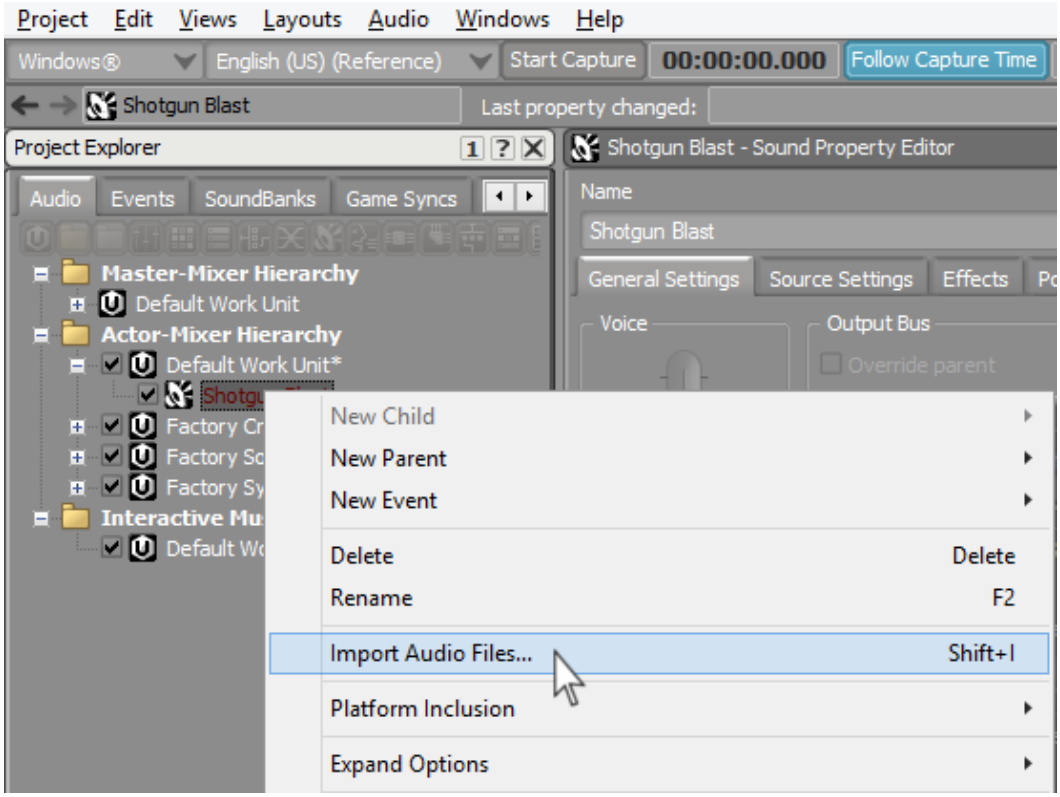
オブジェクト名が赤色で表示されます。



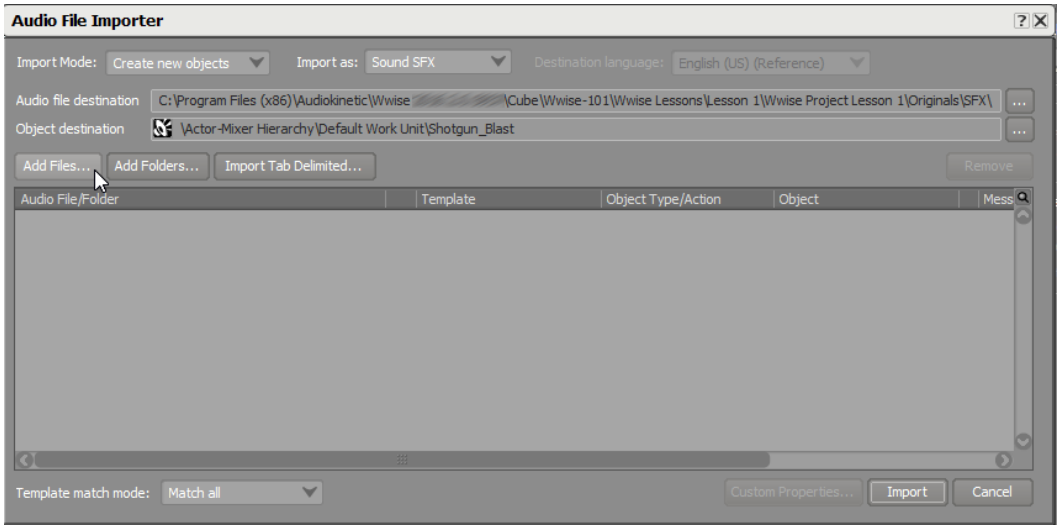
赤色で表示される名前は、SFXオブジェクトに関連付けられたオーディオファイルが存在しないことを意味しています。オーディオファイルを加える必要がありますが、先に進む前にSFXオブジェクトは、オーディオファイルに直接対応しているわけではないことを理解することが重要です。そうではなく、オーディオファイルが再生するチャンネルを示しています。デジタルオーディオワークステーション（DAW）のチャンネルと同等と見なすことができます。チャンネルは、チャンネルを通るオーディオトラックに格納されたオーディオファイルを操作する様々なコントロールを持っています。これを一度理解すれば、Sound SFXオブジェクトにオーディオファイルを追加する準備ができたことになります。

オブジェクト上での右クリックは、Sound SFXオブジェクトにオーディオファイルをインポートすること事を含め、オブジェクトで行うことができる様々なオプションを提供します。

4. Shotgun_Blast SFX オブジェクトを右クリックし、**Import Audio Files**を選択します。



5. **Add Files**をクリックします。



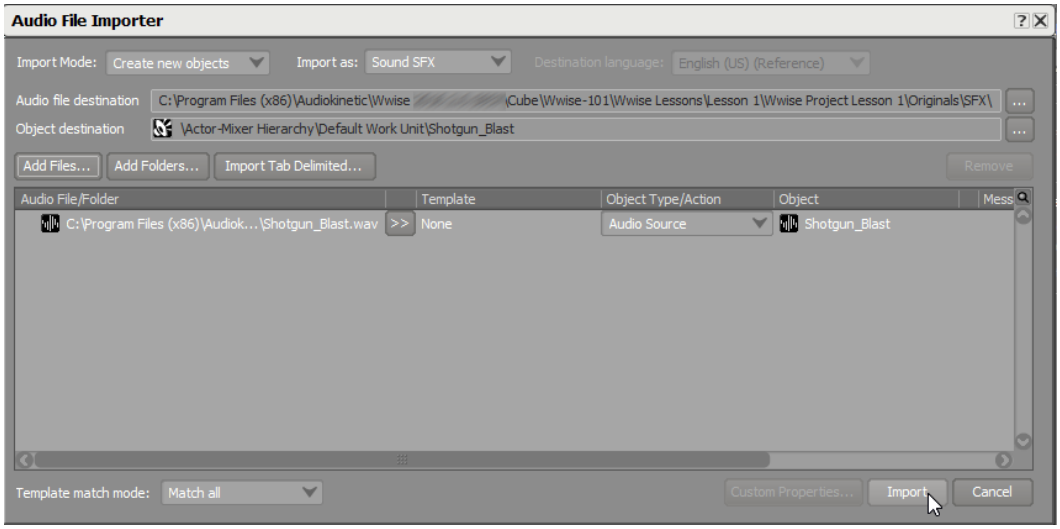
ダイアログボックスが開き、インポートしたいファイルの選択をします。

インポートするファイルは、WAVファイルです。Waveファイルは比較的大きく、MP3ファイルへのコンバート、サンプルレートの変更、及びビット数の変更を最初に考えるかもしれません。今回はそれは不要です。現段階では、サイズを気にすること無く、最も品質の良いオリジナルファイルを波形ファイルとしてWwiseへインポートします。Wwiseの利点として、ゲームにインテグレー

トする前に、ファイルサイズの最適化をどのように行うかを決めることができます。Wwiseには、これに関する豊富な機能があり、レッスン7で詳細を学びます。写真家が良い例になりますが、彼らは常に一番よい25メガピクセルのオリジナルを保存し、第3者にイメージを提供するときに編集や圧縮を考えます。

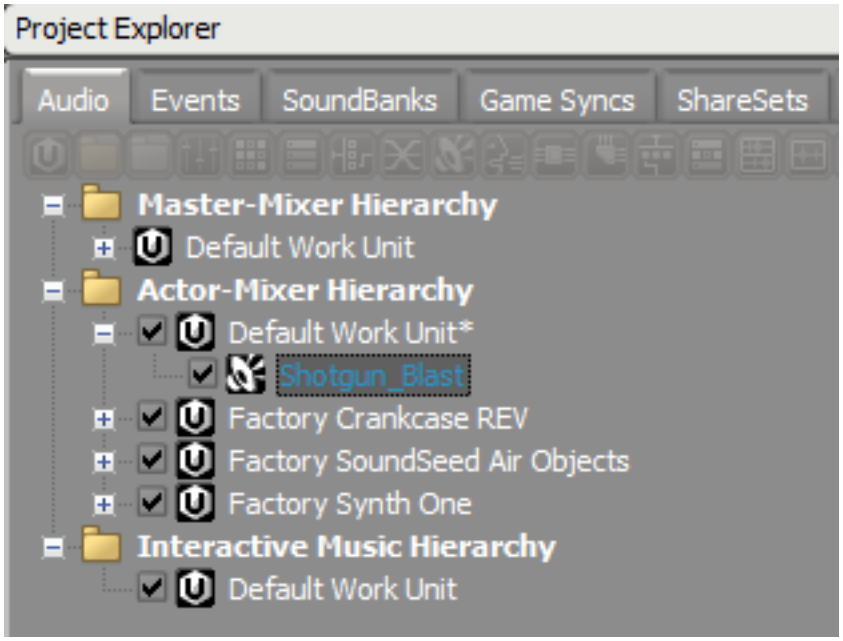
6. `Wwise-101/Lesson 1/Audio files for Lesson 1/`を開き、Shotgun_Blast ファイルを選択し、**Open**をクリックします。

   Audio File Importerが開き、SFX オブジェクトにどのオーディオファイルを取り込むかを確認します。



7. **Import**をクリックします。

   Audio File Importerウィンドウが閉じ、Shotgun Blast SFX オブジェクトが青色に変化したことを確認することができます。



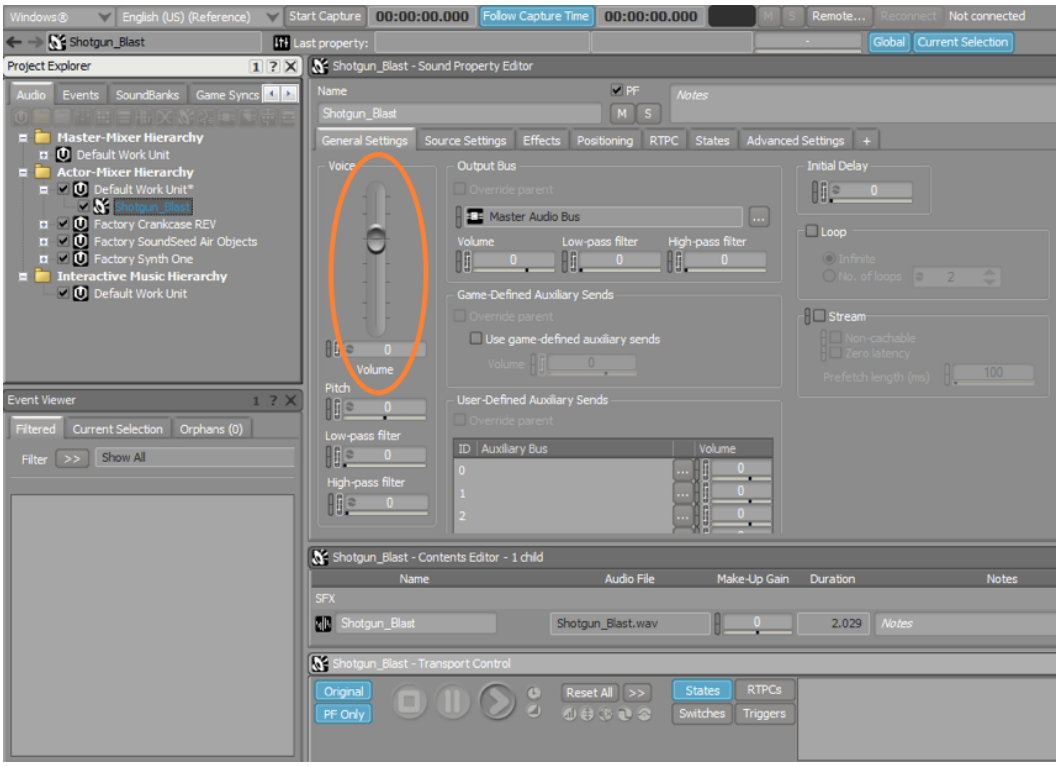
青色は、オーディオソース（今回はwavファイル）がSFX オブジェクトに適切に関連付けられたことを示し、オーディオファイルはインポートされたオリジナルのフォーマットを参照しています。

注記

作業が進むにつれて、ある一部のSound SFX オブジェクトの名前が青色で表示される一方、白色で表示されるものがあります。色は関連付けられたファイルがコンバージョンプロセスを通りして最適化されたかどうかを示します。コンバージョンプロセスは通常サウンドバンク作成時に行われ、後のセッションで学ぶことができます。白色のオブジェクト名は、すでにコンバージョンが行われていることを意味しており、青色のオブジェクトはコンバートが行われていないことを意味しています。この時点では、青色から白色への変更があっても問題はありません。後のレッスン7で最適化や変換のプロセスについて学ぶことができます。

それでは、作業をおこなったサウンドがシステムを通して再生されるかをテストします。

8. Shotgun_Blast Sound SFX オブジェクトをクリックし、選択されていることを確認します。



Property Editor ビューが表示され、Shotgun_Blastのボリュームフェーダーなどのオーディオコントロールが表示されます。

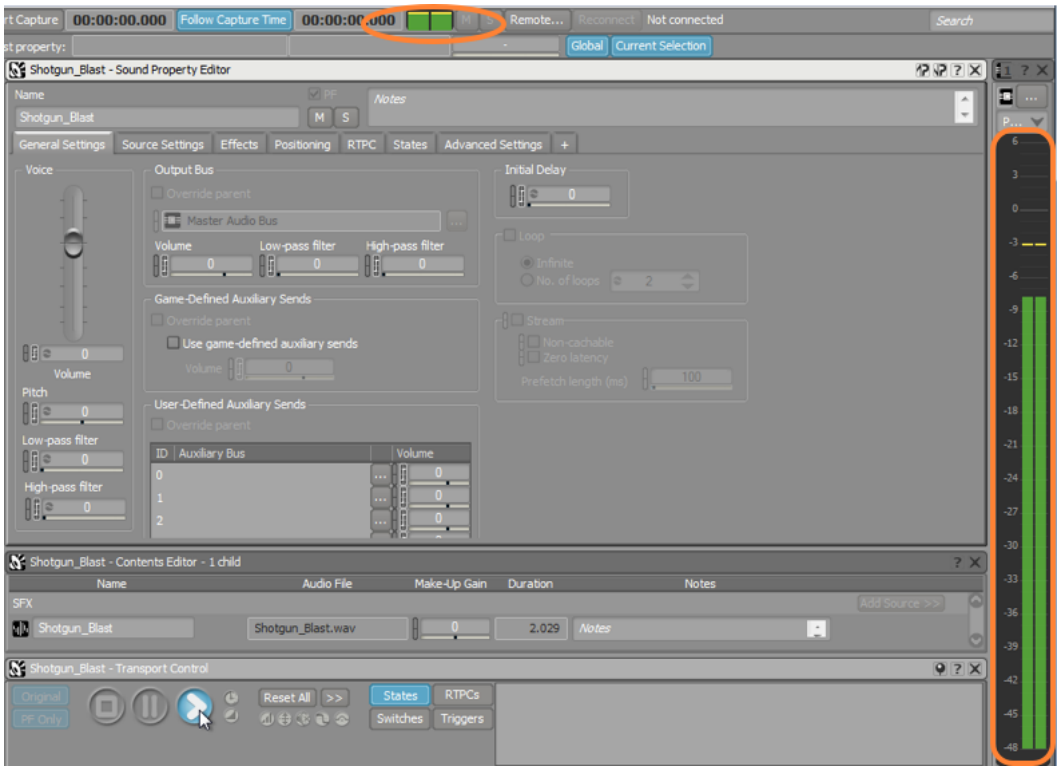
そして、Shotgun_Blast の名前をTransport Control ビューで見ることができます。これは、ゲームでプレイボタンを押したときに、ゲームで再生されるサウンドが聞こえることを意味します。

ティップ

再生ボタンのショートカットはスペースバーです。

9. Transport Control ビューで再生アイコン、もしくはスペースバーをクリックします。

ショットガンの発砲音が聞こえるはずです。プレイバックのレベルがレイアウトの右側にあるメータービュー、及びツールバーのメーターで確認することができます。

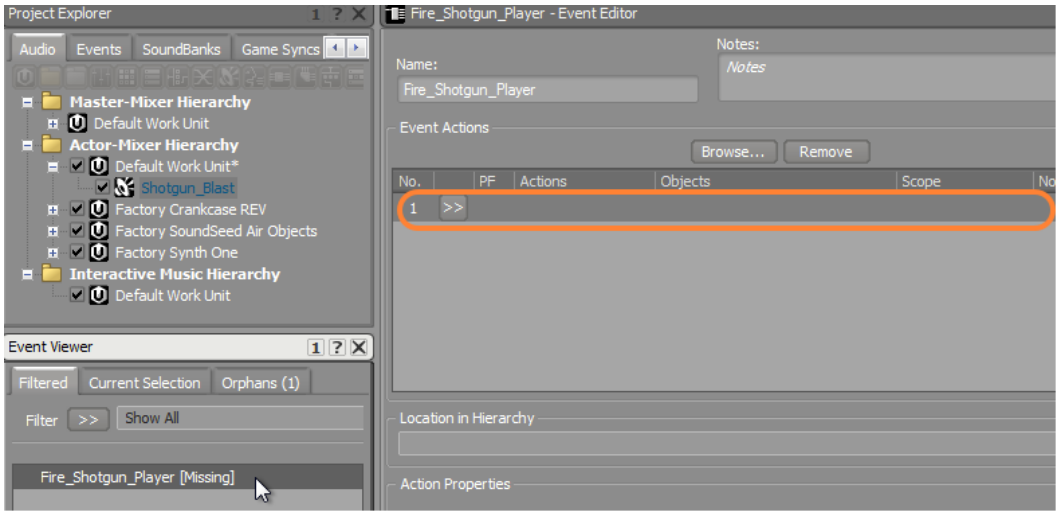


Actionの適用

さきほど、ゲームエンジンから伝えられるゲームコールを受け取るイベントオブジェクトを作成しました。そして今、イベントを受け取るときに鳴らしたいオーディオファイルが含まれるSound SFX オブジェクトを作成しました。そして、これら2つのオブジェクトを関連付けます。これは、選択されたイベントのEvent Editorで作成されたActionsを使用します。

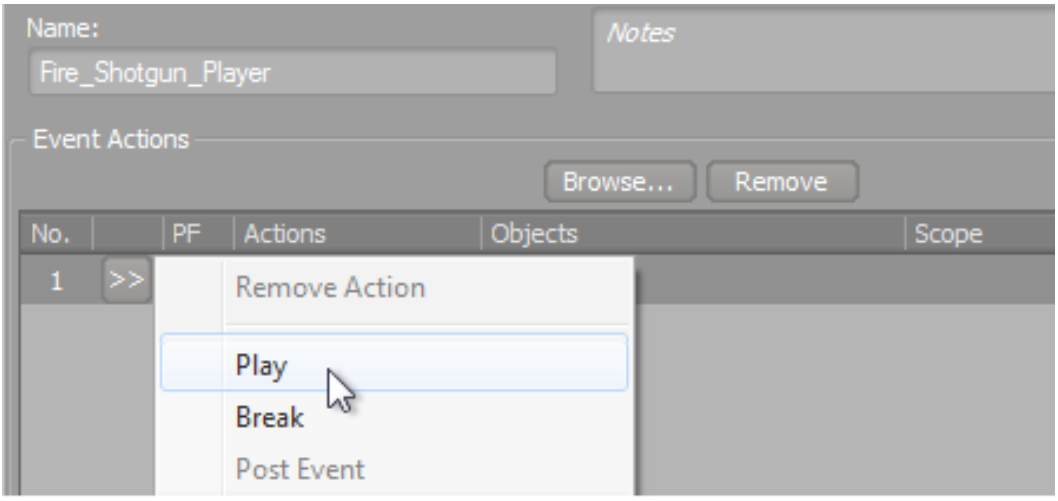
先ほど、Project ExplorerのEventsタブでFire_Shotgun_Playerのイベントを作成したときEvent Editorを使用しました。Eventタブに切り替えていなくても、画面左コーナーにあるEvent Viewerで先ほど作成したFire_Shotgun_Playerイベントがあることが確認できます。そのFire_Shotgun_Playerイベントには、関連付けられたactionsがないため、何の役にもならないことを示しています。

1. Event ViewerでFire_Shotgun_Player[Missing]イベントをクリックします。



Fire_Shotgun_Player Event Editorは、空の列1が含まれていることを示しています。ここでは、ゲームエンジンからFire_Shotgun__Playerイベントを Wwiseが受け取ったときに、どのようなActionを起こしたいかを示す場所となります。

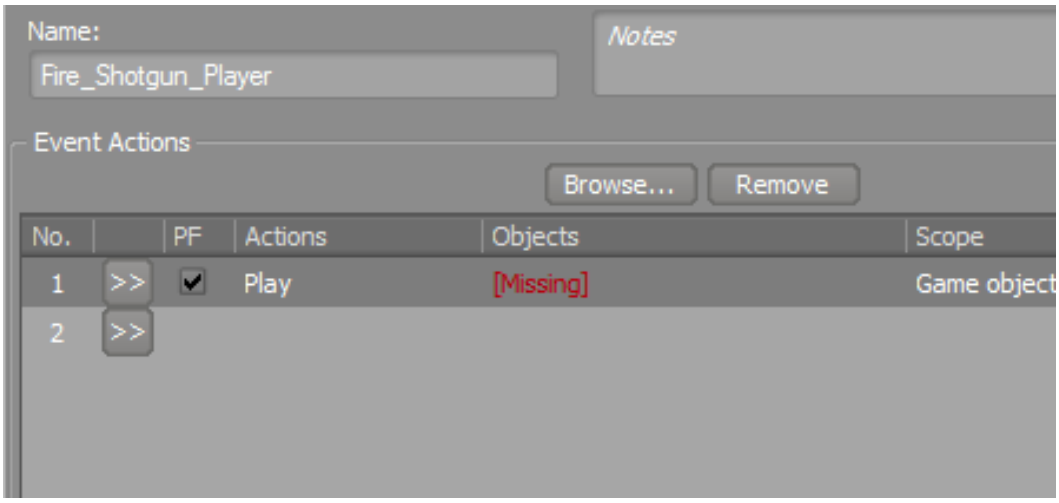
2. 最初の列のセレクターメニューボタン[>>]をクリックします。



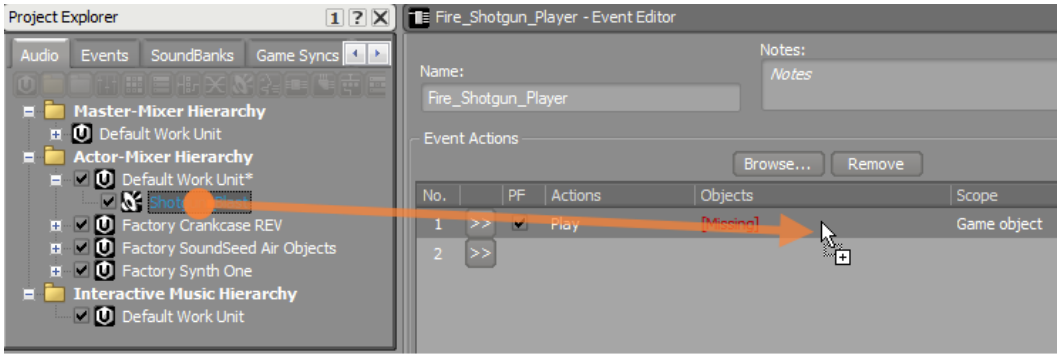
使用することができるactionsリストが表示されます。ご覧のとおり、たくさんのオプションがあります。現在必要なものは、最初のオプションPlayになります。

3. Action Listより**Play** を選択します。

Play actionが追加されますが、Objects列は、このアクションに関連付けられたサウンドが見つからないことを示しています。



以前の演習で作成したShotgun_Blast Sound SFXオブジェクトを再生することをここで設定する必要があります。プロジェクトエクスプローラーからアクションリストの最初のアクションにこのオブジェクトをドラッグし追加します。

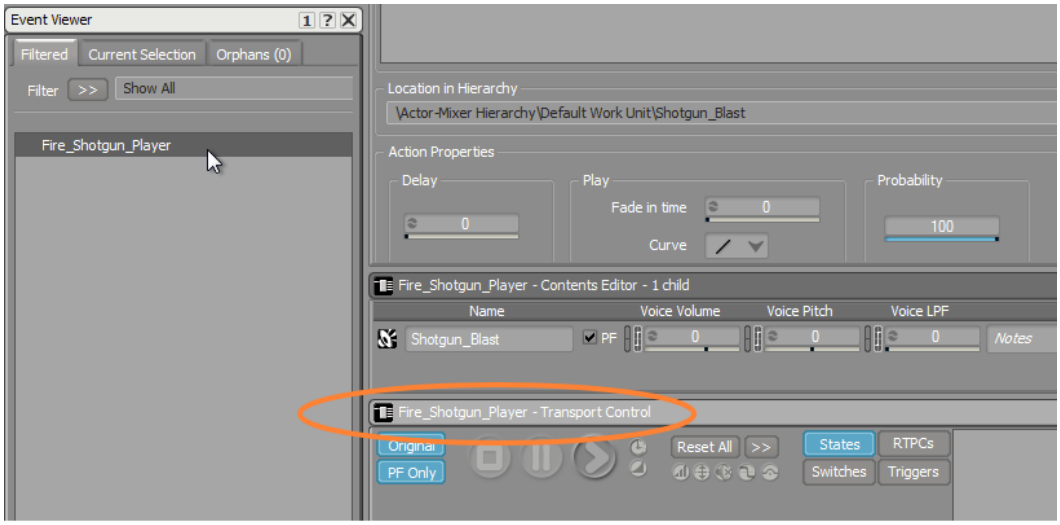


ティップ

作業効率を上げる方法として、空のアクションリストにオブジェクトをドラックすると、自動的にPlayアクションが追加されます。

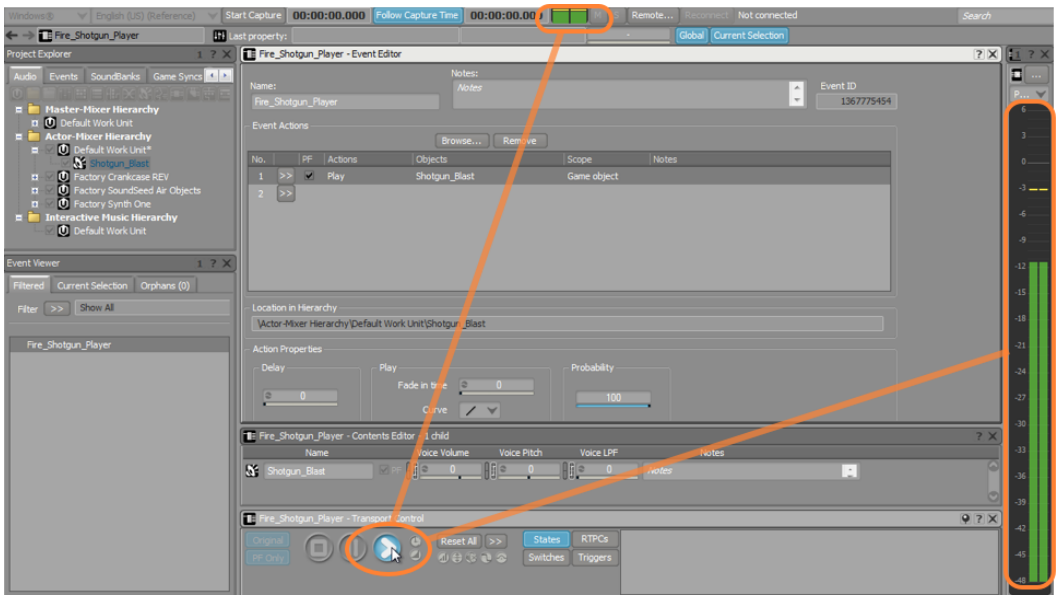
これでようやくShotgun_Blast SFX オブジェクトがFire_Shotgun_Player イベントに関連付けられました。Wwiseでは、Sound SFX オブジェクトの再生をシミュレーションするだけではなく、イベント自体のシミュレーションも行えます。Fire_Shotgun_Player イベントの再生は、ショットガンブラストサウンドをトリガします。これを行うためには、Transport Controlビューで Fire_Shotgun_Player イベントが見えていなければなりません。

4. Transport Control ビューのタイトルバーにFire_Shotgun_Playerと表示されていることを確認してください。もし確認できない場合は、Event Viewerの Fire_Shotgun_Player イベントをクリックします。



5. Transport Control ビューで、再生ボタンをクリック、もしくはスペースバーを押します。

バーッン!ショットガンブラストが聞こえ、Fire_Shotgun_Player イベントが先ほどインポートしたShotgun_Blast サウンドにリンクされていることを確認することができます。



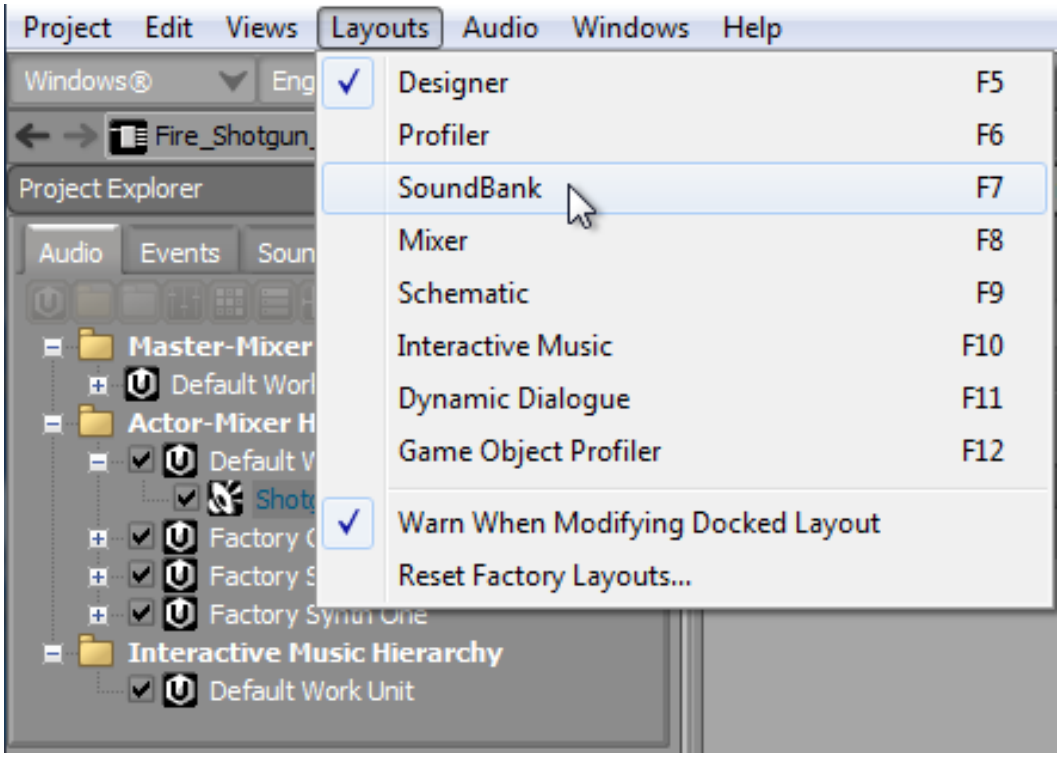
ゲームへのサウンドインテグレーション

ショットガンの音がWwise内で再生できたことで、ゴールに一歩近づきましたが、作業内容をゲームに反映させ、ゲームプレイ時にサウンドを確認する必要があります。多くの点で、この最後のステップはミュージックプロジェクトのバウンス作業に似ていますが、単一のオーディオファイルの作成以上に多くの事を行わなくてはなりません。Wwiseが担当者の代わりにゲームサウンドデザインのコードを書くため、ここがWwise サウンドエンジンの真骨頂であり利点になります。

SoundBankへのEvent追加

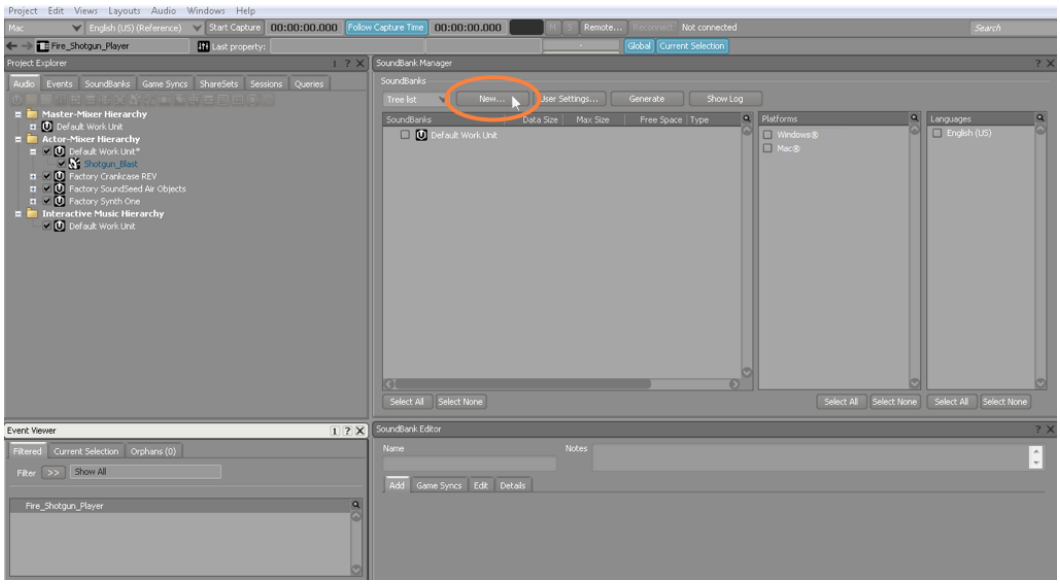
SoundBanksとは、コードとゲームが再生されるときに使われるオーディオアセットのコレクションで、run-timeとしても知られています。Wwiseを使用するゲームは、最低1つのSoundBankを持ち、大きくて複雑なゲームになればなるほど、多くのSoundBanksが存在します。このケースでは、単一のSoundBankを作成し、Cubeデモのゲームエンジンがそれを必要とする場所のディレクトリに保存します。このために、異なるレイアウトを用意しています。

1. メインメニューで、Layouts > SoundBankの順にクリック、もしくはF7を押します。



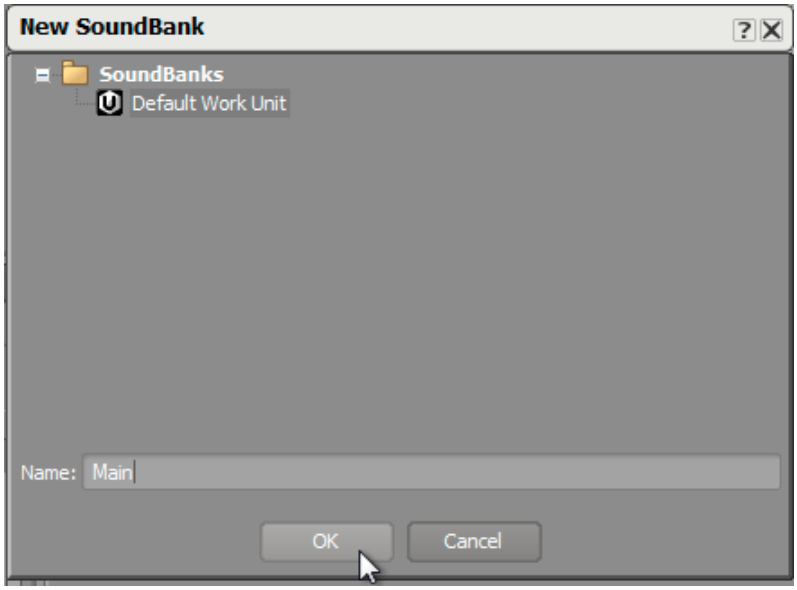
スクリーンの右上にSoundBank Managerを見つけることができます。Wwiseのほかのエリアと同様にDefault Work Unitを見つけることができます。Cubeでは、Mainと呼ばれるサウンドバンクを見つけるようコードされています。そのため、このDefault Wor Unitでこれを作成します。

2. Newをクリックします。

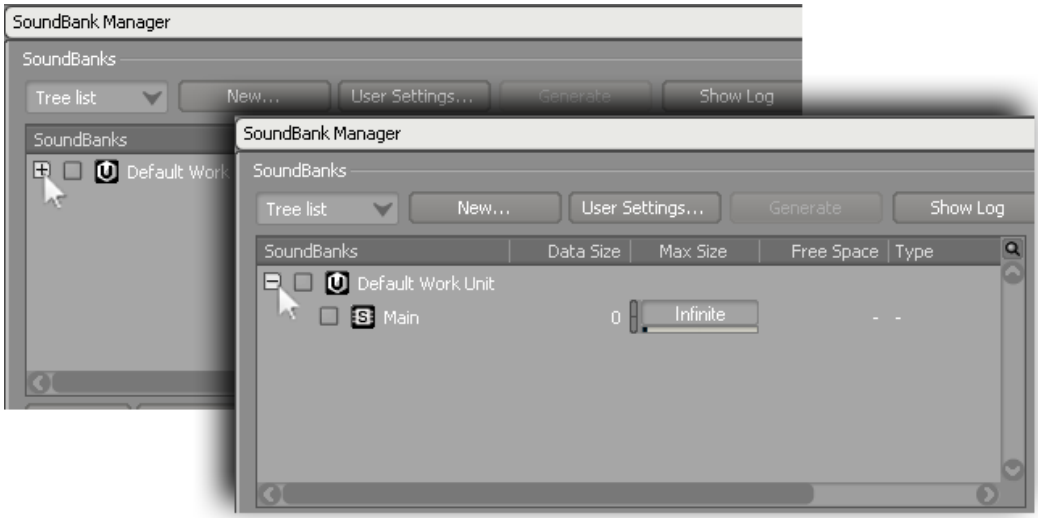


New SoundBank ダイアログボックスが開きます。

3. Nameフィールドに、Mainと記入し**OK**をクリックします。

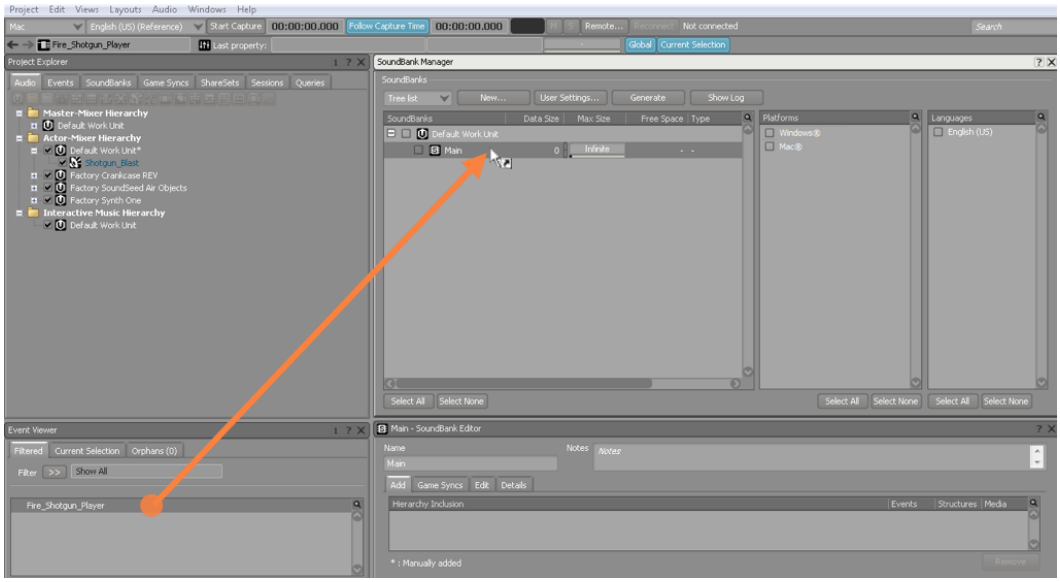


4. Default Work Unitの左に表示されている[+]をクリックします。

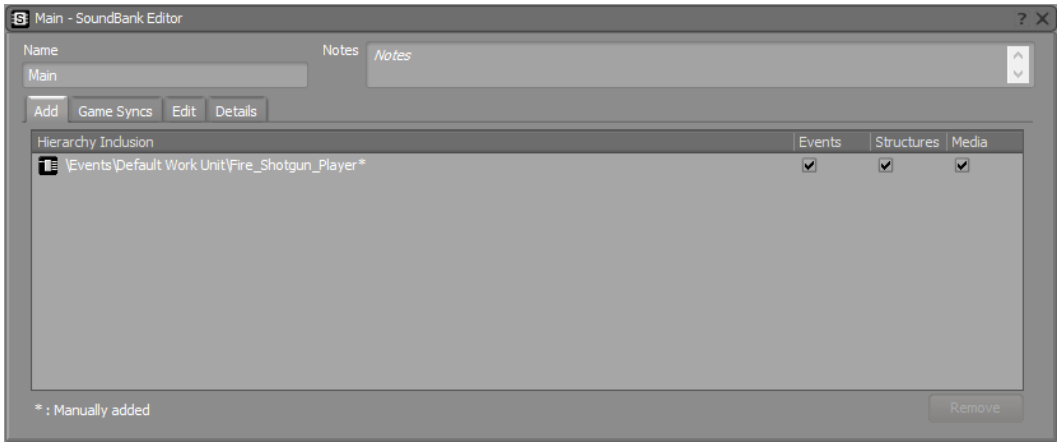


SoundBankのパーツとして、ゲームで使用するイベント群を含める必要があります。希望するSoundBankにイベントをドラッグすることによりアサインできます。

5. Event ViewerにあるFire_Shotgun_Player イベントをSoundBank Manager の Main SoundBankにドラッグします。



右下のビューで、SoundBank Editorに追加されたイベントを見ることができます。



SoundBankの作成（Mac）。

SoundBankの作成

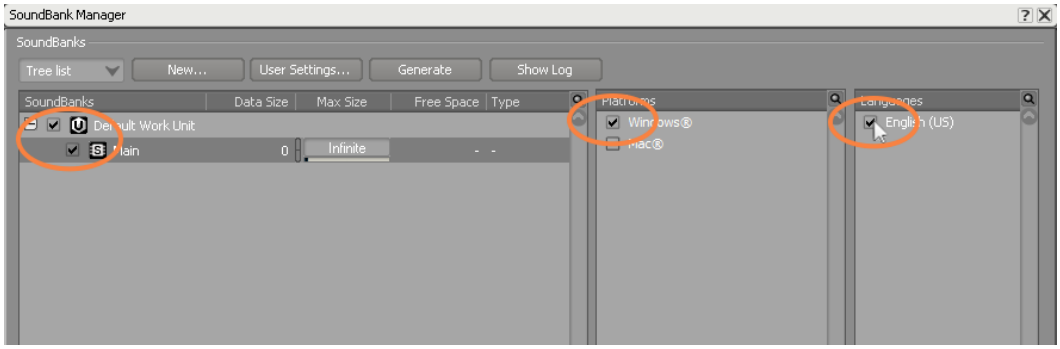
Wwiseは、複数のゲームプラットフォームだけでなく複数言語のSoundBankを、簡単な操作で同時に作成することができます（ダイアログコンテンツを準備してWwiseのローカライゼーション機能を使用した場合）。通常ですとゲームプラットフォームは、それぞれ独自のオーディオプログラムの実装が必要となるため、これは大きな時間の節約になります。要するに、Wwiseを使用することは、各ゲームプラットフォームの特徴を理解した複数のプログラム担当者がチームにいることと云えるでしょう。そのため、どの言語、そしてどのゲームシステムに対してSoundBanksを書き出すかの指定をしなければなりません。

注記

次の演習の手順は、WindowsとMacでは異なります。手順毎のイメージ図はWindowsプラットフォームを使用した場合のものです。Macユーザーは、表示されるMacのオプションを選択してください。

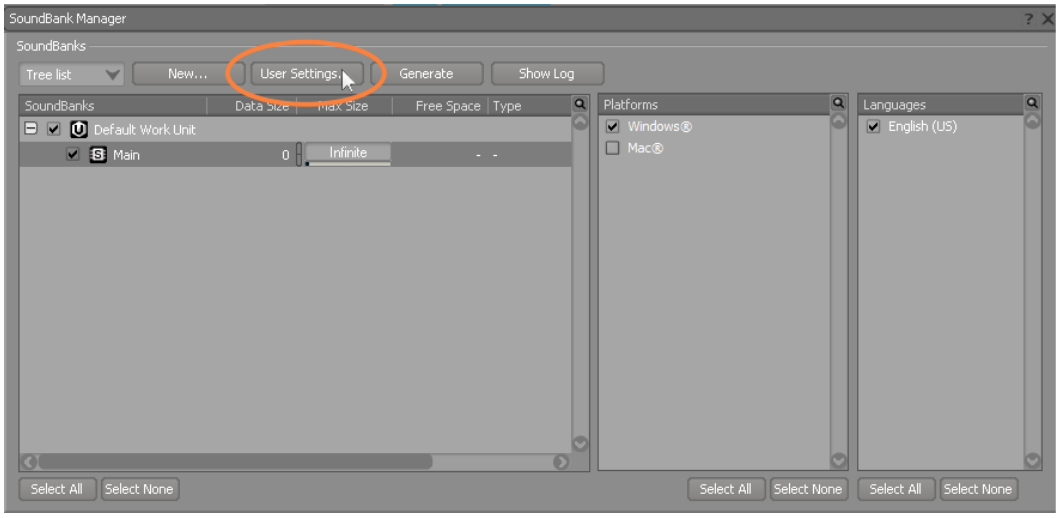
1つのゲームに対して複数のSoundBanksを作成することができるため、最初に作成するSoundBanksを選択しなければなりません。

1. SoundBankはMainを選択して、該当するPlatformと、English (US) のチェックボックスにチェックが付いていることを確認してください。

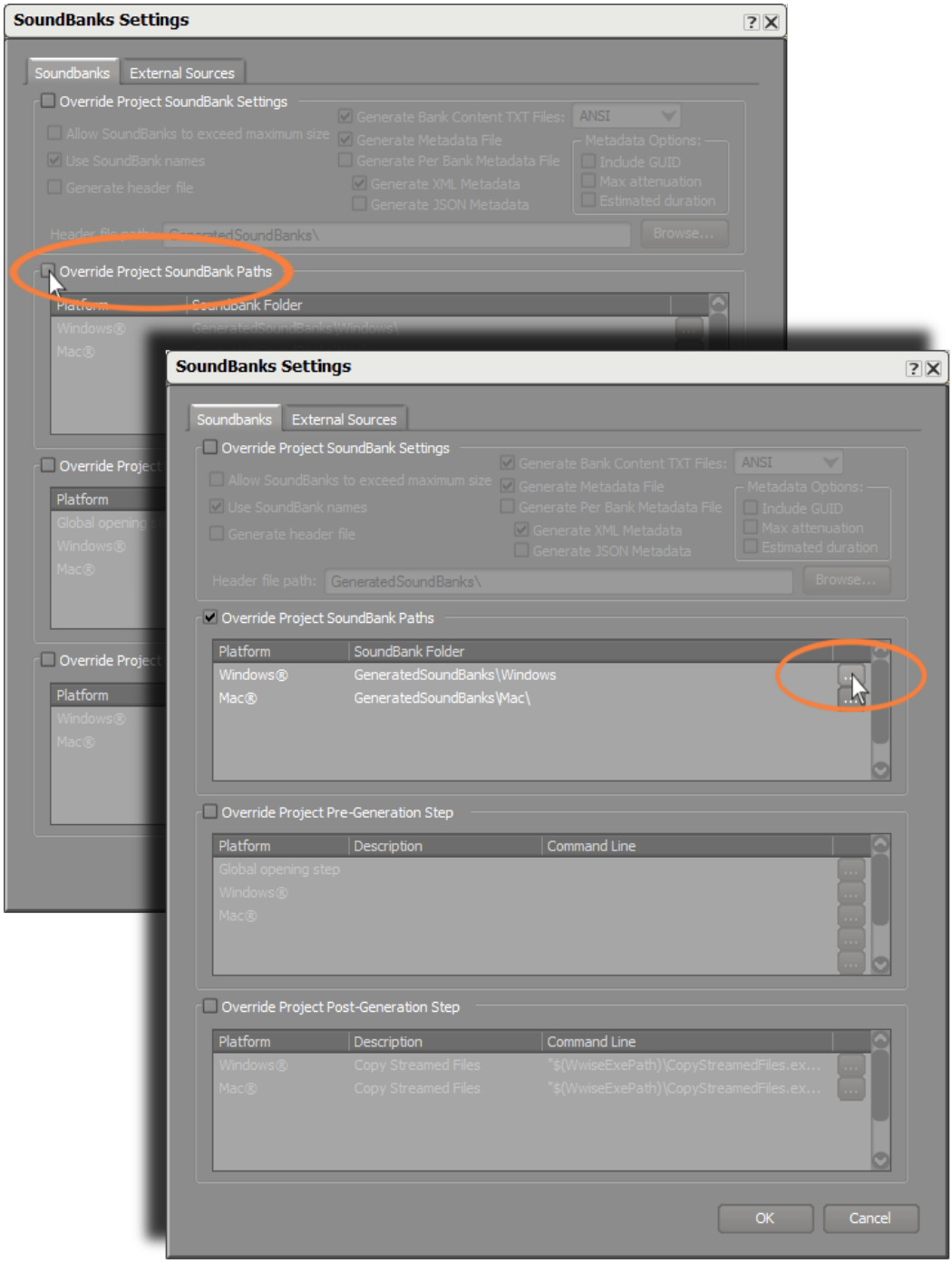


次に、SoundBanksをゲームファイル構成のどこに格納するかを指定するパスを定義しなければなりません。この情報は、通常ゲームプログラマ側から指定されることが多いです。

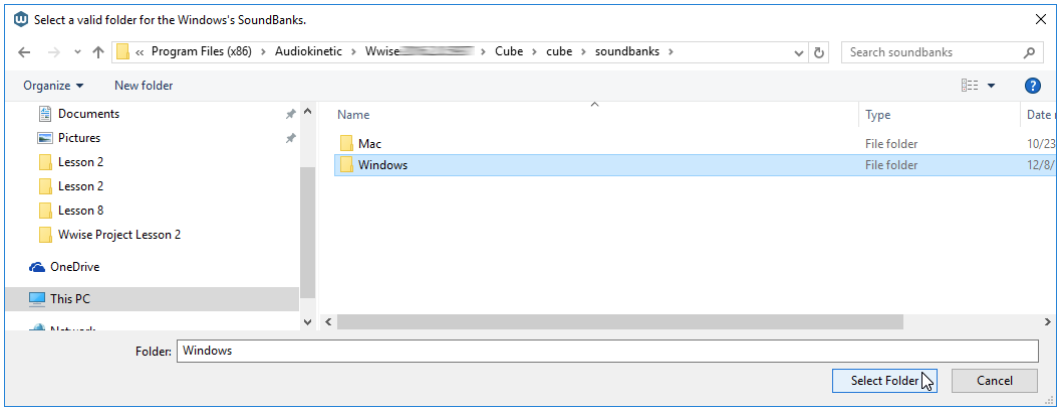
2. SoundBank Managerで、 **User Settings.**をクリックします。



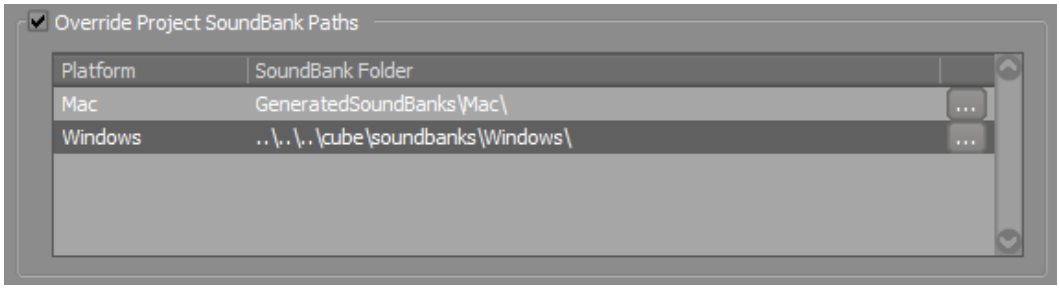
3. Override Project SoundBank Pathsオプションを選択し、あなたのプラットフォームのパスセレクターボタンをクリックします。



4. Wwise Lessonsフォルダまで移動し、`Cube\cube\soundbanks`へ進み、あなたのプラットフォーム用のフォルダを選択し、**Select Folder**をクリックします。

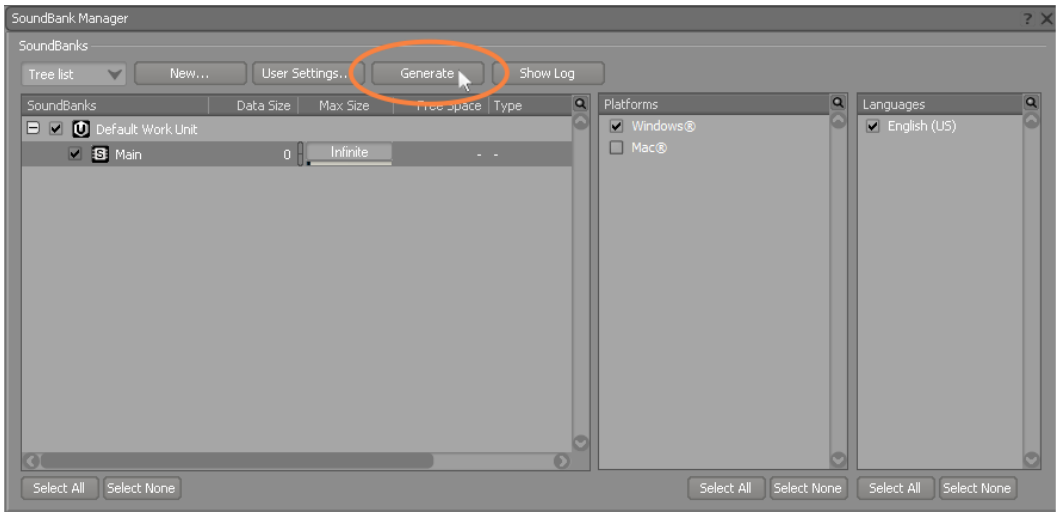


あなたのプラットフォーム用の更新されたSoundBankパスが表示されます。

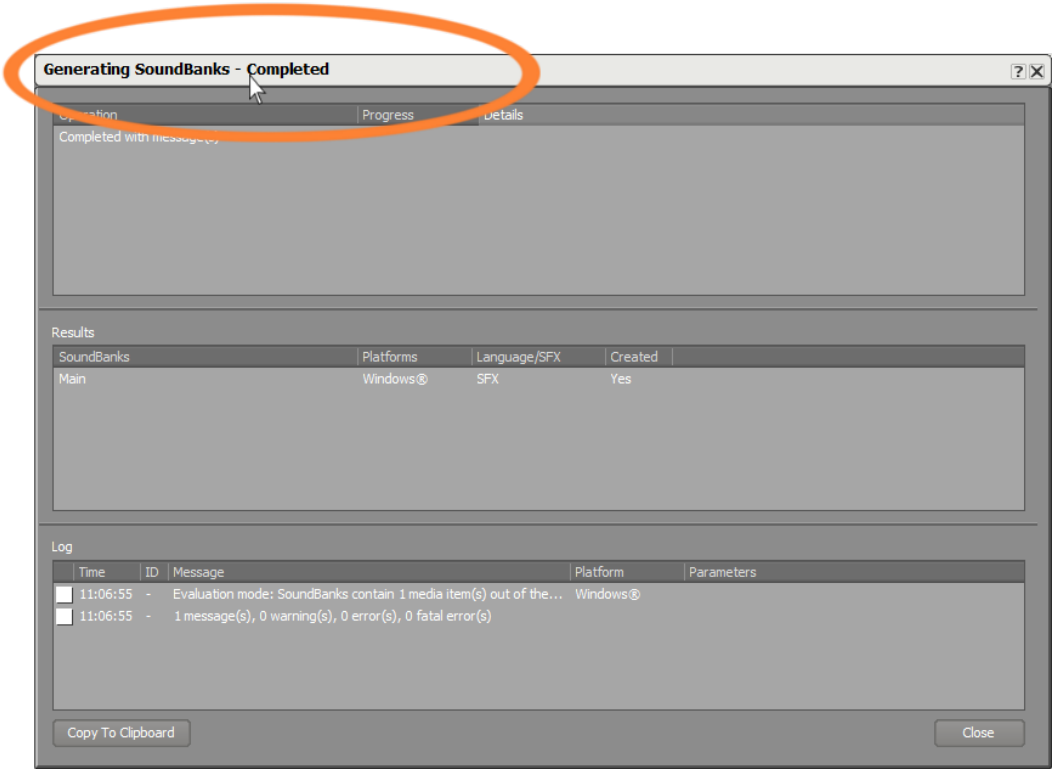


SoundBankを作成します。SoundBankの作成は、今まで行ってきた作業の集大成となります。DAWで云う、ファイルのバウンスのような作業です。その結果としてあなたを含む、皆が、あなたの作業の恩恵を受けられるようになります。

5. **OK**をクリックし、SoundBanks Settings ウインドウを閉じ、SoundBank Managerの**Generate**をクリックします。



SoundBank Generationビューが表示されます。この時点で、ショットガンサウンド実装に必要な全てのプログラムやファイルが作成されます。



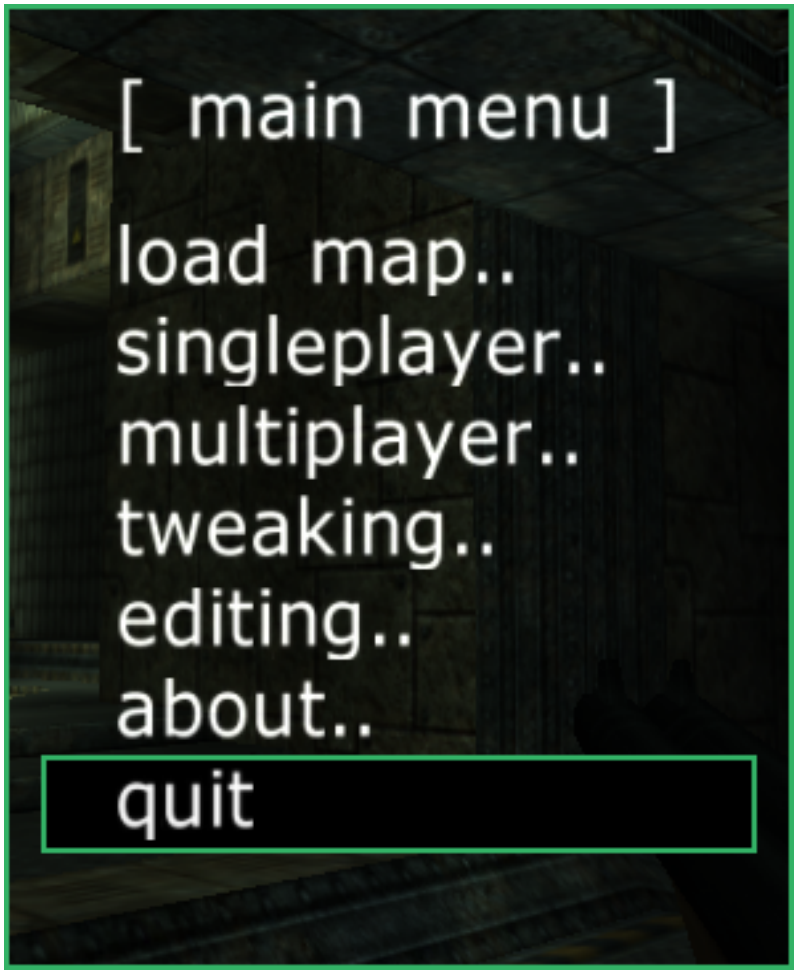
6. **Close**をクリックします。

   SoundBanksはレッスン2から8まで、最後に作成します。これらのレッスンでは、SoundBankフォルダーのパスはすでに設定済みなので、最適なSoundBank、Platform、そしてLanguageのチェックボックスにチェックがあるかの確認をします。

ゲームをプレイする

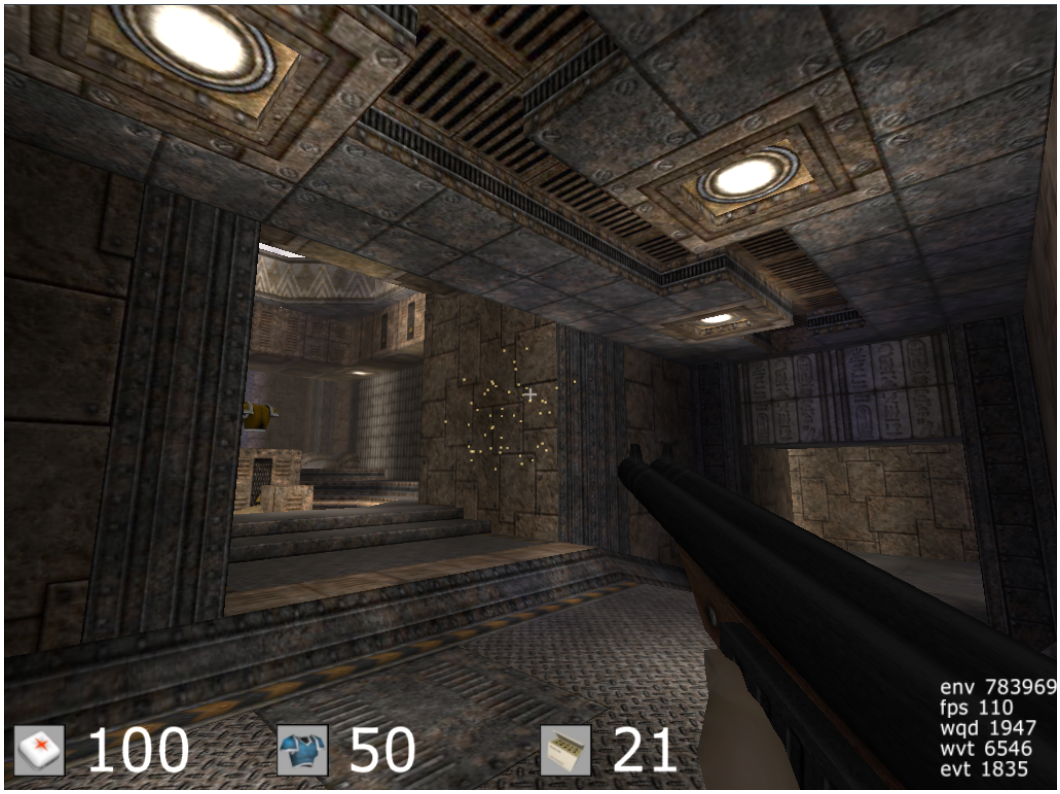
実際のゲームで今まで行ってきた作業の結果を見ることができます。作業を行った部分を反映させるためには、Cubeデモを再起動しなければなりません。現在実行しているゲームを終了させます。

1. Cubeデモに戻り、**Esc**キーを押し、次に矢印キーでQuitを選択し、**Enter**を押します。



2. Cubeデモを再度立ち上げます。

   今回は、先ほど作成したSoundBankが実装されたゲームが起動します。
3. クリックしてショットガンを発砲します。



Wwiseを使って実装したショットガンのサウンドが聞こえます。

レッスン 2：Soundscapeのデザイン

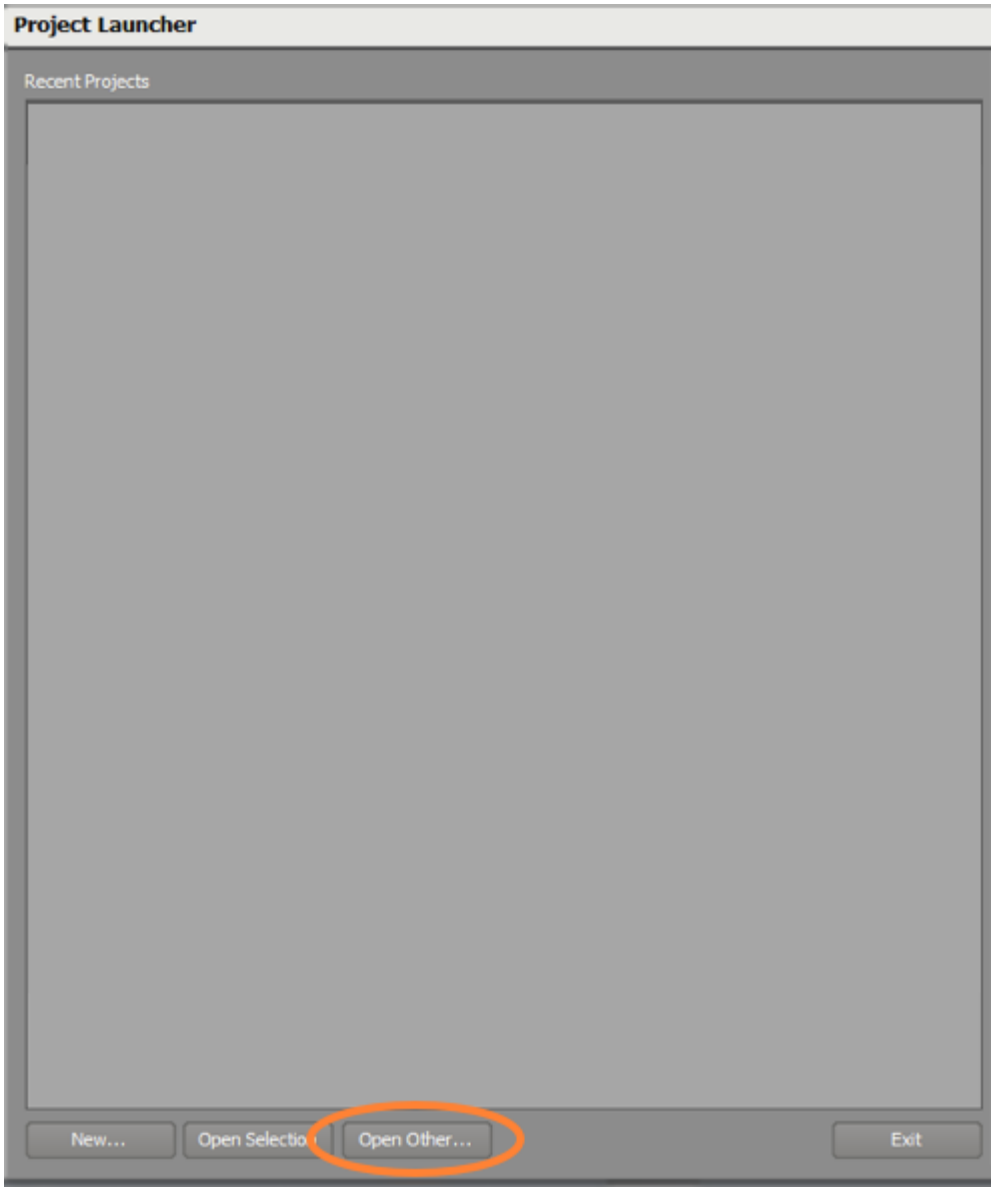
プロジェクトにおけるサウンドの追加 52
単一のサウンドを複数の用途に使用する 59
オブジェクトのパラメーター調整 61
Source Editorの使用 64
単一イベントへの複数アクションの適用 65
ランダマイゼーション（ランダム化）機能の使用 69
サウンドのグラニュラ化 79
作業の確認 97

ショットガン発射のアニメーションにブラストサウンドを実装しました。ここでは、ビジュアルシーケンスのほかの部分にも目を向け、この体験を向上させるために他にができるかを考えます。必要な作業を行う上で、最小限の作業で最大限の効果をを得られる方法を探します。これがゲームオーディオデザイナーの存在価値になります。なぜなら、現在のゲームコンソールシステムはとてつもないパワーを持っていますが、そのパワーは、グラフィクス、フィジックス、そしてAIなどのシステムと共有しなければなりません。そして、多くのゲームはシステムのリソースが限られたモバイルデバイスで再生されることも考慮しなければなりません。ゲームの技術的な制限により、オーディオに使用することができるプロセッサーやメモリー使用量に制限がある場合があります。このため、ゲームオーディオ開発者はいかにして与えられた技術的制約の中でクリエイティビティを発揮するかを考えなくてはなりません。多くの場合、デザイナーの創意工夫で、技術的な制約内で実現するユニークかつ有効なソリューションを提供することになります。幸いなことに、Wwiseはまさにこうしたチャレンジに立ち向かうために設計された広範なツールセットを提供します。

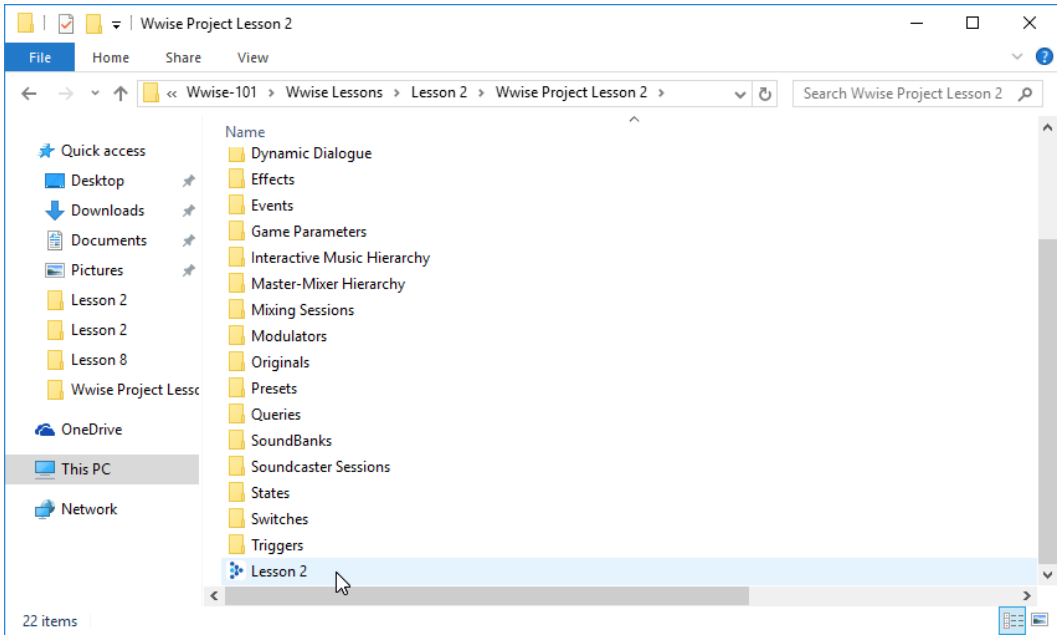
プロジェクトにおけるサウンドの追加

このレッスンでは、すでに前のレッスンで使ったショットガンブラストサウンドが入った保存済みプロジェクトファイルを使用します。

1. Wwiseを起動し、Project Launcherの下部にある**Open Other**をクリックします。

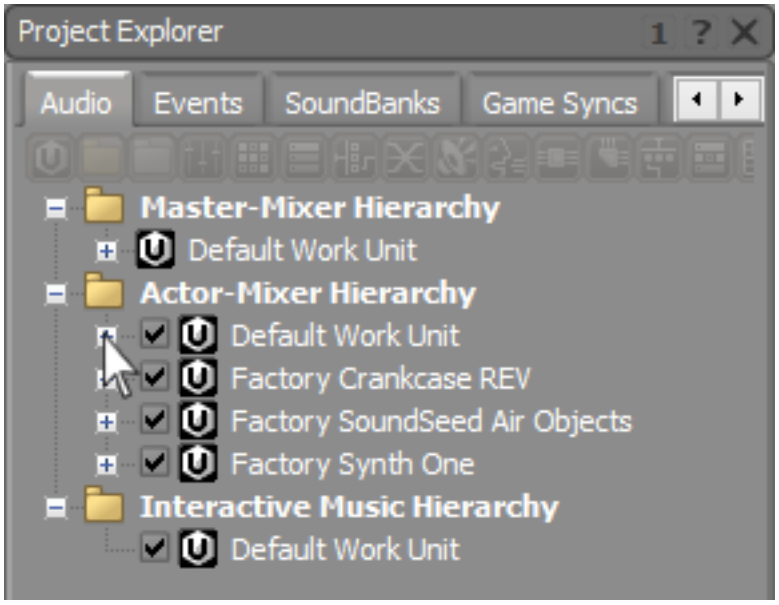


2. **WwiseProjects** > **Lesson 2** フォルダを開きます。



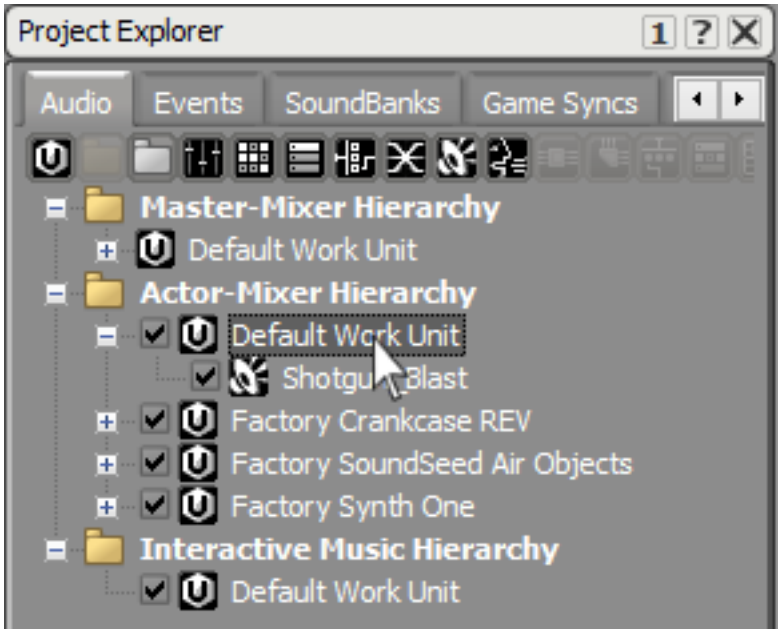
Wwise プロジェクトは、単一ファイルではなく、その中に包含される多くのフォルダ、ファイルの構造体から構成されています。多くの場合、プロジェクトを進めるにつれて必要になるアイテムは自動的に生成、更新されるため、これらに包含されているものを直接作業する必要はありません。プロジェクトは、これらのフォルダと同じレベルに保存されており、Wwise Projectファイルから起動されます。

3. Lesson 2フォルダにあるLesson 2プロジェクトファイルを選択し、**Open**をクリックします。
4. Actor-Mixer Hierarchy内で、Default Work Unit を開いて、前レッスンで作成したShotgun_Blast サウンドSFX オブジェクトを表示させます。



最初にサウンドSFX オブジェクトを生成してから、サウンドを追加するのではなく、ここでは別の方法としてオーディオファイルを希望のWork Unitへインポートするだけで、Wwiseが自動的に必要なサウンドSFXオブジェクトを生成します。

5. アクターミキサー階層で、Default Work Unitを選択します。



注記

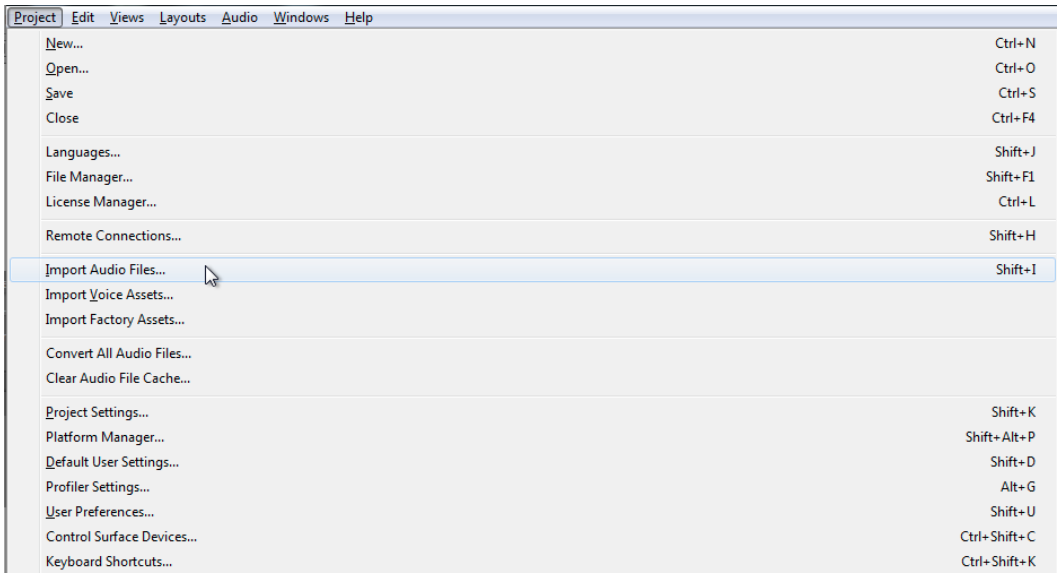
Shotgun Blast Sound SFXオブジェクトが白色では無く、青色で表示されていることに気がつくかもしれません。青色は、参照しているオーディオファイルがインポートされたオリジナルであることを、白色はファイルが変換済みであることを示しています。ファイルが見つからない状態を示す赤色が表示されない限り、このチュートリアルでは色の違いにおける作業の変更はありません。もし希望があれば、メインメニューバーのProject > Convert all Aduio Filesで全てのオーディオファイルをいつでも変換することができます。レッスン7では変換設定の詳細を学ぶことができます。

6. メインメニューで、**Project** > **Import Audio Files**を選択、もしくは**Shift+I**を押します。

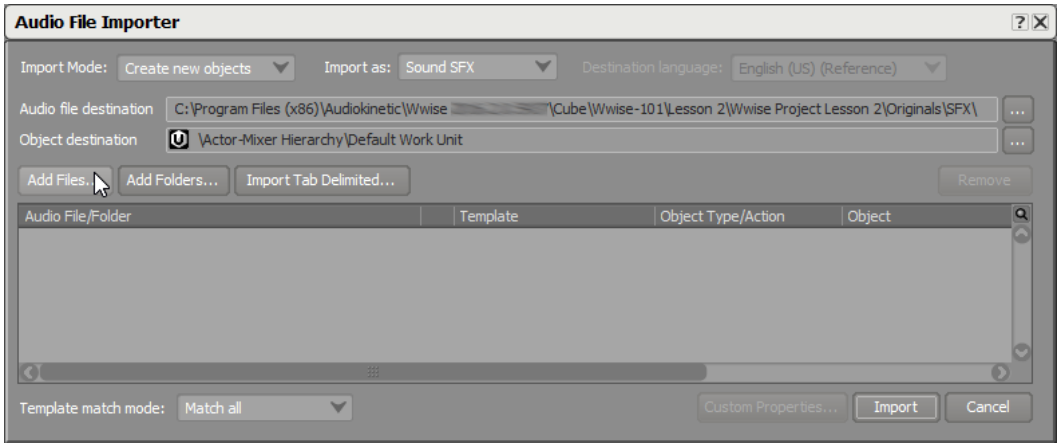
ティップ

定常的に使用する作業にはキーコマンドを使用することで作業効率が良くなります。Wwiseのメニューで表示される機能には、

キーコマンドが用意されていますので、これらを記憶すると便利です。

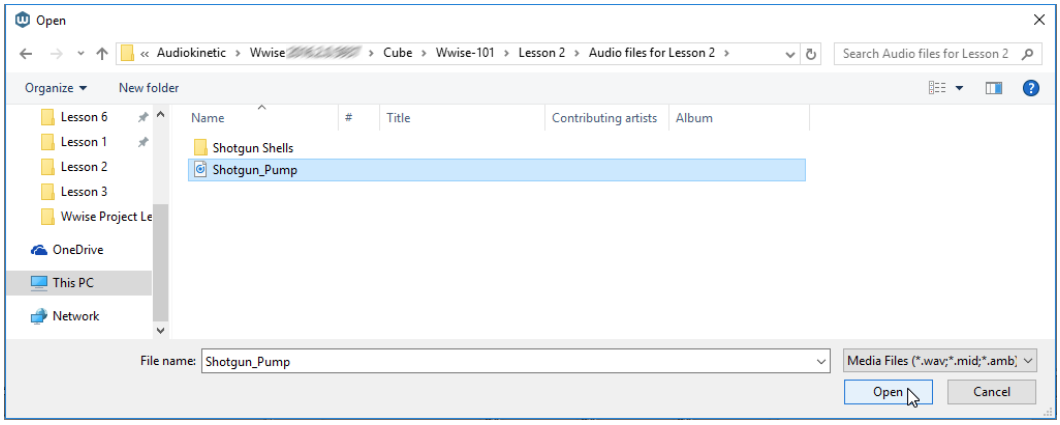


7. Audio File ImporterでAdd Filesをクリックします。



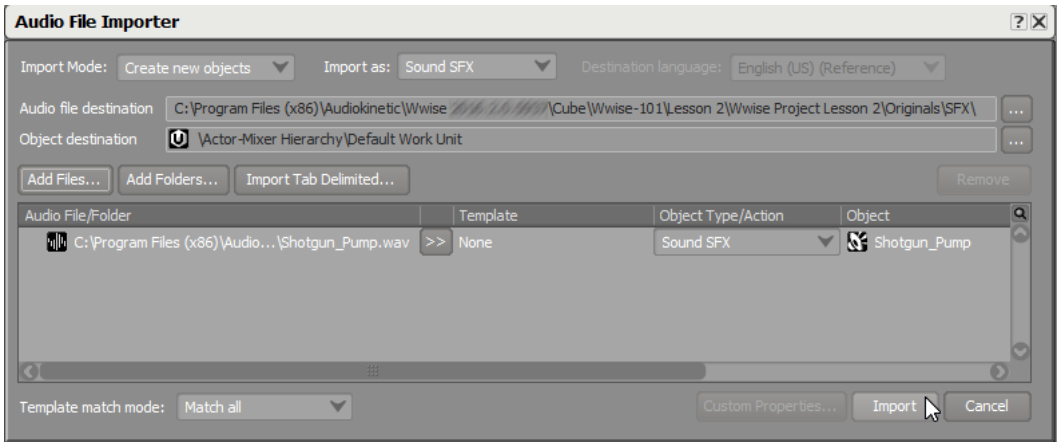
インポートしたいファイルまでナビゲートすることができるExplorerウィンドウが開きます。

8. Wwise-101のLesson 2フォルダを開きます。Lesson 2フォルダのAudio filesを開き、Shotgun_Pumpを選択し、**Open**をクリックします。



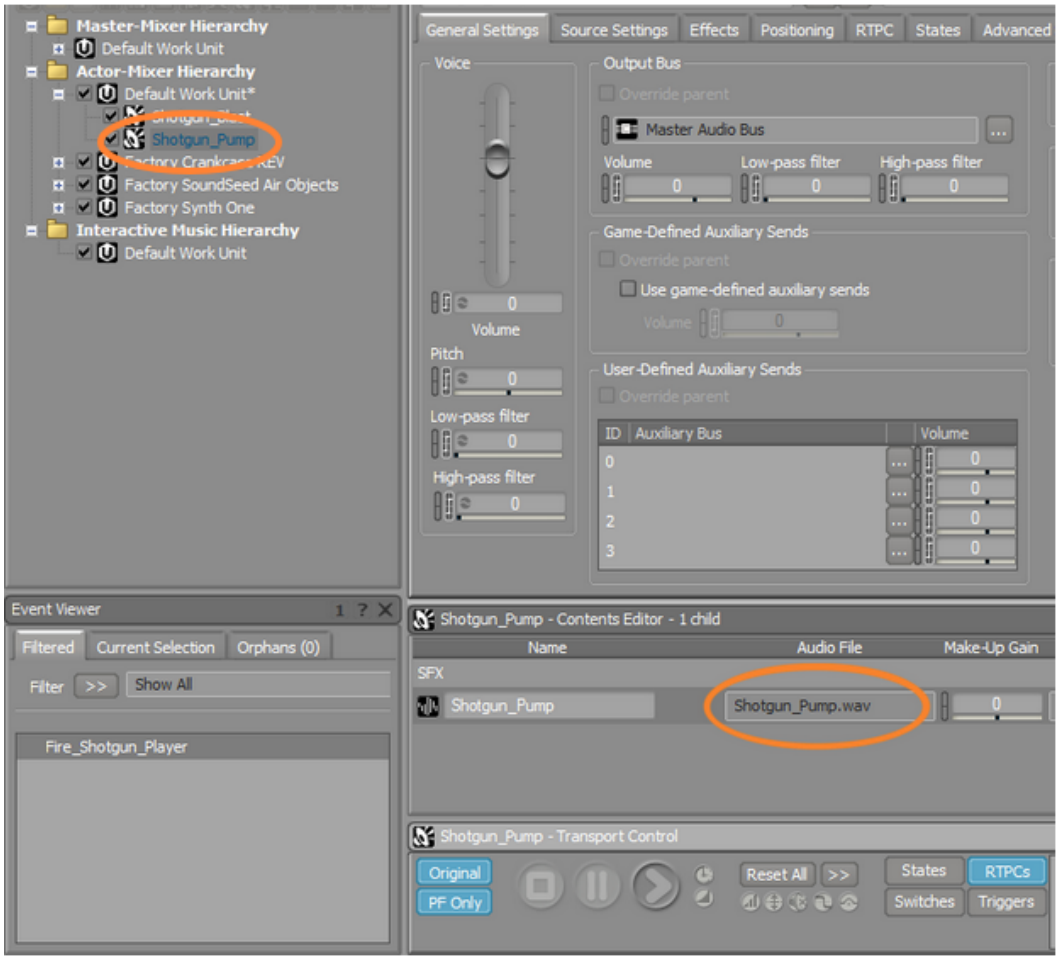
Audio File Importerウィンドウが現在のロケーションを表示します。Object Type/Actionのプルダウンメニューが表示され、インポートされたファイルには自動的にSound SFXオブジェクトが作成され、包含されます。

9. **Import**をクリックします。



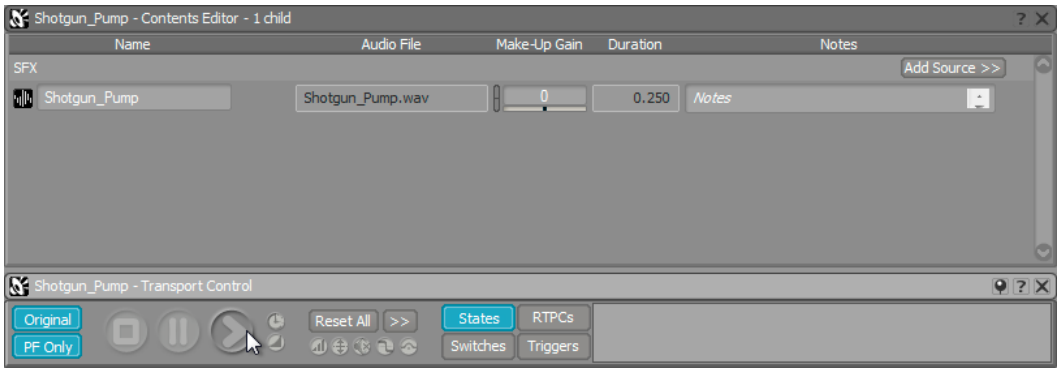
Shotgun_Pump サウンドSFXオブジェクトは、アクターミキサー階層のDefault Work Unit内に表示されます。

10Shotgun_Pump SFX オブジェクトをクリックし、Sound Property Editorで新しく作成したオブジェクトのパラメーターを確認することができます。



Contents Editor（Sound Property Editorの下）では、Shotgun_Pump.wavファイルがShotgun_Pump Sound SFX オブジェクトのソースとして表示されます。

11新しくインポートしたオーディオファイルを試聴するためには、Shotgun_Pump Sound SFXオブジェクトがProject Explorerで選択された状態で、Transport Controlビューで再生ボタンをクリック、もしくはスペースバーを押します。



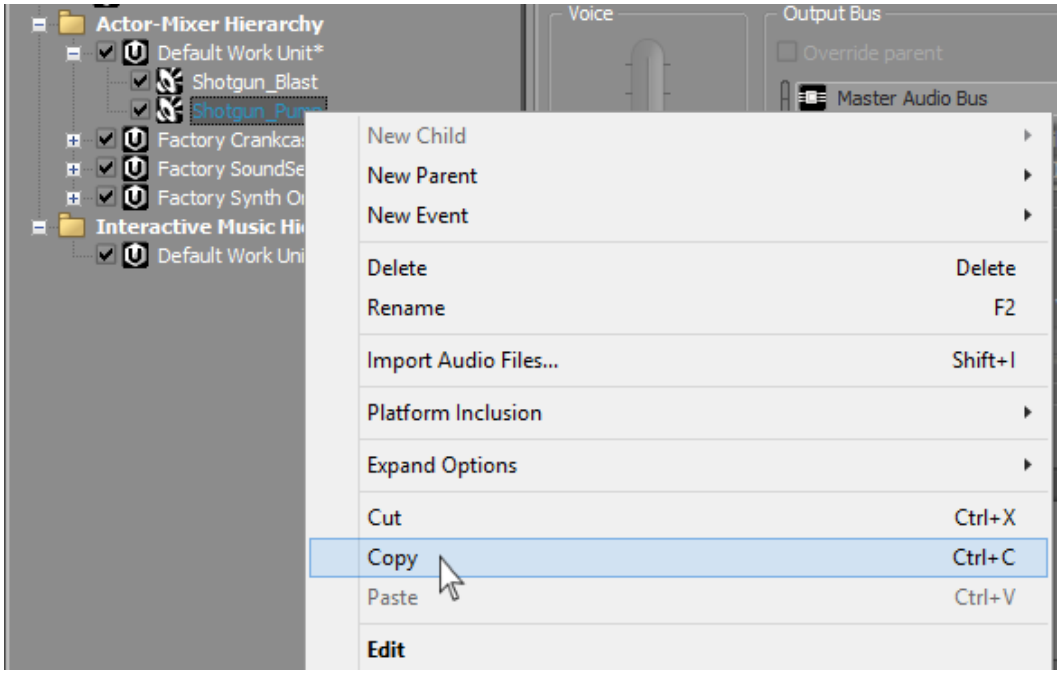
ショットガンがパンプされた機械的なクリックサウンドが聞こえます。

単一のサウンドを複数の用途に使用する

プレイヤーのショットガンアニメーションシーケンスを注意深く観察すると、ショットガンのパンピングアクションには実際には2フレーズあることに気がつきます。まずはじめに、プレイヤーキャラクターはショットガンを手前にパンプし、空のシェルを排出します。そしてチャンバーに次のラウンドを装填するために外へパンプします。それぞれの動きは、機械的ななクリックノイズを発生します。しかしながら、インポートしたオーディオファイルには2つのクリックが含まれていません。

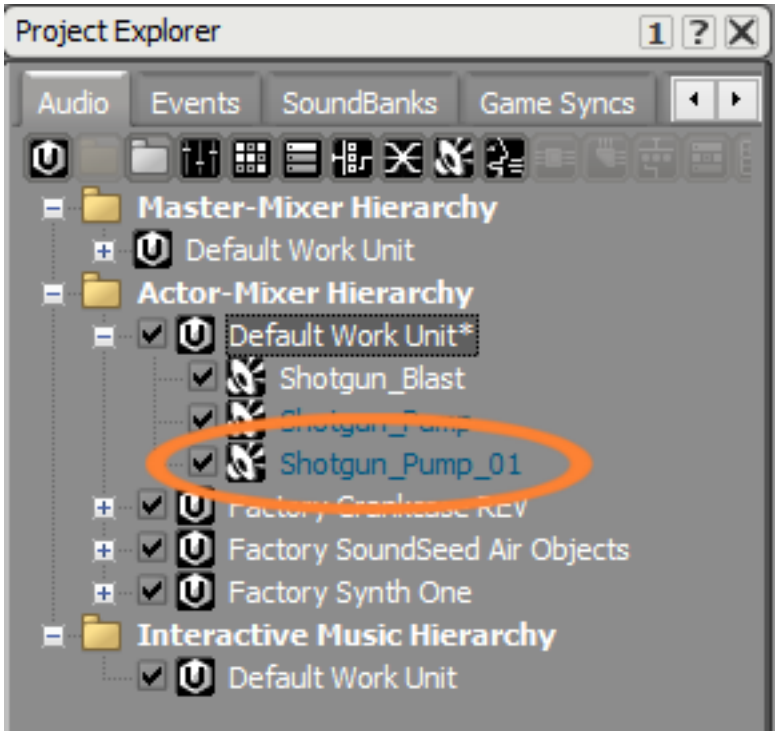
アニメーションシーケンスにあう2種類のパンプサウンドをレコーディングすることも可能ですが、オーディオファイルを長くし、ゲームにロードする際、多くのメモリーを消費します。メモリの節約は常に考慮すべきことですので、これは良い戦略とは云えません。そしてゲーム開発が進むにつれてアニメーションシーケンスのタイミングが調整されることも発生します。そのため、ここでは違うアプローチを採用し、メモリの節約、そしてタイミング設定における柔軟性のため、単一のメカニカルクリック音をパンプインとパンプアウトサウンドとして使用します。同じサウンドファイルを使用しますが、２つの独立したSound SFX オブジェクト（パンプインとパンプアウト）を使用します。

1. Shotgun_Pump Sound SFXオブジェクトを右クリックし、**Copy**を選択します。



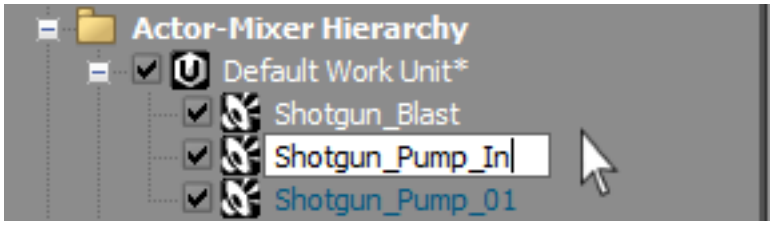
同じWork Unit内にオブジェクトのコピーを作成します。

2. Actor-Mixer Hierarchyで、Default Work Unitを右クリックし、**Paste**を選択します。



WwiseはSound SFXオブジェクトを複製しますが、_01という拡張子を追加します。

1つのShotgun_Pump オブジェクトはパンプインサウンドを作成し、もう1つはパンプアウトサウンドとなります。そのためネーミングもあわせます。



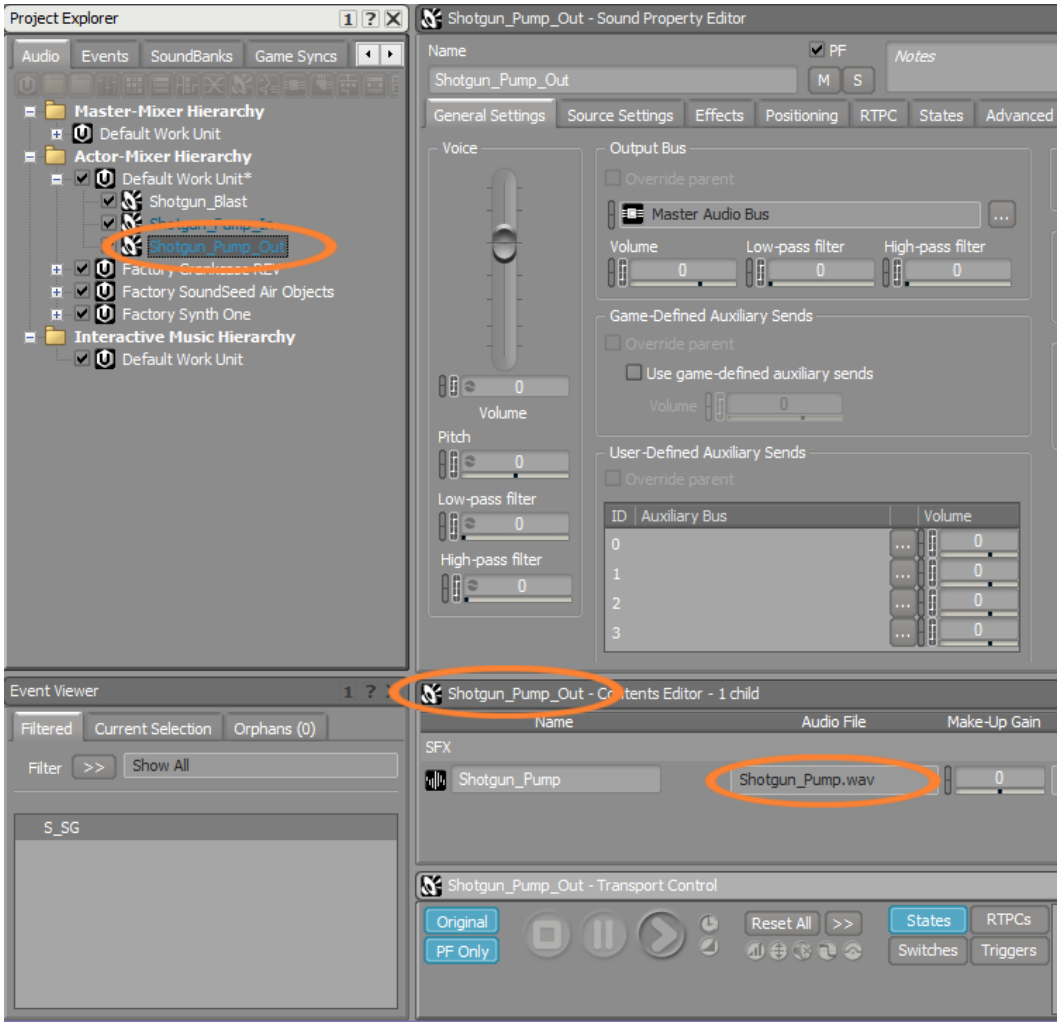
3. Shotgun_Pump オブジェクトをクリック、ポーズ、そして再度クリックしShotgun_Pump_Inとリネームし**Return**を押します。

ティップ

F2キーを押してオブジェクトのリネームを行うこともできます。右クリック後のリネーム、もしくはProperty EditorのName fieldに直接記入することもできます。

4. Shotgun_Pump_01オブジェクトをShotgun_Pump_Outにリネームします。

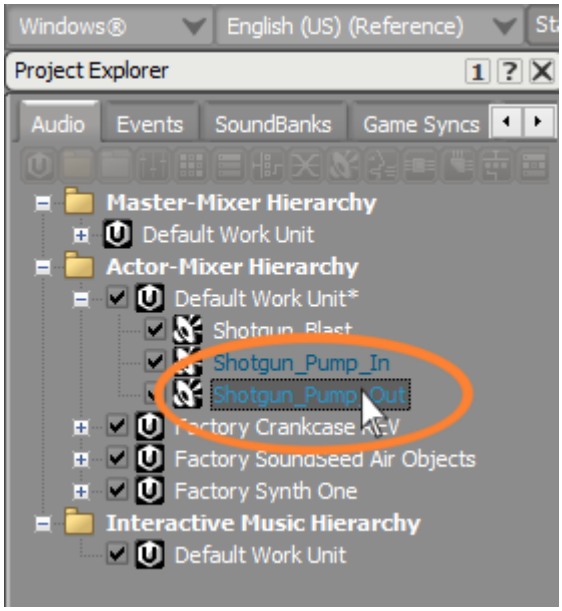
Shotgun_Pump_Out オブジェクトが選択され、コピーされたこのオブジェクトは同じShotgun_Pump .wav ファイルを参照しています。



オブジェクトのパラメーター調整

どちらのShotgun_Pumpオブジェクトも同じ名前を参照していますが、Shotgun_Pump.wav オーディオファイルが複製されたわけではありません。これは、同じオーディオファイルが違うサウンドオブジェクトによって参照されているだけで、これらのオブジェクトを再生するのに必要なメモリー使用量が倍増しているわけではありません。DAWでもこれと似たようなことが行われており、単一のオーディオファイルが、編集目的、もしくは様々なプロセス用に複数のトラック上に配置されることがあります。同様に、今からShotgun_Pump_Outオブジェクトのパラメーターを変更し、パンプインとの差をつけます。

1. Shotgun_Pump_Outオブジェクトが選択されている事を確認し、もし選択されていなければ選択します。

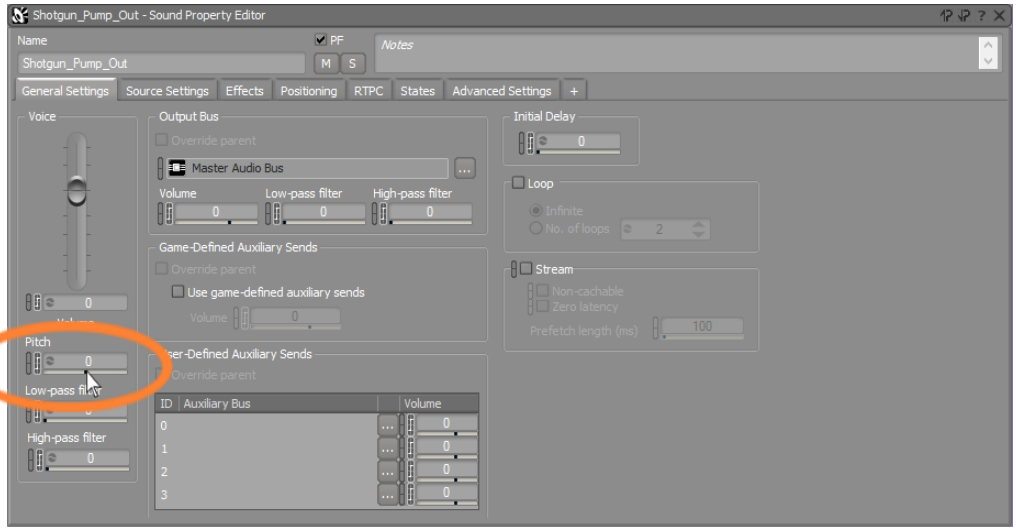


右のSound SFX Property Editorビューでは、選択されたオブジェクトのオーディオファイルがどのように再生されるかを決めるセッティングが表示されます。このレッスン後半、そしてこのチュートリアル全体で様々なパラメータを学ぶことになりますが、ゲームオーディオで重要になるパラメーターのひとつはPitchです。PitchパラメーターはVolumeフェーダーの下に位置します。これはゲームオーディオデザイナーにとって非常に便利なものであり、オーディオプロダクションツールとして頻繁に使用しますが、必ずしも手に届くところに表示されているとは限りません。ピッチの調整はゲームオーディオで頻繁に活用され、単一のサウンドからピッチを変更させることによって、様々な異なるサウンドを作り出すことができます。この戦略は大切なメモリーを節約することができます。この手法を用いてパンプインとパンプアウトのサウンドに違いを持たせます。

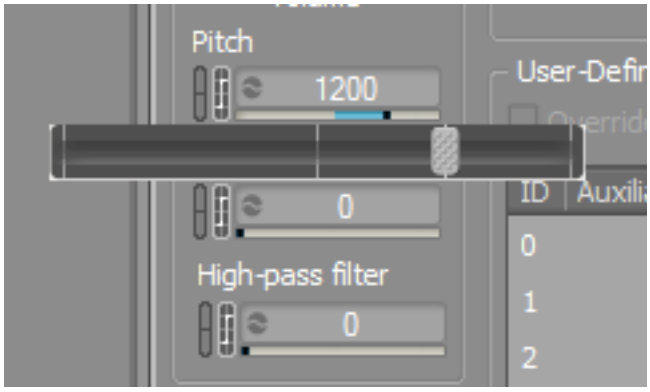
Pitchパラメーター値は、音楽用語のセントを使用します。ピアノでは、白鍵、黒鍵の区別なく、並んでいる2つのキーの間には100セントのピッチの違いがあります。Pitchパラメーターのデフォルト値は0になり、オブジェクトに関連付けられているオーディオは、そのオリジナルのピッチで再生することを意味しています。

Pitchの値を調整するには、小さい黒色のドットがPitchプロパティーの下にあり、クリックするとスライダーを変更することができます。

2. 現在のPitch値の下のスライダーヘッド（黒のポイント）をクリックしホールドします。



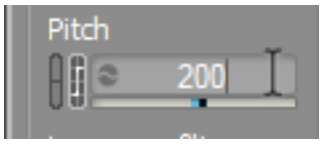
この状態でスライダーを右や左に動かし、プレイバックのピッチを調整することができます。1200値変更すると、音楽でいうオクターブに当たります。



3. **スペースバー** を押して違いを試聴します。

   Shotgun_Pumpオーディオファイルが 1 オクターブ高い状態で再生され、音が2倍の高さになり、2倍のスピードで再生されます。これは明らかにやりすぎであり、もう少し現実的な値を設定する必要があります。今回は直接値を入力します。

4. 現在の値をクリックし、200と入力し**Return**を押します。



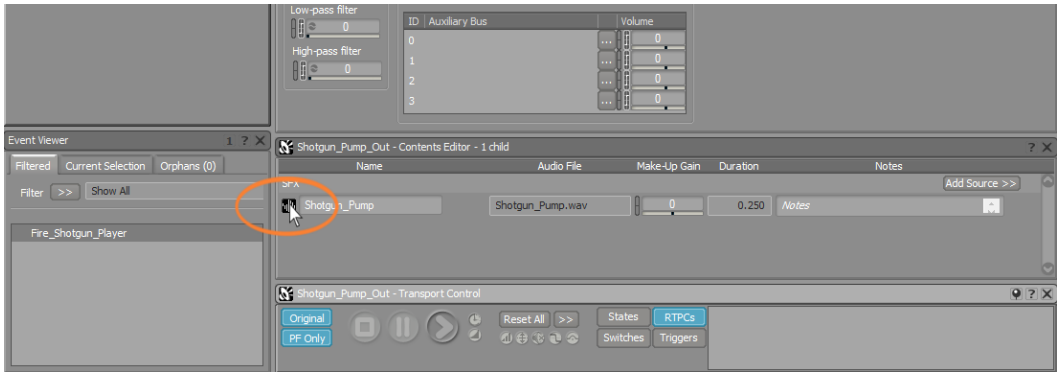
5. オブジェクトを再度再生し、変化を確認します。

   パンプインとは多少異なる現実的なパンプアウトを作成することができました。

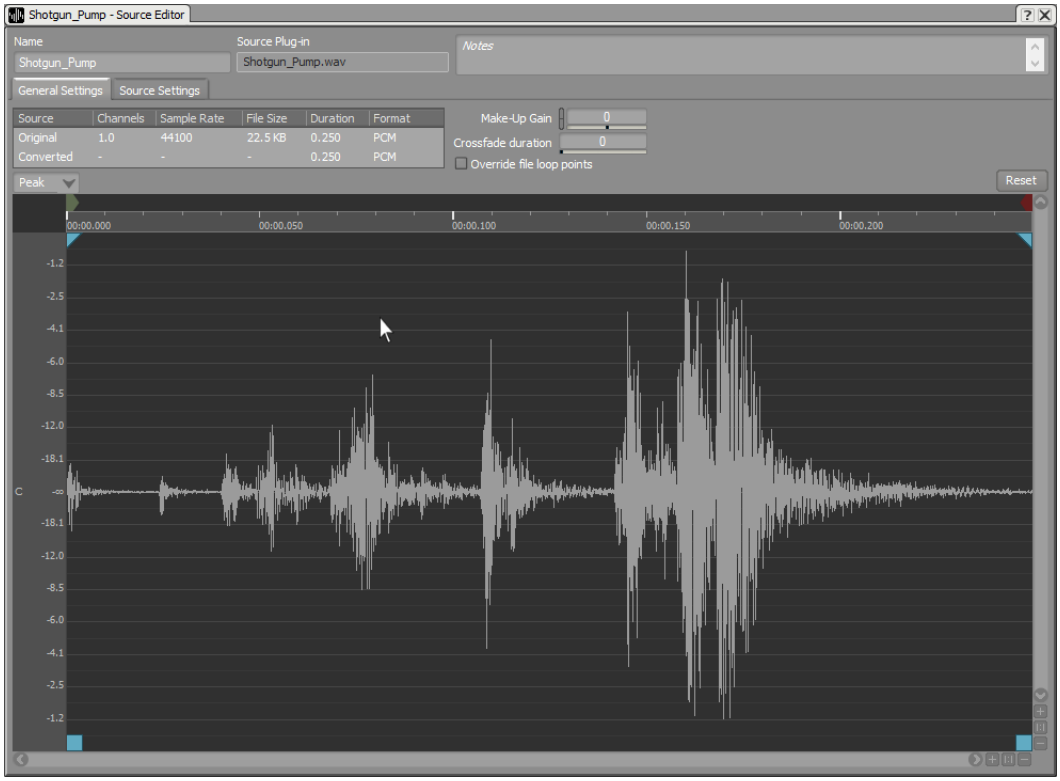
Source Editorの使用

パンプアウトサウンドが多少高く再生されますが、他のすべては同じです。短いサウンドですが、パンプメカニズムによって生み出されるリズムがあります。多くの方はこれらに気がつきます。そのためサウンドの再生させる場所を変更させることにより、パンプアウト独自のサウンドにします。これはSource Editorを活用し、実現することができます。

1. Shotgun_Pump_Out SFXオブジェクトが現在も選択されている事を確認し、Contents EditorでShotgun_Pump オーディオファイルのオーディオソースアイコンをダブルクリックします。



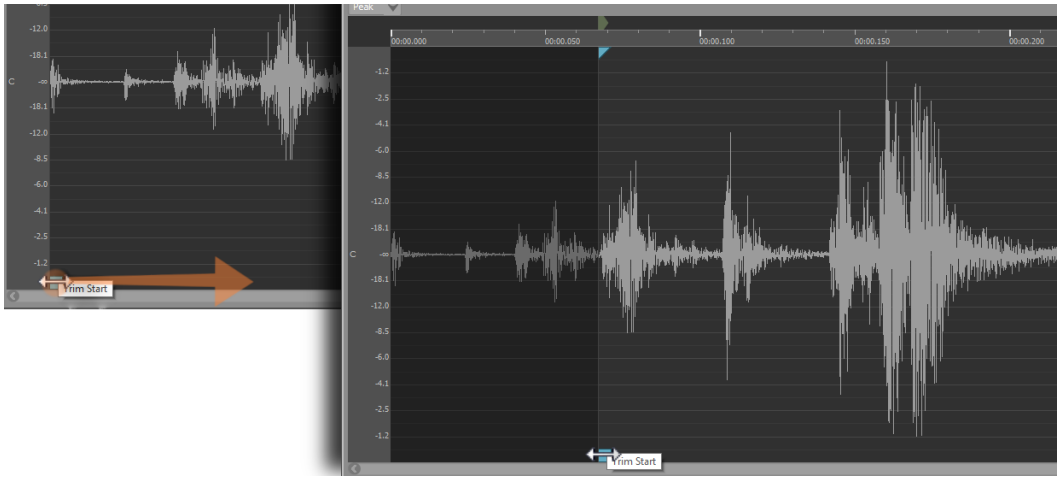
Source Editorが開きます。



Source Editorでは、ソースオーディオジェネレーターに関連した詳細を確認することができます。Wwiseでは、トーンジェネレーターやSoundSeedシンセサイザーを含むいくつかのサウンド生成方法を提供します。これらのツールが使われている場合は、Source Editorをクリックすることにより、これらのジェネレーターのインターフェースを表示することができます。このケースでは、ソースオーディオジェネレーターがオーディオファイルのため、オーディオシグナルの波形ビューが表示されます。このディスプレイは、シンプルな波形エディターとしても使用することができ、再生のスタートポイントとエンドポイントを調整することができます。

パンプアウトでは、オーディオファイルの再生位置を後ろに変更します。Trim Startコントロールを再生させたい場所にドラッグすることにより簡単に実現することができます。波形を解析することにより、メカニカルノイズのリズムパルスがどこにあるのかを見つけることができます。これらのセクションの1つ前から再生を始めるのが理想です。

2. Trim Start コントロールを波形の1/4までドラッグします。波形にある複数のピークの一つの直前です。



このスタート位置の変更は、Shotgun_Pump_Outサウンドオブジェクトから再生させる場合にのみ適用され、Shotgun_Pump_Inオブジェクトの聞こえ方は変更されません。まとめとして、サウンドSFXオブジェクトは、同じオーディオソースファイルを使う可能性がありますが、それぞれに独自のスタートタイムを設定することができ、オブジェクトごとに違いをつけることができます。

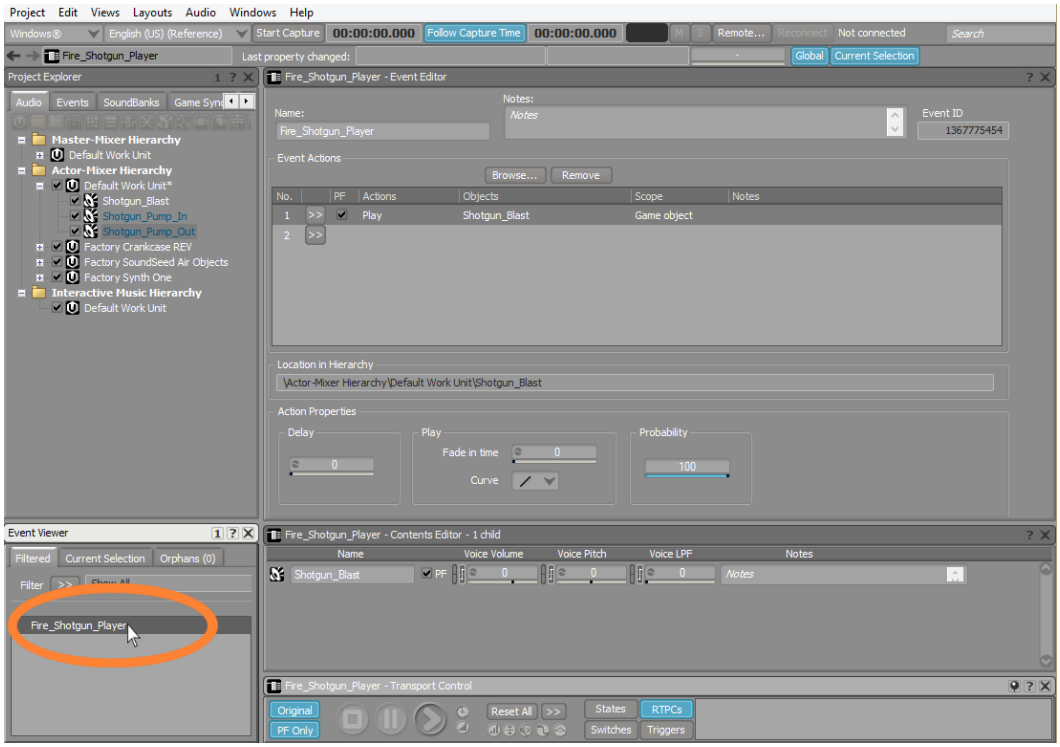
3. Shotgun_Pump_Outを再生し、違いを確認した後、Source Editor ウィンドウを閉じます。

単一イベントへの複数アクションの適用

現在、パンプサウンドを設定しました。それらを正しいタイミングで再生する必要があります。視覚的には、プレイヤーがショットガンを発射したときに、閃光が見え、パンプイン、そしてパンプアウトと続きます。ゲームエンジンプログラマーにパンプイン、及びパンプアウト用のゲームコールをお願いすることもできます。そうすることによりショットガンブラストのFire_Shotgun_Player イベントと同じように、関連するイベントを作成することができます。もしくは、プログラマーの

手を煩わせずに、これを実現させる方法を検討します。こうすることでオーディオ担当者、そしてプログラマ、どちらにとっても貴重な時間を節約することができます。ショットガンブラストとパンプは全て同じビジュアルアニメーションであるため、アニメーションが変更されない限り、ブラスト、パンプイン、そしてパンプアウトのビジュアルのタイミングは予測することができます。ショットガンブラスト用のFire_Shotgun_Player イベントはすでに作成済みのため、パンプサウンドをイベント後の予め決まったタイミングでトリガーする必要があります。これは、Playアクションを遅延させることで実現できます。

1. Event Viewerで、Fire_Shotgun_Playerイベントを選択します。

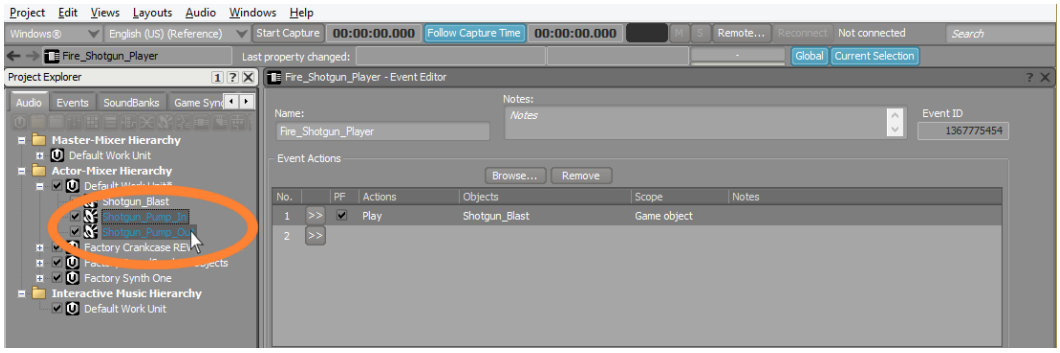


前のレッスンで設定したPlay Shotgun_Blast アクションをEvent Editorで見ることができます。単一のイベントにより実行可能なアクション数には実質的な制限はないので、Shotgun_Pump オブジェクトをアクションリストに加えます。今回は、Playアクションを作成し、オブジェクトを加える方法の代わりに、Project Explorerから直接オブジェクトをアクションリストにドラックします。これを行うためには、両方のShotgun_Pump オブジェクトを選択しなければなりません。

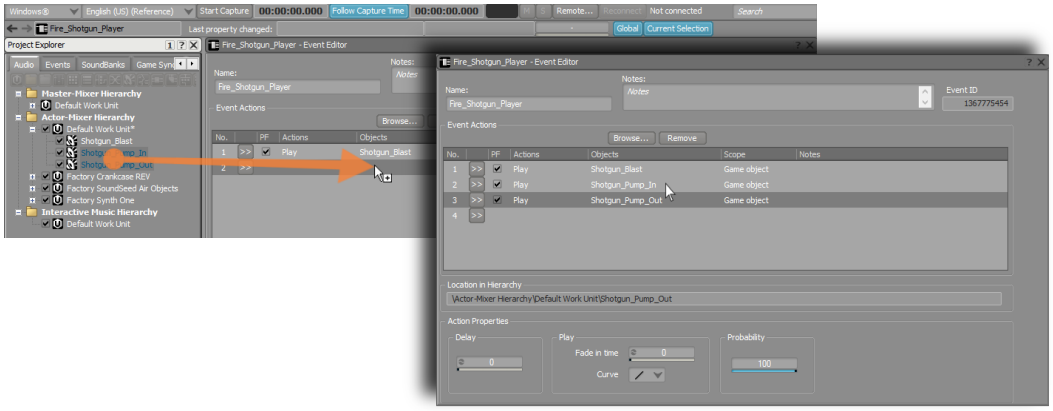
2. **Shift**キーを押した状態でShotgun_Pump_InとShotgun_Pump_Outオブジェクトを選択します。

注記

もしShiftキーを最初に押し忘れた場合、Shotgun_Pump オブジェクトの一つをクリックした瞬間に、Event EditorがProperty Editorに変わり、次のステップへ進むことができません。



3. 選択したオブジェクトのどちらかをドラッグし、Event Editorのアクションリストにドラッグします。

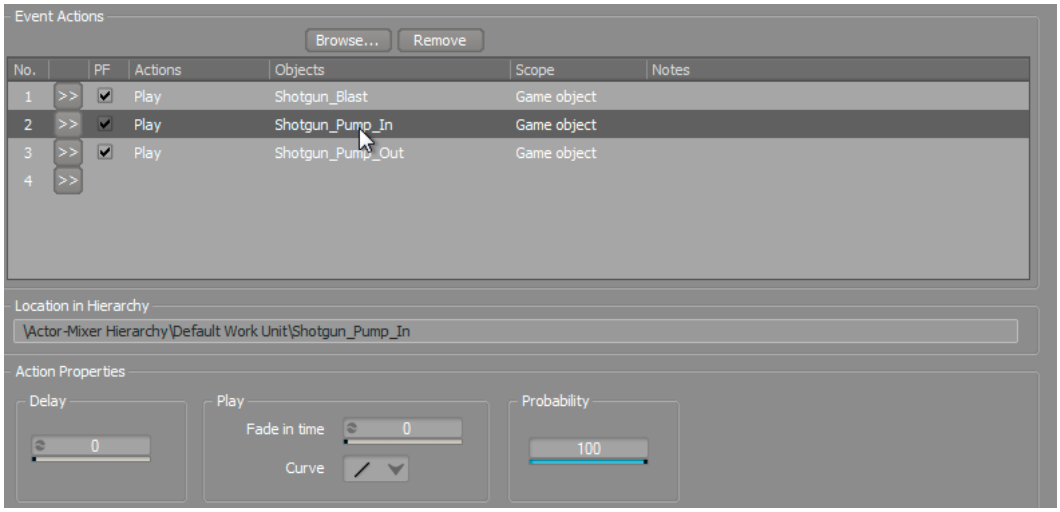


オブジェクトは、Event EditorのShotgun_Blastの下に追加されます。これらのアクションは番号順に並んでいますが、重要な点はこれらのアクションは番号順に実行されるのではなく、追加の設定を行わない限り、同時に実行されます。

4. Fire_Shotgun_Player イベントオブジェクトを再生します。

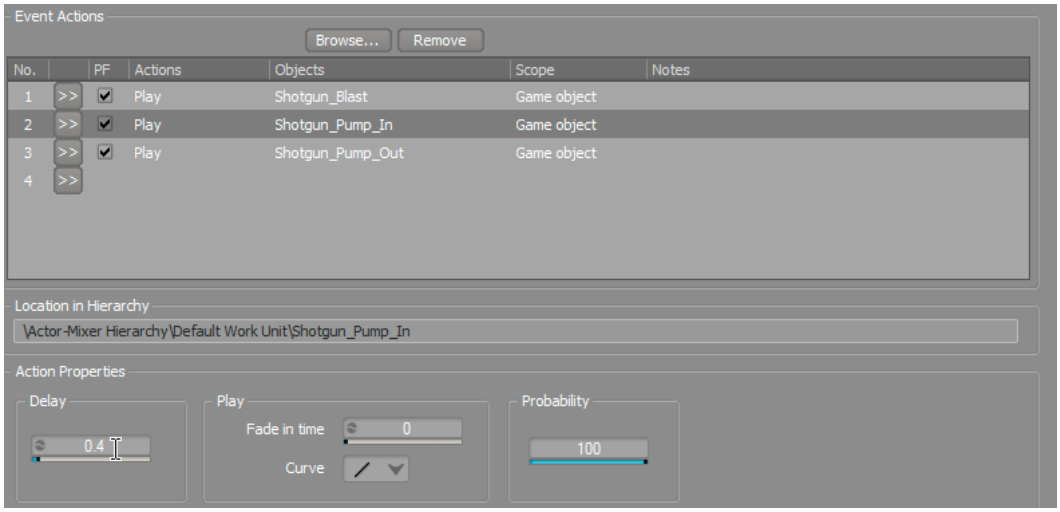
   ブラストとパンプサウンドが同時に再生されたことに気が付きます。パンプサウンドをアニメーションのシーケンスに合わせるために、それぞれのパンプサウンドに数値を与えて遅延させます。

5. Event Actions リストで、2行目のShotgun_Pump_Inを選択します。



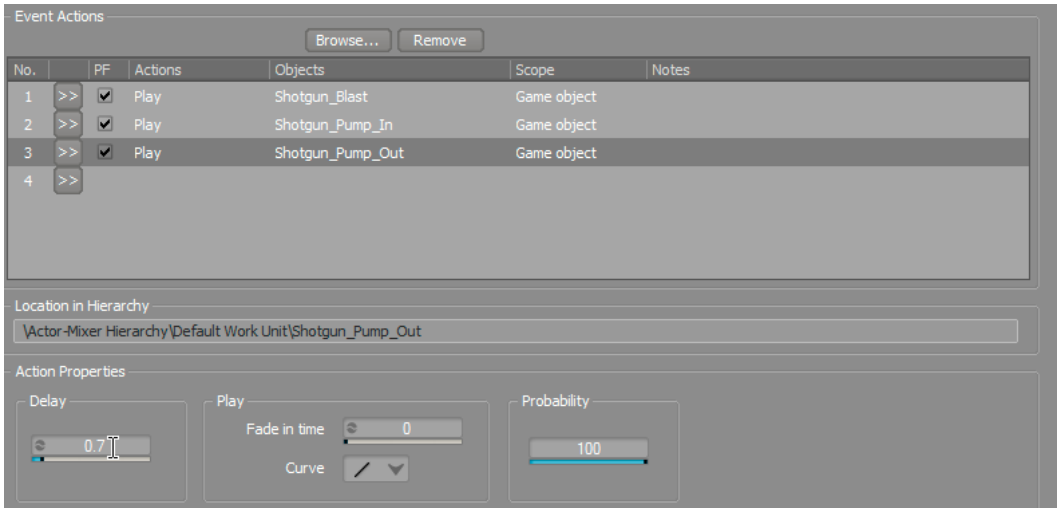
Event Actions リストの下のAction Propertiesエリアで、Delay プロパティーを見つけることができます。パンプインサウンドは、最初のブラストから0.5秒以内に聞こえなければなりません。Delay値は秒単位で指定します。.4は400ミリセカンドと等しく、0.5秒より少し短くなり、ちょうどよいタイミングとなります。

6. Delay パラメーターをクリックし、.4と記入し**Enter**を押します。



パンプアウトサウンドは、最初のブラストから0.5秒以降に聞こえなければならず、ここでは.7がよい選択になります。

7. 3行目のShotgun_Pump_Out アクションを選択し、Delay値を.7へ変更し**Enter**を押します。



ショットガンパンプが入った全体のブラストシーケンスをテストすることができます。

8. Fire_Shotgun_Player イベントを再生します。

ランダマイゼーション（ランダム化）機能の使用

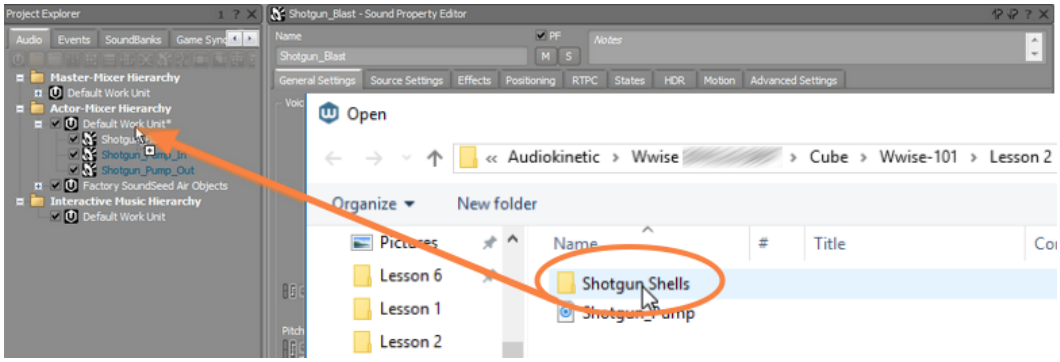
オーディオがビデオゲームに与える大きな要素の一つとして、ビジュアルだけでは伝えられない部分をオーディオで補うことができます。オーディオの役目は、スクリーンで見えるグラフィックのサポートをするだけではありません。ゲームで見えている以外の部分を考えることが重要になります。そして何より重要なことは、見えない部分にフォーカスすることです。例えば、ショットガンのパンプの目的は、使用済みのシェルを出し、次を装填することです。ここで問題になるのは、シェルがどこに飛ぶかです。もちろん地面に落ちますが、これを示すショットガンのアニメーションはありません。これは、シェルを加えてはならないという意味ではありません。加えるべきなのです。これが、プレイヤーの没入感につながります。

ショットガンのシェルが地面に落ちるサウンドを加えます。シェルが地面に落ちるたびにシェルは違うはね方をし、違うリズムを生み出します。そのため、同じシェルがはねる録音を何度も繰り返し聞くことは不自然で、プレイヤーにとって不快なものになります。これを避けるために、ショットガンのシェルを地面に落としたときの複数のオーディオファイルをインポートします。ショットガンが発射されるたびにインポートされた複数のうちの1つを選択することができるオーディオオブジェクトRandom Containerを使用すます。

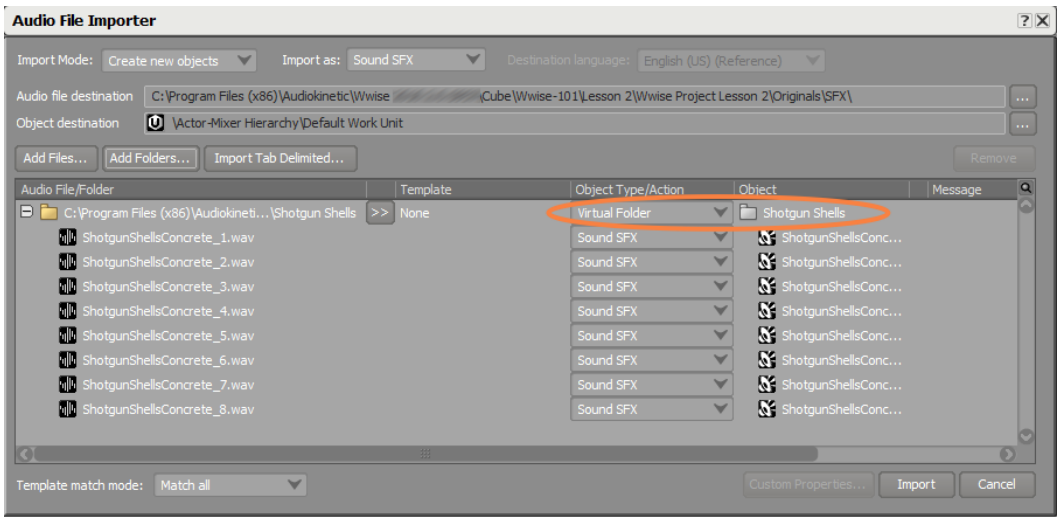
オーディオファイルフォルダーのインポート

前回のレッスンでは、ショットガンのブラストサウンドをWwiseのインポートプロセスを使い追加しました。システムウィンドウからWwise Projectへ直接オーディオファイル、もしくはオーディオファイルを含んだフォルダーをドラッグすることもできます。

1. オペレーティングシステムで、`Wwise-101 > Lesson 2 > Audio files for Lesson 2`のオーディオファイルまで移動し、Shotgun Shellsフォルダーをを Actor-Mixer HierarchyのDefault Work Unitへドラッグします。

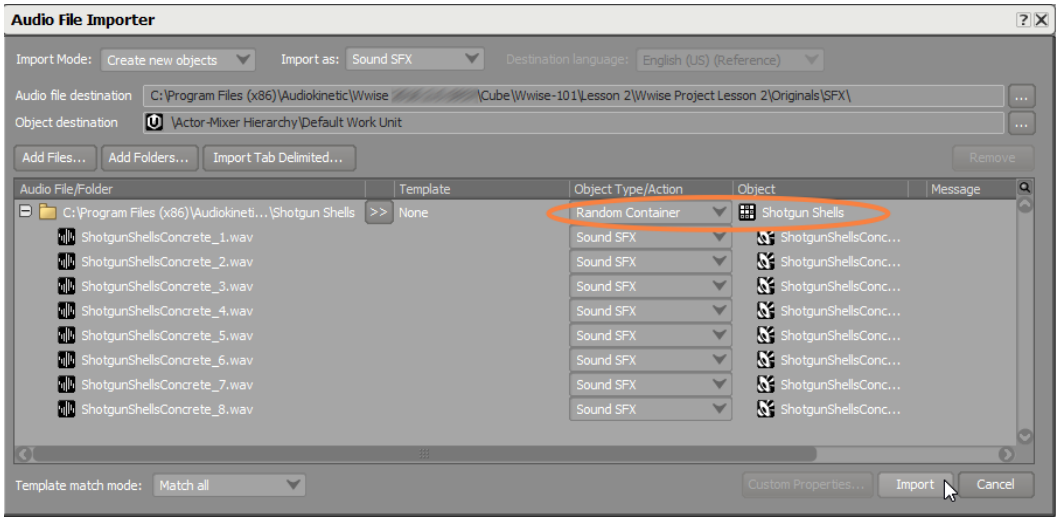


Audio File Importerが開きます。

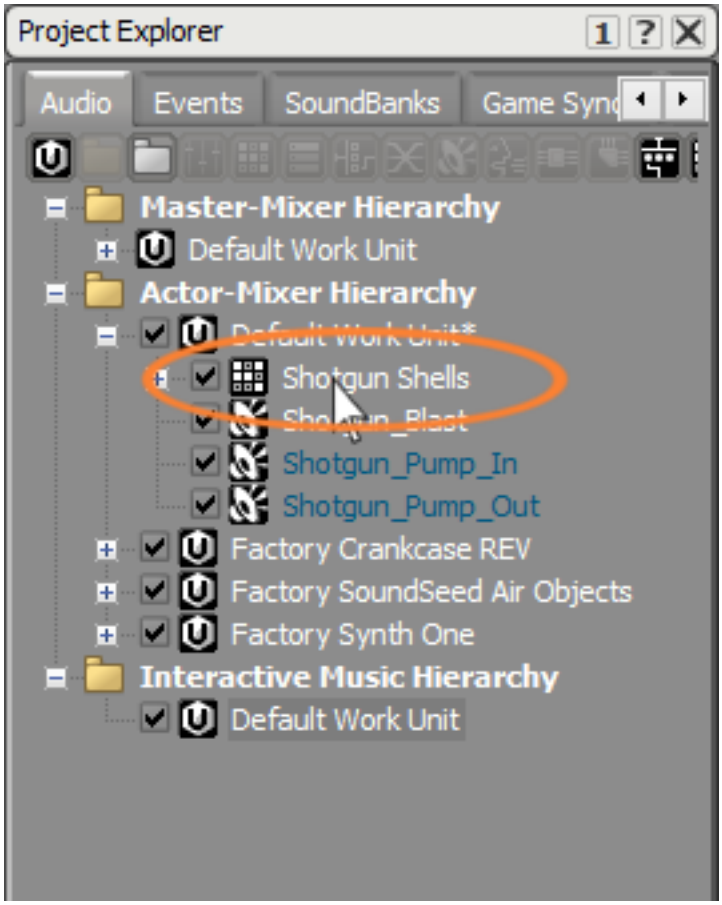


これからインポートするフォルダ内のオーディオファイルが、Sound SFXオブジェクトとして取り込まれることが分かります。最初の行に、フォルダ自体が階層構造にインポートされるオブジェクトとして表示されています。このフォルダはデフォルトでVirtual Folderとしてインポートされますが、これは単純にWwise階層内でオブジェクトを整理するための設定です。オブジェクトタイプをRandom Containerに変えることもできます。Random Containerオブジェクトとは、追加の機能があるフォルダだと考えてください。Random Containersは、Sound SFX オブジェクトのようにTransport Controlのプレイボタンを使い再生することができます。このケースでは、含まれているオブジェクトがランダムで選択され再生されます。これは、同じオーディオファイルが毎回再生されたら、繰り返しが気になるであろサウンドにバラエティを追加する最適な方法です。

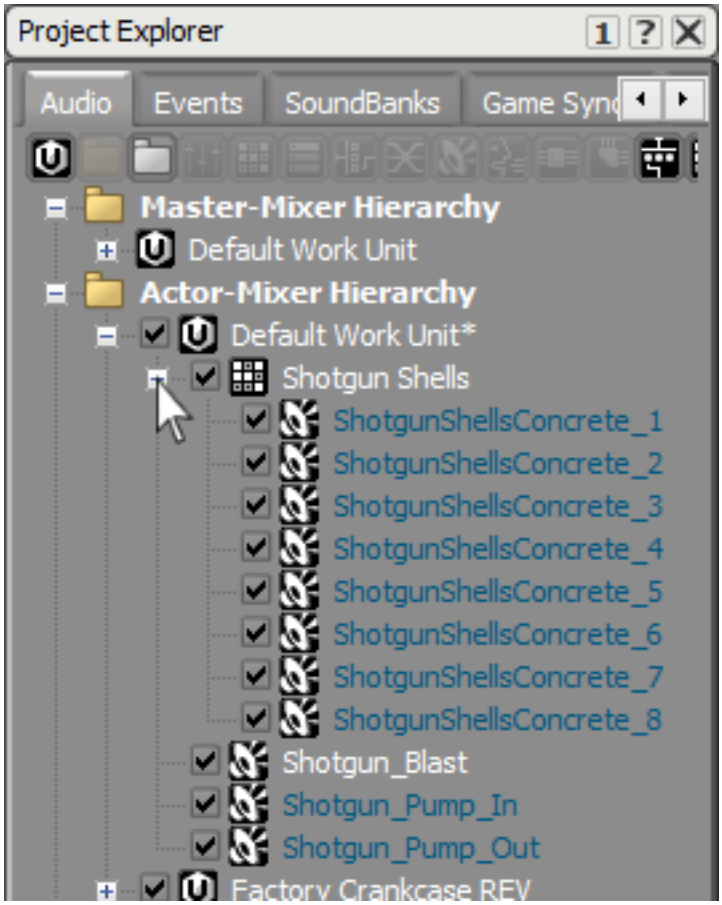
2. 最初の行のObject Type/ActionをRandom Containerに変えて、**Import**をクリックします。



プロジェクトエクスプローラーで、Shotgun Shellsという名前の新しいオブジェクトを見ることができます。これは今まで使ってきたSound SFX オブジェクトとは違うアイコンです。

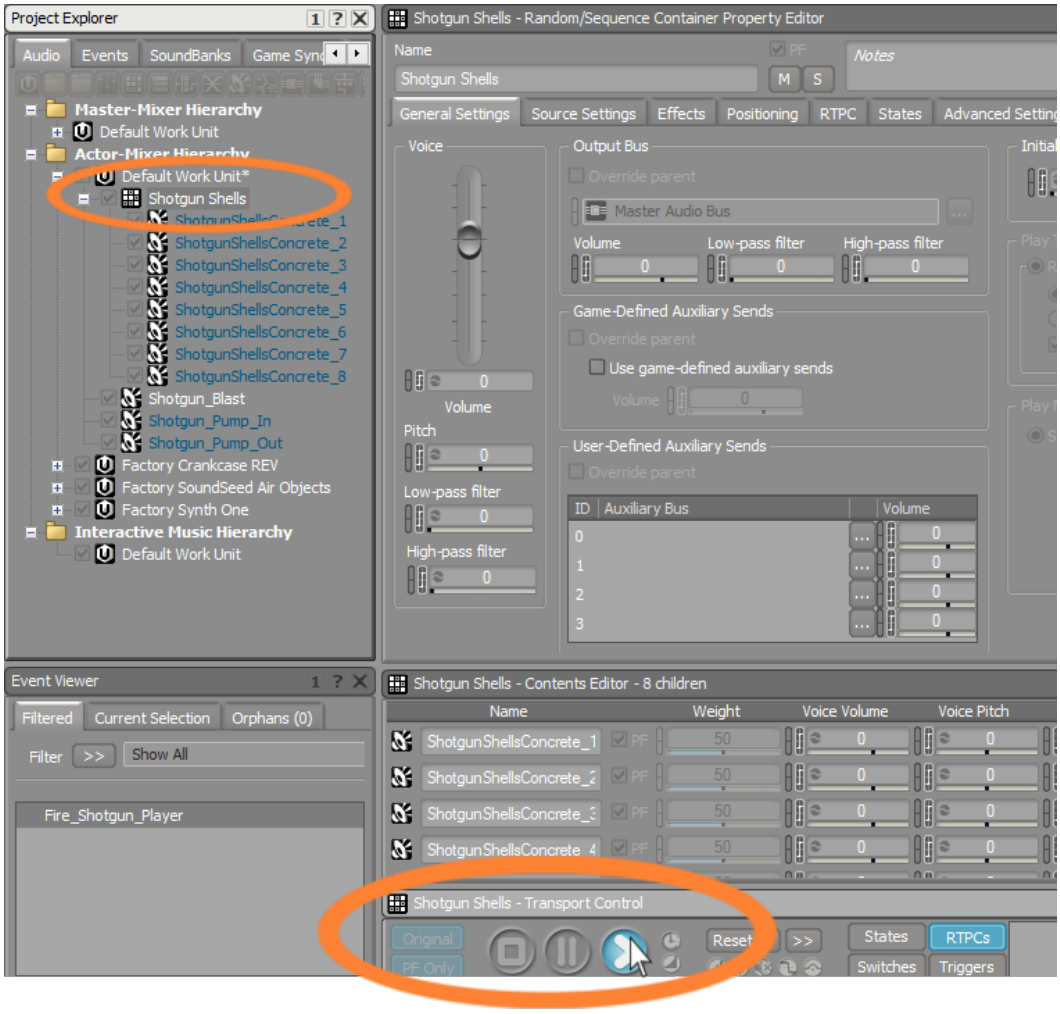


3. Shotgun Shells Random Containerオブジェクトを展開します。



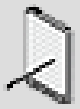
Shotgun shellの録音すべてがSound SFX オブジェクトとしてインポートされ、これらはShotgun Shells Random Containerに内包されます。

4. Shotgun Shells Random Containerを選択し、オブジェクトを複数回再生します。



シェルのサウンドが違うことに気が付きます。Shotgun shells オブジェクトを再生するたびに、WwiseがShotgun shell音をランダムに選択しているからです。

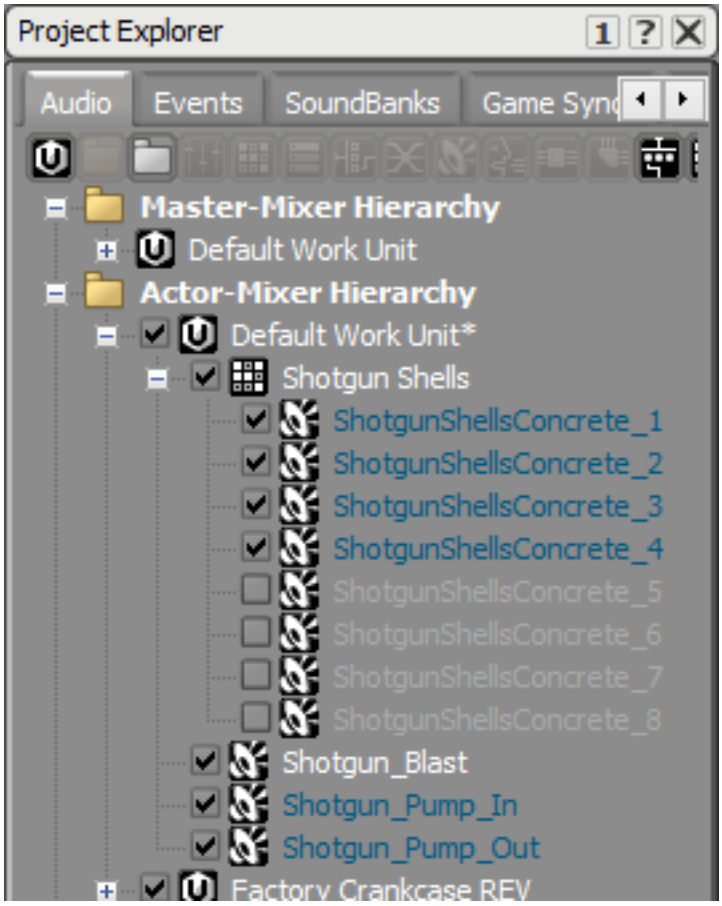
シェルの音の一部が気になる、もしくはランタイムのメモリーを消費する多彩なサウンドを持つことを心配するかもしれません。Wwiseのプロジェクトでは、特定のシェルの録音をWwiseプロジェクトから永久に削除せずに、インクルード/エクスクルード（含める/含めない）を選択することができます。これは、プロジェクトエクスプローラーのオブジェクトの左にあるチェックボックスでプラットフォームインクルージョン/エクスクルージョンの設定を行えます。これらのチェックボックスは、ミキシングコンソールのミュートボタンと考えることができるかもしれません。チェックボックスのチェックを外すことにより、そのサウンドスケープからその役割を除外し、後に戻したい場合はチェックボックスに再度チェックを入れます。



注記

レッスン7のプロジェクト最適化でプラットフォームのインクルージョン/エクスクルージョンチェックボックスを学ぶことができます。

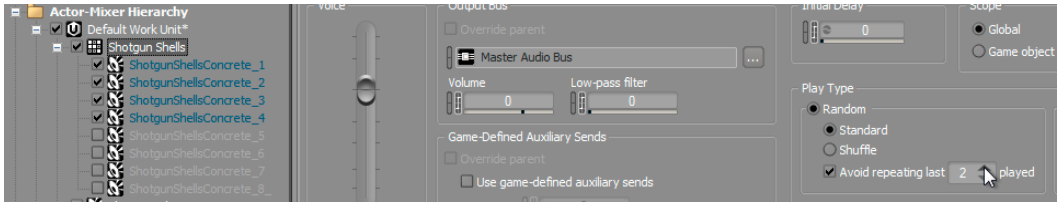
5. 最後の4つのシェルのプラットフォームインクルージョン/エクスクルージョンチェックボックスを外します。



ランダマイゼーションの操作

Shotgun Shellsはランダマイズ化されましたが、まだ連続して同じShellサウンドが再生される可能性があります。これを防ぐために、ランダマイゼーションをどのように行うかをコントロールするプロパティーがあります。

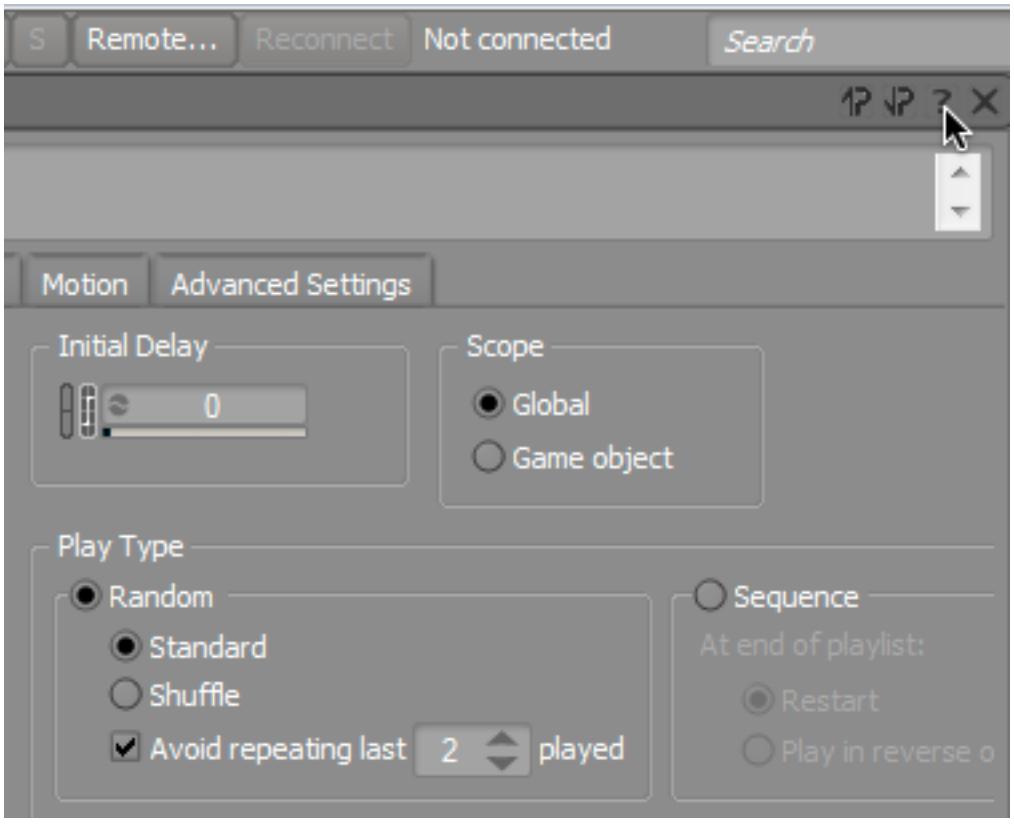
1. Shotgun Shells Property Editorで、**Avoid repeating last** チェックボックスを選択し、Last 2 playedに変更します。



これにより、まったく同じシェルのサウンドを連続して聞くことがなくなります。

ここで気になることは、StandardとShuffle randomization radio ボタンで、どのように使われるのかということです。インターフェースに関する質問に素早くアクセスするには、プログラムのすべてのビューにあるクエッションマークシンボルボタンを押すことにより実現します。そのビューに表示されているすべての機能に関する説明、および使われ方を見ることができます。

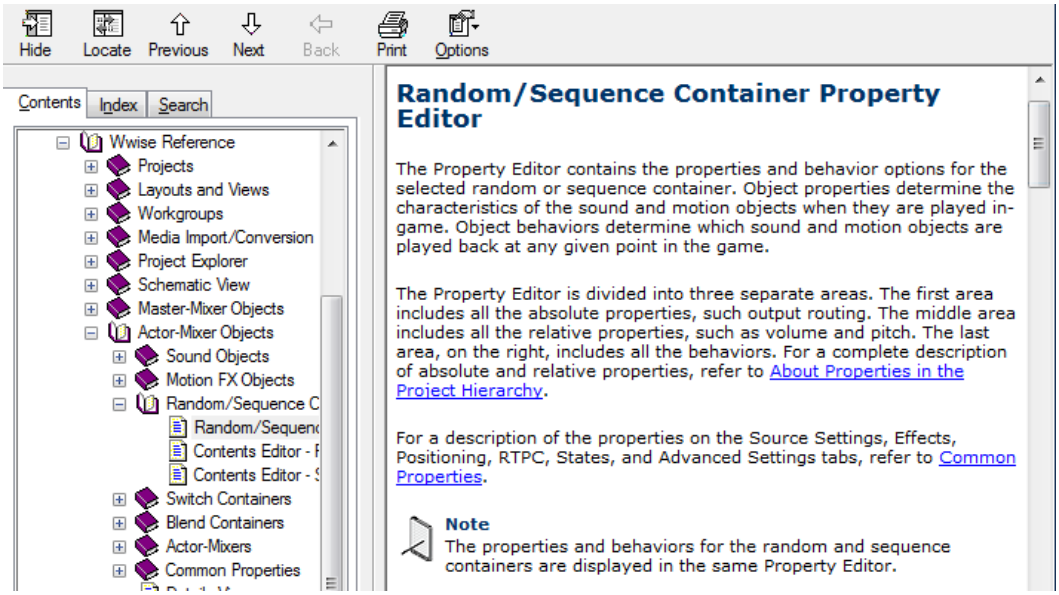
2. Property Editor ビューの右上のコーナーにあるヘルプボタンをクリックします。



Wwiseコンテクストヘルプがプロパティエディターに表示されているプロパティの全てについての情報を表示します。

ティップ
User Preferences ダイアログで、コンテクストヘルプから Audiokinetic のオンラインヘルプを開くのか、それともWwise インストール時に入れることができる CHM を開くのか、設定します。

3. ページの最後までスクロールダウンして、How do I?... の下にある「Random Containerの作成」 リンクをクリックして、 Random Containerの活用例 と題されたセクションまで移動します。事例の手順6を読み、Standard とShuffle の違いを理解してください。



Shuffleは、Shotgun Shellsを選択することにより、同じサウンドを繰り返し聞く可能性が低くなります。

4. Shuffle ラジオボタンをクリックし、Shotgun Shells Random Containerを再生し効果を確認します。

プロパティーのランダマイゼーション

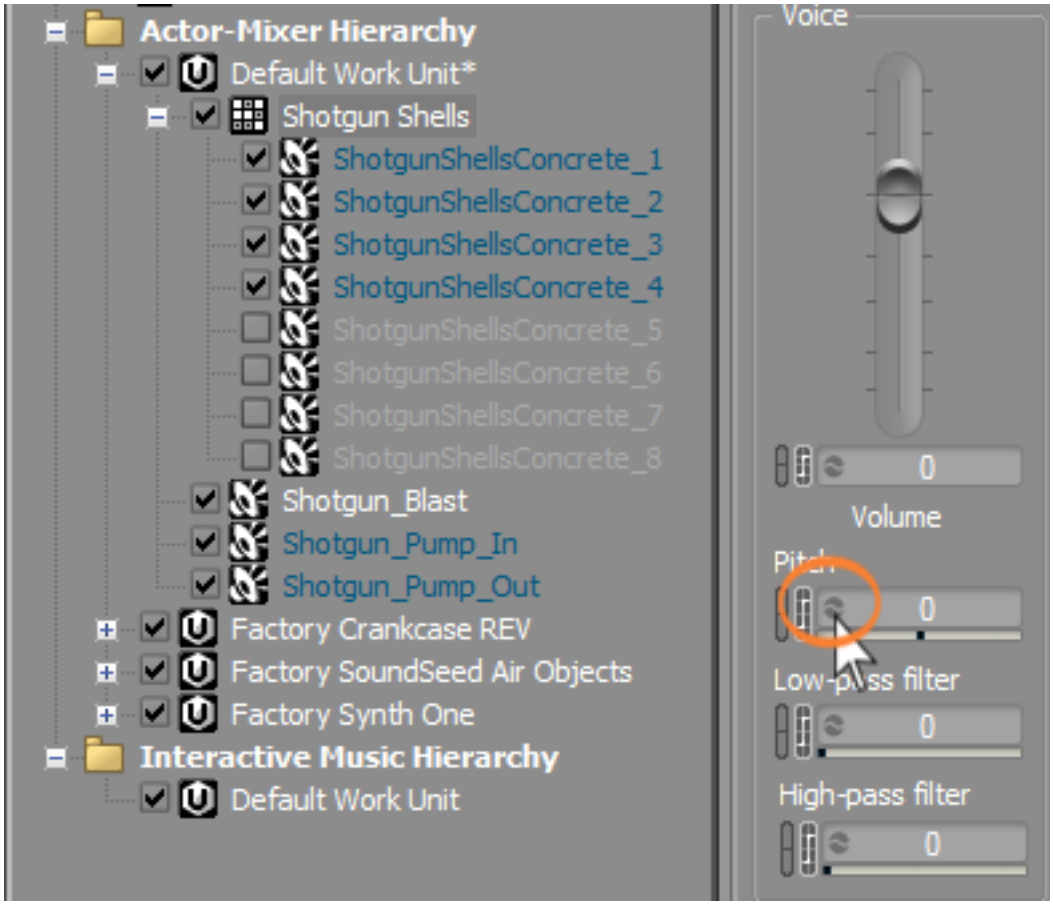
サウンドセクションをランダム化する他に、ボリューム、フィルター、そしてピッチなどのほとんどのプロパティ値をランダム化させることができます。そのため、

サウンドのランダム化と組み合わせることにより、ランタイムのメモリー使用量を増やすことなく、サウンドの種類をたくさん作ることができます。

ランダマイザーボタンはプロパティ値の左に見つけることができ、グレーのサークルに曲がった線が入っています。

Shotgun shellの録音にピッチランダマイジングを適用させ、Shotgun Shellsにさらなる変化を加えます。

1. Shotgun Shells Property Editorで、Pitch プロパティー用のランダマイザーアイコンをダブルクリックします。

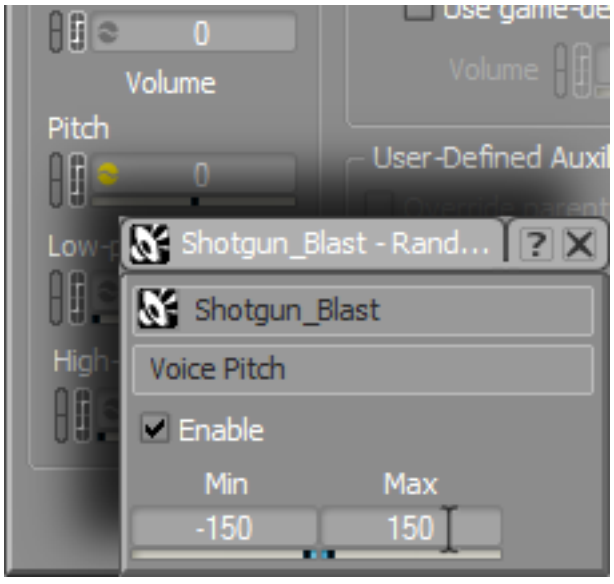


ランダマイザーウィンドウが開き、ランダマイザーを有効にするチェックボックス、そして最少値と最大値用のセッティングが表示されます。設定している内容を反映させるためには、最初にピッチプロパティのランダマイゼーションを有効にする必要があります。

2. **Enable**チェックボックスをクリックします。

ランダマイゼーションが有効になったことを示す黄色のピッチプロパティ用のランダマイザーアイコンを確認します。ランダマイゼーションが有効になっていても、Min/Max値を変更しない限り、違いを聞くことはできません。このMin/Max値は、相対オフセット値の範囲を示し、このプロパティのUIで設定した値を中心に設定されます。ピッチの場合は、値はセントで示されます。音楽的な半音は100セントであり、オクターブは1200セントです。

3. MinとMaxフィールドに、それぞれ-150と150を入力します。



Shotgun shellが再生されるたびに、ピッチは、レコーディングされているピッチから上下150セントの間の値をとります。

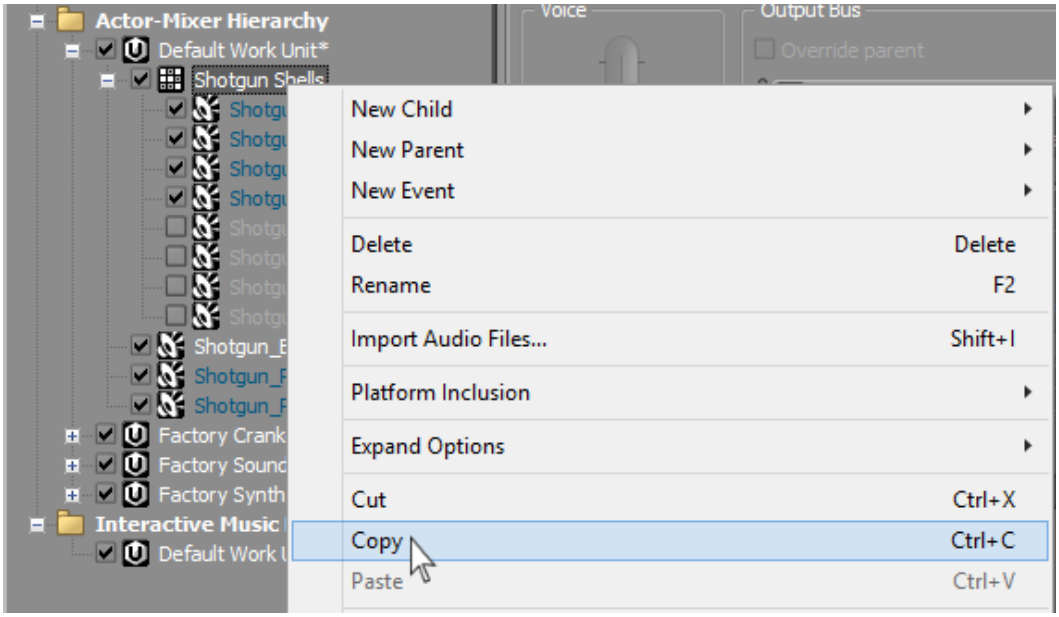
4. Shotgun Shellsを何回か再生し、再生するたびにピッチに変化があることに気が付きます。Pitch Randomizerウィンドウを閉じます。

サウンドのグラニュラ化

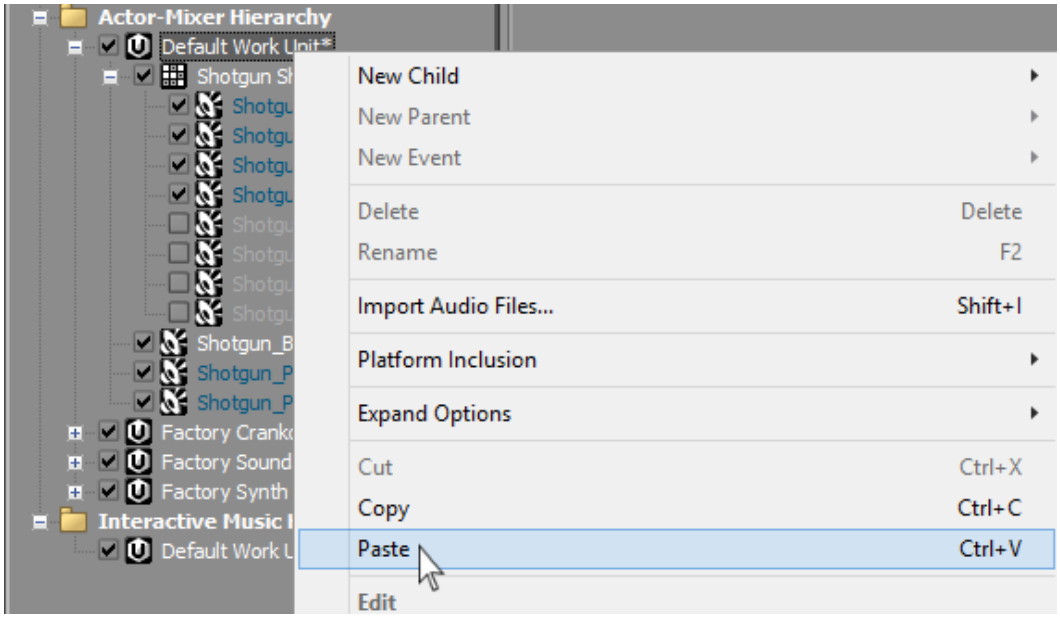
Shell の録音を2つのパートに分けることにより、Shotgun Shell サウンドにさらなる違いを付けることができます。最初に地面にヒットしたサウンドをヘッドとし、シェルが地面をはねているときのサウンドをテイルとします。 サウンドを複数のパートに分けることにより、それぞれ違うヘッドサウンドとテイルサウンドをランダムに組み合わせ、さらなるバリエーションを作り出すことができます。

すでに作業を行った場所のコピーを作成することから始めます。

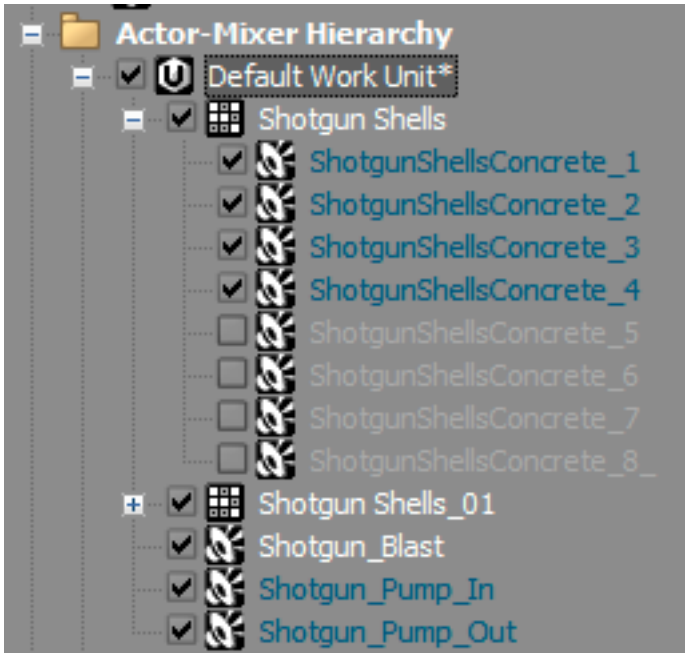
1. Shotgun Shells オブジェクトを右クリックし、**Copy**を選択します。



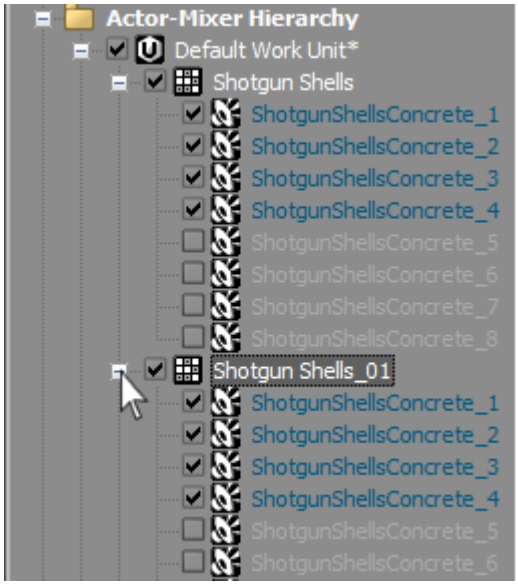
2. Actor-Mixer Hierarchyで、Default Work Unitを選択し、**Paste**を選択します。



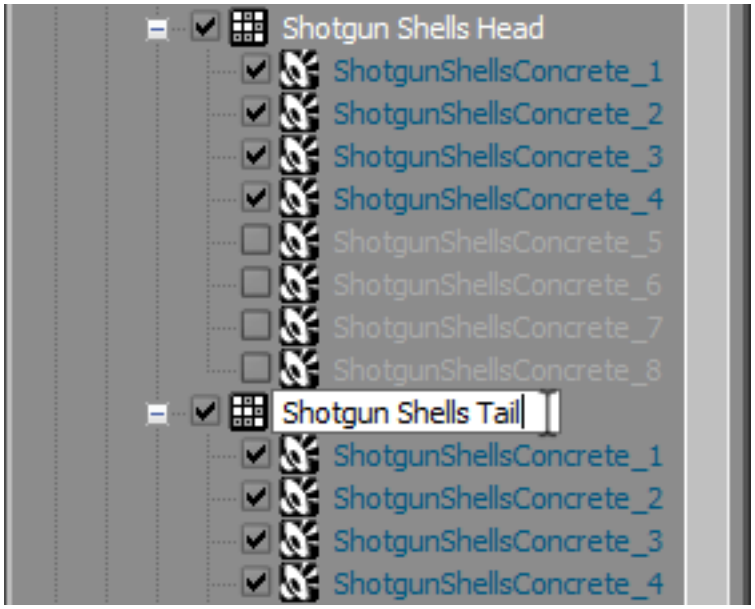
Shotgun Shell_01という名のRandom Containerが複製されます。



3. 内容を確認するためにShotgun Shells_01を開きます。

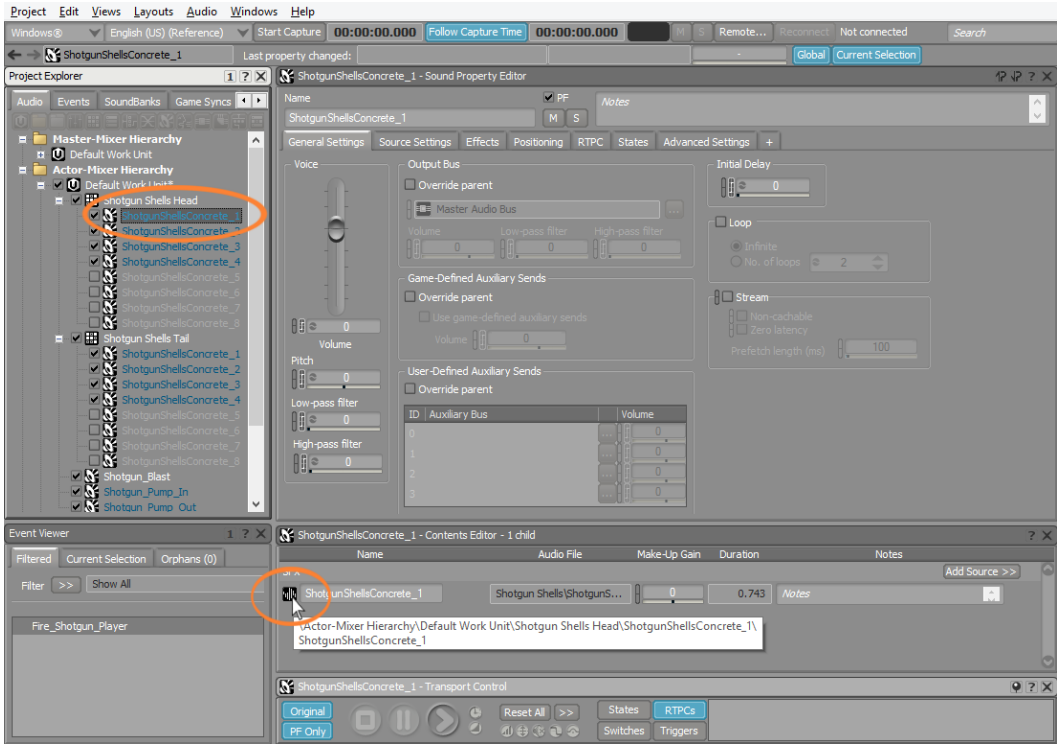


4. 2つのShotgun Shells Random Container オブジェクトをShotgun Shells Head とShotgun Shells Tail へとリネームします。



まったく同じ内容のRandom Containerが2つ作成されます。インポートしたオーディオファイルの一部分だけを再生させるために、内包されているSound SFX オブジェクトを編集します。これを行うために、内包されているSound SFX オブジェクトのオーディオファイルの波形を確認します。

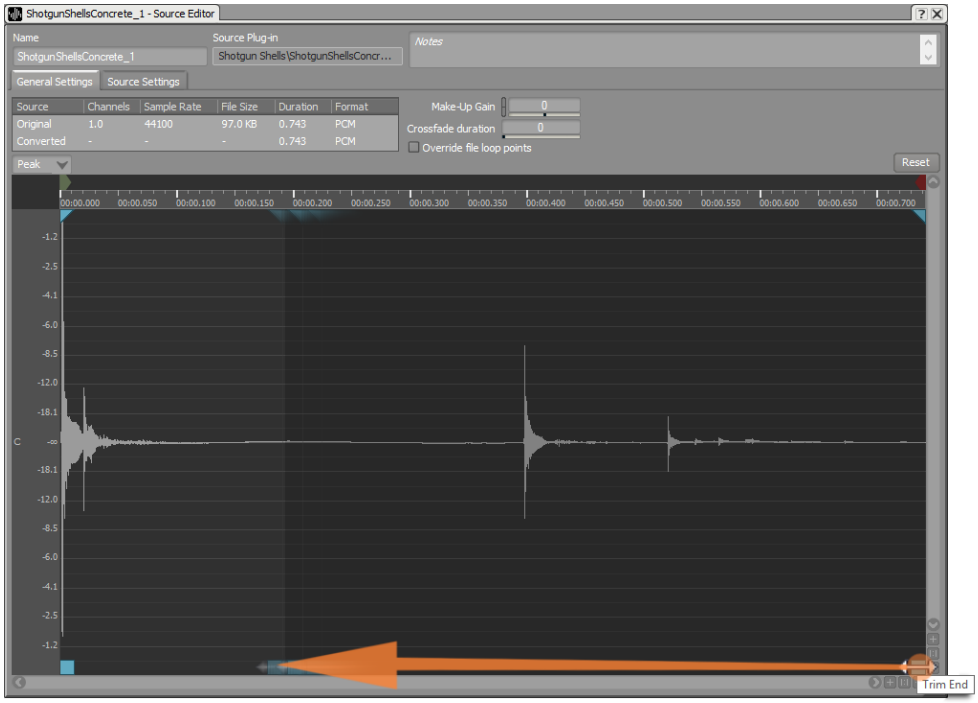
5. Shotgun Shells Head Random Containerでは、ShotgunShellsConcrete_1 オブジェクトを選択し、Contents Editorで、ShotgunShellsConcrete_1 ソースオブジェクトアイコンをダブルクリックします。



ソースエディターが表示され、ウェーブフォームとして使用されるオーディオファイルが表示されます。Shotgunのシェルが地面にはねるたびに、トランジェントを見ることができます。

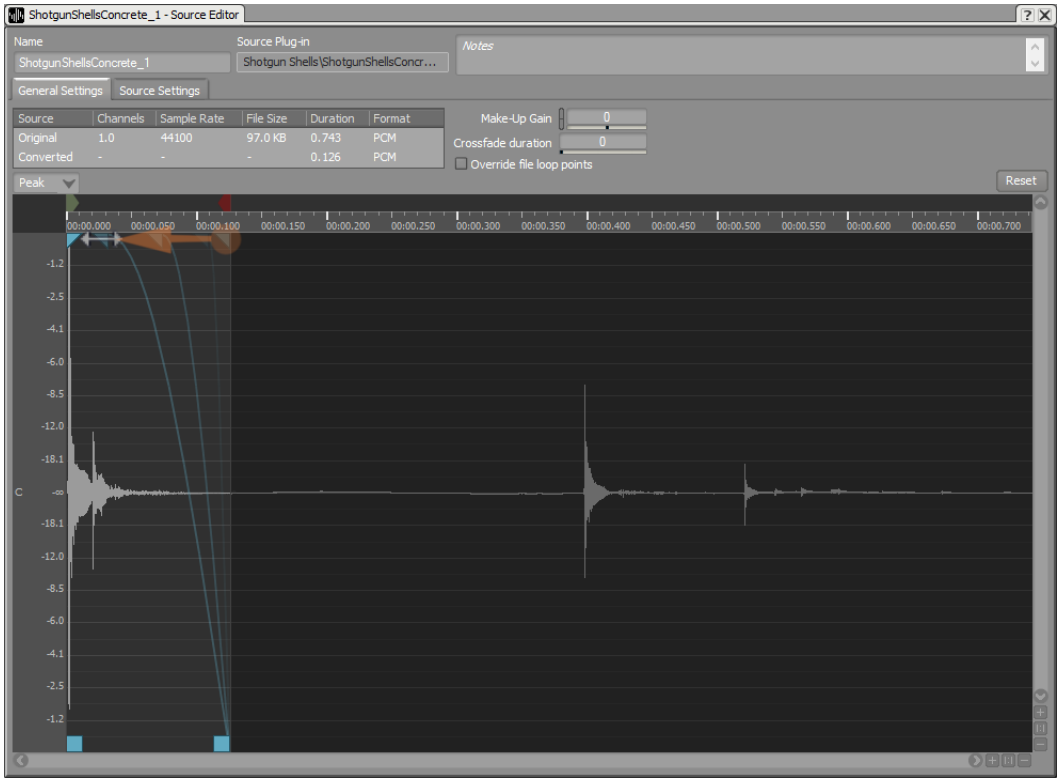
波形表示の下部に、再生されるオーディオの範囲を示すブルーの四角が表示されます。これらは動かすことができ、オーディオファイルの決められたセクションを再生します。Head Random Containerのすべてのシェルは、地面に当たった最初のシェルサウンドのみ再生します。

6. エンド位置にある四角をドラッグし、最初の薬きょうが落下した場所のみ選択し、選択した部分を再生します。



サウンドを再生した際に、クリックやポップノイズが聞こえる場合があり、フェードインやフェードアウトを適用したい場合があります。このようなクリックやポップを削除、もしくはフェードインやフェードアウトを適用されるためには、ウェーブフォームディスプレイの上部にある青い三角を調整します。

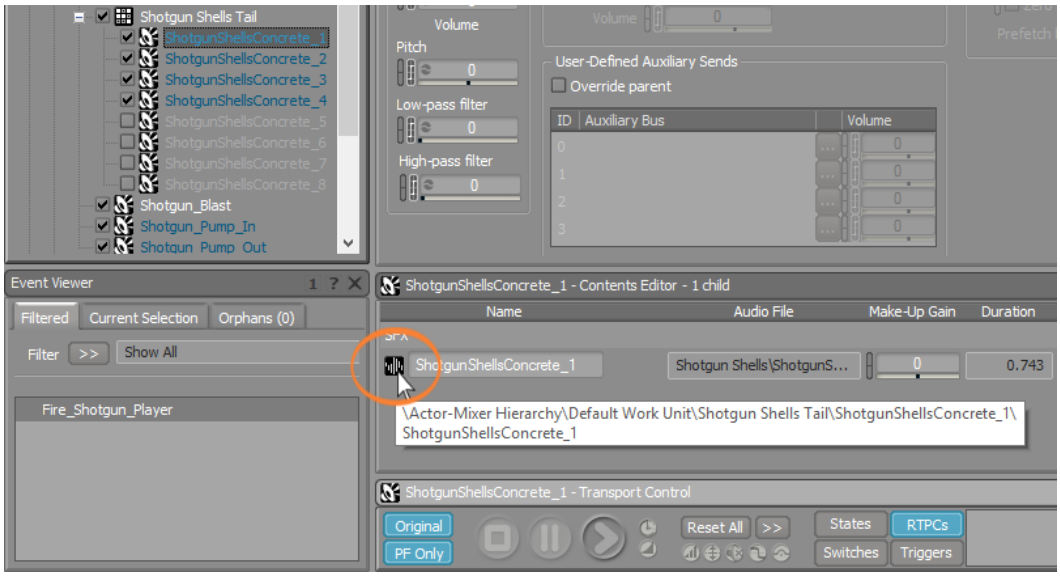
7. 右にある三角を左にドラッグすることにより、エンディングをスムーズにすることができます。ファイルを何度か再生し、この調整を行います。



8. Shotgun Shells Head オブジェクト内のの残りのオブジェクトに対しても、5から7のステップを繰り返します。

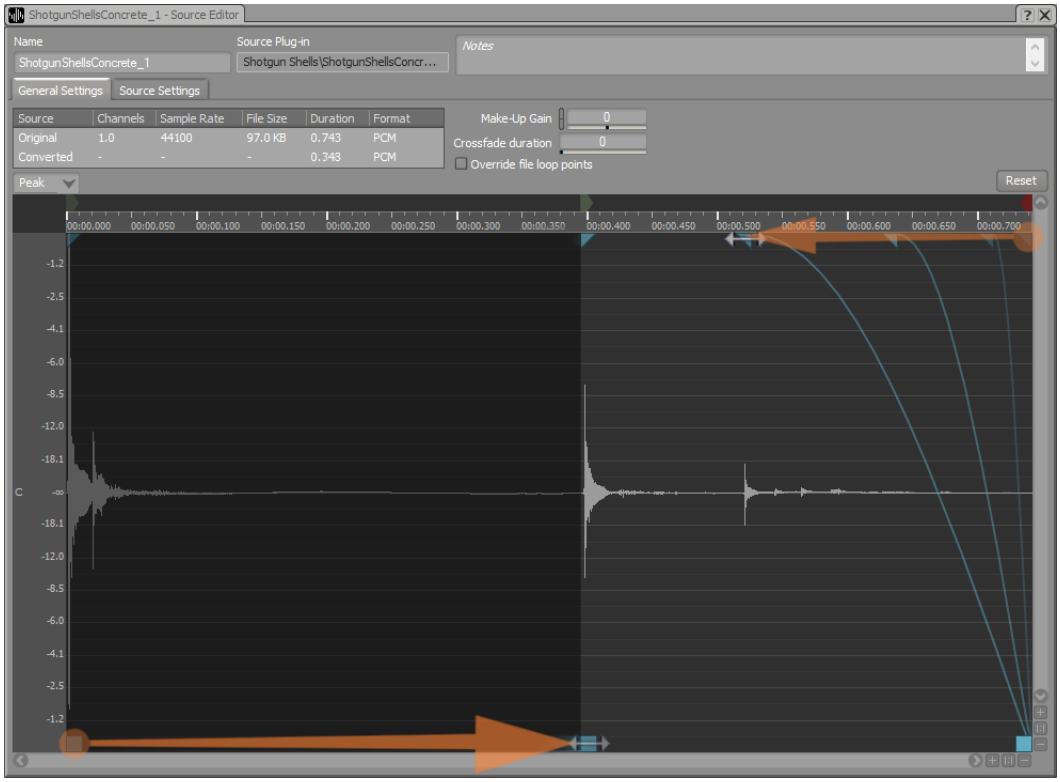
   同じようなことをShotgun Shells Tail Random Containerにも行います。

9. Shotgun Shells Tail Random Containerを開き、ShotgunShellsConcrete_1 オブジェクトを選択します。



すべてのShell Tail Sound SFX オブジェクトにおいて、最初の部分を聞こえなくし、最初の部分以外でシェルが地面を跳ねているサウンドが聞こえるように、Source Editorを使い調整します。

10Shotgun Shells Tail Random Containerにあるすべてのサウンドオブジェクトのスタート部分を調整し、後半部分の直前から再生されるようにします。スムーズなサウンドにするために、必要な箇所にフェードを適用します。



シーケンスの作成

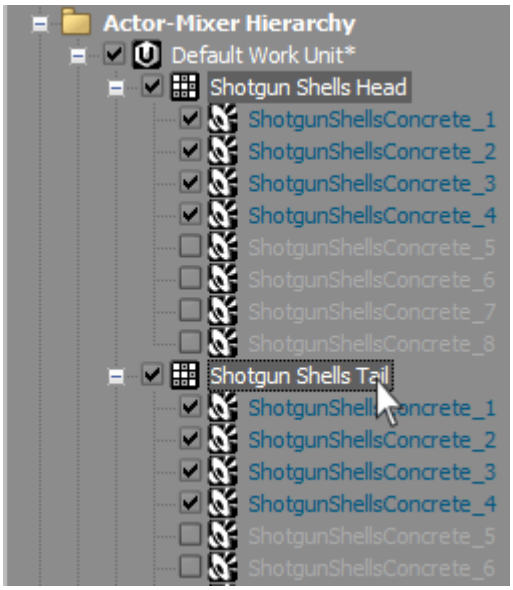
現在、HeadとTailのセクションがあります。今からこれらをつなぎ合わせることにより、ランダムに選択されたHead、そしてそのあとにランダムに選択されたTailがきこえるようにします。

Sequence Containerという異なるタイプのオブジェクトを使い、HeadとTailをつなぎ合わせます。Random Containerと同じように、これは他のオブジェクトを内包します。しかしながら、ランダムにそれらを再生する代わりに、オブジェクトをどの順番に再生するかを決めることができます。このケースでは、Head Shellサウンドを最初に再生し、そのあとにTail Shellサウンドを再生します。

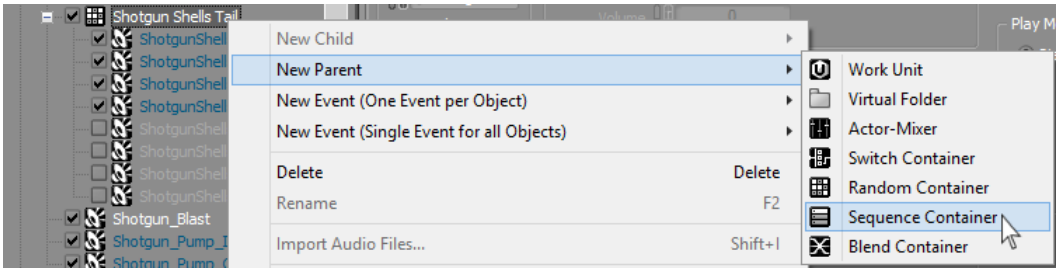
それぞれのHeadとTailのランダムコンテナーに適用されたランダム化と、ピッチのランダム化により、リスナーは使われているシェルの録音数が実際には非常に少ないことに気づくことはないでしょう

Shotgun Shells HeadとTailのランダムコンテナーをシーケンスコンテナオブジェクトに入れるところから始めます。

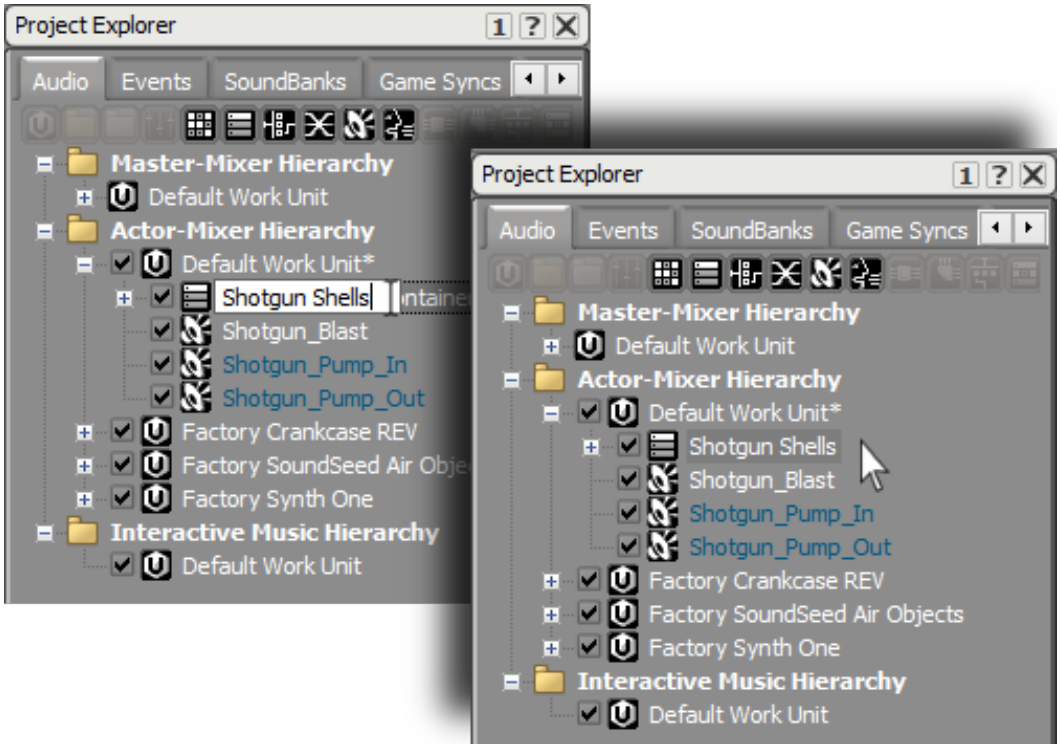
1. **Ctrl** キーを押したまま、Shotgun Shells Head と Shotgun Shells Tail のランダムコンテナーを選択します。



2. ２つの選択されたオブジェクトのうちの一つを右クリックし、**New Parent** > **シーケンスコンテナーを選択します。**

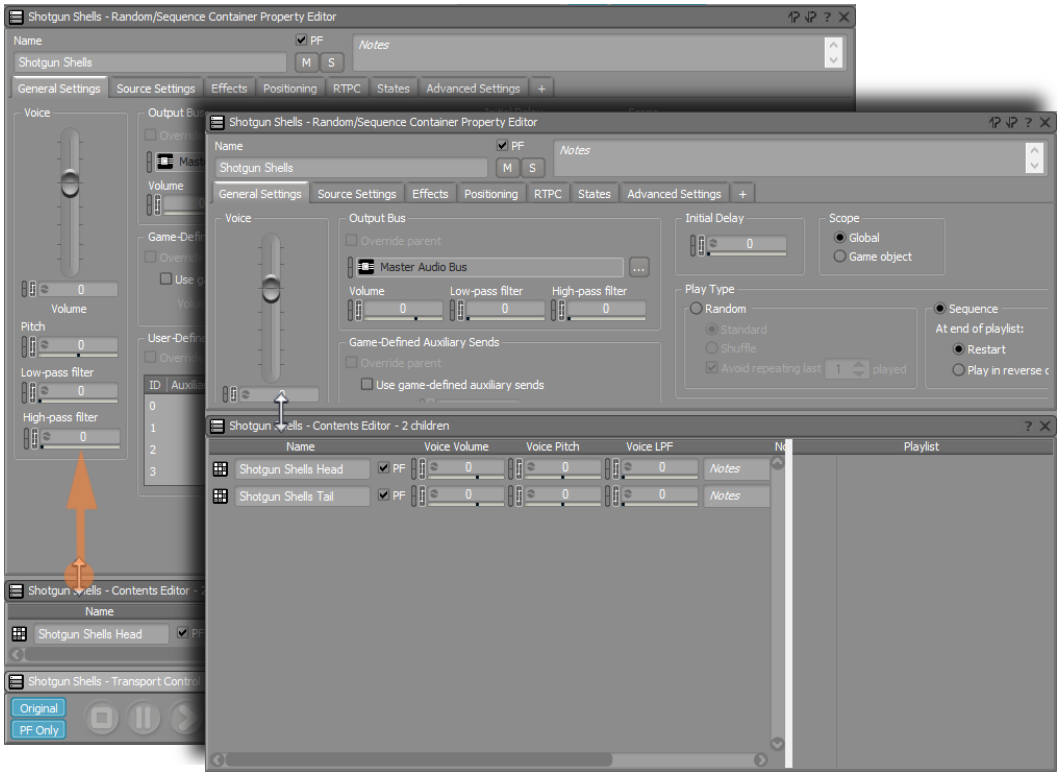


3. 新しく作成したオブジェクトをShotgun Shellsとリネームします。



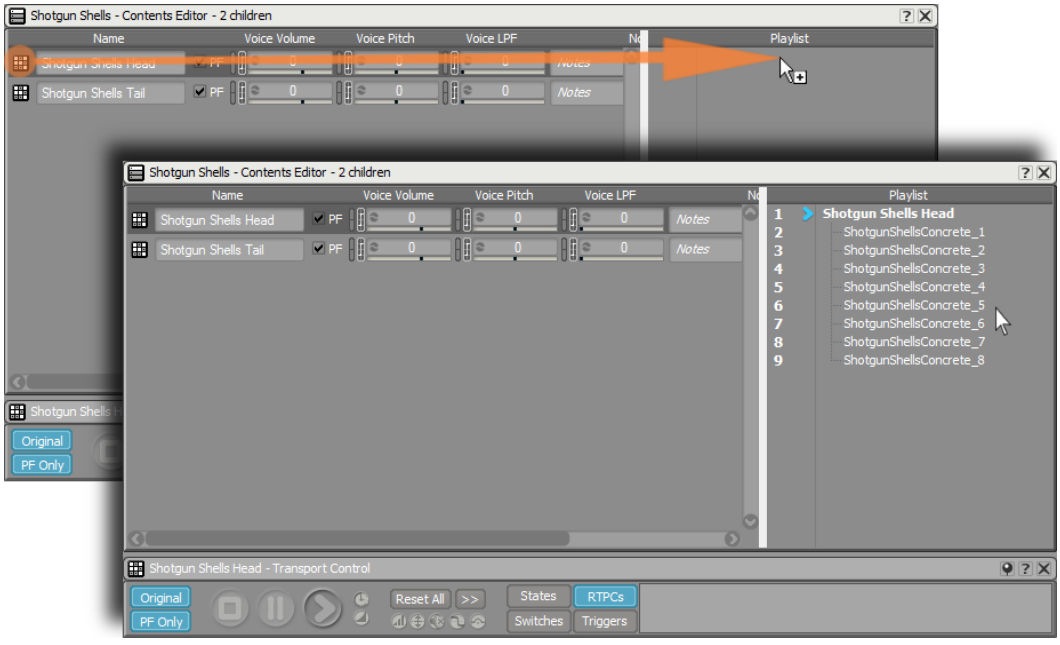
次に、包含されたオブジェクトを再生する順番を指定する必要があります。作業を行うためには、多少広いスペースが必要になるため、Shotgun Shells Contents Editorのエリアをリサイズします。

4. Contents Editorの内容を表示するために、リサイズします。



Contents Editorの左側には、Shotgun Shells Sequence Containerに2つのRandom Containersが表示されています。表示されている順番は、再生される順番を示しているわけではありません。Contents Editorの右側に表示されているPlaylistにより、再生される順番をコントロールします。可能なオブジェクトを左の欄から右のPlaylistにドラッグし、アイテムを追加します。

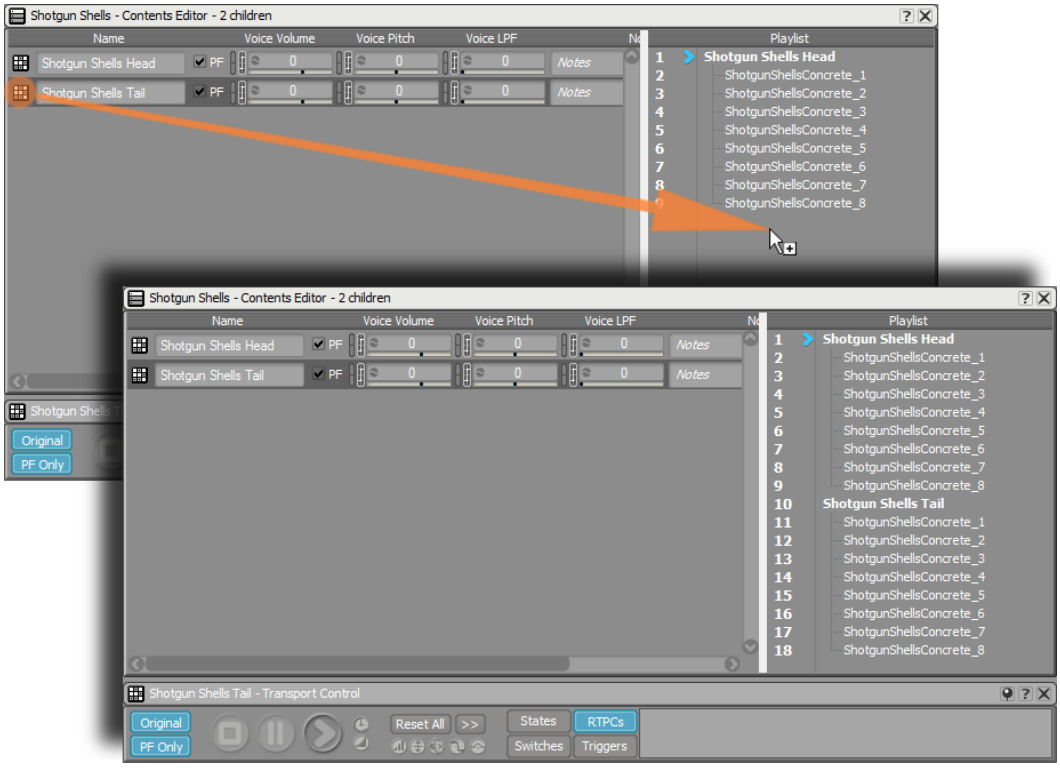
5. Shotgun Shells HeadをPlaylistにドラッグします。



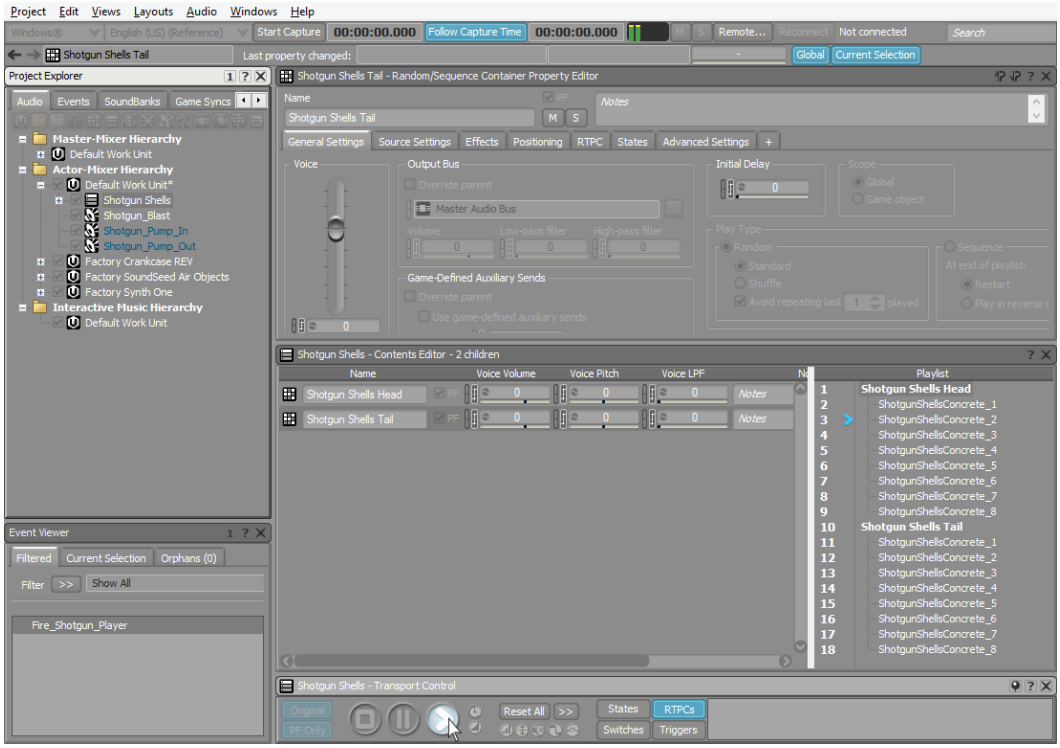
Shotgun Shells Head Random Containerだけではなく、Playlistは自動的に内包されているSound SFX オブジェクトを展開し、内容を見ることができます。

そしてShotgun Shells Tail Random ContainerをPlaylistの最後に追加します。こうすることによりHeadが最初に再生され、そのあとにTailが続きます。

6. Shotgun Shells Tailオブジェクトをplaylistの下半分にドラッグします。

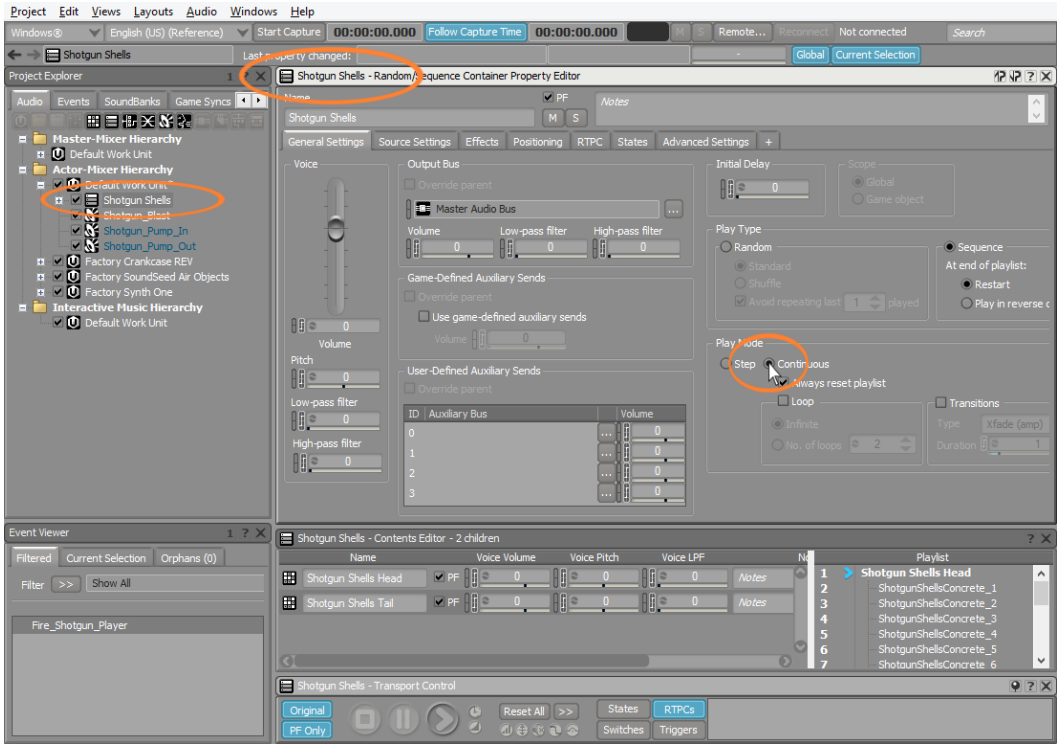


7. Shotgun Shells Sequence Containerをダブルクリックし、オブジェクトを複数回再生します。



Head オブジェクトだけが再生され、次に再生されるまでTailオブジェクトが再生されないことに気が付きます。もし左側が何も変更されない場合、ゲームでも同じことが適用され、プレイヤーがShotgunを発射するとHeadだけが聞こえ、2回目に発射するとTailが聞こえます。これは意図するサウンドのシーケンスではありません。これを直すにはPlayModeのStepをContinuousに変更する必要があります。

8. Shotgun Shells Sequenceオブジェクトが選択されていることを確認し、Continuousラジオボタンをクリックします。



9. Shotgun Shells Sequence Container を複数回再生します。

   ランダムに選択されたHead Shellの後にランダムに選択されたTail Shellが、オブジェクトを再生するたびに聞こえます。

Silenceの使用

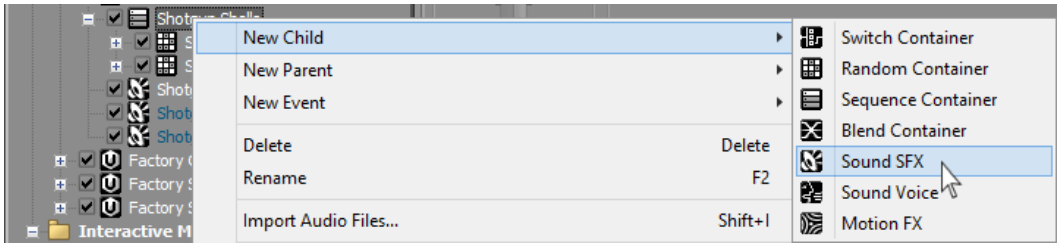
最初のHeadとTailサウンドの間にスペース、もしくはサイレンスがないため、HeadとTail Shell シーケンスサウンドが多少ぎこちなく聞こえます。先ほど行った編集で、Headの後、もしくはTailの前に空白を残しておけば、この問題は起きませんでした。しかしながら、これらを編集するために戻るよりも、さらに柔軟性がある違うアプローチをとることができます。

別オブジェクトを作成します。しかしながら、Shotgun Shells Sequence Containerの全体を通してこのオブジェクトをどのように使用するかオーディション試聴する必要があります。Transport ControlがShotgun Shells オブジェクトを選択し続けるために、Transport Controlのピン機能を使用します。これは、トランスポートを現在選択しているオブジェクトにロックします。

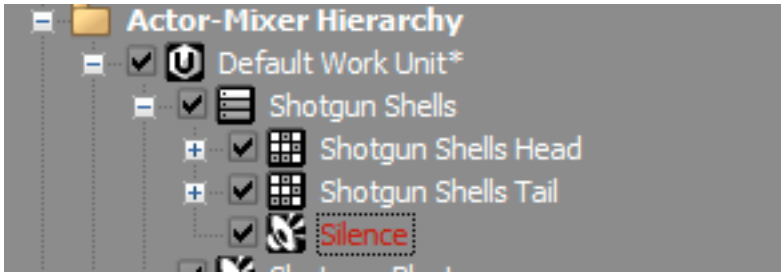
1. トランスポートをShotgun Shells オブジェクトにロックするために**Pin**をクリックします。

Shell HeadとShell Tailにサイレンスを追加するために、サイレンスを作り出すオブジェクトを作成し、Sequence ContainerのPlaylistにあるHeadとTail オブジェクトの間にインサートします。

2. Shotgun Shells Sequence オブジェクトを右クリックし、**New Child** > **Sound SFX**を選択します。



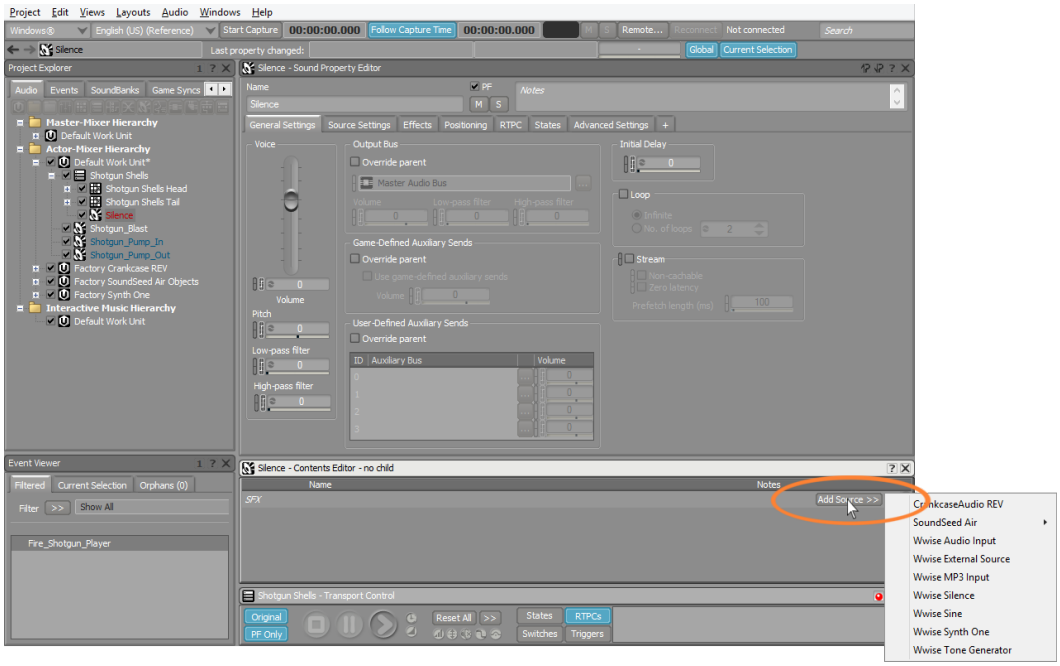
新しいオブジェクトをSilenceと命名します。記載した名前は赤く表記されます。なぜなら、Sound SFX オブジェクトに関連するソース（ファイルやジェネレーター）が存在しないからです。



オブジェクトの名前が赤く表記されているのは、オブジェクトを通しては有効なオーディオソースがないことを意味します。

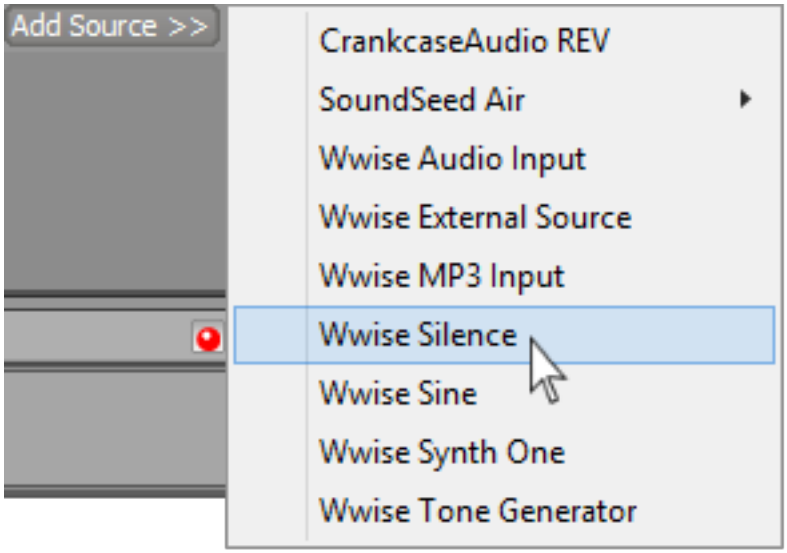
ここまでは、Sound SFXオブジェクトに対してサウンドソースを使用してきました。しかしほかのソースタイプも存在します。それらは、Synth One シンセサイザーと呼ばれ、一からサウンドを生成します。この場合は、Wwise Silence をサウンドソース（もしくは、無音）として選択します。

3. Silence オブジェクトのContents Editorで、**Add Source**をクリックします。

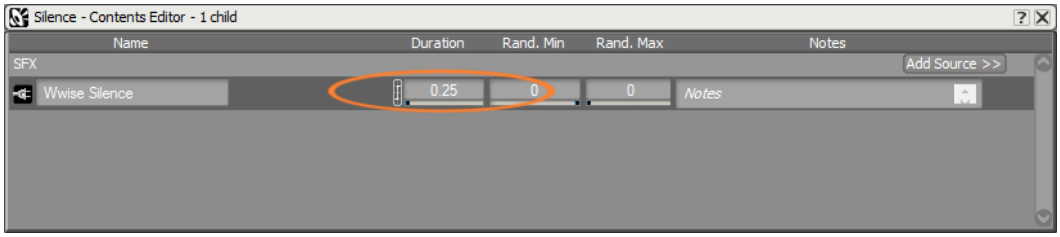


使用可能なオーディオソースのリストが表示されます。

4. Wwise Silenceを選択します。

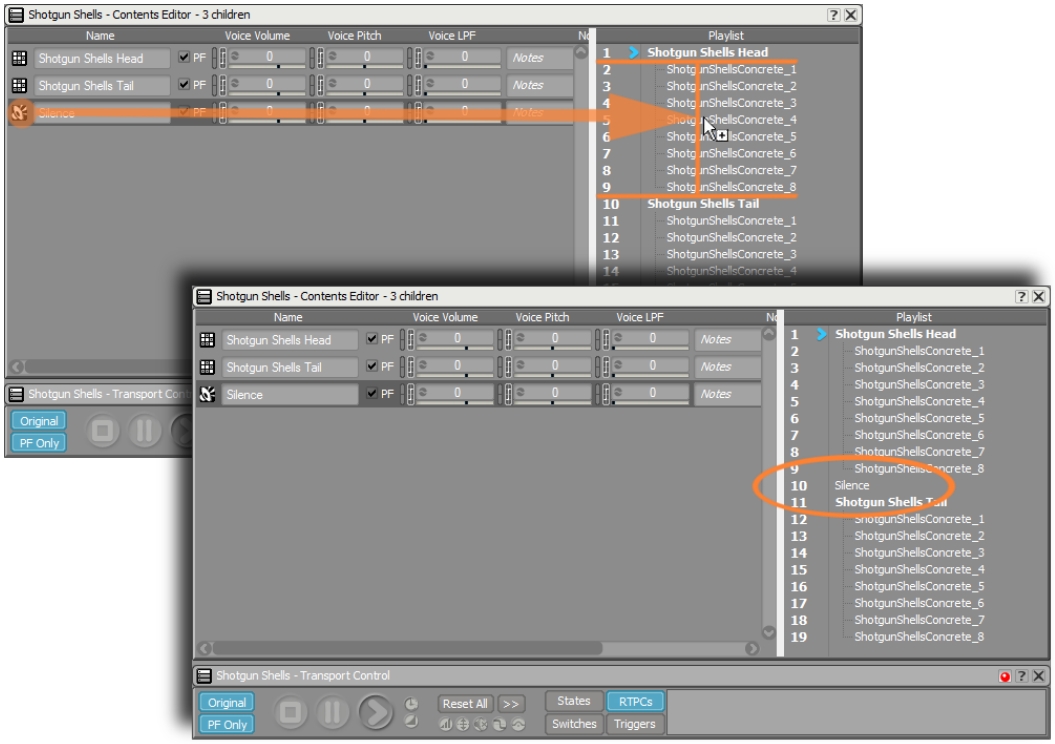


新しいSilence Audio Source オブジェクトがContents Editorに表示され、Durationのパラメーターが表示されます。1秒がDefault値となります。Duration を0.25秒と変更します。



ここでは、Silence オブジェクトをSequence Containerに追加します。そうすることで、PlaylistのShotgun Shell HeadとShotgun Shells Tailオブジェクトの間にSlienceを入れることができます。Container オブジェクトのPlaylistに新しいオブジェクトが追加されます。Shotgun Shells Headの後にSilenceを追加するためには、Silence オブジェクトを2行目から9行目のどこかにドラッグする必要があります。

5. Silence オブジェクトをPlaylistの2行目から9行目の間にドラッグします。



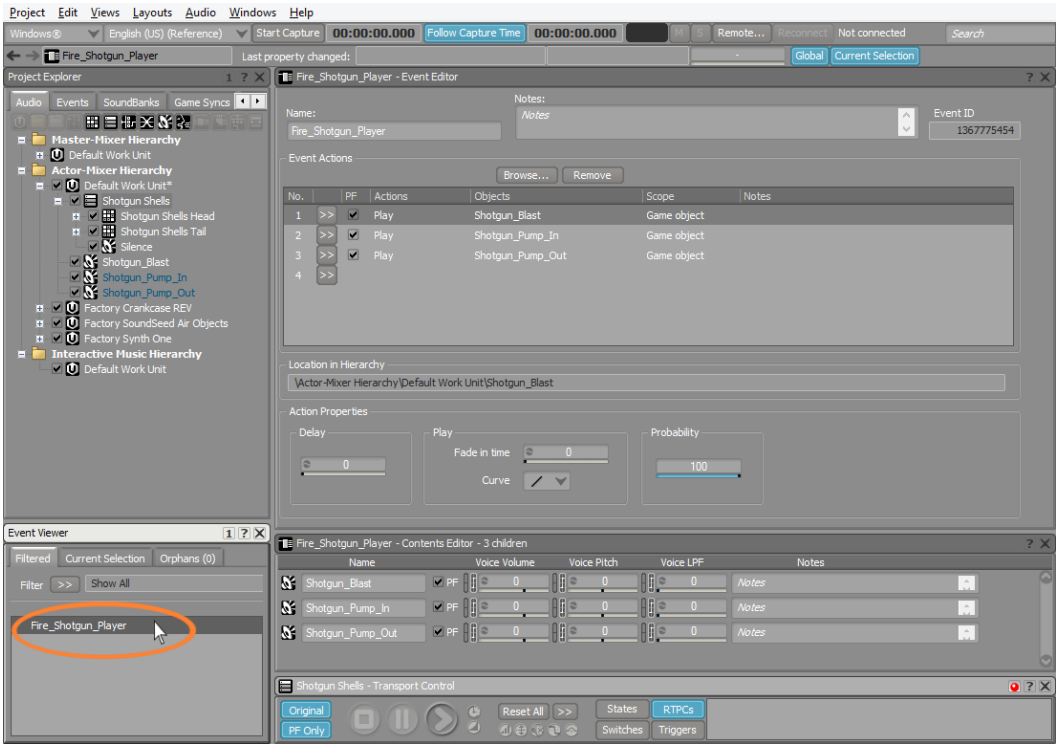
Silence オブジェクトがShell HeadとTail Random Containersの間に位置しているのが確認できます。

ティップ

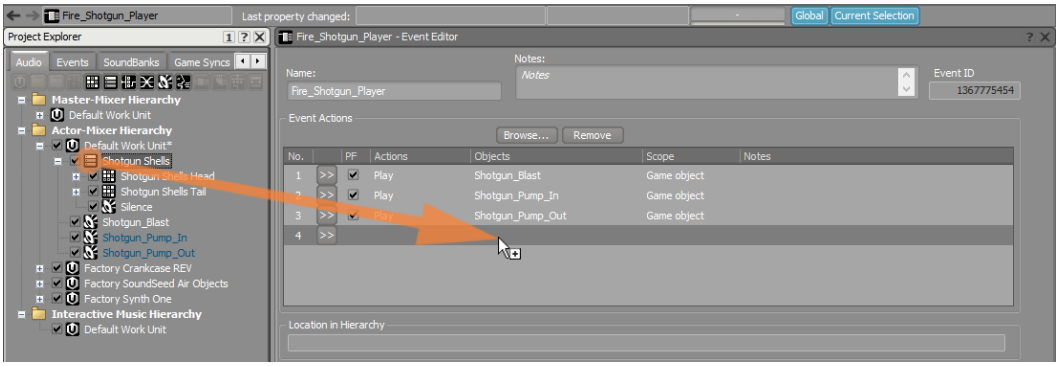
Silence Sound SourceのDuration Propertyを変更することによりSilenceの量（秒）を調整することができます。

ここで、すべてのShells Sequence をFire_Shotgun_Player イベントに追加し、どのように聞こえるか確認します。

6. Event Viewerで、Fire_Shotgun_Playerイベントを選択します。

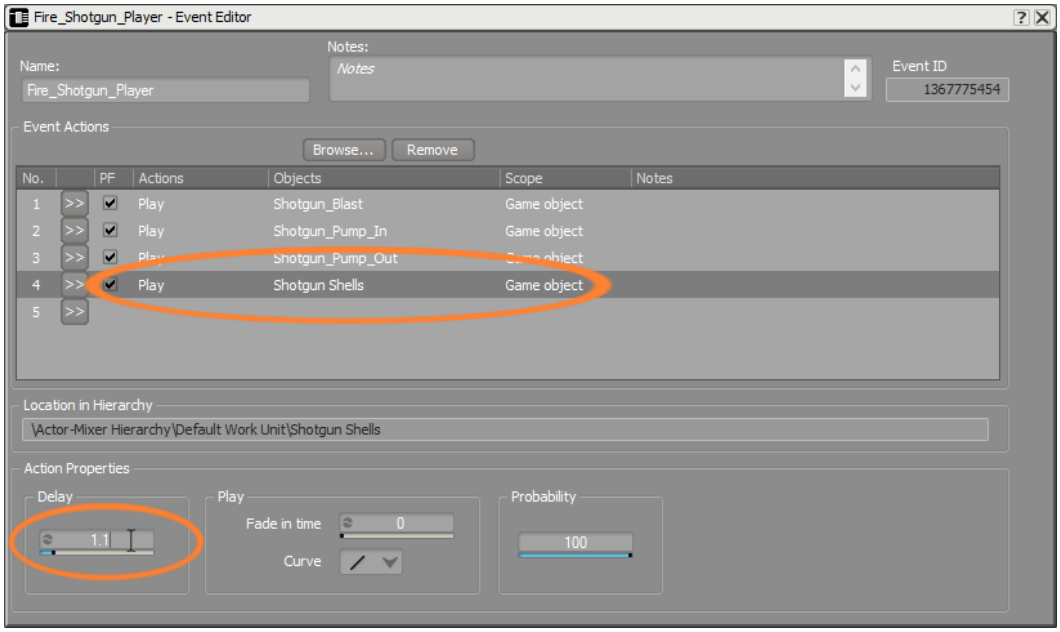


7. Shotgun Shells SequenceオブジェクトをFire_Shotgun_Player Event Editorにドラッグします。



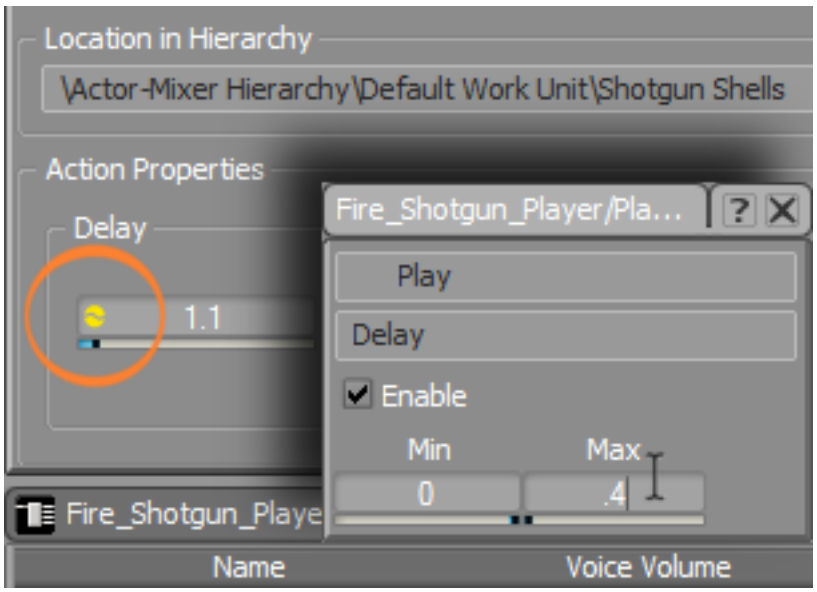
この設定では、地面に落ちるShellsのサウンドがShotgun Pump-inサウンドが聞こえる前に発生します。これは本物のショットガンではありえないことです。これを避けるために、Shellsに遅延を加えます。

8. 新しく作成したActionにDelay parameter1.1秒をセットします。



Shotgun からShell が押し出された時、それらはそれぞれ違う方向で飛び出すため、地面に到達するまでの時間は同じではありません。Delay Propertyをランダマイズすることにより、これをシミュレートすることができます。

9. Randomizer Effectを有効にし、最大値を0.4に設定します。

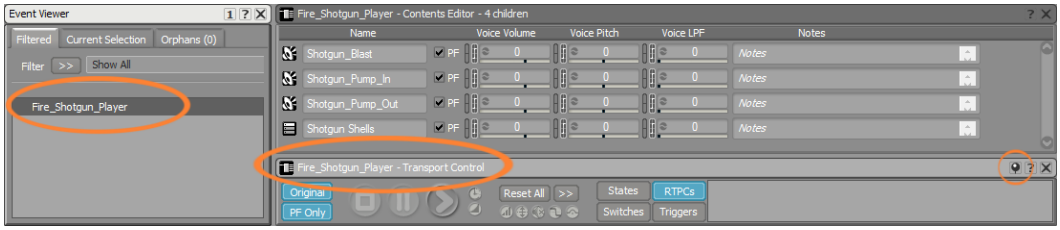


ここでの値はオリジナル値からのオフセットであり、このケースでは、地面にShellが当たるまでに1.1秒から1.5秒までの値をとることを意味します。しかしながら、Minimumオフセットがないため、トリガーをひいた時点から.7秒以内にシェルが地面に落ちることがありません。

先ほど、TransportをShotgun Shells Sequence Containerにピンしました。最初にTransportをFire_Shotgun_Playerイベントにフォーカスするためにピンを

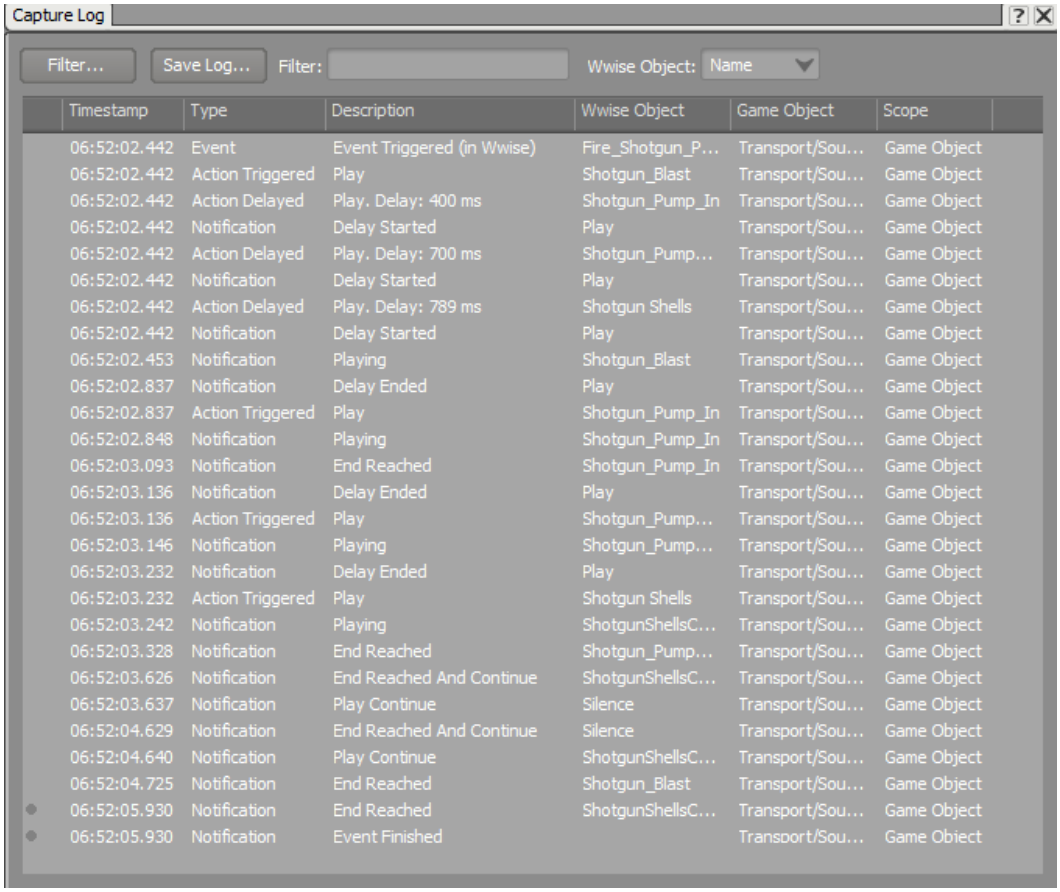
外します。そうすることによりShotgunに関連したサウンドの全体像を聞くことができます。

10.Transport Controlのピンを外し、**Event Viewer**にてFire_Shotgun_Player イベントを選択後、完成したShotgun シーケンスを再生します。



作業の確認

1. Main Menu バーで**Views** > **Capture Log**を選択します。ツールバーで、**Start Capture**をクリックし、Fire_Shotgun_Playerイベントを一度再生し、キャプチャーをストップします。

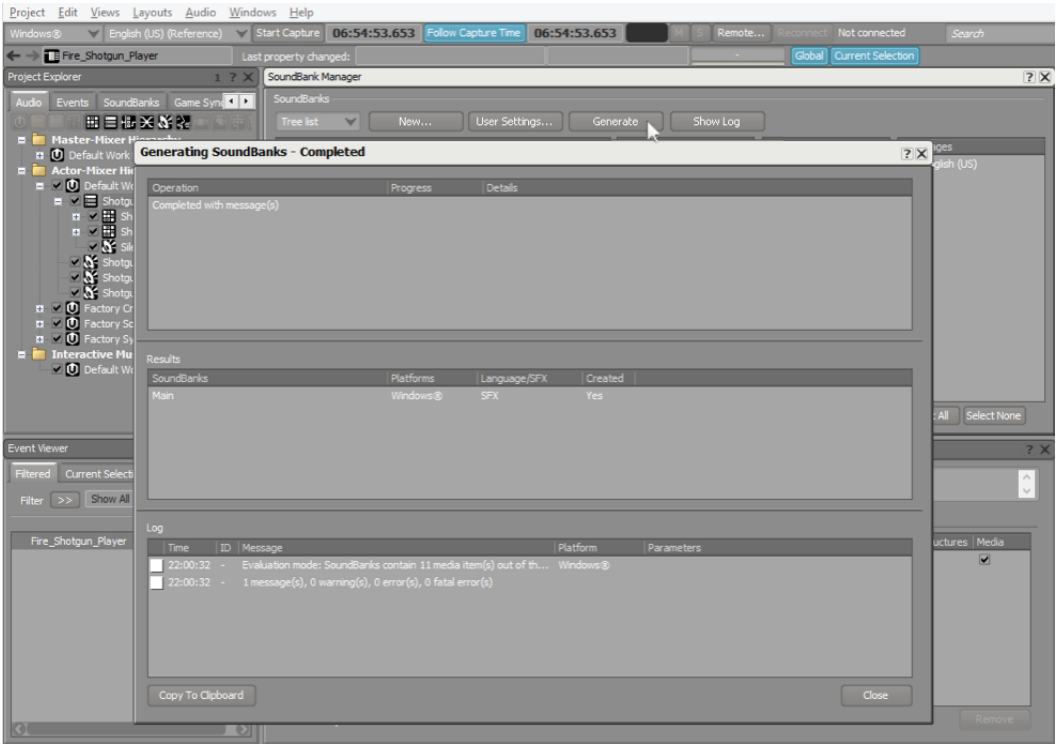


様々な情報が表示されます。一番最初は、あなたが再生を押した時、もしくはゲームプレイ中にプレイヤーがトリガーを引いた時に渡される最初のイベントゲームコールです。その後に、PlayとDelay Actionsが含まれた様々なアクションを見ることができます。通知が、意図通りのアクションが行われた、もしく

は完了したことを知らせます。この複雑なアクション群が、このレッスンで作成したサウンドスケープの結果となります。

サウンドデザインにそったコードをビルドし、サウンドをゲームに渡します。これはボタンを押すことで行うことができます。

2. Capture Logを閉じ、Main Menu Barの**Layouts** > **SoundBank**を選択します。Main SoundBank、システム用のプラットフォーム、そしてEnglishをチェックし、ゲーム用のSoundBanksを作るために**Generate** をクリックします。



3. Cubeデモを起動し、ゲームをプレイして、ゲームプレイ内にあなたがデザインしたサウンドを聞きます。

レッスン 3：Game Syncの理解

スイッチの使用 100
Game Parametersの使用 114
Statesの活用 126
CubeデモへのGame Syncsインテグレーション 132
ProfilerでのGame Syncsの確認 135

前のレッスンで学んだとおり、ゲームコールはゲームエンジンとWwise オーディオエンジン間のメッセージになります。今まではプレイヤーがショットガンを発射したことを知らせるシンプルなイベントゲームコールで作業を行いました。しかしながら、ゲームの特有なコンディションを知らせるために、今まで以上の詳細を知らせる必要があります。例えば、「プレイヤーがどのような地面を歩いているか」や「どれだけのHPが残っているか」、「プレイヤーは生きているか、もしくは死亡しているか」などです。このようなすべてのコンディションが、ゲーム中のプレイヤーがきくサウンドに大きなインパクトを与えます。

Wwiseはイベントタイプゲームコール（前レッスンで使ったFire_Shotgun_Playerイベント）を受け取るためにイベントオブジェクトを使用します。それと同様に、ゲームの様々な状況をWwiseに通知するメッセージを受け取る特別なオブジェクトを作成する必要があります。

Wwiseにおいて、このような状況を定義する作業は、Game Syncsと呼ばれる様々な種類のオブジェクトにより実現することができます。. これらのGame Sync オブジェクトは、プログラマによって組み込まれた特別なゲームコールの受け側になります。

様々なタイプのGame Syncオブジェクトが存在し、それぞれに特別な機能を持っています。下記の演習では、その中の3つを学びます。

スイッチの使用

キューブデモでは、プレイヤーは走り、敵を追いかけ、時には敵から逃げます。多くの1人称視点のゲームでわかるとおり、プレイヤーの足を見ることはありません。これは、プレイヤーの足がないことを意味しているのでしょうか？もちろん違います。プレイヤーがゲーム中に浮遊しているわけではなく、足が地面についていることを表すことができるのは、オーディオの仕事になります。これは、プレイヤーの動きに足音を付けることによって実現させることができます。

そのためには、ゲーム側からWwiseへ、プレイヤーがいつ移動しているかの情報を伝える必要があります。これは、シンプルなイベントゲームコールを使用すること可能ですが、様々な方法があります。一つのアプローチは、プレイヤーの動き初めにコールを送り、そしてプレイヤーが止まるときに別のコールを送ります。Cubeデモにおいては、プレイヤーの一つ一つの足どりは、一つのイベントとして伝えられます。もし、足どりのイベントが伝えられない場合、プレイヤーは動いていません。

Shotgunと同様に、足音を足どりのイベントに関連付けることができます。複数の足音を動きに合わせてランダムで再生することをお勧めします。そうすることで、同じサウンドの繰り返しを防ぎます。前の2つのレッスンで学んだ情報を活用することにより、先ほど説明したすべてを行うことができます。しかし、プレイヤーがどのような地面を歩いているかを考慮しなければなりません。Cubeデモにおいて、プレイヤーはコンクリートやダート、メタルなどの様々な場所を歩きます。プレイヤーがコンクリートの歩道を歩いている時に砂利の足音が聞こえると、没入感がそがれ違和感につながります。

地面の変化に適応させるために、特別なパラメーターを作成し、そのパラメーターでオプションを設定し、その情報を使用しゲームのサウンドを変化させます。

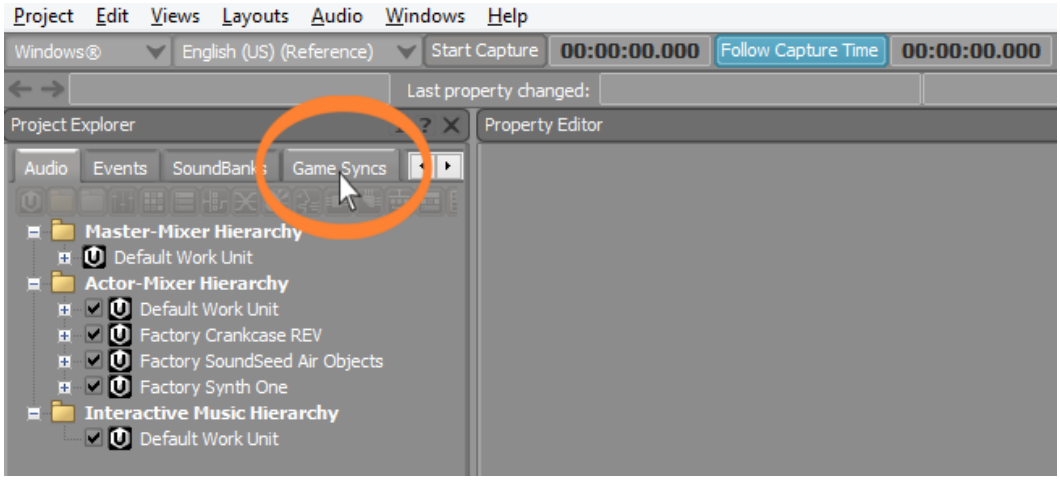
例えば、ゲーム中にプレイヤーが存在し、3つのプリセットボタンがあるラジオがあると想定します。選択されるプリセットによって違うミュージックを再生させます。この設定は3つのプロセスから成り立ちます。最初にSwitch Group Objectのパラメーターを設定します。次にSwitch Objectsのパラメーターのオプションを設定し、最後にどのサウンドをSwitch Container Objectを使用して聞きたいかをアサインします。Switch Group ObjectはRadioと呼ぶことにします。Switch Groupにおいて、Preset 1、Preset 2、Preset 3のSwitch Objectsを3つ作成します。最後に、Switch Containerを使用して、現在選択されているプリセットによってどのオーディオファイルを聞きたいかを決めます。たとえば、Preset 1はロック、Preset 2はジャズ、Preset 3はクラシックを再生するように設定します。

Switchグループの作成

1. レッスン3のWwise プロジェクトを開きます。

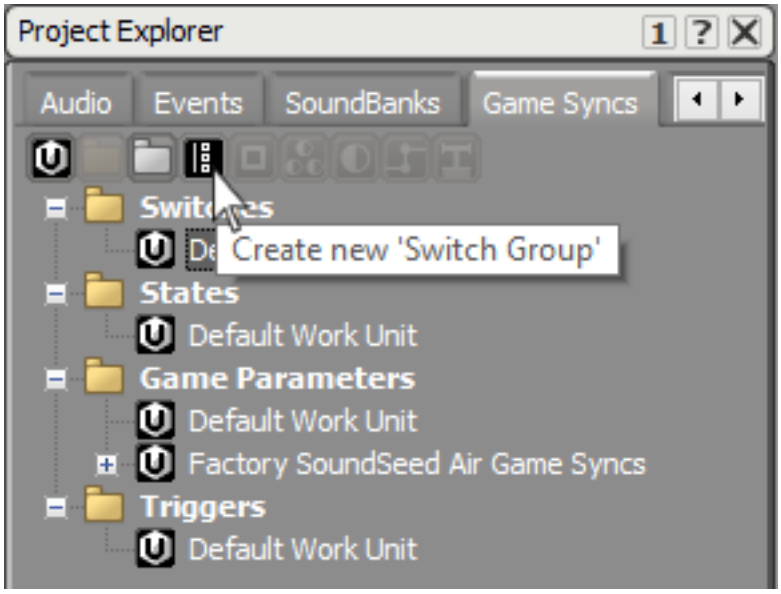
   最初にSwitch Group Objectを作成します。イベントオブジェクトと同様に、これはゲームエンジンによって伝えられる情報を受ける受け側となります。ー

2. Designer Layoutで、プロジェクトエクスプローラー内のGame Syncsタブを選択します。



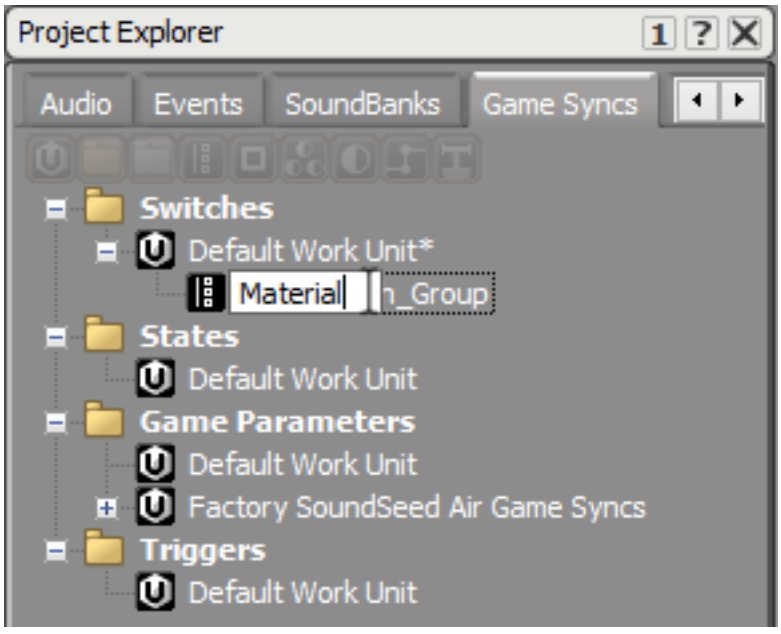
   Wwiseが受け取ることができる特殊なタイプのGame Syncsを表したいくつかのフォルダーがあります。各フォルダーは違う種類のGame Syncsを作り出す機能があり、それぞれに特殊機能が存在し、ゲームプレイでよくみられるコンディションベースのシナリオと情報伝達することができます。プレイヤーがどこの地面を歩いているかを設定する目的では、Switchesが一番適した選択になります。

3. Switchesフォルダ内で、 Default Work Unit を選択し、 **Create New Switch Group** アイコンをクリックします。



Switch Groupと呼ばれる新しいオブジェクトが作成され、それに名前を付けます。イベントと同様に、あなたとゲームエンジンプログラマはSwitch Groupメッセージの名前に関し共通認識を持たなければなりません。あなたが、このように足音に変化をつけようとする一方で、このSwitch groupによって影響を与える他のサウンドが多数存在します。プレイヤーが歩いている地面のマテリアルによって、ショットガンのシェルが地面に落ちる時のサウンドを変化させることもできます。こうした理由により、このオブジェクトをMaterialと命名します。

4. Switchを**Material**とリネームし、エンターを押します。



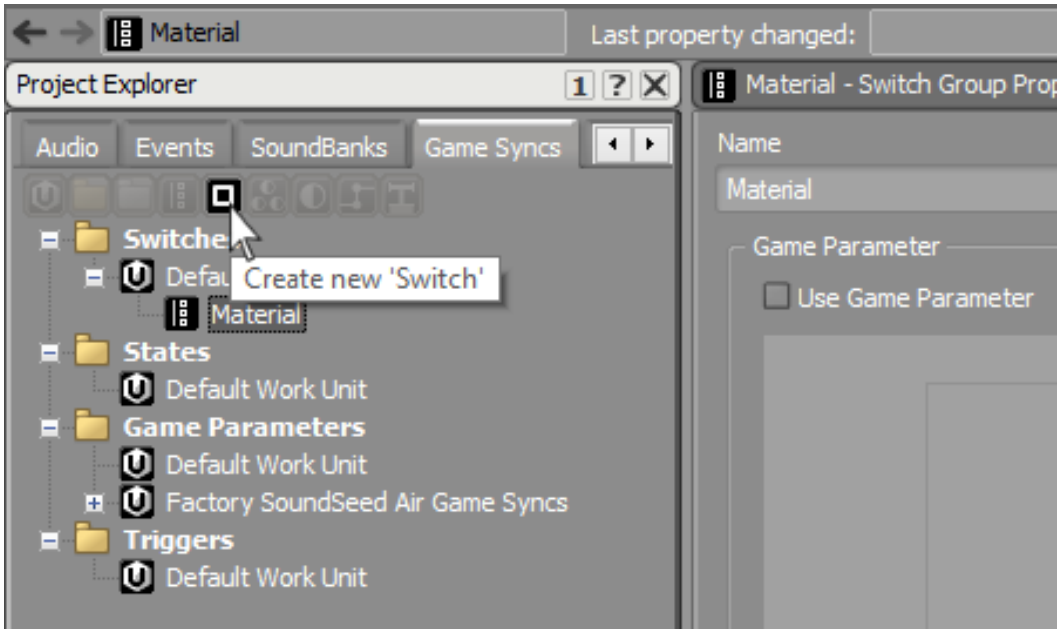
Switch Groupでのオプション設定

Switch Groupを作成し、そのGroupにおけるポジションの可能性について確認しなければなりません。この場合、あなたはゲームプレイ中にプレイヤーが存在しうる全ての地面の種類を把握し、これら地面に対する名称を決めておきます。これらの名前は、先ほど作成したMaterial Switch Groupに含まれるSwitch Objectsとして示されます。Switch Groupと同様に、これらのSwitch名は、プログラマと共通認識を持たなければなりません。そうすることにより、ゲームエンジンはプレイヤーが歩いている地面を的確に扱うことが出来るようになります。

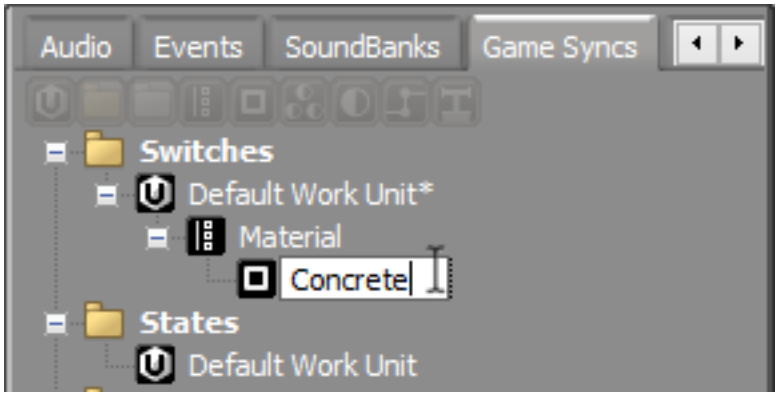
注記

Cubeデモの場合は、ゲーム用のコードがすでに書かれているため、Game Syncsの名前付けを正確に行う必要があり、誤りがあると動作しません。

1. 作成したMaterial Switch Groupオブジェクトを選択し、**Create new Switch**アイコンをクリックします。

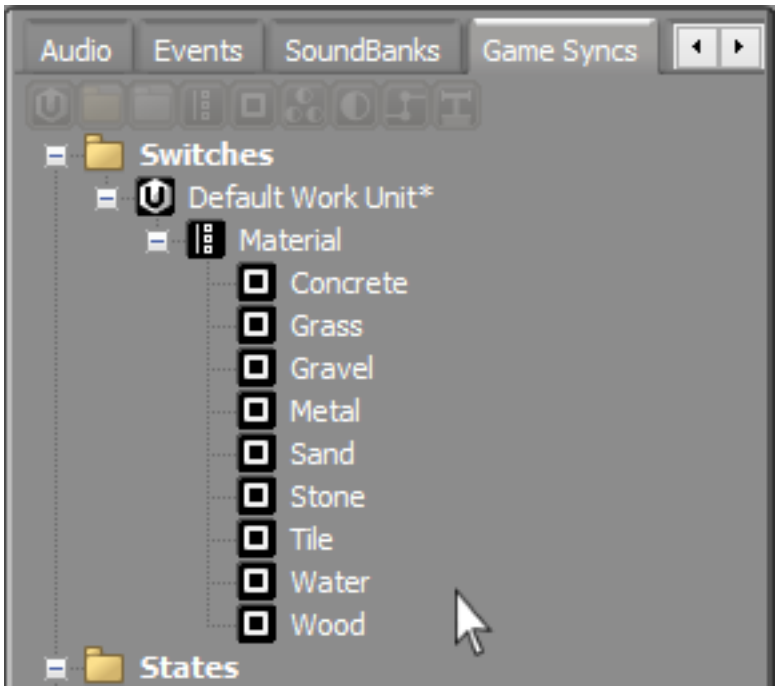


2. Switch オブジェクトを **Concrete**と命名します。



Cubeデモでは、合計で9種類の地面マテリアルがあり、それらをプレイヤーが通過する可能性があります。このSwitch Groupで異なる種類の地面を示さなければなりません。

3. 下記のイメージ通りSwitch オブジェクトを追加し、名前を付けます。



これでSwitch Groupはゲームエンジンから情報を受け取る準備が整いました。

Switch Group、もしくはSwitch Objectをダブルクリックすると、Property Editorが開きますが、ほとんど情報がありません。これは正しいです。これらのオブジェクトの目的は、受け側であり、ゲームエンジンから送られるコールを受け取ることにあります。

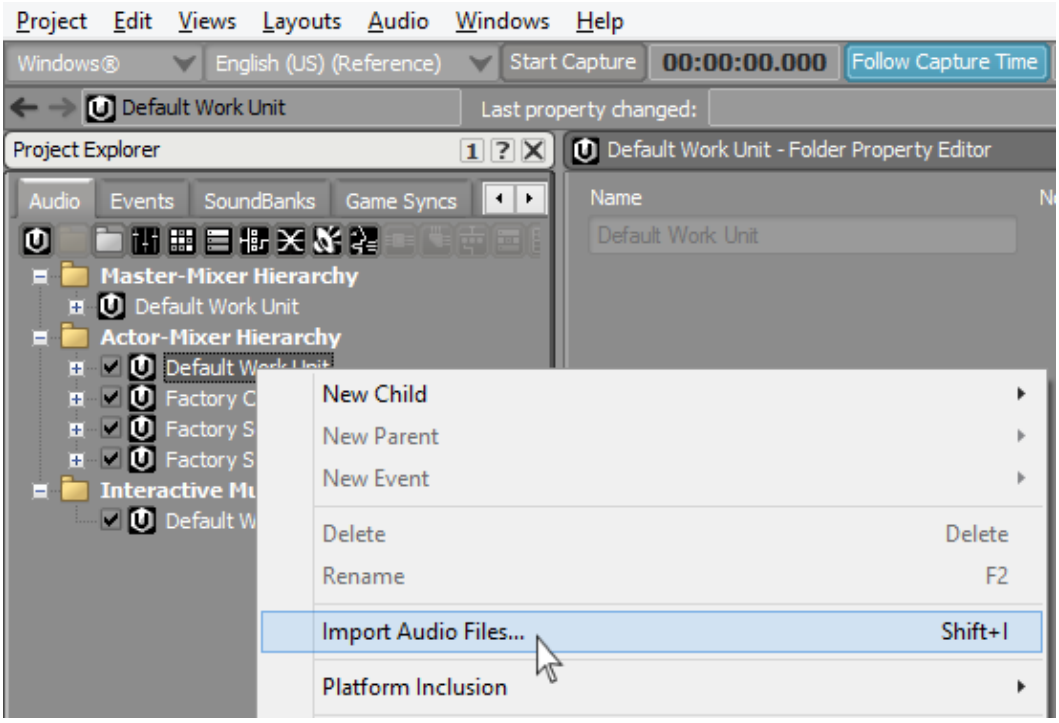
ティップ

Switch Groupを確認すると、Game Parameter グラフにアクセスすることができます。ここではゲームパラメーター（RTPC）を使い、Switch の変更をコントロールすることができます。このレッスン後半にGame Parameter グラフの詳細について学ぶことができます。

Switch Containerの作成

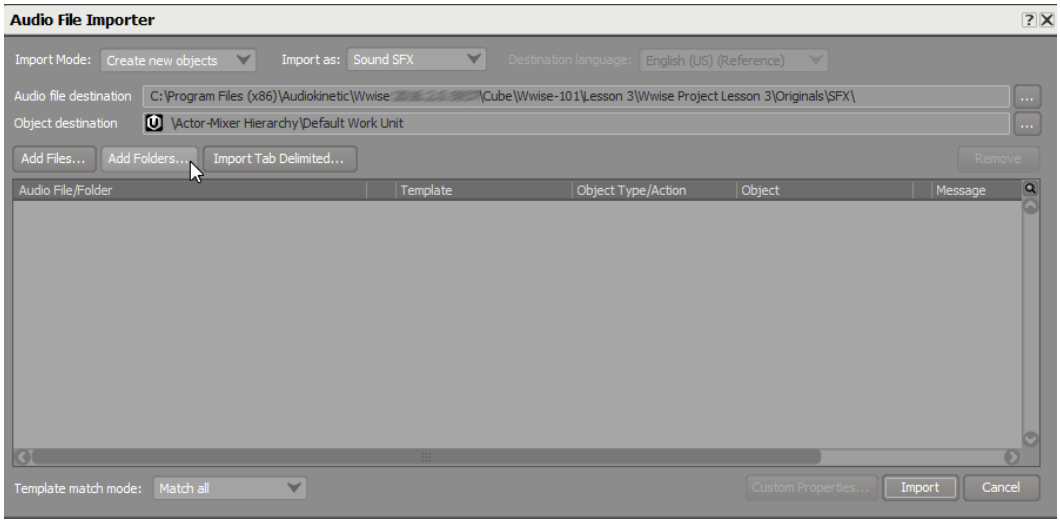
Game Callを受け取るオブジェクトができましたが、これに対しゲームで使用するサウンドを関連付ける必要があります。9種類の地面を作成し、最終的にはそれらに一致するサウンドを関連付けなければなりません。これは、Actor-Mixer HierarchyでSwitch Container オブジェクトを作成することで解決します。

1. プロジェクトエクスプローラーのAudio tabで、Actor-Mixer Hierarchyの Default Work Unitで右クリックし、**Import Audio Files**を選択します。



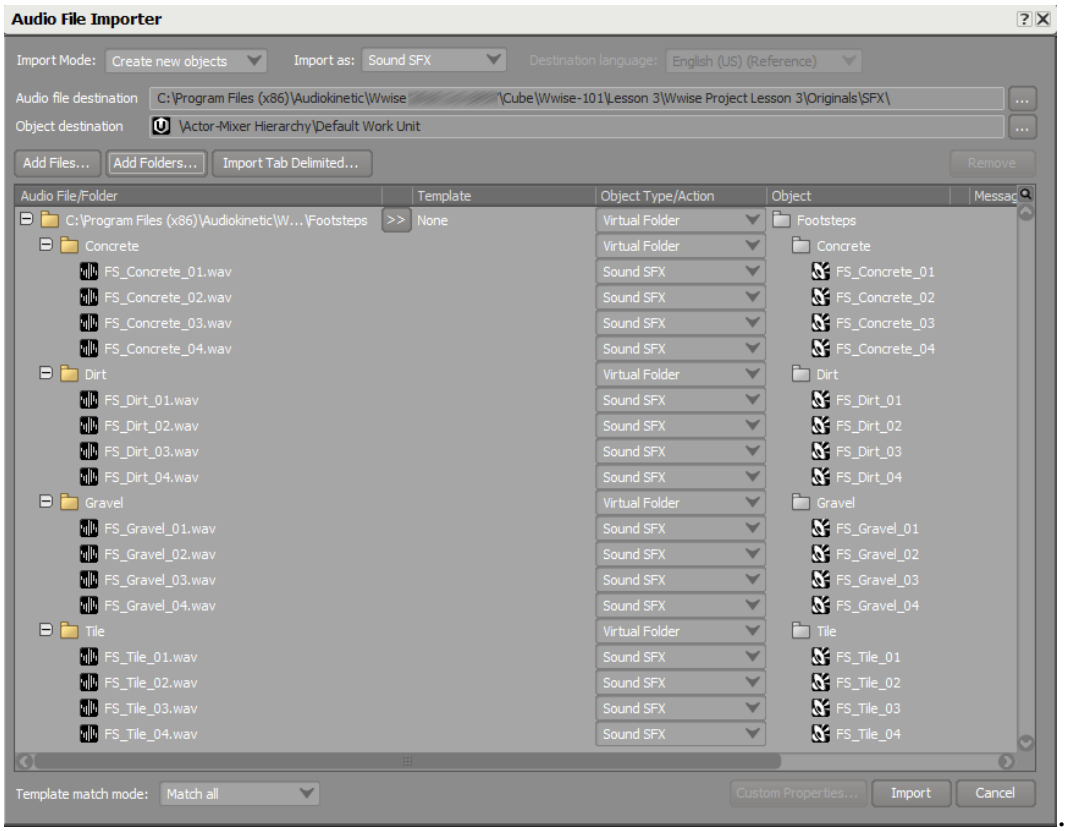
個々のオーディオファイルをインポートする代わりに、フォルダーをインポートします。このように行うメリットは、Wwiseは一つのプロセスで適切なRandom ContainersやSound SFX オブジェクトを効率的に作り出すことができます。

2. **Add Folders**をクリックします。



3. `Wwise-101 > Lesson 3 > Audio files for Lesson 3` へナビゲートし、Footstepsフォルダーを選択、そして**Select Folder**をクリックします。

   Audio File Importerウィンドウが開きます。

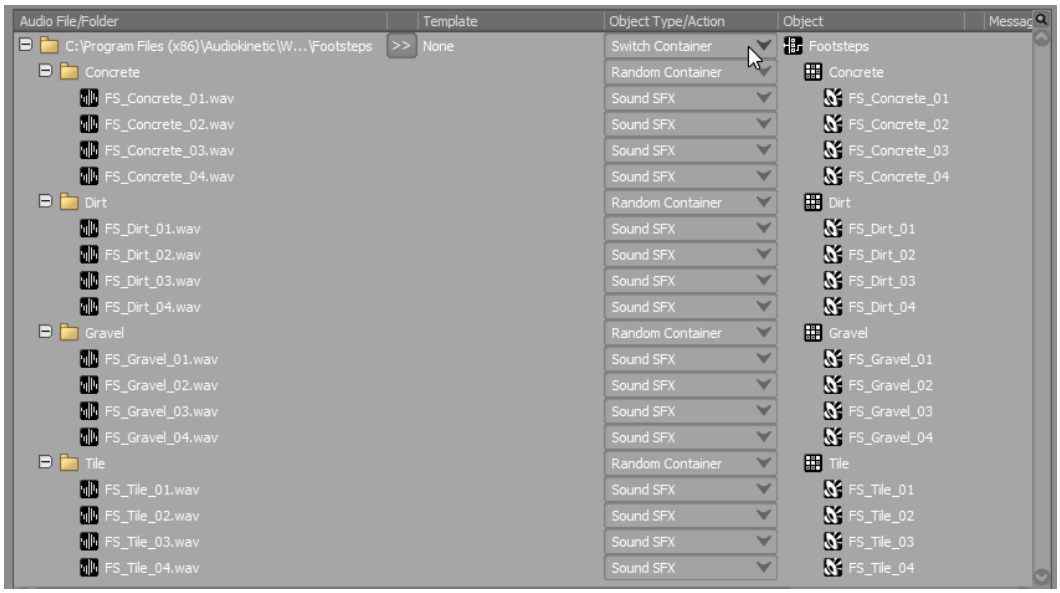


Footstepsフォルダー内に違う種類の地面の足音が含まれた4つのフォルダーがあります。先ほどのエクササイズで設定した9のつの地面を表すフォルダーがないことに違和感を覚えるかもしれません。実際のゲームプロダクションでは、使用したいすべてのサウンドがあるとは限りません。可能な地面の数に対

し、それぞれ違う地面のサウンドを提供する十分なリソースがないかもしれません。次のエクササイズで、同じセットのサウンドを違う地面に適用させる方法を学びます。

Object/Typeの欄を見ると、Wwiseが自動的にそれぞれのフォルダー用にVirtual Folderを作成していることが確認できます。この場合には、Switch Containerを選択します。Switch Containersは、ゲームからの特別なインプットのステータスによってどのサウンドを再生するかを選択することができます。このレッスンの後半でSwitch Containersの詳細を学ぶことができます。さらに各マテリアルフォルダが、インポートした時点でそれぞれRandom Containerとなるようにします。これは一回でまとめて行えます。

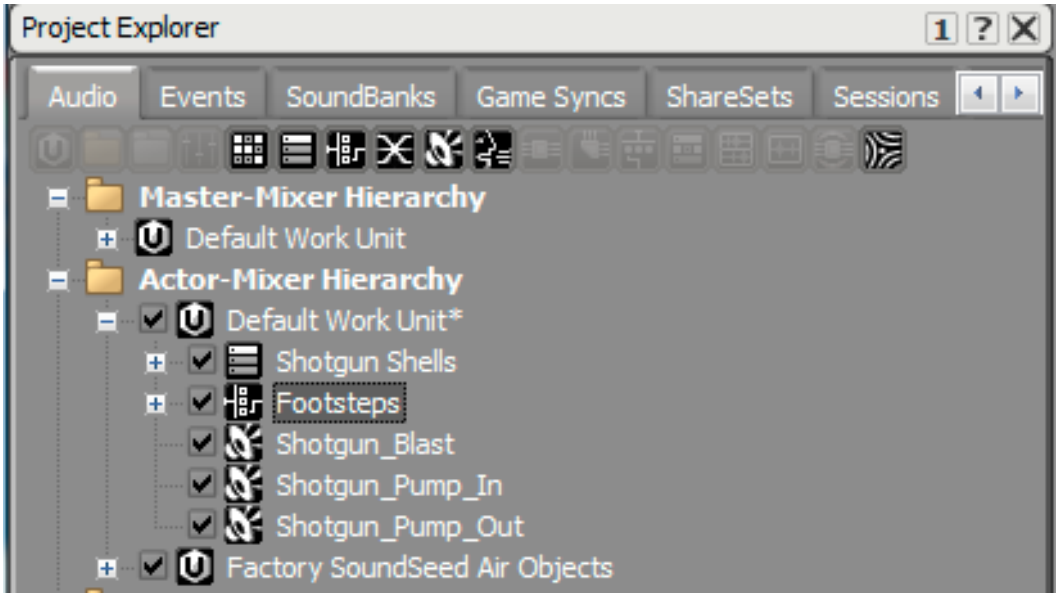
4. Footsteps Object Type/Actionプルダウンメニューを Switch Containerに変更します。



FootstepsをSwitch Containerに変更すると、自動的に中に含まれる各フォルダがRandom Containerとなり、残りのオブジェクトはSound SFXオブジェクトとなります。

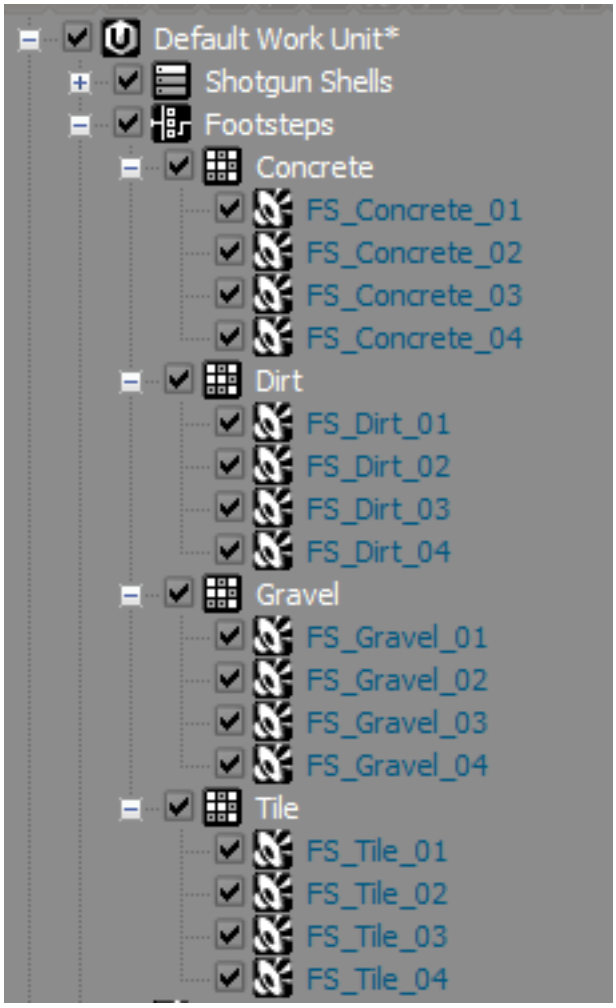
5. **Import**をクリックします。

新しいFootsteps Switch Containerが作成されました。



Footsteps Switch Containerの中には、Sound SFXオブジェクトとして設定しRandom Containersに含めたサウンドファイルとなり、構成済みのフォルダーです。

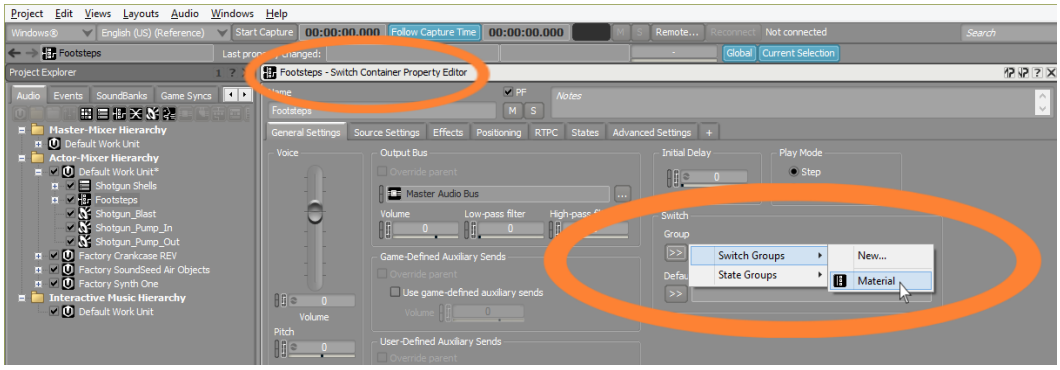
6. Footsteps Hierarchyを開き、フォルダーインポートプロセスの結果を確認します。確認後、先に進む前にFootsteps Switch Containerを閉じます。



SwitchesとSwitch Containersの接続

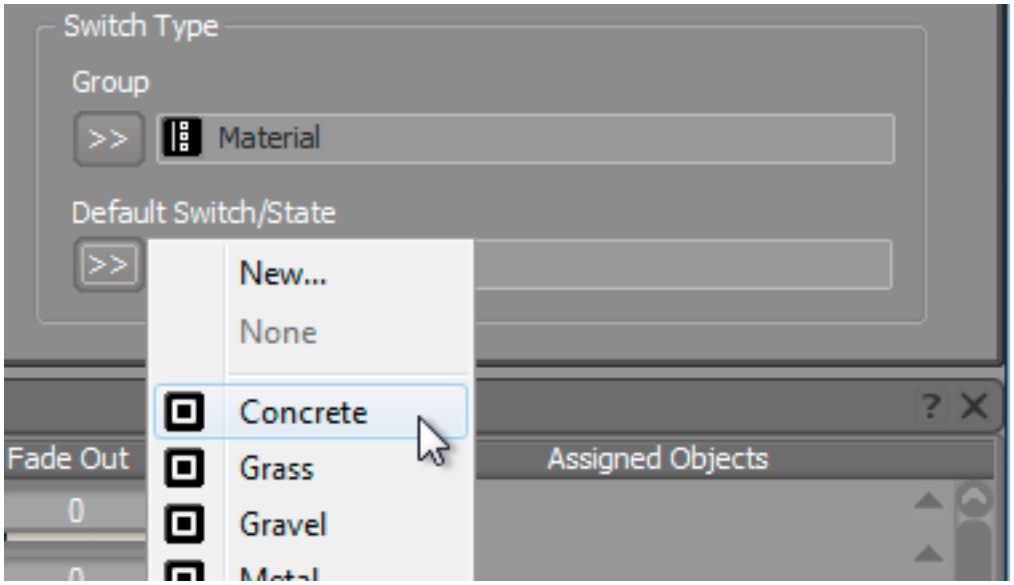
地面マテリアルサウンドの4つのセットをインポートしました。ただし、ゲームでは9つのマテリアルがあります。そのため、プレイヤーが歩くであろう違う種類の地面マテリアルに同じサウンドを適用する必要があります。このアサインは、先ほど作成したSwitch Containerで行うことができます。これは、最初にSwitch Containerと特殊なSwitch Groupを結び合わせることにより実現させることができます。ゲームオーディオ開発を進めるにしたがって、他のアプリケーション用にたくさんのSwitch Groupsを作るかもしれません。しかし、一つのSwitch GroupのみSwitch Containerと関連付けることができます。Switch ContainerのProperty EditorでSwitch GroupをSwitch Containerへアサインします。

1. Footsteps オブジェクトを選択し、Property EditorでGroupボタンをクリックします。そして**Switch Groups** > **Material**を選択します。



これでFootsteps Switch Containerは、プレイヤーが歩く可能性がある地面を認識します。なにかの理由でゲームエンジンが地面情報をオーディオエンジンに伝えられない場合、デフォルトの地面を設定し、これが使われます。ここではConcreteが設定されます。

2. Default Switch/Stateをクリックし、**Concrete**を選択します。

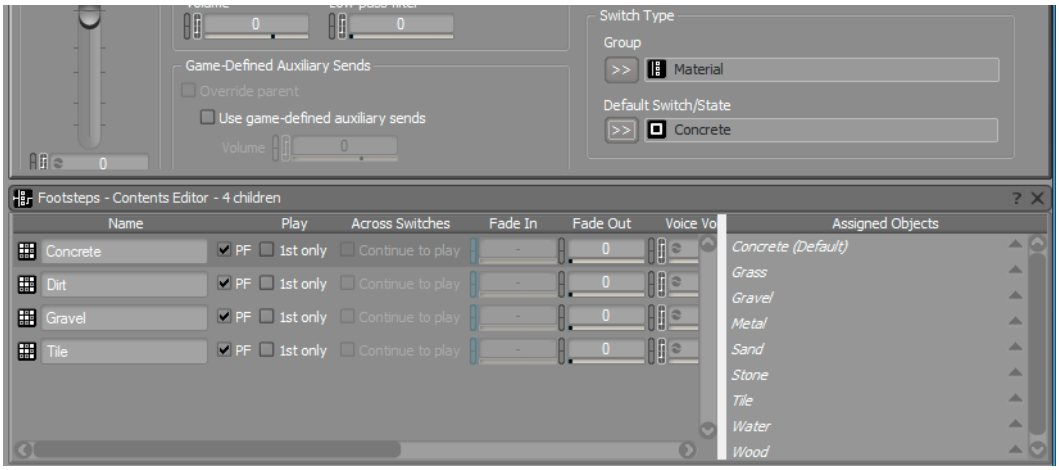


Assigned Objectエリアに先ほどエクササイズで作成したSwitch Objectsが現れます。

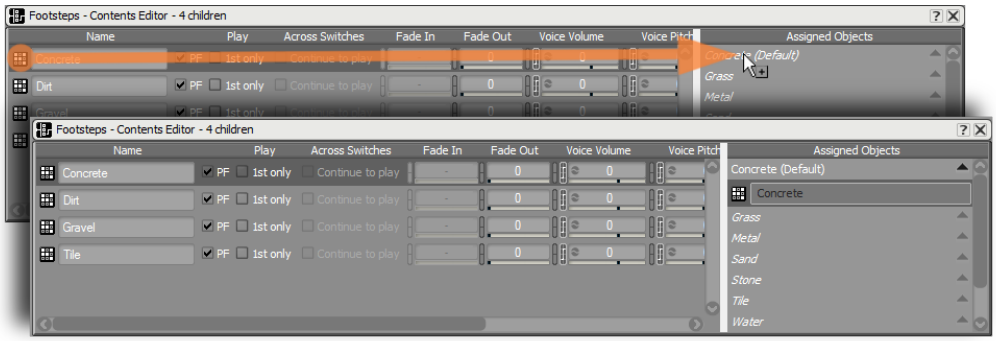
Footsteps Contents Editorでは、左側にインポートしたサウンドのRandom Containersが表示され、右側にプレイヤーが歩く地面を示すSwitch Objectsのリストが表示されます。地面にサウンドをアサインする場合、オブジェクトを左の欄から、右側の目的の地面までドラッグします。

ここで、現在使用可能なサウンドがいくつあり、そしてゲームで発生する地面でどのサウンドが一番適切かを考慮する必要があります。Concreteサウンドはの地面にアサインされ、関連付けは非常に明確です。しかしながら、理想ではないもが存在する可能性があります。例えば、インポートした4つのタイプのサウンドでWaterに関連付けられるサウンドはなんですか？この

時点ではありません。しかしここでPlaceholder（仮音）をアサインします。後に、Waterの足音を録音した際、Waterの地面をアップデートすることができます。このアプローチを適用し、可能性がある地面を事前に構築することにより、後のサウンドを変更したいときにプログラマを困らせることはありません。



3. Footsteps Contents Editorで、Concrete Random Container オブジェクトを Assigned Objects エリアのConcreteスイッチにドラッグします。



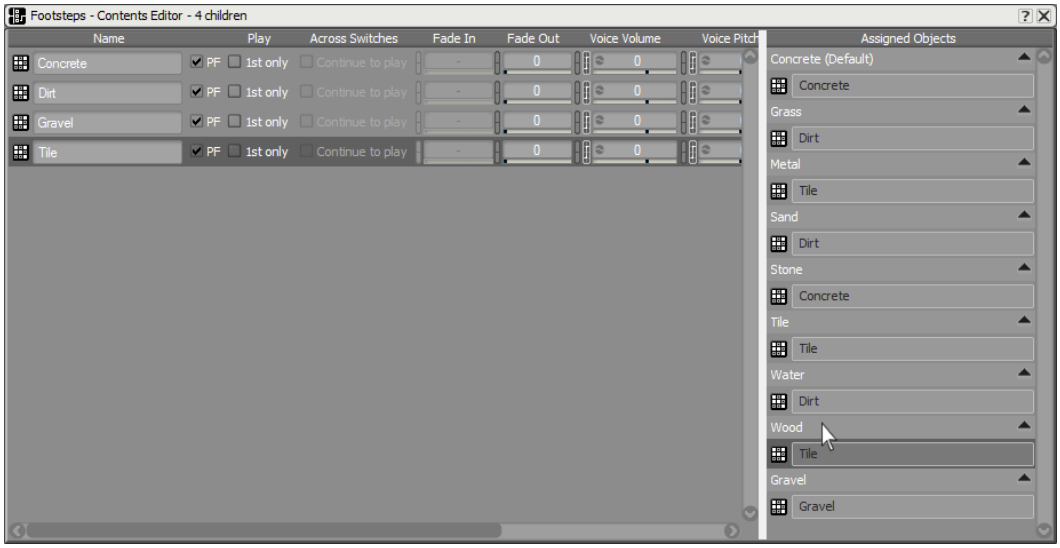
4. Dirt Random ContainerオブジェクトをGrassスイッチにドラッグし、この作業をすべてのスイッチに対し行い、全てのスイッチが関連オブジェクトを持つまで続けます。

ティップ

もし左の欄でRandom Containerを選択した場合、選択されたものはTransportにアサインされ、どの地面にサウンドをアサインするかを決める前にサウンド試聴することができます。

注記

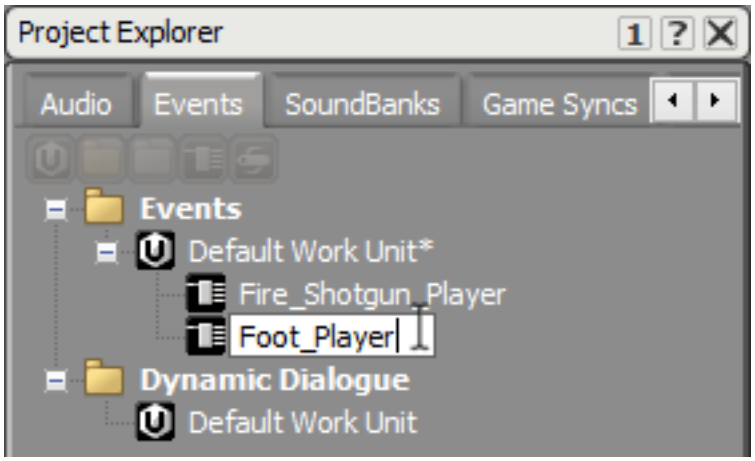
一つ以上のオブジェクトを同じ地面にアサインすることが可能で、その場合、プレイヤーがその地面を歩いた時、2つ同時に再生されます。



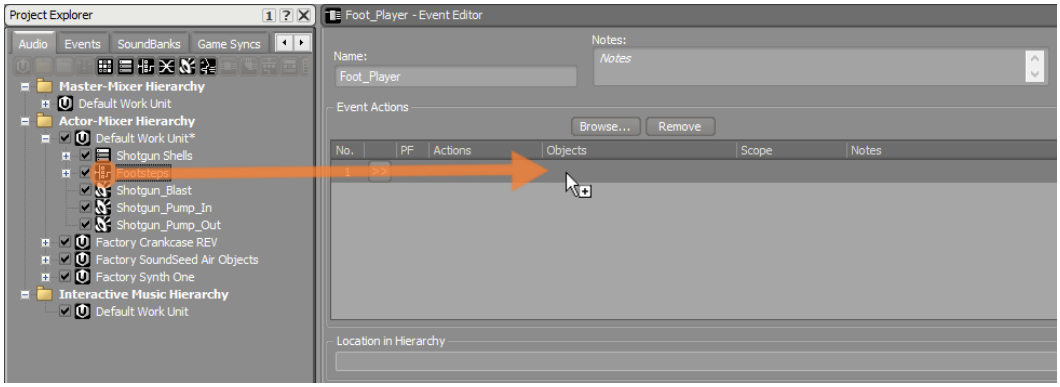
Switch Containerでの地面シミュレーション

Switchは、プレイヤーがどの地面にいるかを示しますが、Footstepは行われたかどうかは示しません。これを伝えるには、Shotgunと同様にEventを作成し、Footstepの情報を受け取らなくてはなりません。

1. プロジェクトエクスプローラーのEventsタブ、そしてEventsのDefault Work Unitをクリックし、Foot_Playerと呼ばれる新しいEventを作成します。



2. Project ExploreのAudioタブより、Footsteps Switch Containerオブジェクトを Event EditorのActionリストにドラッグします。



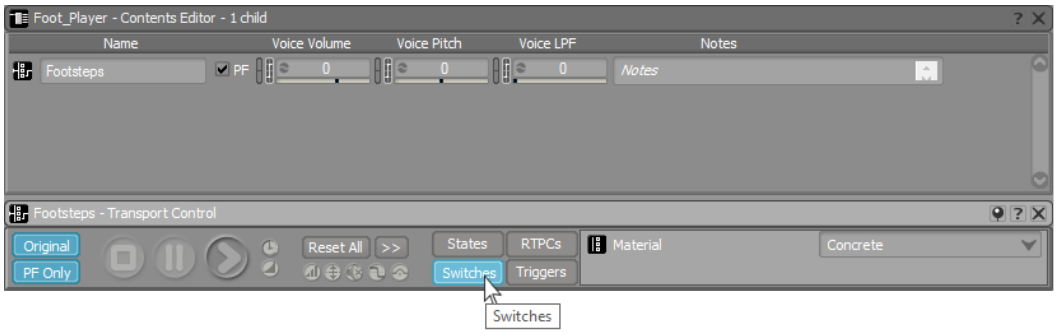
Foot_Player Eventを再生すると、Switch ContainerのFootstepサウンドの一つが再生されます。Switch Containerに何かない限り、設定したConcreteのDefault 地面タイプが使われます。

3. Foot_Playerイベントを再生します。

   ランダムに選択されたConcreteのFootstepサウンドの一つが聞こえます。

   ほかの地面のFootstepsサウンドを視聴し、地面のスイッチングシステムが正しく動作しているかをテストします。これは、Transport Controlビューにある Switches ボタンを使います。

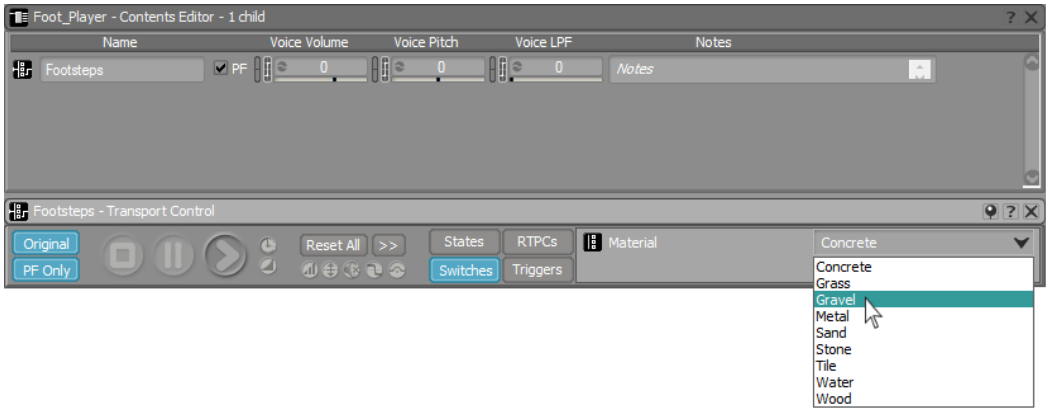
4. Transport ControlビューのSwitchesをクリックします。



イベントに関連付けされているすべてのSwitch ContainersがTransport Controlの右側に表示されます。

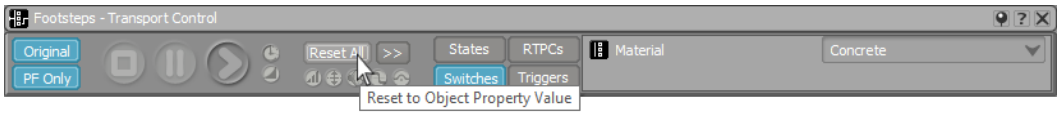
そして、プルダウンメニューが表示され、特定の地面を選択し、ゲームにおける地面の移り変わりを効果的にシミュレートすることができます。

5. プルダウンメニューでGravelを選択し、同じFoot_Playerイベントを再生します。そして違う地面もシミュレートします。



TransportビューのReset AllをクリックすることによりDefault の地面に戻ることができます。このボタンは、すべてのSwitches、もしくは次のレッスンで作成する他のGames Syncsをデフォルト値に戻すことができます。

6. Transport Controlの **Reset All**をクリックします。



Game Parametersの使用

他の重要なGame Syncタイプとして、Game Parameterが存在し、RTPC（Real Time Parameter Control）と呼ばれます。Game Parametersはゲームプレイ中、数値として変化するどのような情報に対しても有効です。Game Parametersは、車のスピードやエンジン回転数（RPM）を示すこともありますし、ミッションの達成率や、時刻としても活用することも可能です。

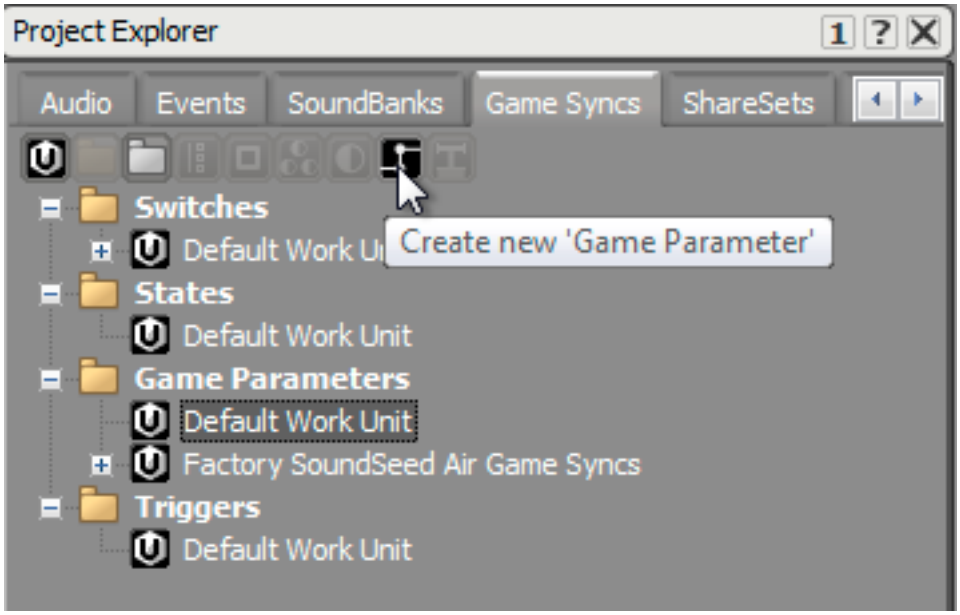
Cubeデモでは、重要な変数としてプレイヤーのHPがあります。プレイヤーはHP値が100で始まり、敵の攻撃を受けHP値が0になるとプレイヤーが死亡します。

ゲームのプレイ中に画面下部でHP値を見ることができます。しかしながら、戦闘中はHPメーターに気を配るのは難しいことです。プレイヤーのHP値に関連したオーディオによるフィードバックはとても役に立ちます。ハートビートが大きくなり気が付く程になればなるほど、プレイヤーの状態が悪化しています。

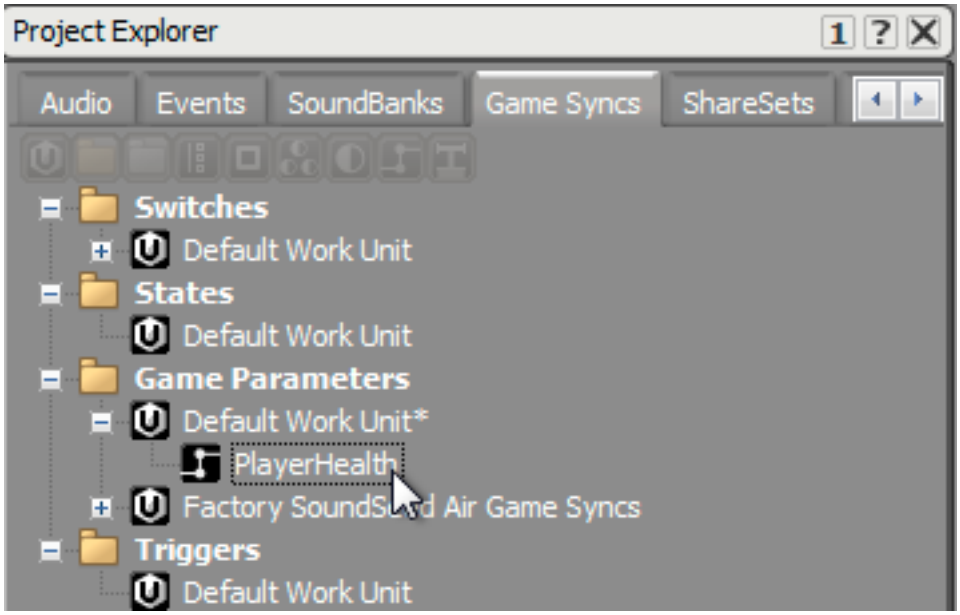
Game Parameterの作成

前の演習のSwitch Containerオブジェクトのように、Game Parameterは特化したGame Callsを受け取るために設置されたオブジェクトです。

1. プロジェクトエクスプローラーのGame Syncsタブを選択し、Game ParametersフォルダーのDefault Work Unitを選択します。そして **Create New Game Parameter**をクリックします。



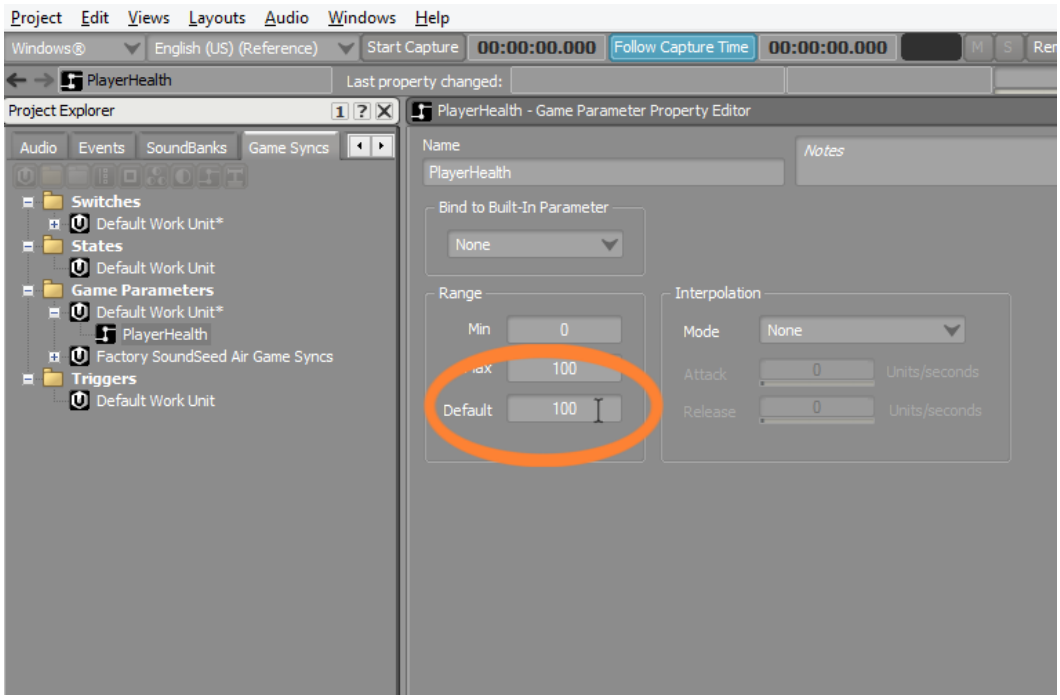
2. 新しく作成したGame ParameterオブジェクトをPlayerHealthと命名します。



Game Parametersは設定された範囲の数値を伝えます。この数的な範囲においては、様々な値をとることができます。例えば、車のスピードのシナリオを使うと、範囲は0から300になり、これはkm/時を意味します。

Cubeデモの場合では、Healthパラメーターは0から100までとなり、これはデフォルト値の範囲であるためこれらのPropertiesを変更する必要はありません。ここで必要なのは、デフォルト値を決めることです。そうすることで、ゲームが現在の値を伝えなかった時に、Wwiseはどの値からスタートすべきかを知ることができます。. プレイヤーはHPがフルの状態から始まるため、デフォルト値は100になります。

3. PlayerHealth Game Parameterのデフォルト値を100に変更します。

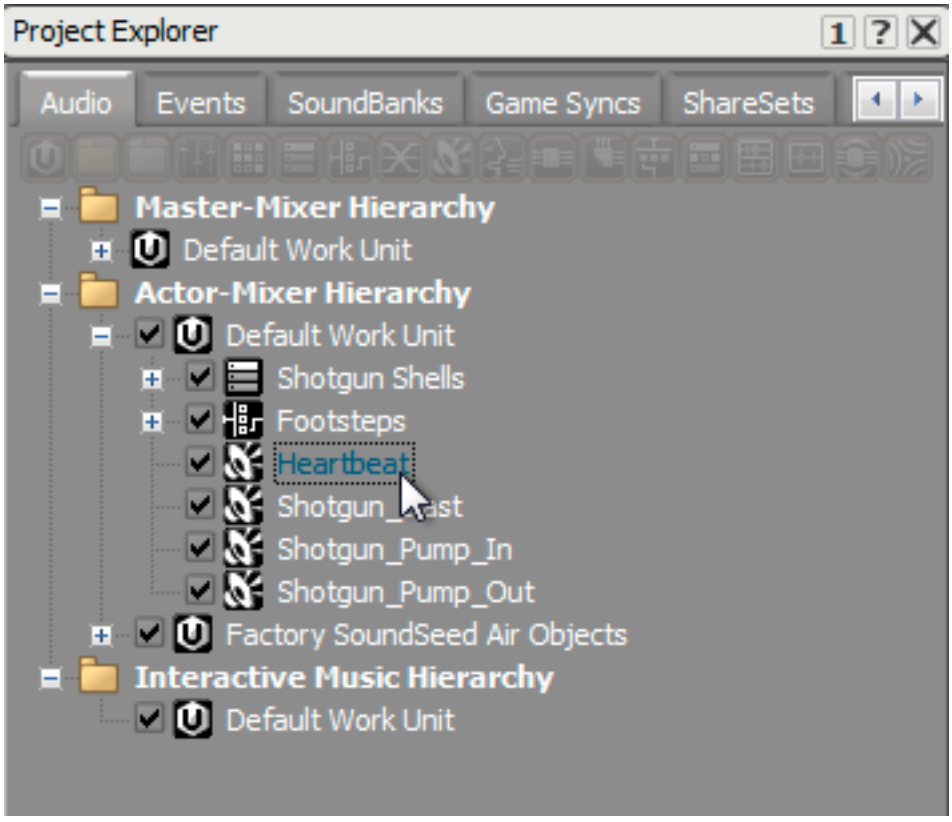


Game ParametersでObject Propertiesを変更

Heartbeatで使用するオーディオファイルをインポートします。

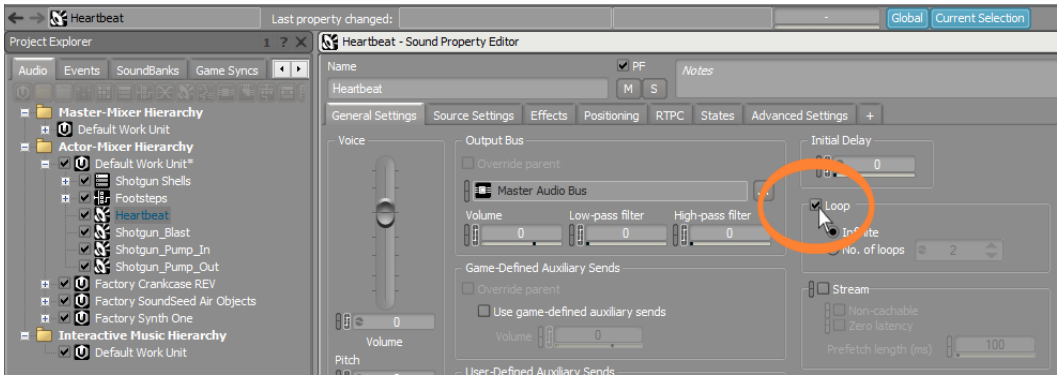
1. Audioタブをクリックし、Audio Files for Lesson 3フォルダーのHeartbeatオーディオファイルを**Actor-Mixer Hierarchy**のDefault Work Unitにドラッグします。

   Heartbeat（心拍音）と呼ばれる新しいSound SFXオブジェクトが作成されます。



このオーディオファイルは、約30秒ほど続きます。プレイヤーが新しいレベルを始めた時に、一度だけ1つのEventを使用してHeartbeat音をトリガーします。プレイヤーのHeartbeat音が短かく終わってしまってはいけないので、このファイルを無限にループ再生させる必要があります。これは、Sound Property Editorで有効にすることができます。

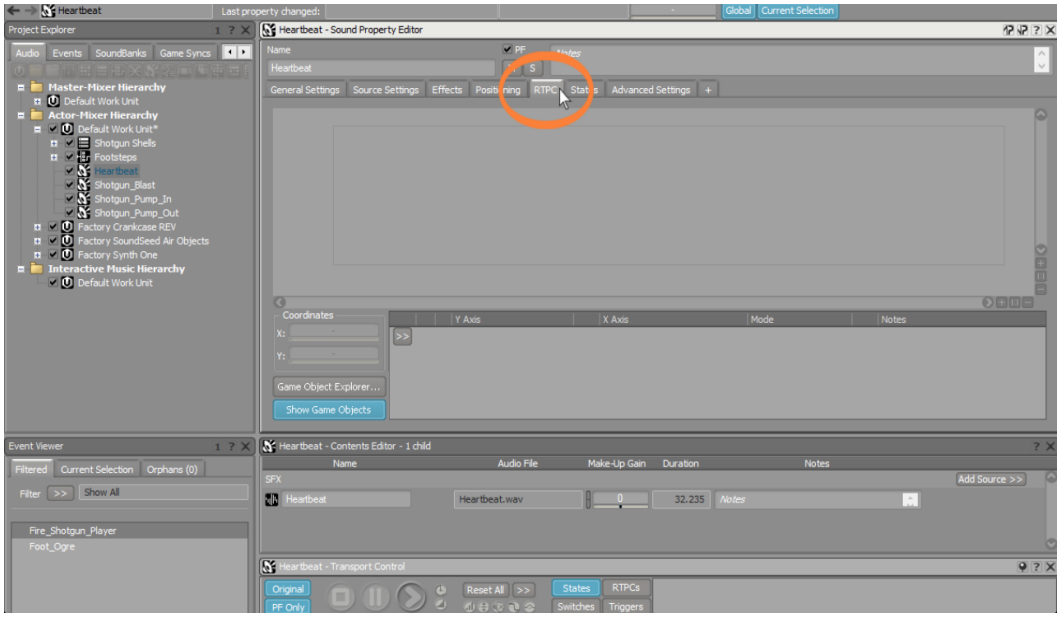
2. Heartbeat Sound SFX オブジェクトを選択し、 **Loop**のチェックボックスをクリックします。



Heartbeat音は、事実上プレイヤーがゲーム中に存在する限り鳴り続けます。これを継続して聞き続けると大変不快になります。ここで思い出していただきたいこととして、Heartbeat音の目的はプレイヤーにHPの危険を知らせるためです。ある一定のレベル以下にHPが到達した時のみ、Heartbeat音が聞こ

えるようにします。要するに、PlayerHealth Game Parameterの値を使用して、Heartbeat Sound SFX オブジェクトのボリュームを変化させます。Sound Property EidtorのTRPCタブでこれを設定します。

3. Sound Property Editorで、RTPCタブをクリックします。



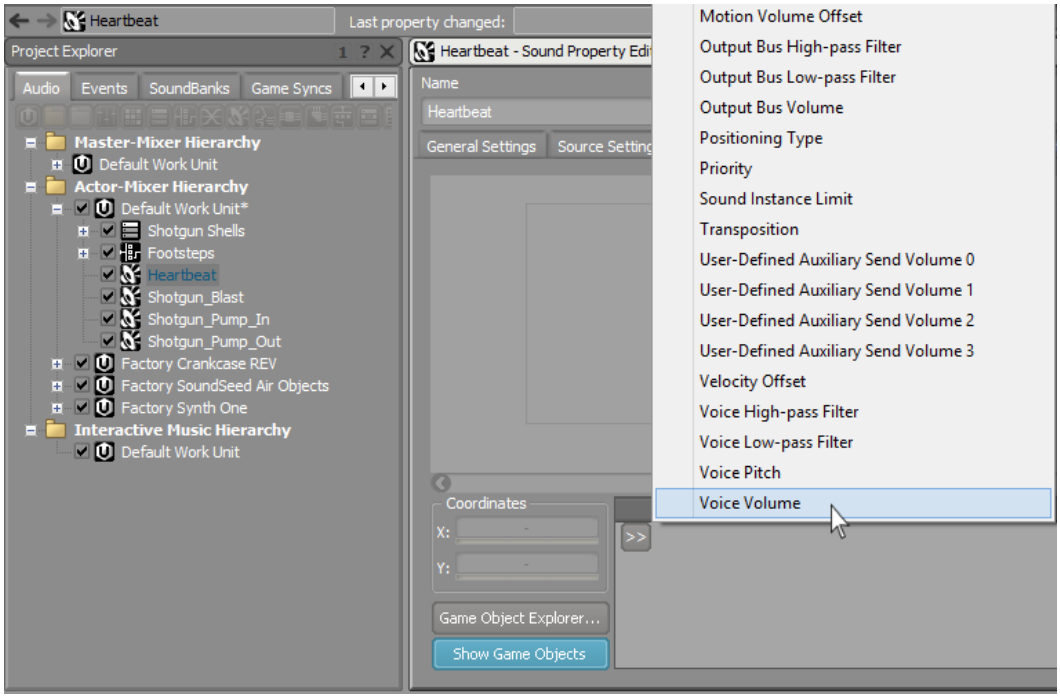
Property EditorのRTPCタブが表示されます。

注記

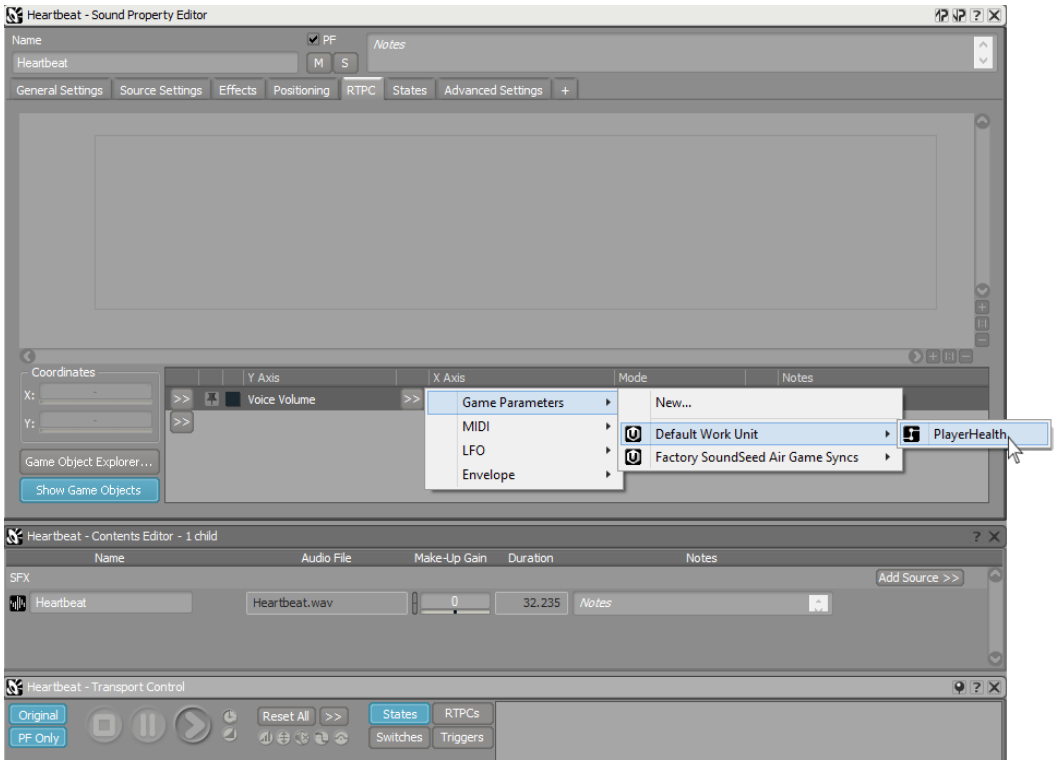
必要であれば、Sound Property Editorの下部分をドラッグし、ディスプレイ上でGraph Viewが見やすくなるように調節します。

Game Parameterと影響を与えるPropertiesを設定する場所は、Property EditorのGraph Viewになります。。現在、Graph Viewには何も情報がありません。なぜならGame Parameterのインプットで変更したいObjectのPropertyを示していないためです。

4. **[>>] Selector** メニューボタンをクリックし、Voice Volumeを選択します。

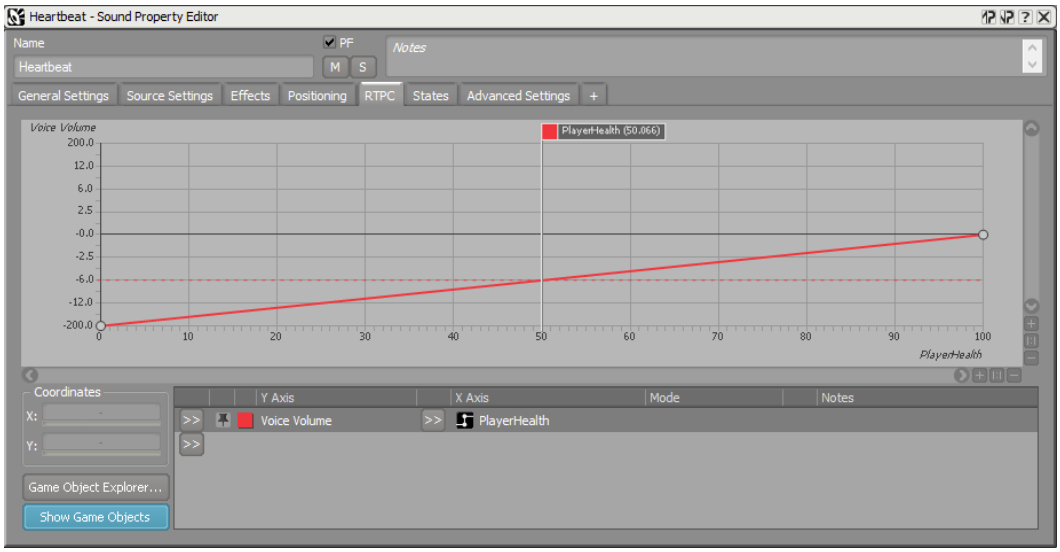


5. X軸の欄で、Selector MenuのPlayerHealthを選択します。



X軸がプレイヤーのHP、そしてY軸がHeartbeatのVoice Volumeを示すグラフを見ることができます。斜めの赤線は、RTPCカーブと呼ばれ、プレイヤーのHPに対応する心拍音のボリュームを示します。Voice Volume 欄の最低

値は-200dBを意味し、何も聞こえません。そのためプレイヤーのHPがゼロと考えることができ、心拍音は聞こえません。実際にはこれは全くの逆であり、PlayerHealth Game Syncsは、100でプレイヤーのフルHPを示し、0でプレイヤーが死亡していることを示すスケールを使用しています。

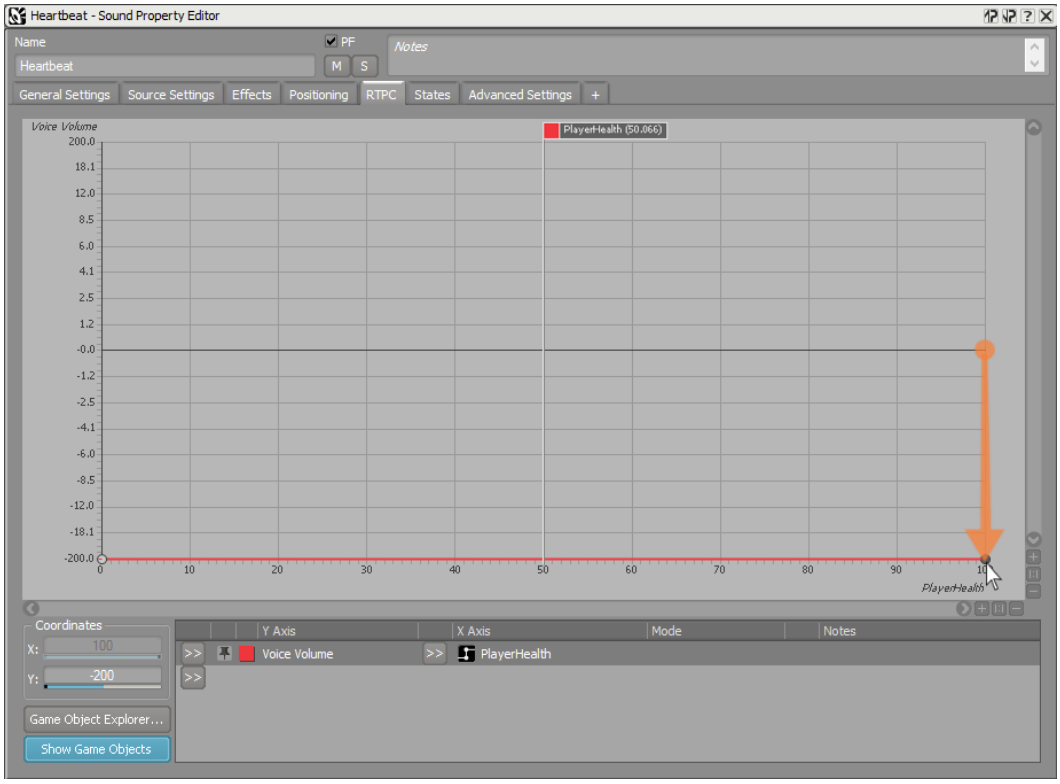


Graph ViewにおけるGame Parameterの調整

プレイヤーのHPが良い状態のときには、Heartbeatが聞こえない仕様にしなければなりません。これは、Graph ViewのRTPC カーブを変更することにより実現することができます。

カーブの両脇にある白いドットのコントロールポイントを動かして調整することができます。

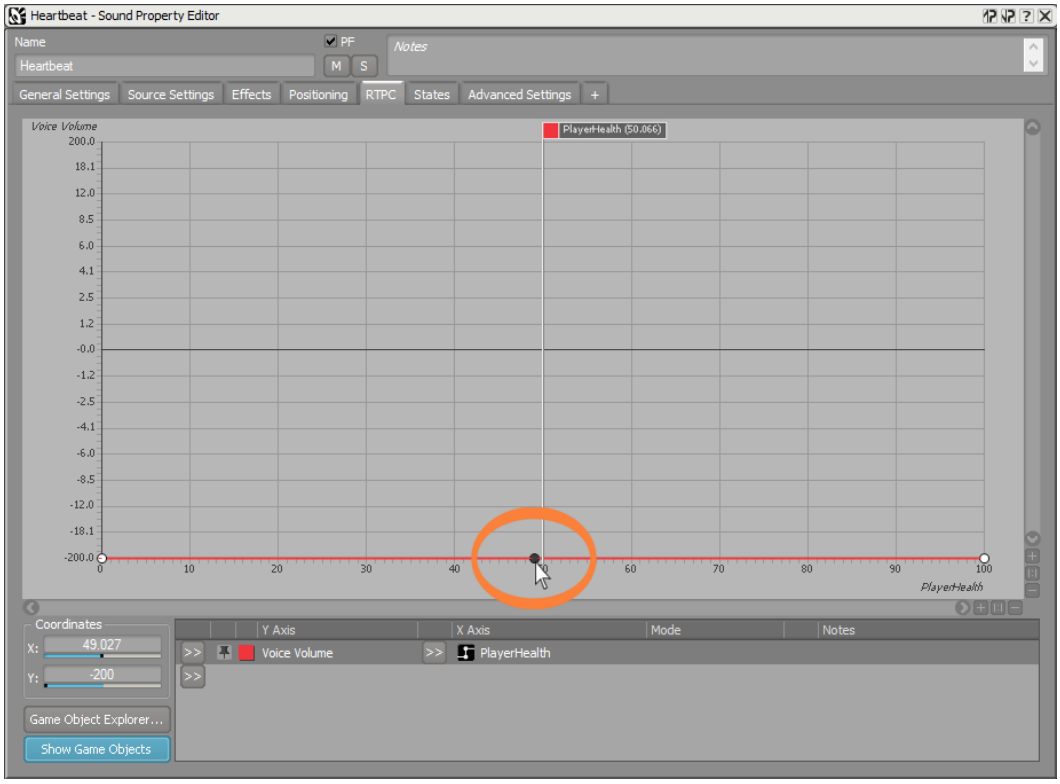
1. 右にあるコントロールポイントを下までドラッグします。



結果は、プレイヤーのHPがどのような状態であろうと、200 dBのボリュームが心拍音の通常再生レベルから減算されます。多くの場合、80 dBものボリュームが引かれるとサウンドは聞こえなくなり、Heartbeat音は全く聞くことができないといっても過言ではありません。そのため、ここで違う調整を行います。

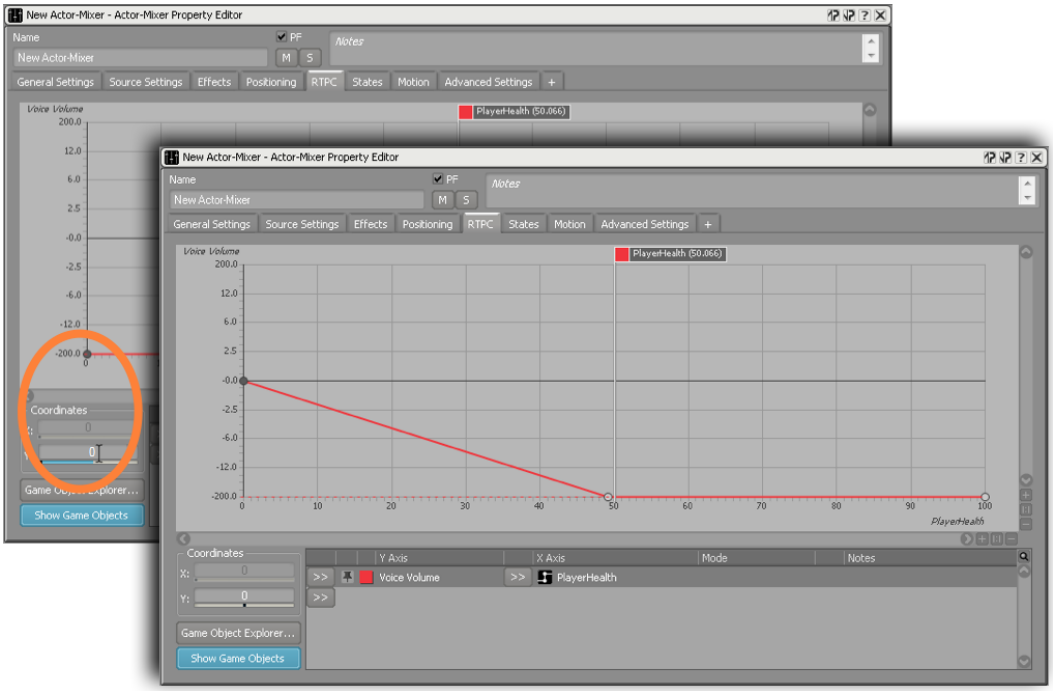
プレイヤーのHPが50以下に落ちるまで、Heartbeat音は聞こえてほしくはありません。そのためこれを適用するカーブを作らなくてはなりません。これはコントロールポイントを追加することで簡単に実現することができます。.

2. 数値50周辺でダブルクリックし、新しいコントロールポイントを作成します。



新しいコントロールポイントが作成されます。

3. 左のコントロールポイントをクリックし、Y座標に0を打ち込み、**Enter**を押します。

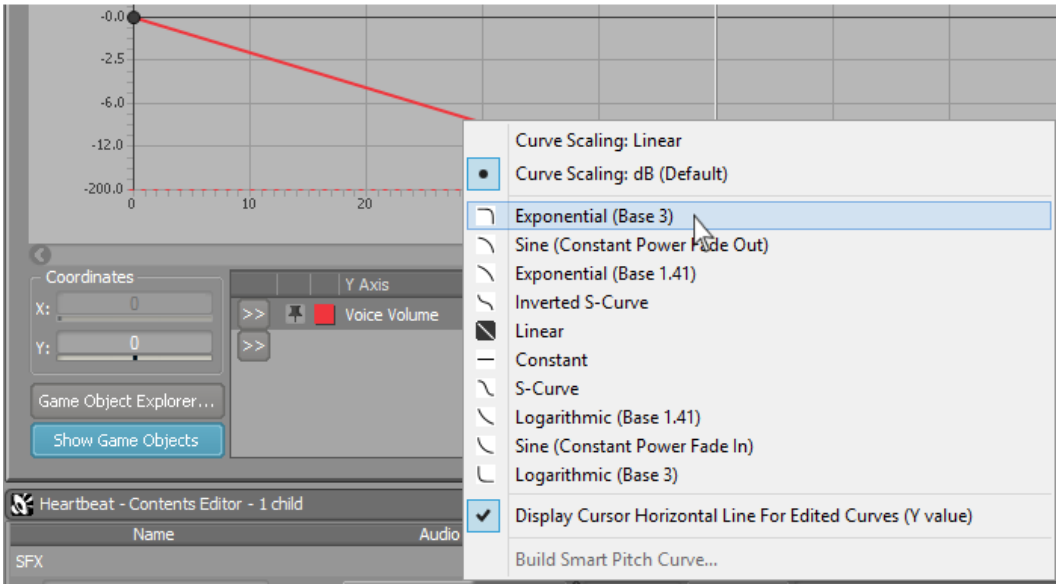


ティップ

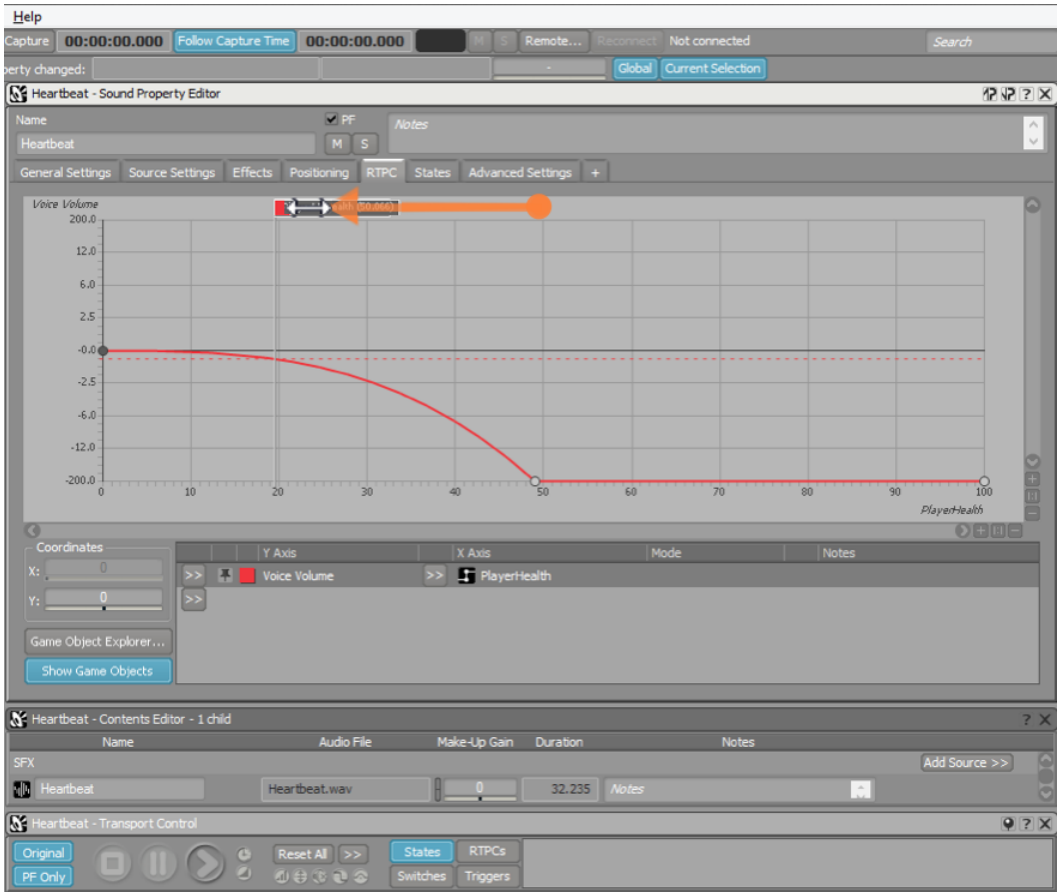
グラフの右下のコーナーにある＋と-のシンボルを使うことにより、カーブのセクションをズームすることができます。＋と#シンボルの中間に位置するアイコンは、カーブ全体を表示するGraph Viewのリセットボタンです。Z＋クリック - ドラッグでズームイン、もしくは、**CTRL＋マウスホイール**でズームイン、そして**CTRL＋SHIFT＋マウスホイール**でズームアウトです。詳細に関しては、Wwise Help MenuのWwise Keyboard Shortcuts Quick Reference Cardをご覧ください。

Heartbeat音のボリュームは、プレイヤーのHPが50以下に落ちるまで上がり始めません。線形の遷移設定では、ボリュームの変化において我々が求めるスムーズな遷移を実現することができないこともあるので、必要に応じてカーブの編集が必要になります。これは、コントロールポイントを追加することにより実現することができます。もしくは、様々なプリセットのカーブスタイルを使用して、さらに複雑な遷移曲線を素早く作成することができます。このケースでは、HPが50以下に落ちた時に急にボリュームが上がるようにカーブを編集します。そしてプレイヤーのHPが0に近づくにつれて、心拍音のボリュームが徐々に上がるカーブを作成します。

4. 今動かしたコントロールポイントの右側のカーブどこかで右クリックし、Exponential (Base 3)を選択します。



Game Parameterの変化をシミュレーションでどのように聞こえるかを試聴しないでカーブの設定をすることは非常に困難です。これを行うためには、Heartbeatオブジェクトをプレイし、グラフのトップにあるPlayerHealth Game Parameter Cursorを右や左に動かします。



5. Heartbeat音をプレイしPlayerHealth Parameter Cursorを左や右に動かし、Game Parameterの効果を確認します。

   Heartbeat音の全体のボリュームを変化させる以外に、徐々にHeartbeatに気が付かせる方法として、Low-Pass Filterを使用します。Low-Pass Filterは、設定された周波数以下の音のみを通す機能があります。低周波数帯域のサウンドのみを聞かせるためのLow-Pass Filter設定は、こもったような音にすることで、これはプレイヤーにとってHeartbeat音に徐々に気が付かせる最適な方法になります。徐々にLow-Pass Filterを開くことにより、Heartbeat 音をさらに目立たせ、明確にしていきます。Wwiseは、一つのGame Parameterをほかの Propertiesにマップすることができるので、先ほど作成したVolume カーブに Low Pass Filterのカーブを加えます。

6. 二つ目のPropertyをマップするために、次のSelector Menuボタンをクリックします。設定は、**Voice Low-pass Filter** からPlayerHealth Game Parameterになります。

2つのラインが見えます。赤色のラインがVolumeカーブを示し、青色のラインがFilterカーブを示します。

Low Pass Filer Property値は0から100になります。どれだけの割合の周波数がフィルターされるのかと考えることができ、スタートは高周波数帯域からです。言い換えますと、数値20は、可聴周波数帯域の高域20パーセントがフィルターされ、サウンドが暗く、もしくははっきりしなくなります。Default値は0のため、現在のオーディオには変化がありません。すべての周波数帯域が聞こえます。

注記

Low-Pass Filter値に関する詳細記事は上記にあります。

ボリュームカーブを作成したように、周波数帯域すべてをフィルターするところから始めます。そしてあるポイントに到達したら、徐々にフィルターを開き、すべての周波数帯域が聞こえるようにします。

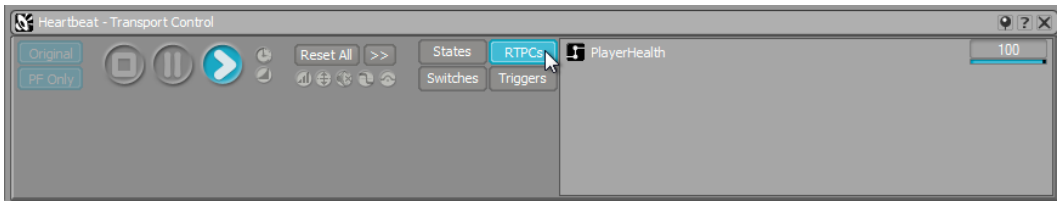
7. 右側のコントロールポイントを上にあげ、2つ目のコントロールポイントを数値60の周辺で作成します。この新しいコントロールポイントを上にあげます。

8. Heartbeat音を聞きながら、Graph ViewでPlayerHealth Parameterの調整を試みます。

TransportでのRTPC値の調整

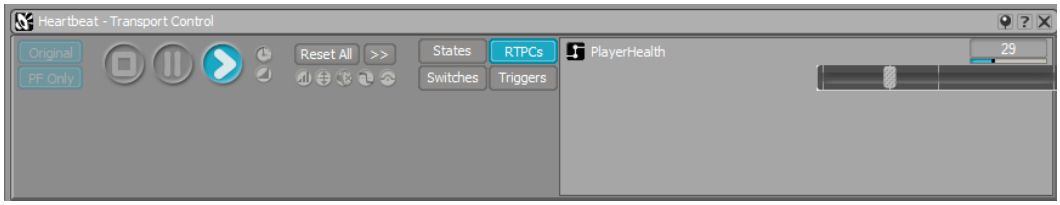
このレッスンのはじめに行ったSwitch Groupのセッティング調整のように、Transport ViewでGame Parametersを便利に調整することができます。

1. Transport Controlで、**RTPCs**をクリックします。



PlayerHealth RTPC とその値が表示されます。

2. 心拍音を聞いている間に、PlayerHealth Parameterをクリックし、PlayerHealth スライダーを調整します。



Statesの活用

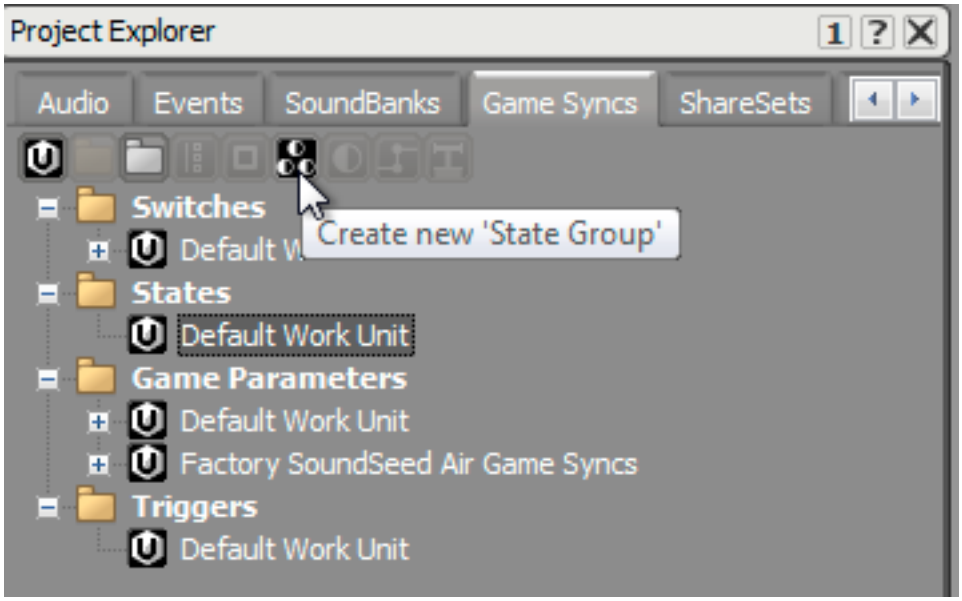
Statesは別種類の特別なGame Syncで、典型的には、ゲーム中のグローバルなイベントに対応してゲームオーディオに一括して変化を反映させるのに使用しま

す。例えば、Stateはプレイヤーの意識のあり/なしや、水の上/下の区別に使えます。Stateのステータスは、例えば先ほど使ったLow Pass Filter など、オブジェクトのProperty値をオフセットして、ゲームの音の特性を変えるために使えます。

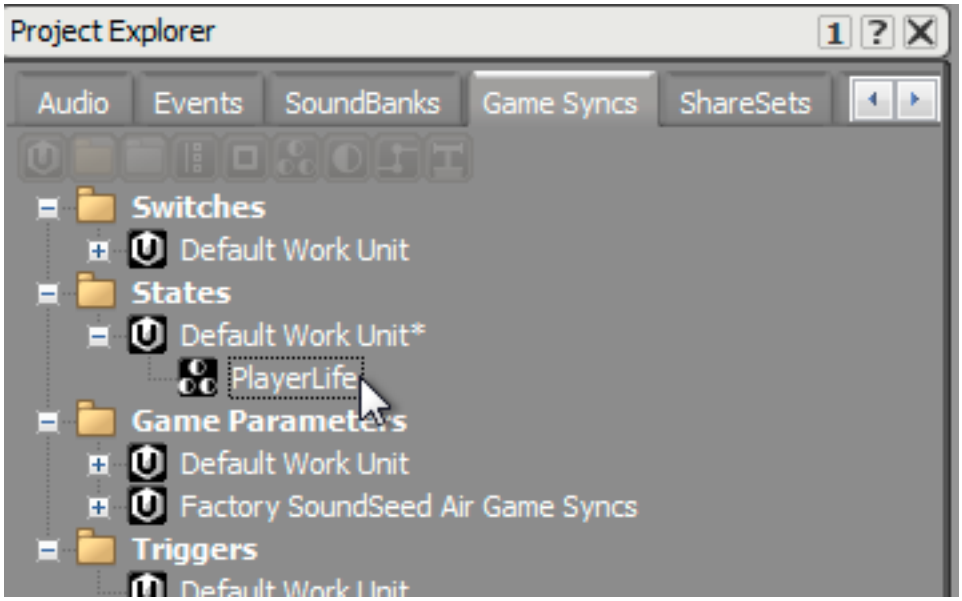
この演習では、プレイヤーが生きているか死んでいるかを示すStateを設定します。このStateをプレイヤーが死亡したときに心拍音を変化させるために使います。ただサウンドを停止するだけではなく、さらに興味深いものにします。このStateはレッスン 4：イベントの作成の作成</e1>でゲームのサウンドの他の種類の変更に使用します。

新しいState Groupの作成

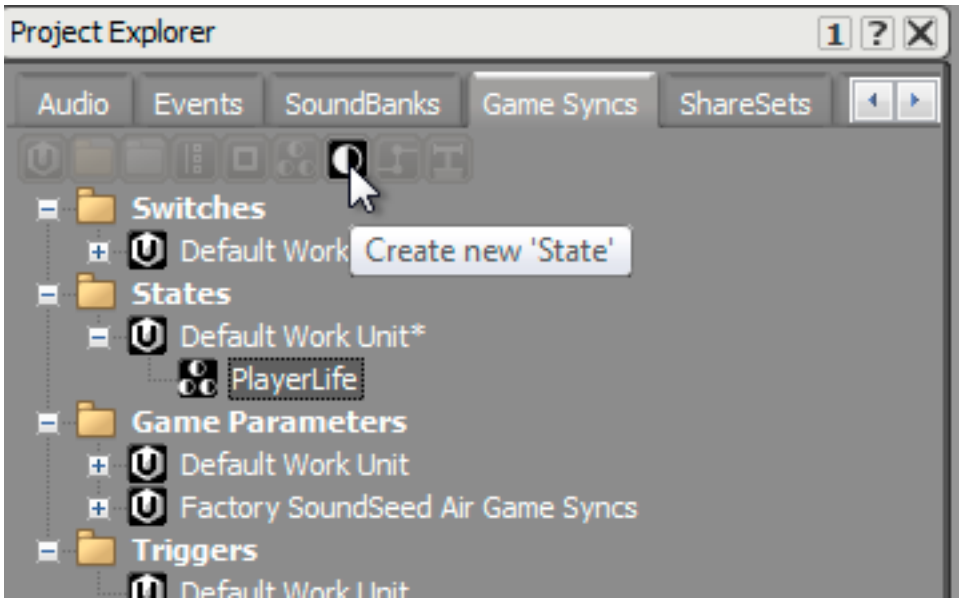
1. プロジェクトエクスプローラーのGame Syncsタブをクリックします。States フォルダーのDefault Work Unitを選択し、 **Create new State Group**ボタンをクリックします。



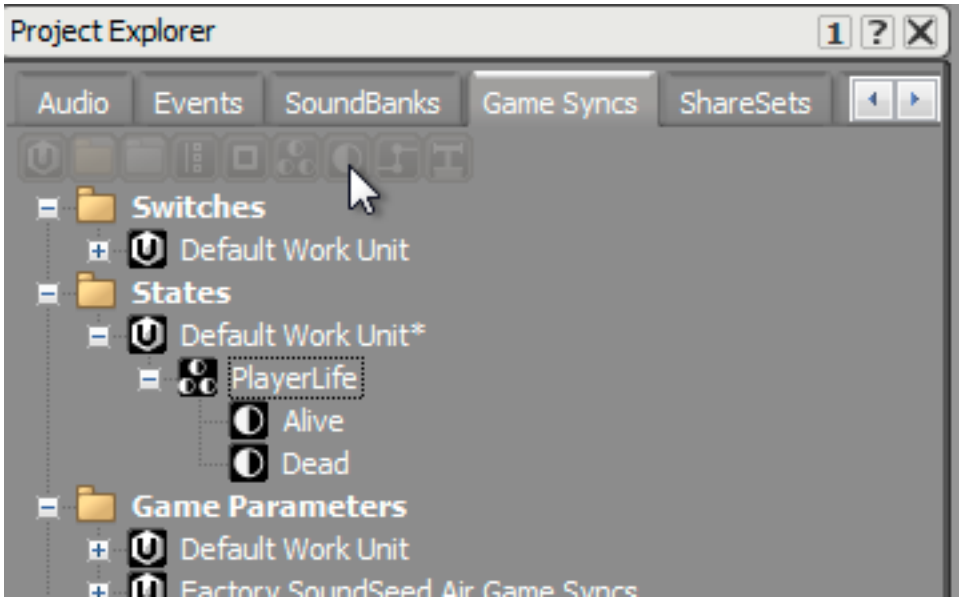
2. 新しいState GroupをPlayerLifeとします。



3. PlayerLife State Groupが選択された状態で、**Create new State button**ボタンをクリックします。



4. 新しく作成したstateをAlive、そしてもう一つ作成し、Deadと命名します。



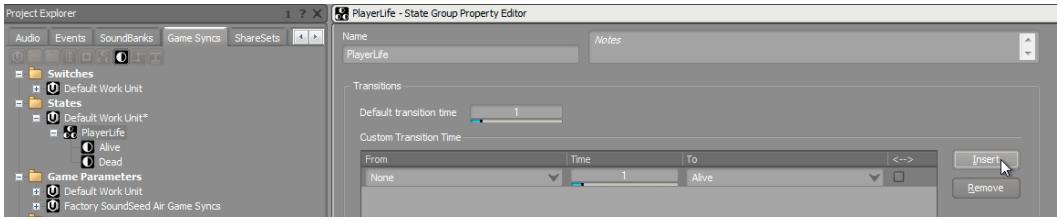
今まで作成してきた他のGame Syncsと同様に、State Groupとそれに含まれるState Objectsの名前は、ゲームコールがこれらのObjectsとマッチするように、ゲームエンジンプログラマに正確に伝えられなければなりません。

State Transitionの作成

Stateは、予め設定した時間に沿って徐々に適用されるOffsetを提供することができます。例えば戦争ゲームにおいて、手榴弾がプレイヤーの近くに落ちた際に、気絶する程のショックの感覚を表現するためにStateを使用し、ゲームのほとんどのサウンドをLow Pass Filterに通す手段として使うことができます。ショックなしのStateからショックのStateの遷移時間は瞬間的だったとしても、ショック状態からショックなしの状態に戻る際は、15秒以上かけて回復させ、爆弾ショックの効果が薄れるまで、徐々にフィルターの影響力を減少させて、すべての周波数帯域が聞こえるようにできます。

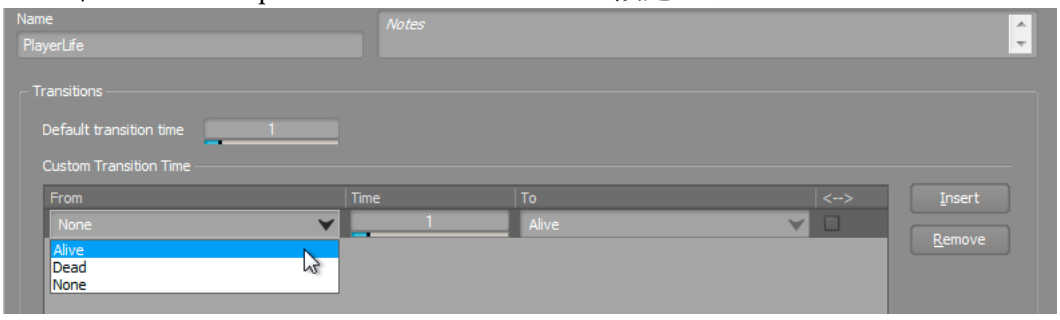
この演習では、AliveからDeadまでの遷移が5秒かかるように設定します。

1. 作成したPlayerLife State Groupをダブルクリックします。State Group Property Editorで、Insertボタンをクリックします。

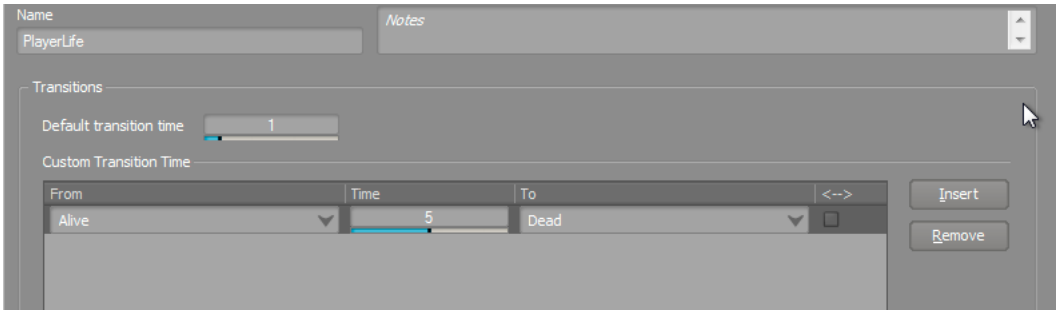


Custom Transition Time エリアで、FromとToの欄があり、Time値がその間に位置します。

2. Custom Transition Time エリアで、Fromのpull-down menuをAliveに、そしてTopull-down menuをDeadに設定します。



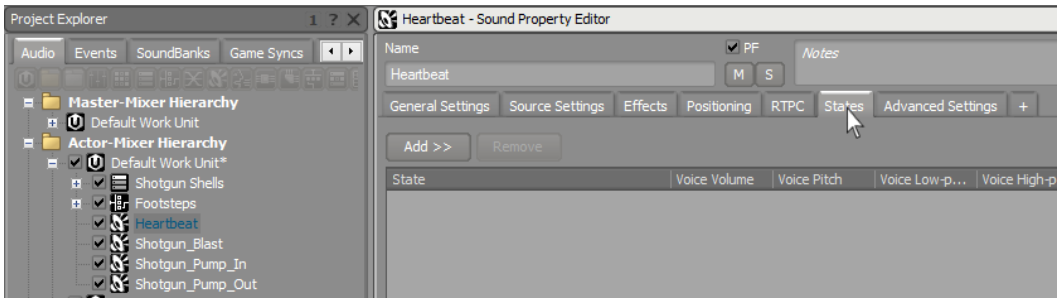
3. Transition timeを5秒に設定します。



State変更値の設定

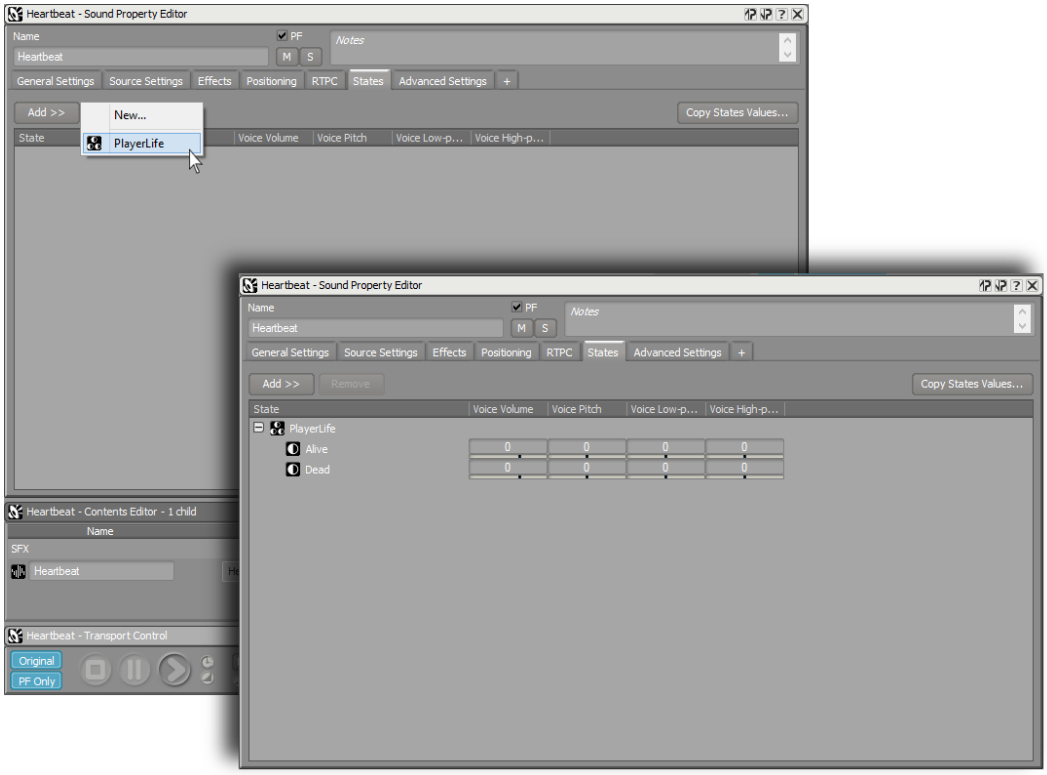
希望するTransition Timeが設定されたState Groupを作りました。ここでPlayerLife Stateが変更された時、Heartbeat Sound SFX オブジェクトのどのPropertiesを変更するかを決めなくてはなりません。

1. プロジェクトエクスプローラーのAudioタブで、Heartbeat オブジェクトが選択されているのを確認し、Statesタブに切り替えます。



Stateの変更によってどのPropertiesが影響をうけるかを設定する前に、どのState GroupがHeartbeatオブジェクトを変化させるかを指定しなければなりません。

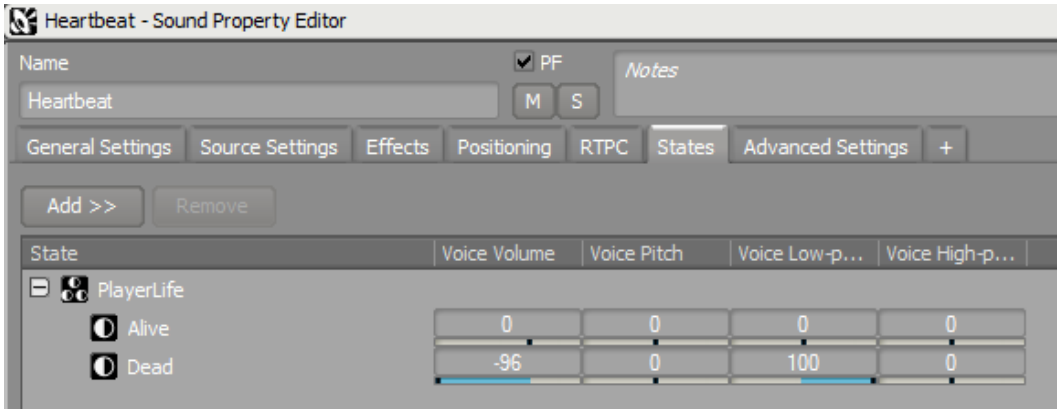
2. **Add**をクリックし、PlayerLifeを選択します。



PlayerLife State Groupがが表示され、そのStatesが下に表示されます。各Stateは、関連するStateが有効になった時にOffsetできるParameters群を表示します。

Playerが生きている時にはOffsetをなくし、Playerが死亡した時には、心拍音のボリュームがフェードアウトするします。ボリュームを低くすると同時に、Low-Pass Filterの値を上げます。

3. Dead Stateでは、Voice Volumeを-96にし、Voice Low-pass Filterを100にします。



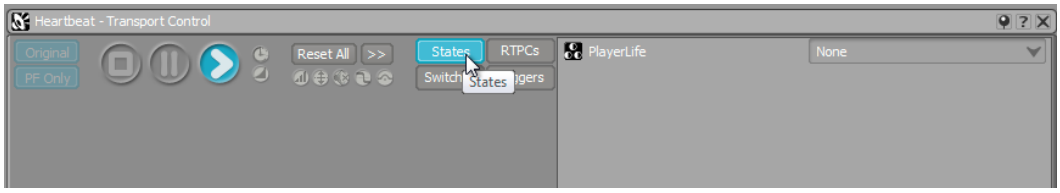
Transport Controlを使用したState Transitionのシミュレーション

AliveからDeadへの遷移時間が5秒間ということを覚えていてください。そのため、ボリュームと心拍音のフィルターの変更は、5秒間で徐々にクロスフェードします。心拍音を聞かせ、死亡した時点で徐々に消えます。

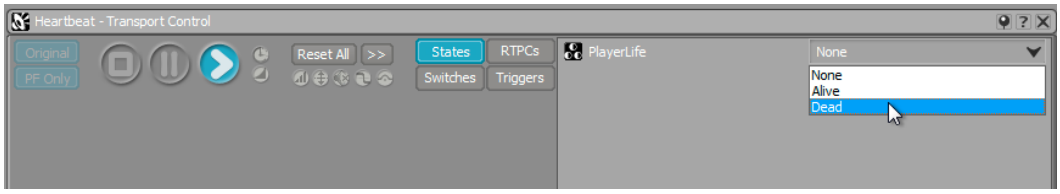
SwitchesやGame Parametersで行ったTransport Controlでのテストと同様に、Statesでも同じことが行えます。テストを行う前に、Heartbeat音が聞こえなければなりません。そのためPlayerHealth RTPCを0に設定し、Heartbeat音がしっかりと聞こえる設定にします。

1. Transport Control Viewで、PlayerHealth RTPCを0に設定します。

2. **States**をクリックします。



3. PlayerLife State GroupをDeadに変更します。



プレイヤーの心拍音が徐々に消えますが、PlayerLife StateをAliveに変更することで、生き返らせることができます。DeadからAliveへの遷移時間を設定していないため、デフォルトで設定されている1秒の遷移時間がここでは使われます。

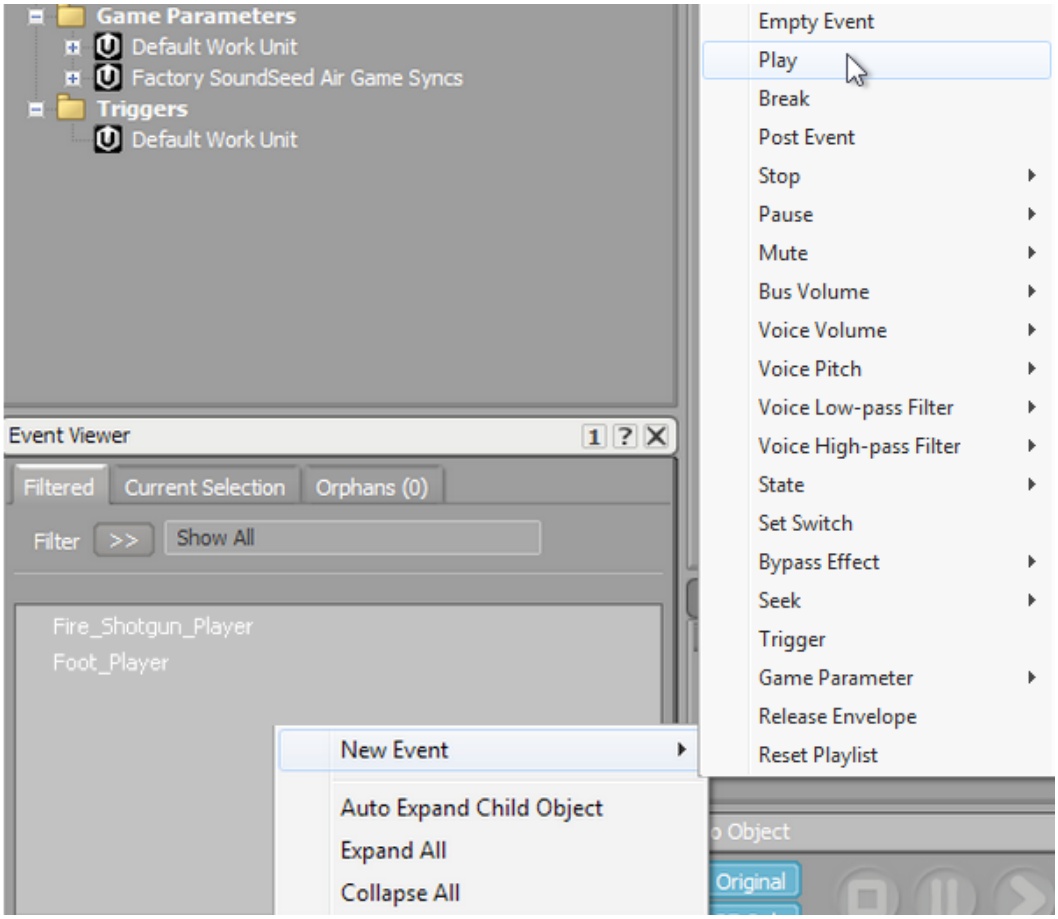
CubeデモへのGame Syncsインテグレーション

今まで行ってきた作業をCubeデモに組み込みます。前の2つのレッスンと同様に、SoundBankを再生成することで行ないます。しかしまずここではどのようにしてプレイヤーのHeartを再び動かすかを考えなくてはなりません。Cubeデモには、プレイヤーのHeartをスタートさせるGame Callがありません。プログラマに追加を依頼することもできますが、プログラマを介することなく同じことを実現できないかを考えます。ゲームがプレイされる際に一度だけ起こるイベントが必要になります。

幸い、レベル開始時にミュージックをスタートするのに使用されるイベントが既にあります。これは心拍音用ではありませんが有効です。

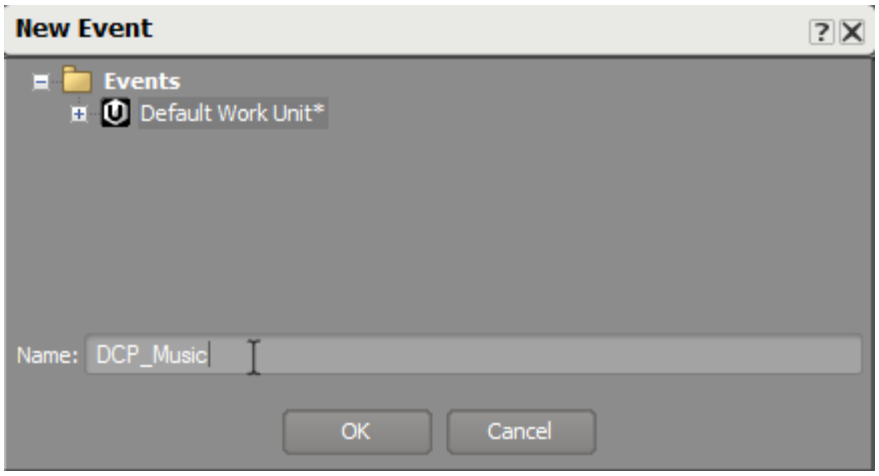
今回は、プロジェクトエクスプローラーのEventタブに戻る代わりに、ショートカットを使います。

1. Event Viewerのメインエリアで右クリックし、**New Event** > **Play**を選択します。

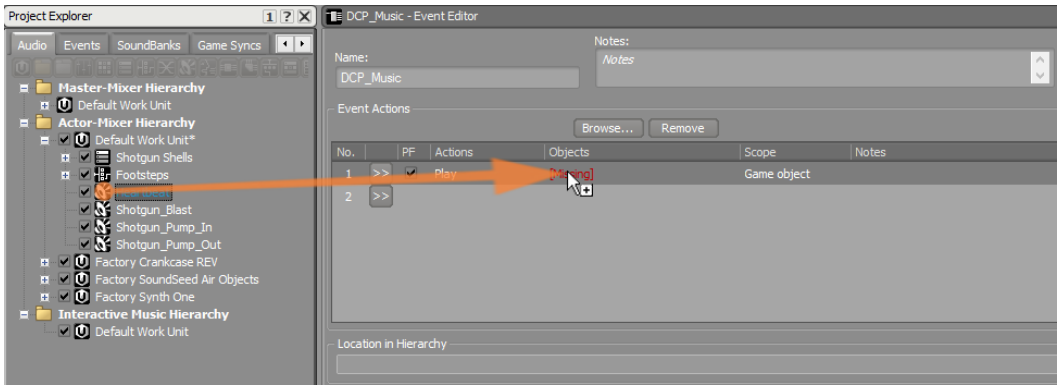


Eventsフォルダーを表示した新しいEvent Dialog Boxが開きます。

2. Nameフィールドに**DCP_Music** と記載し、**OK**をクリックします。

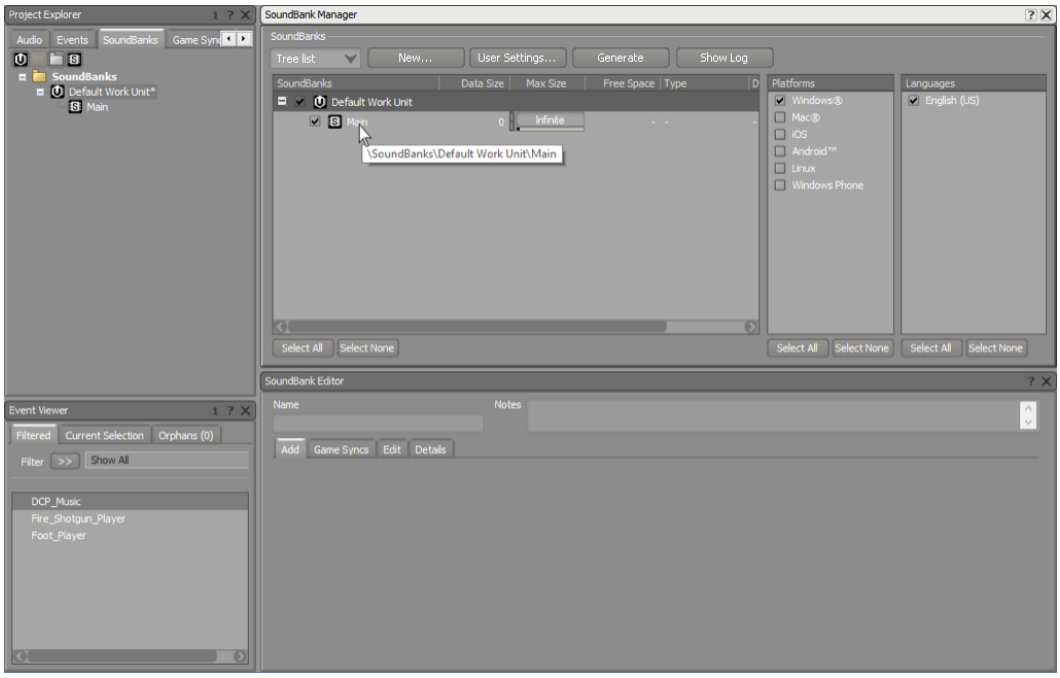


3. ActionリストのObjects欄にHeartbeat Sound SFXオブジェクトをドラッグします。



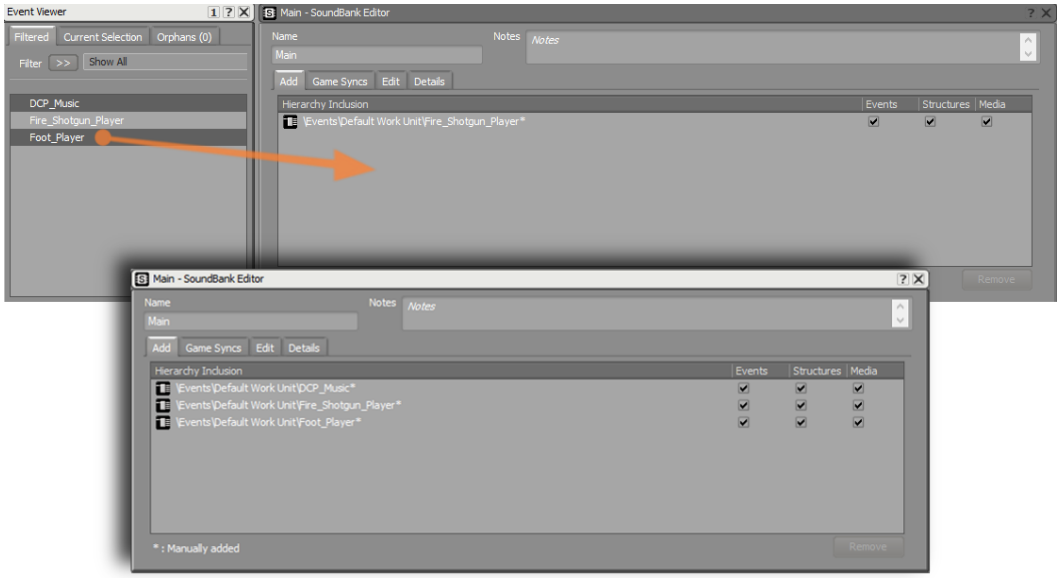
これで、SoundBankのアップデートの準備が整いました。

4. メインメニューバーで、 **Layouts** > **SoundBank**を選択、もしくは **F7**を押します。

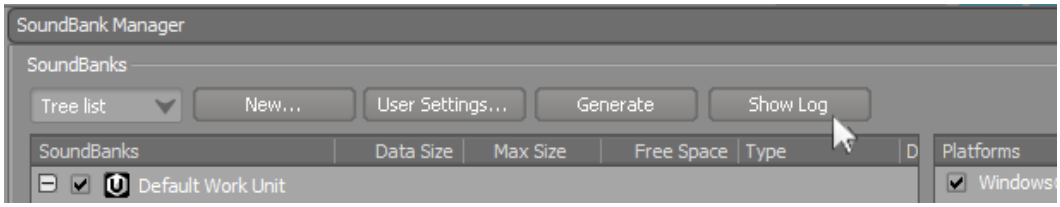


Foot_PlayerとDCP_Musicイベントを SoundBankに追加します。

5. Event Viewerで、DCP_MusicとFoot_Playerを Main SoundBankの SoundBank Editorにドラッグします。



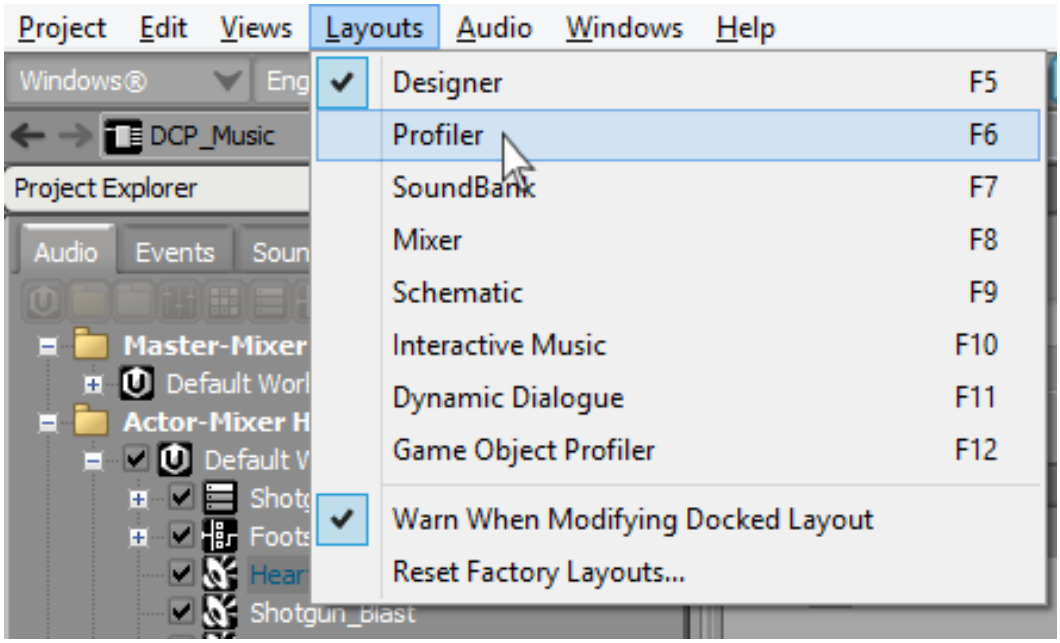
6. SoundBank Managerで、Game Sync Integrationが含まれたSoundBankを**Generate**を押して作成します。



ProfilerでのGame Syncsの確認

Profilerを活用し、このレッスンで設定したGame Syncsを見て、正しく動作しているかを確認します。

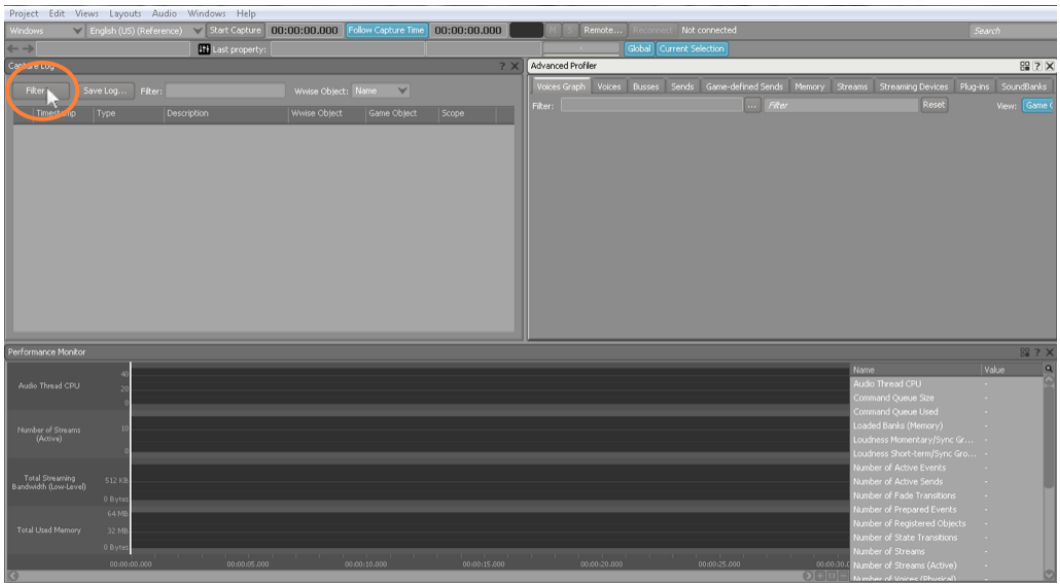
1. メインメニューで、 **Layouts** > **Profiler**を選択します。



Profilerレイアウトが表示されます。

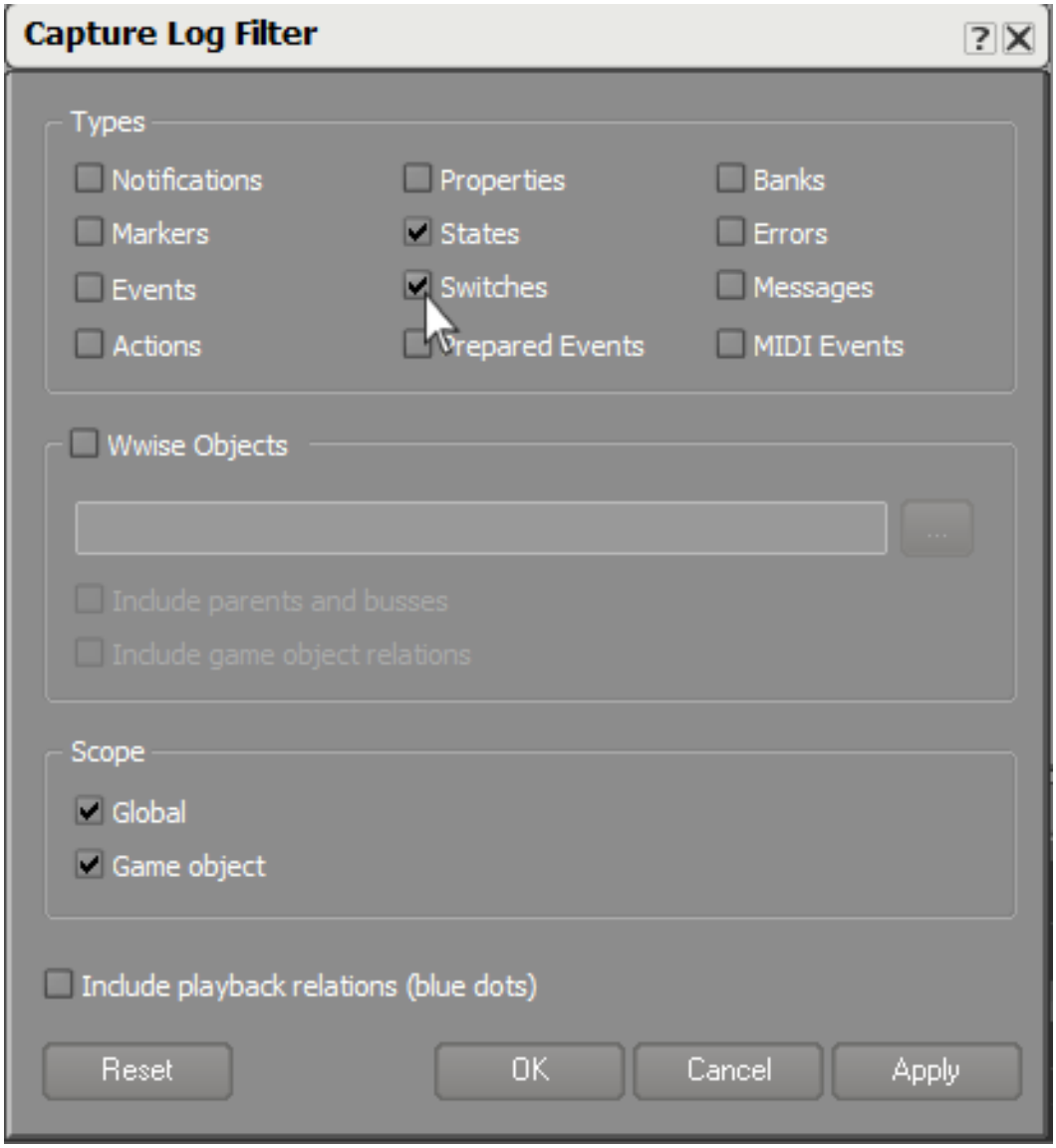
Profilerで様々な情報を確認することができます。しかしながらこの情報は時に多過ぎることがあります。ここでは、Game Syncの情報だけが必要なため、Filterを使い、確認したい事柄のタイプを選択します。

2. Profiler Layoutの左上エリアにあるCapture Log Viewの**Filter**をクリックします。



Capture Log Filterが表示され、確認したい情報項目それぞれにチェックボックスが用意されビューをフィルタすることができます。

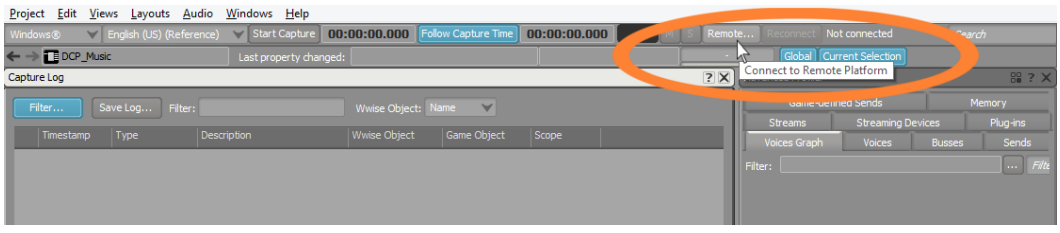
3. StatesとSwitches以外のすべてをクリアし、OKをクリックします。



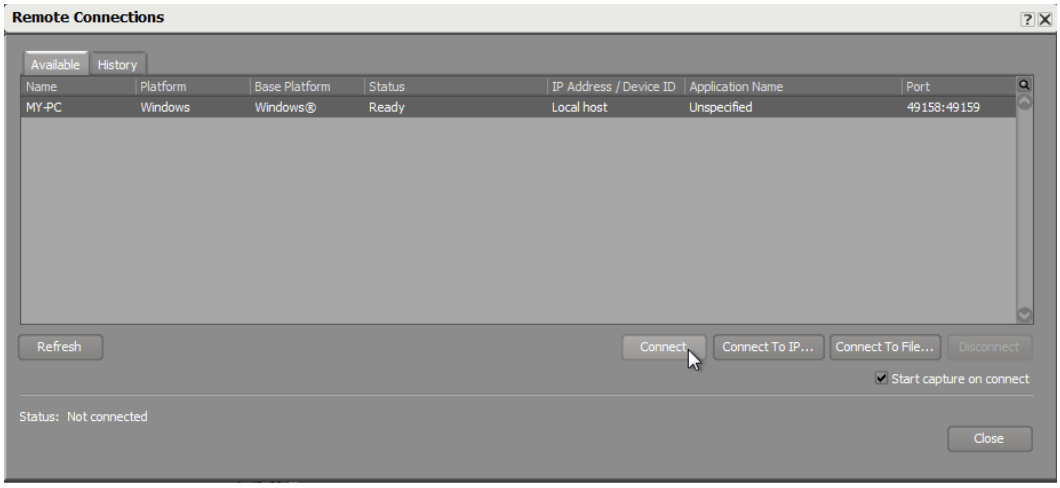
4. Cubeデモを起動します。

   Wwiseに直接つなぐことができる特別なProfilerバージョンのCubeデモを起動していることを思い出してください。Cubeデモが起動したら、Wwiseとゲームの接続を確立します。

5. Wwise側で、**Remote**をクリックします。

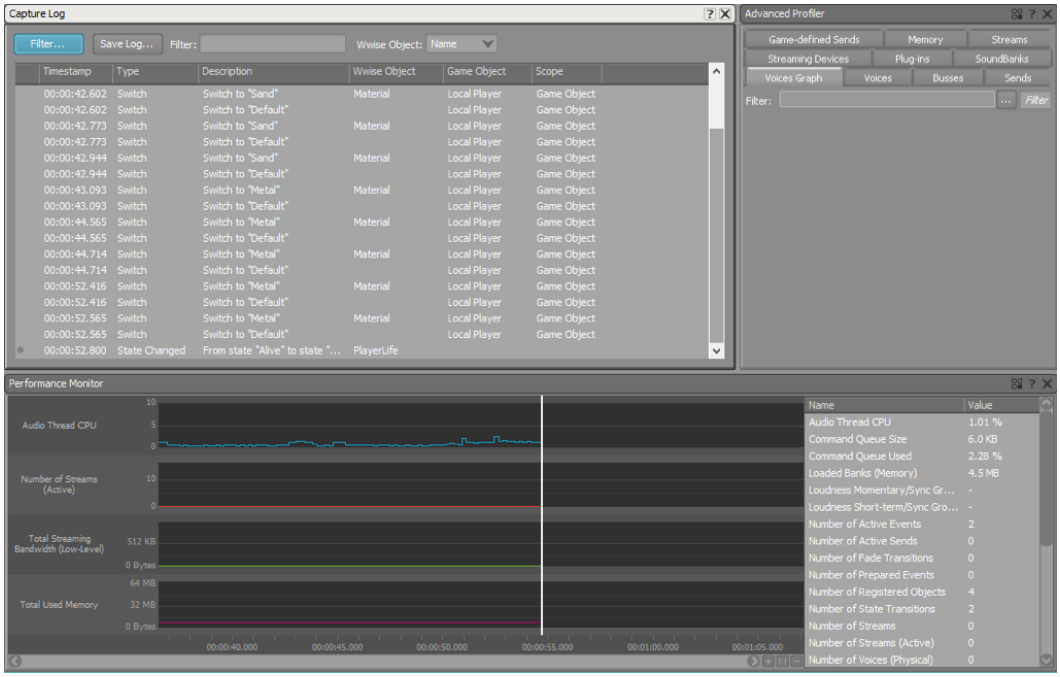


   コンピューターを選択し、**Connect**をクリックします。



Wwiseがゲームに接続します。Start capture on connectのチェックボックスにチェックが入っているため、Wwiseは瞬時にログのキャプチャを開始します。

6. ProfilerのCapture Logでキャプチャーしながら、Cubeデモをプレイし、このレベル内を走りまわります。



プレイヤーがレベル内を移動すると、Wwiseはプレイヤーが今いる地面を示すSwitch Callsを受け取っていることを表示します。これはCapture Logで確認することができます。

レッスン ４：イベントの作成

3D Game Definedの使用 140
3D User Definedを理解する 160
2Dパンニングの使用 169
ゲームをプレイする 174

目を閉じて、あなたをとりまく音に耳をすませてみてください。おそらく遠くに鳥のさえずり、頭の上を飛ぶ飛行機、時を刻む時計の音などが聞こえるでしょう。目を閉じた状態で、これらの音がどこからくるのかどうやっって分かりますか。その音は右、それとも左から、近くから、または遠くから聞こえますか。あなたがこれらを判断する能力は物理法則によるものです。あなたの右から聞こえる音は、左の耳に到達するよりも早く右の耳に到達し、より大きく聞こえるので、あなたの脳はこれらの差異を素早く認識し、音が右からくることを認識します。遠くのオブジェクトは常に、全体的な音量においてのみならず、階調においても比較的静かに聞こえるもので、それは音の特定の周波数は他の帯域と比べ容易に伝搬されずに、あなたとの距離に従って音の全体的なクオリティにも影響を受けます。与えます。

オーディオエンジニアはボリュームとパンニング制御などにより、録音された音源をスピーカーセットから再生して聞く際に、音空間を再現しようとします。音楽エンジニアはボリュームとパンニング制御により、リスナーに認識してもらいたい音像を描きます。例えば、ギタリストがステージの左側、もしくは右側に立っているか。映画やTV向けのオーディオでは、スクリーン上に映るイメージがあり、サウンドエンジニアは上記の制御を使用してビジュアルにマッチする聴覚イメージを制作します。仮に、画面の右側から話をしている男性の声が左側から聞こえたら没入感がそがれることでしょう。ビデオゲームにおいても、映像がスクリーン上にある映画やTVのためのオーディオと同様ですが、大きな違いはゲームでは表示されるイメージがどのようなものになるか予測できず、それはゲームをプレイするプレイヤーによって決定されます。こうした背景から、ゲームオーディオエンジニアは従来の方法でボリュームやパンニングの制御を使用することができません。ゲームの非予測性に対応するために、Wwiseのようなオーディオエンジンはゲーム自体が自動的にオーディオがリアルタイムにミックスされる制御を可能にする特別なシステムを使用します。

3D Game Definedの使用

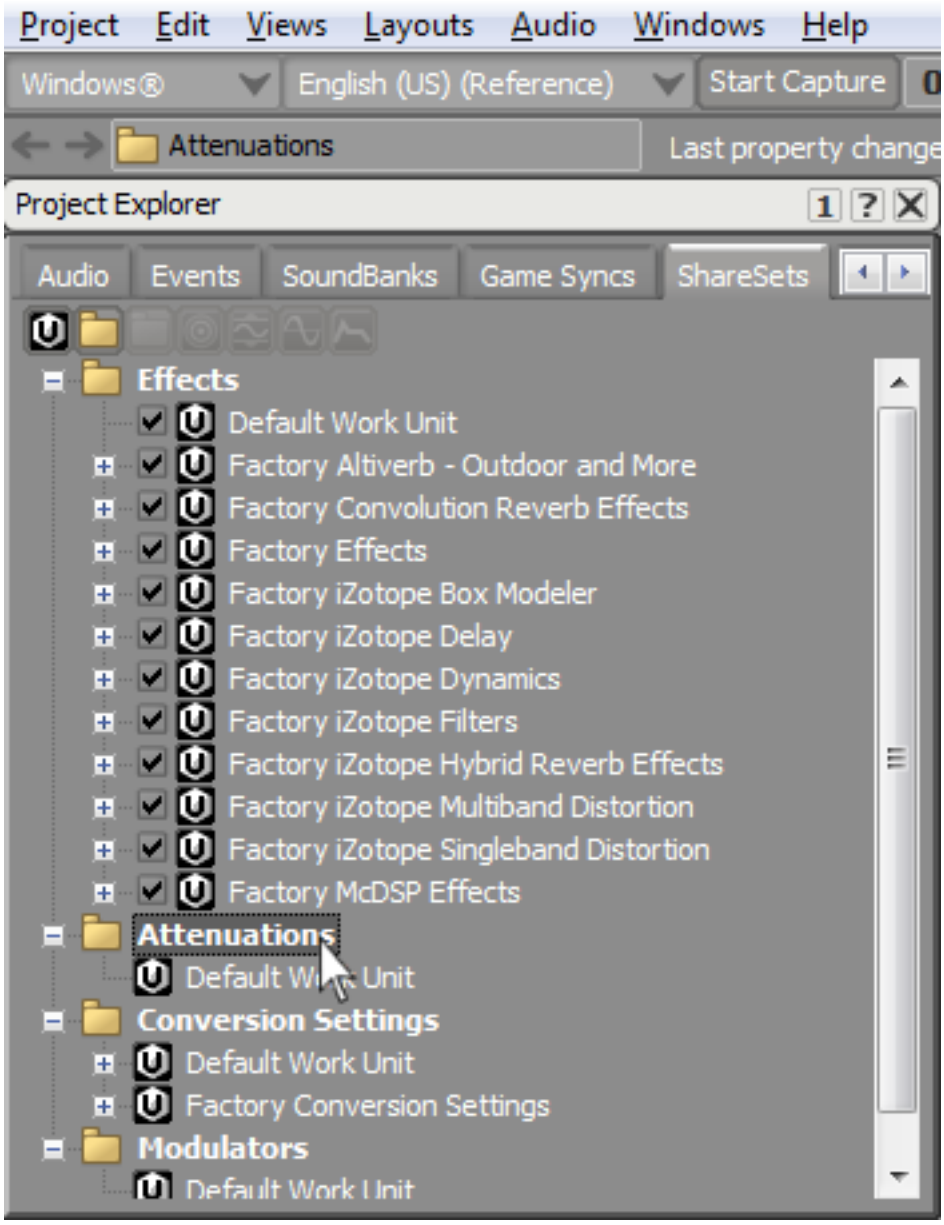
現実世界おいてと同様に、ゲーム内のオブジェクトも、近づく敵の足音などの、さまざまな音を発します。ゲーム内で音を発するものは全て概念的に エミッターと呼ばれます。ゲームにおいてエミッターを持つこと自体、これら音を受ける、もしくは聞くものがなければ有用ではありません。リスナー は通常、一人称視点のゲームでは、キューブデモにおいてあなたが操作するキャラクター同様に、メインキャラクターと関連付けられています。エミッターとリスナーはゲームオブジェクトに関連付けられ、x, y, z座標と、オブジェクトの向きを示す情報を持っています。Wwiseでは、これら全ての情報を使用し、エミッターとリスナー間の空間的な位置関係にもとづき各種サウンドのボリュームやパンニングの適切なプロパティ設定を実現します。 これら全ては即座に処理され、ゲーム内で様々なオブジェクトが移動し、空間的な位置関係が常に変更されるのに対応して、各種プロパティが更新されます。これらの処理判断は3D空間におけるオブジェクトの位置に従うため、このタイプのプロパティは3D Game Definedと呼ばれています。

3D Game Definedを使用すること自体が、サウンドデザイナーが不要になるということではありません。実際、これら空間的な位置関係にもとづいて、Wwiseがどのように振る舞うかを定義する基盤をなす重要なデザイン判断をあなたはゆだねられています。例えば、あなたはエミッターがリスナーに対してどれくらい離れているかによって、サウンドのフェードのかかり具合を指定したりします。

Attenuation Curve ShareSet（減衰カーブシェアセット）の作成

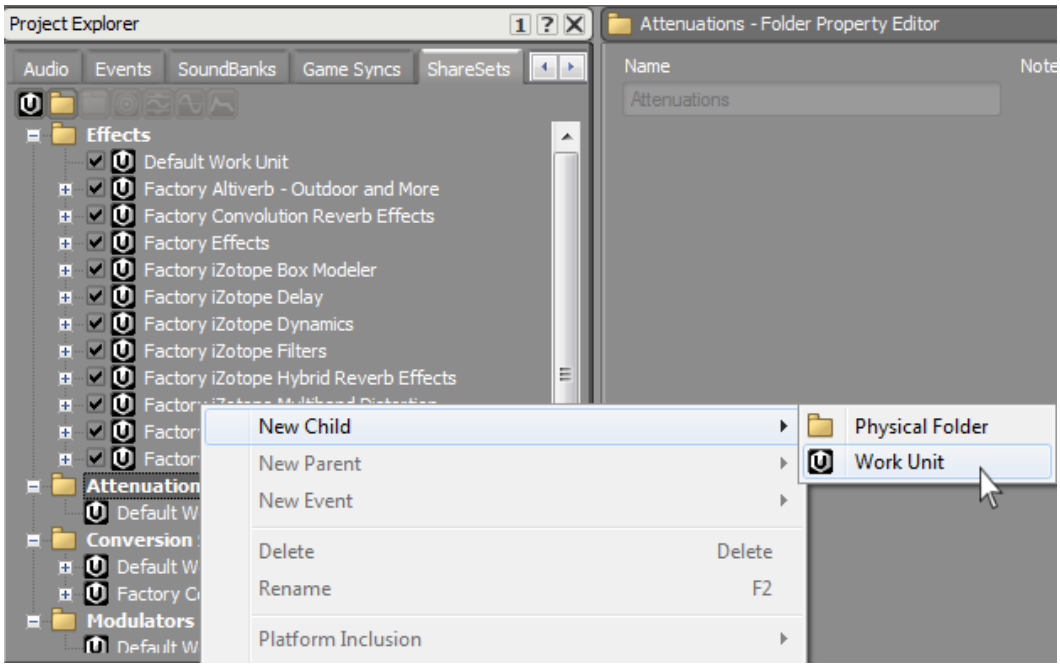
位置情報を元にサウンドのプロパティを変更することは、減衰カーブと呼ばれるものを使用して実現します。次の演習では、減衰カーブを作成する方法を学びますが、まず減衰カーブShareSetを設定する必要があります。ShareSetはプリセットのようなもので、頻繁に使う設定や、プロジェクトを構築する際に複数のオブジェクトに対して各種タイプの効率的な適用を可能にします。

1. レッスン4のWwiseプロジェクトを開きます。
2. プロジェクトエクスプローラーで、ShareSetsタブをクリックします。

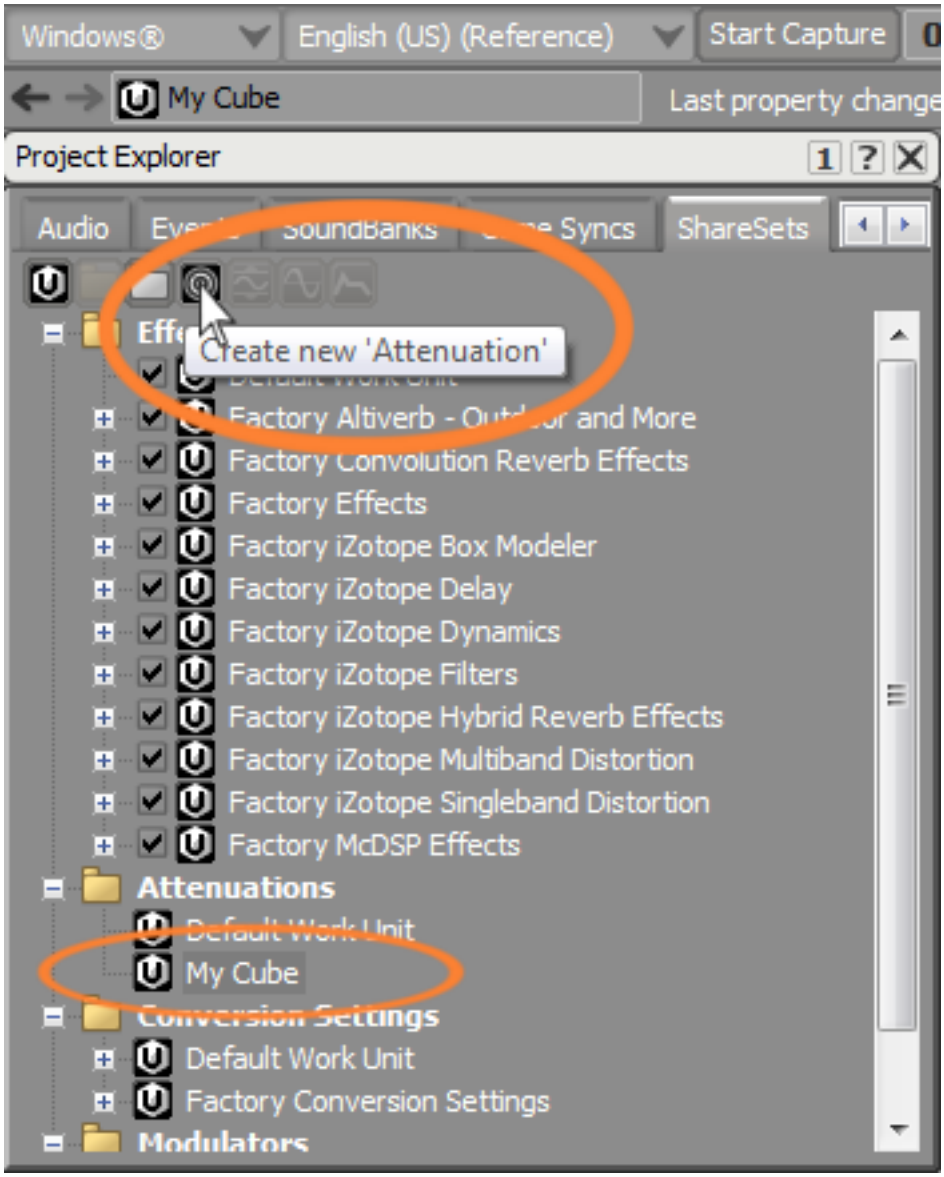


タイプ別に異なるフォルダへ分割された各種ShareSetワークユニットが確認できます。

3. Attenuationsフォルダを右クリックし、New Childを選択し、My Cubeという名称で新規ワークユニットを作成します。

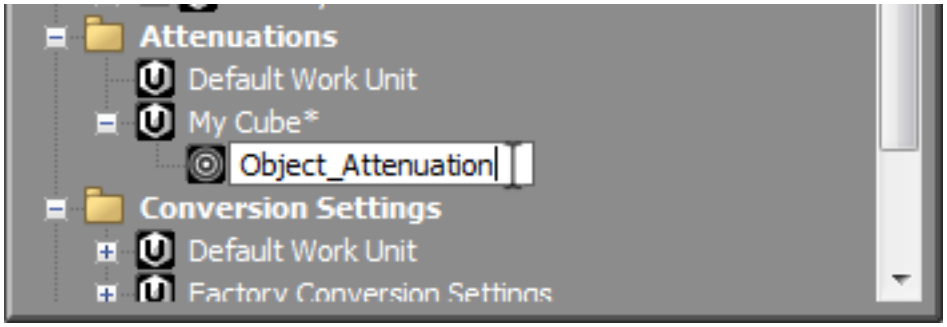


4. 新規ワークユニットを選択した状態で、**Create new Attenuation** アイコンをクリックします。



環境の物理特性はあらゆるタイプのサウンドに対して平等に影響を与えるので、たいていの場合、ゲーム内で使用する大多数のサウンドオブジェクトに対して使用できる単一の減衰カーブを生成することになります。

5. その減衰オブジェクトをObject_Attenuationと命名します。



減衰カーブをオブジェクトへアサインする

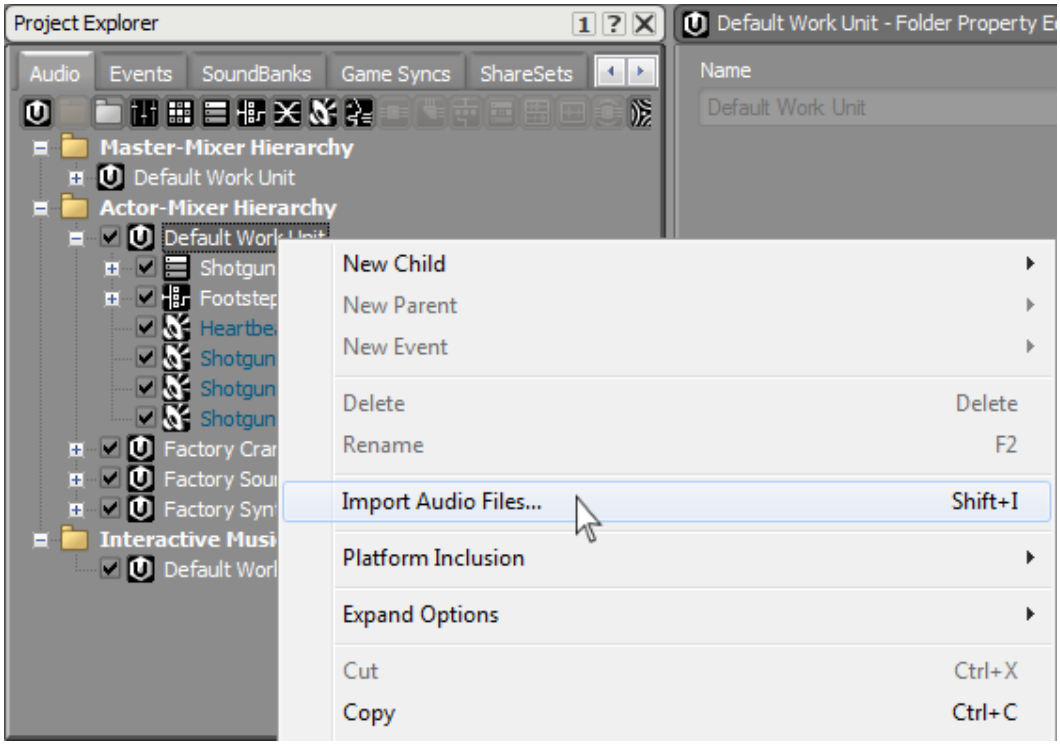
減衰カーブは Property EditorのPositioningタブを使用して、Actor-Mixer階層内のオブジェクト適用することができます。どのように減衰カーブが作用するかを確認するために、Cubeデモ内にあるテレポーターへサウンドを追加します。各テレポーターが環境に対して、一定のサウンドを発信していると考えて下さい。あなたがそのサウンドを聞こえるようになるにはどれだけ近づく必要がありますか？この課題は、減衰カーブが対処するものの一つです。

まず、テレポーター用のサウンドが必要です。

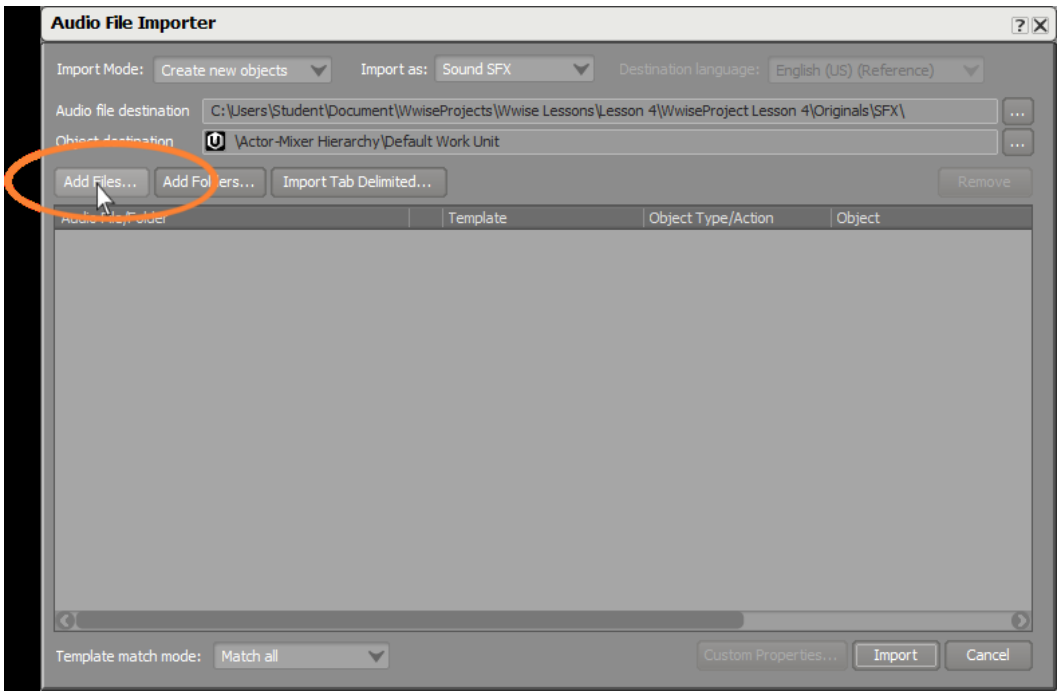
注記

減衰カーブはインタラクティブミュージック階層内のオブジェクトにも適用できます。

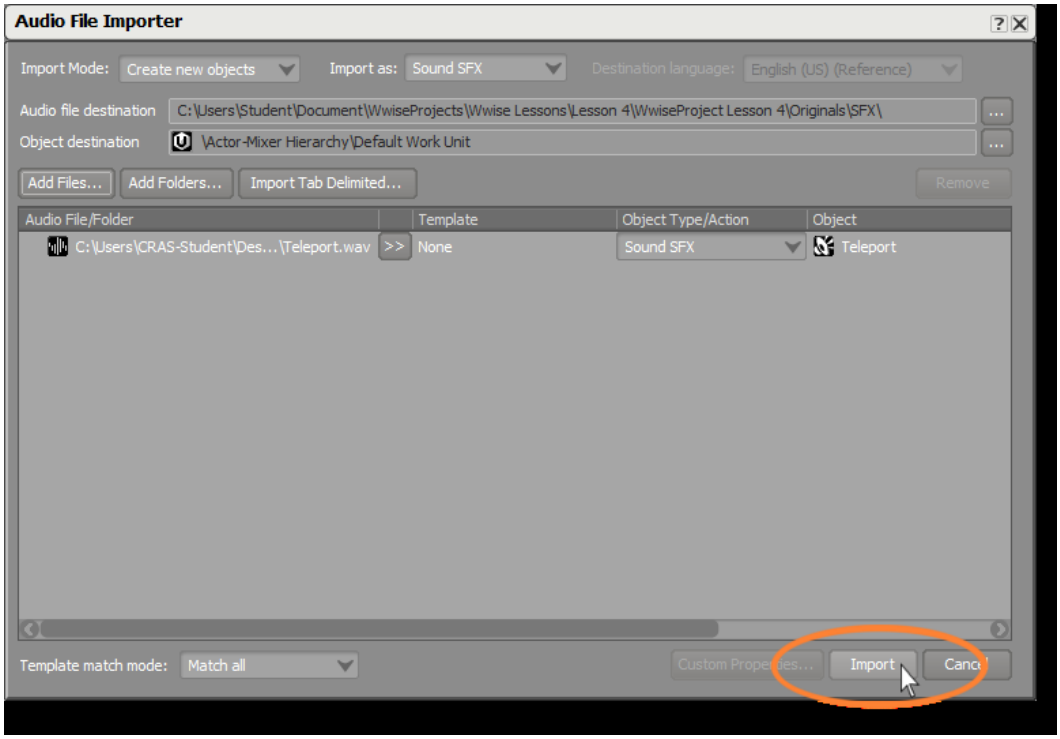
1. プロジェクトエクスプローラーのAudioタブで、Actor-Mixer階層のデフォルトワークユニットを右クリックし、**Import Audio File.**を選択します。



2. **Add Files.**をクリックします。

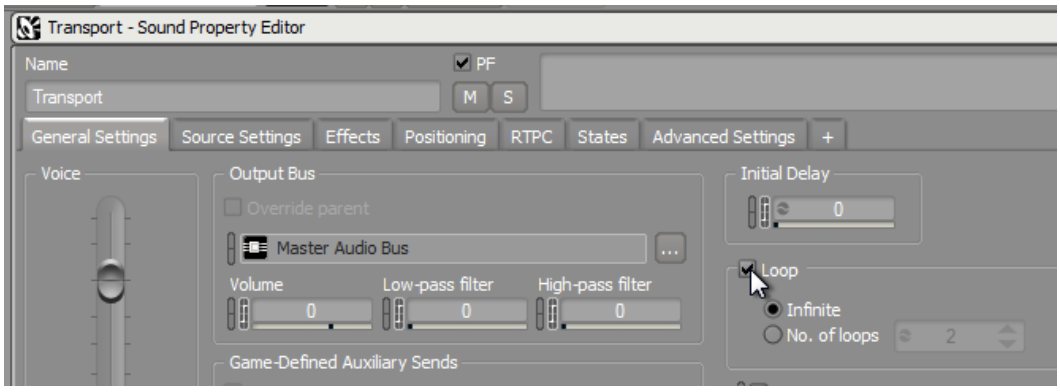


3. あなたのレッスン4フォルダまでナビゲートし、レッスン 4フォルダのAudio Files内にあるTeleport wavファイルをselect/openし、次に **Import**をクリックします。

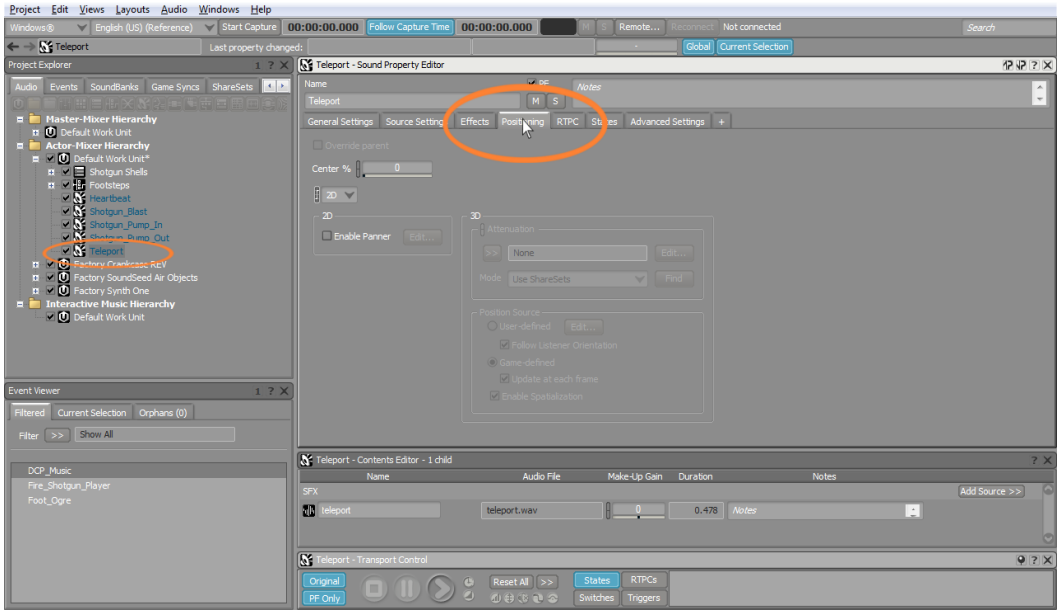


いまあなたがインポートしたファイルは非常に短いものです。テレポーターはコンスタントなサウンドをエミットするものなので、オブジェクトをループする必要があります。

4. 今、作成したTeleport SFXオブジェクトを選択し、General Settingsタブ内のloopチェックボックスをチェックします。

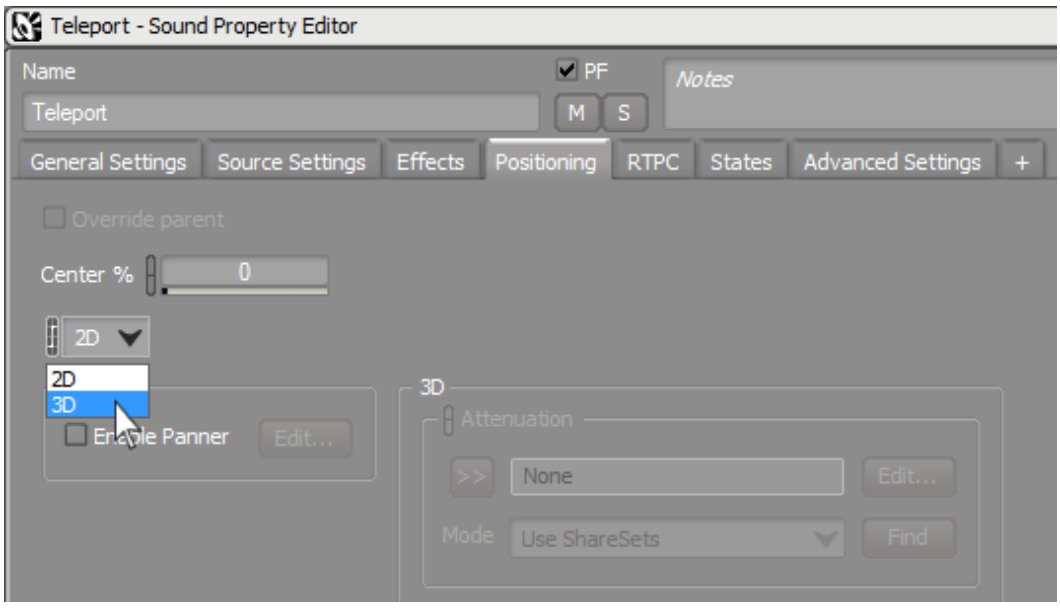


5. 作成したTeleport SFXオブジェクトを選択し、Sound Property エディター内のPositioningタブへ切り替えます。

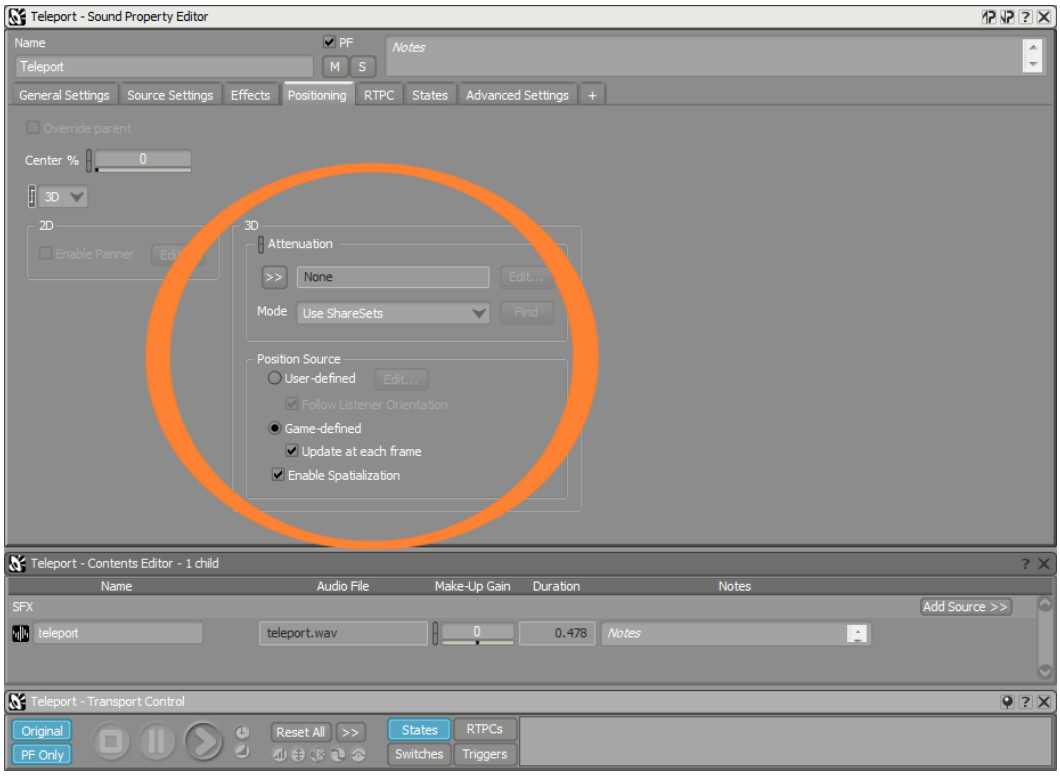


Positioningタブで最も重要な制御項目のうちの一つは、プレイヤーのスピーカーに対してサウンドが2Dもしくは3D手法でレンダリングされるかを定義するものです。端的に云えば、2Dではサウンドは、現在の音量やパンニング位置の変更無しに、スピーカーから単純に音を再生します。2Dを使用する場合の応用や、オプションについては本レッスンの後半で学びます。3Dオプションを選んだ場合には、テレポーターのようなサウンドを発するオブジェクトの座標を使用して、自動的にサウンドプロパティが変更され、聴覚体験が視覚体験と一致するようにします。

6. Teleport Sound Propertyエディタで、2D設定から3Dに変更します。

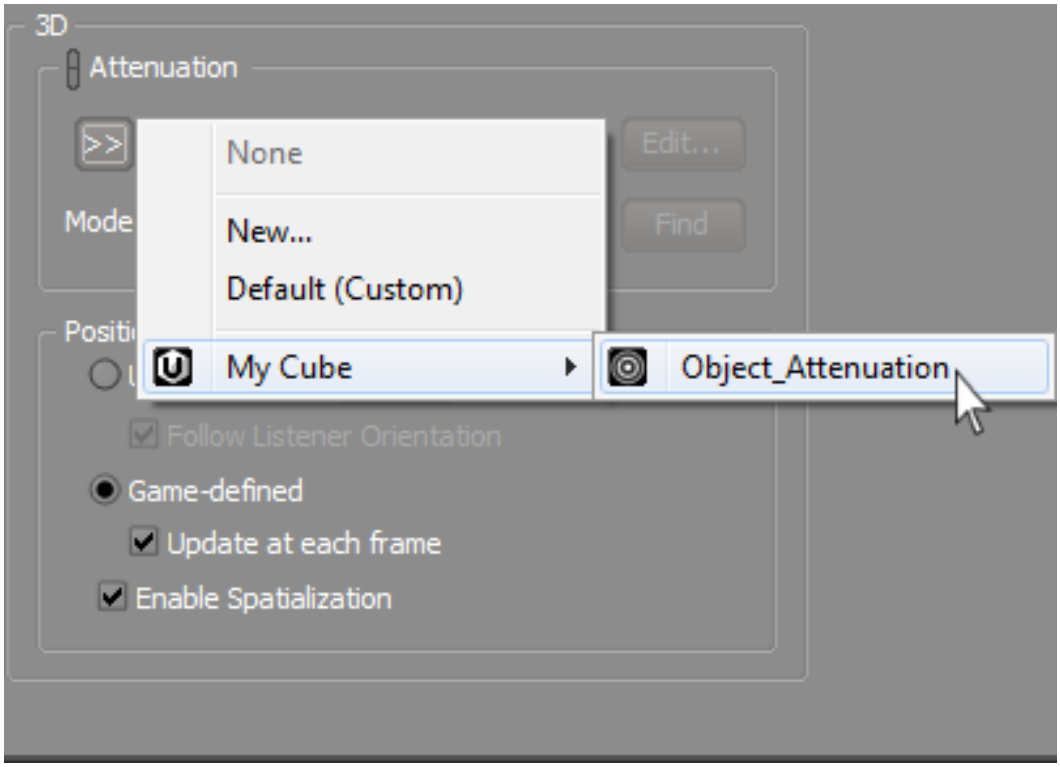


3Dを選択したことで、3Dパラメータオプションが使用できるようになりました。



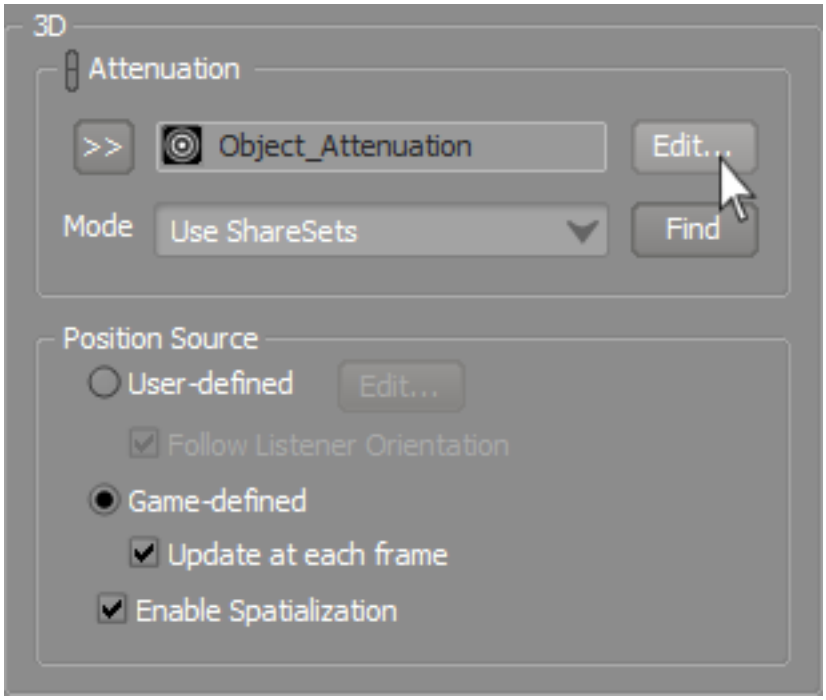
Object_Attenuation ShareSetのコンテナー内で、3Dパラメータの調整を行ないます。テレポーター専用の減衰カーブを作成することも可能ですが、あなたはテレポーターを使用して、距離に対してワールド内のサウンドがどのように変わるのかをテストします。空間距離を伝搬するサウンドの影響は、ワールド内の全てのオブジェクトに対しておおよそ均一であろうことから、ShareSet 内でこれらの設定をすることは、ゲームの開発を進めるにあたり多くのその他オブジェクトに対して、これら設定を適用する簡便な方法になります。

7. Selectorボタンをクリックし、**Object_Attenuation**を選択します。



Object_Attenuation ShareSet が選択されました。ShareSet内の各パラメータはサウンドに適用されましたが、現状デフォルト値が使用されています。 次にこれらの値を、あなたの好みに応じて調整する必要があります。

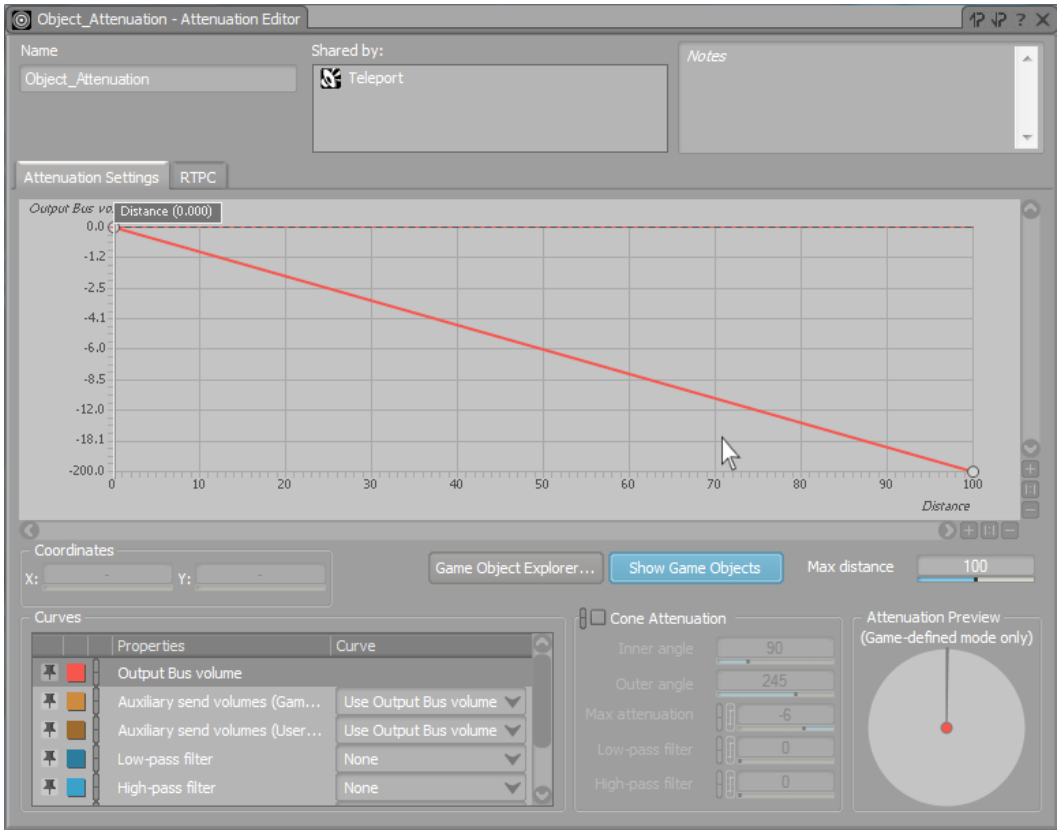
8. Object_Attenuationの隣にある **Edit**をクリックします。



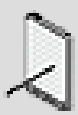
減衰エディター（Attenuation Editor）が開きます。

ここで、減衰エディターが前レッスンで使用したRTPCエディターに類似していることを確認して下さい。基本的には同じものですが、サウンドの特定のパラメータに対してgame syncを設定するのでは無く、オブジェクトとリスナーの位置間の距離が自動的に設定されます。その他にも、距離値によって影響を受ける固定パラメータ群があります。

減衰は、リスナーから遠ざかるに従って、音声信号が自然に減衰するのをシミュレートするのに通常使用します。Wwiseは一連のカーブを使用してボリュームやローパスフィルターなどのWwiseプロパティ値を、特定の距離値に対してマッピングします。これらのカーブを使用して、あなたはサウンドやミュージックオブジェクトの距離に従った繊細なロールオフを作成することが出来ます。



前レッスンで使用したRTPCカーブ同様、赤いラインは操作パラメータが変化するの従い、オブジェクトのボリュームがどのように影響を受けるかを表しています。



注記

減衰カーブは7種類あり、それぞれ色が異なります。

このシナリオでは、サウンドを発するゲームオブジェクトとリスナー間の距離がボリュームを決定します。Wwiseはサウンドが発せられた時点で、エミッターとリスナーの相互のX/Y/Z座標を比較することで距離を判断します。

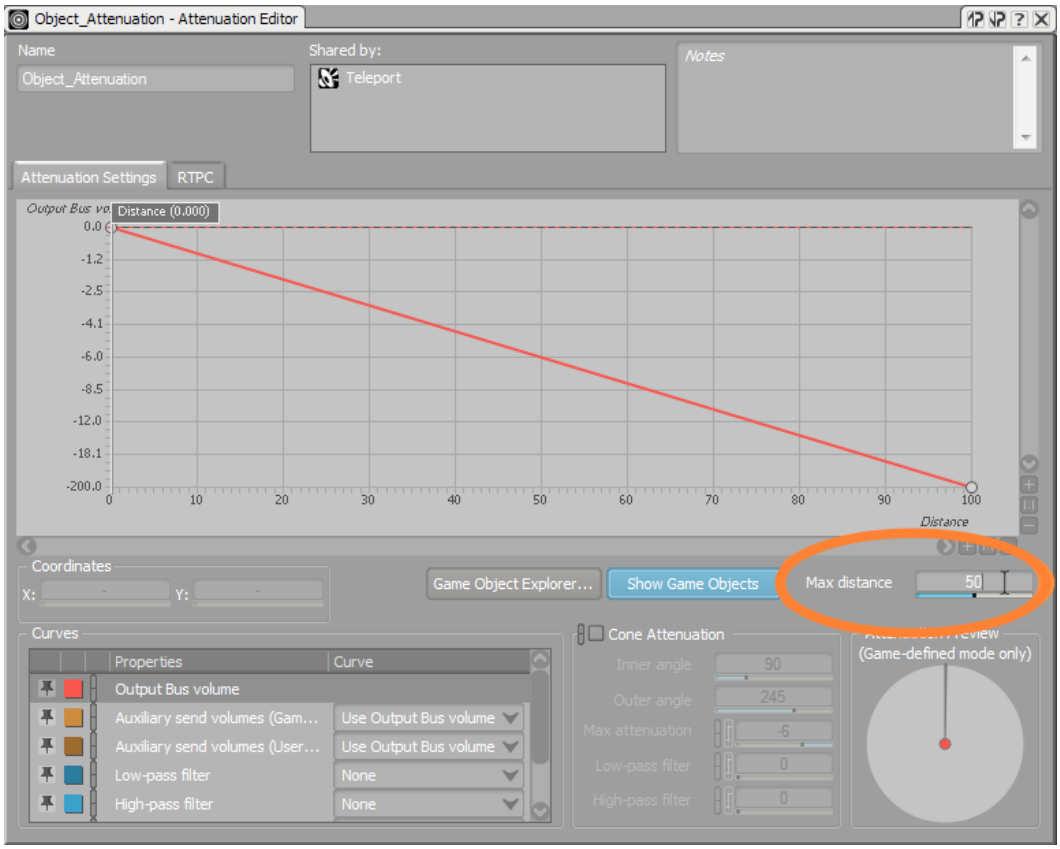
音源が遠ざかる際にサウンドにおいて最も顕著な変化は、より小さな音になることで、デフォルトでは、グラフビューにおいて赤い対角線で示されるように、現在のカーブはボリュームに影響を与えています。

減衰カーブを調整する

減衰カーブを適切に調整するには、ゲーム距離単位が適切な判断を下すことの意味を理解する必要があります。距離単位はゲーム毎に定義され、ゲーム開発の初期に決定されます。例えば、アリになるゲームでは、距離単位はミリメートルになるかもしれません。その一方で銀河系宇宙旅行のゲームでは、距離単位は光年になるかもしれません。Cubeデモでは、距離単位はおよそ0.25メートルになります。

グラフのX軸は距離を表し、グラフに表すことの出来る距離値は最大距離プロパティ（Max distance property）を調整することで、どのような範囲にも対応することが出来ます。

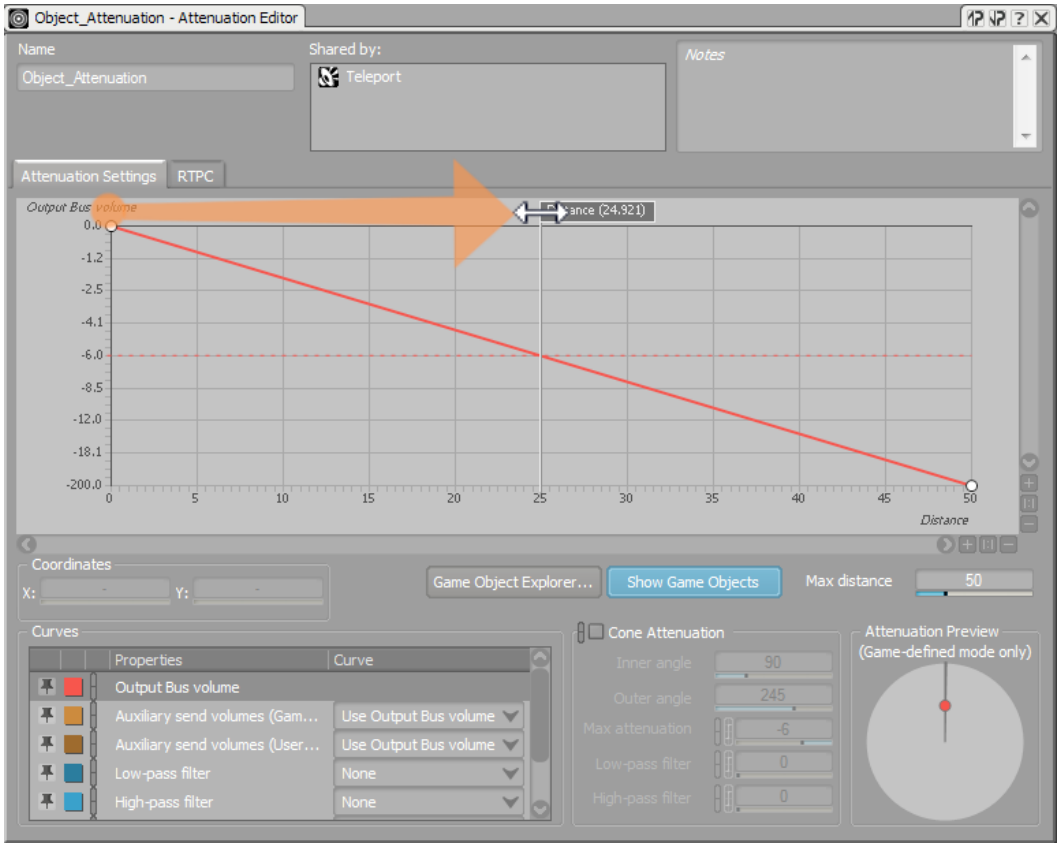
1. 最大距離プロパティを50に変更しEnterを押します。



現在、減衰カーブはサウンドエミッターであるテレポーターと、リスナーであるプレイヤーの間の距離が増加するにつれ、サウンドが小さくなることを示しています。

この距離の変化がどのようにサウンドに影響を与えるか、サウンド再生時にDistanceパラメータカーソルを左右にドラッグすることでオーディション試聴することができます。

2. テレポーターのサウンドを再生し、Distanceパラメータ値をおよそ25程度に調整します。



注記

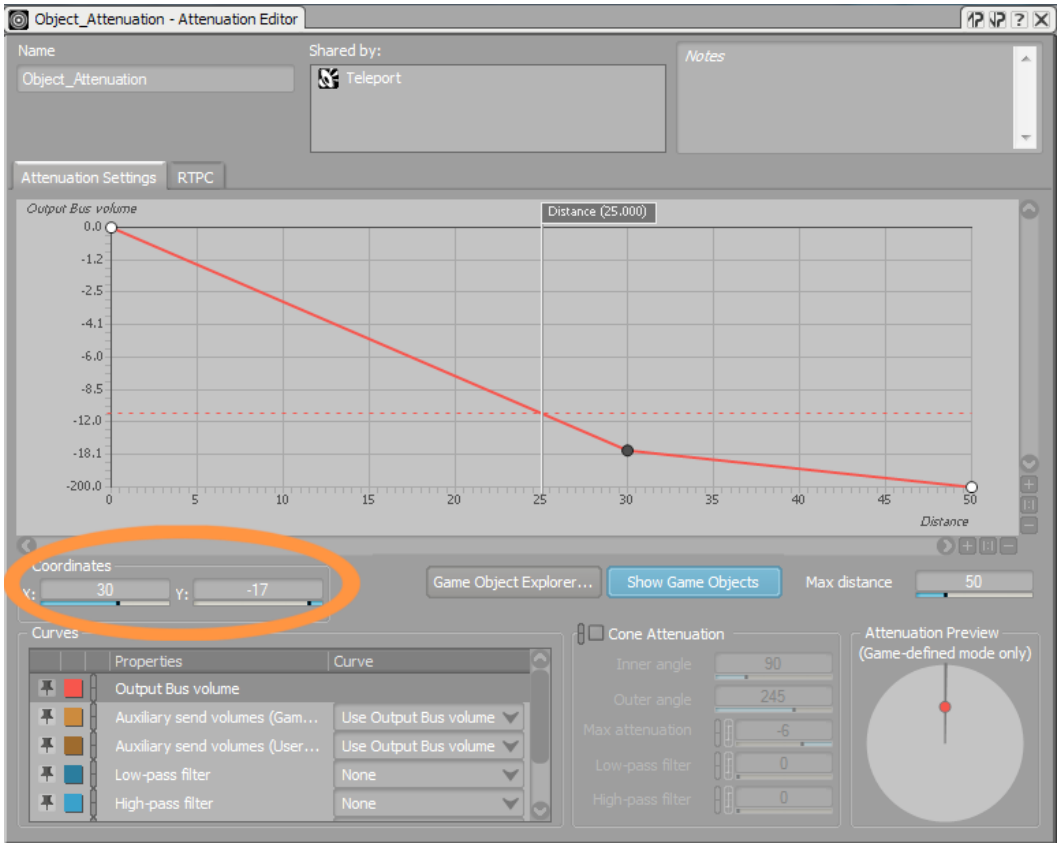
テレポーターのサウンドは無限ループにセットします。これは本演習において、調整をする際にサウンドをオーディション試聴するために繰り返し再生命令を行なわずに済むので、非常に便利です。 再生をストップさせるにはトランスポートの四角いストップボタンを押すだけだと覚えておいて下さい。また、スタートさせるに一度だけサウンドをプレイしてください。プレイボタンを何度も押すと、同時に複数のサウンドのインスタンスが再生されることになります。

RTPCカーブと同様に、減衰カーブの形状も変更することが出来ます。距離が離れた場所ではサウンドが聞こえないように、サウンドを調整し、テレポーターからリスナーが30mの時にボリュームが大幅に減少するようにします。

3. パラメータカーブをダブルクリックして、新しいコントロールポイントを作成し、それを30の距離にセットし、 Output Bus Volume を -17にします。

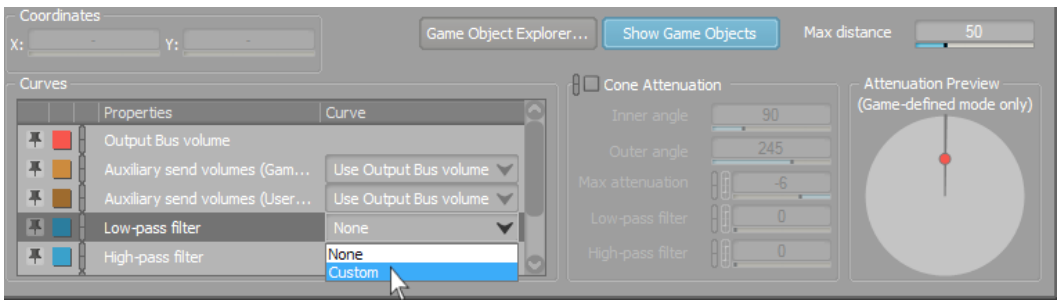
ティップ

X、Y座標値を使用して、正確な値を確認します。

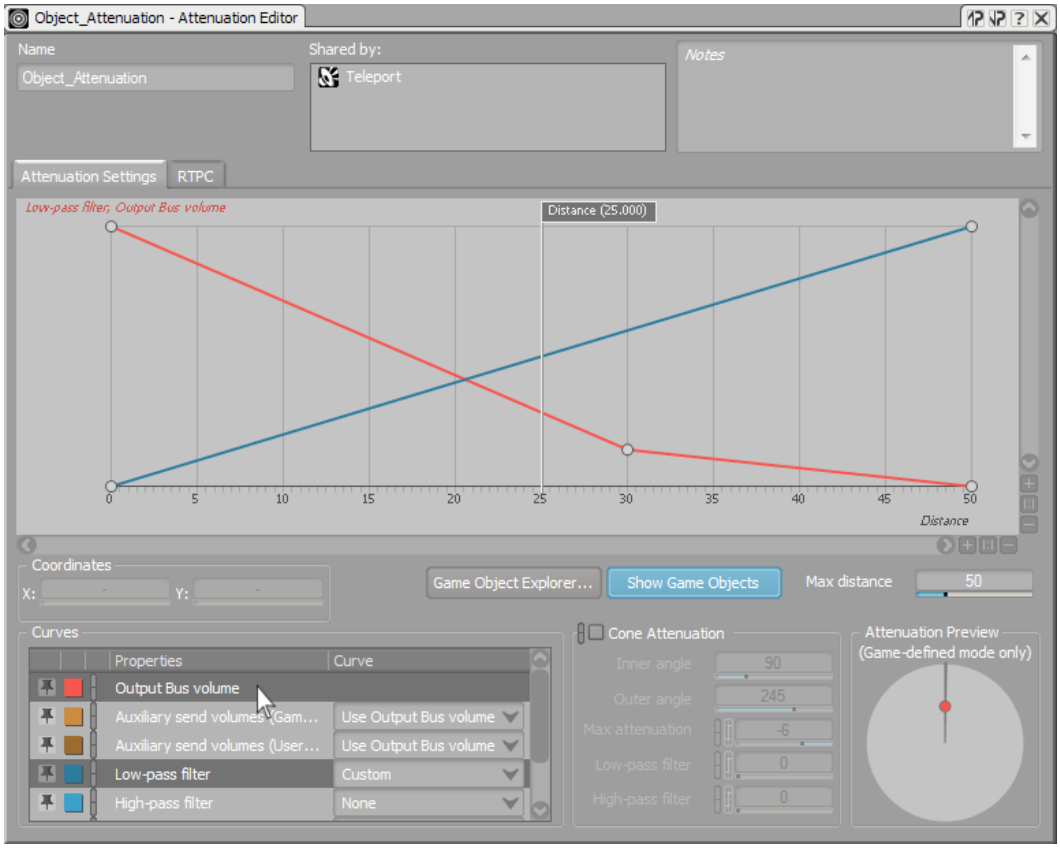


ボリュームの次に、サウンドにおける重要な変化のひとつは、大気中を伝搬するにあたり高周波数帯域が距離に従って減衰することです。これは、距離パラメータをローパスフィルターにマッピングすることで実現します。

4. カーブエリアで、ローパスフィルターを選択し、Curve typeをCustomに設定します。

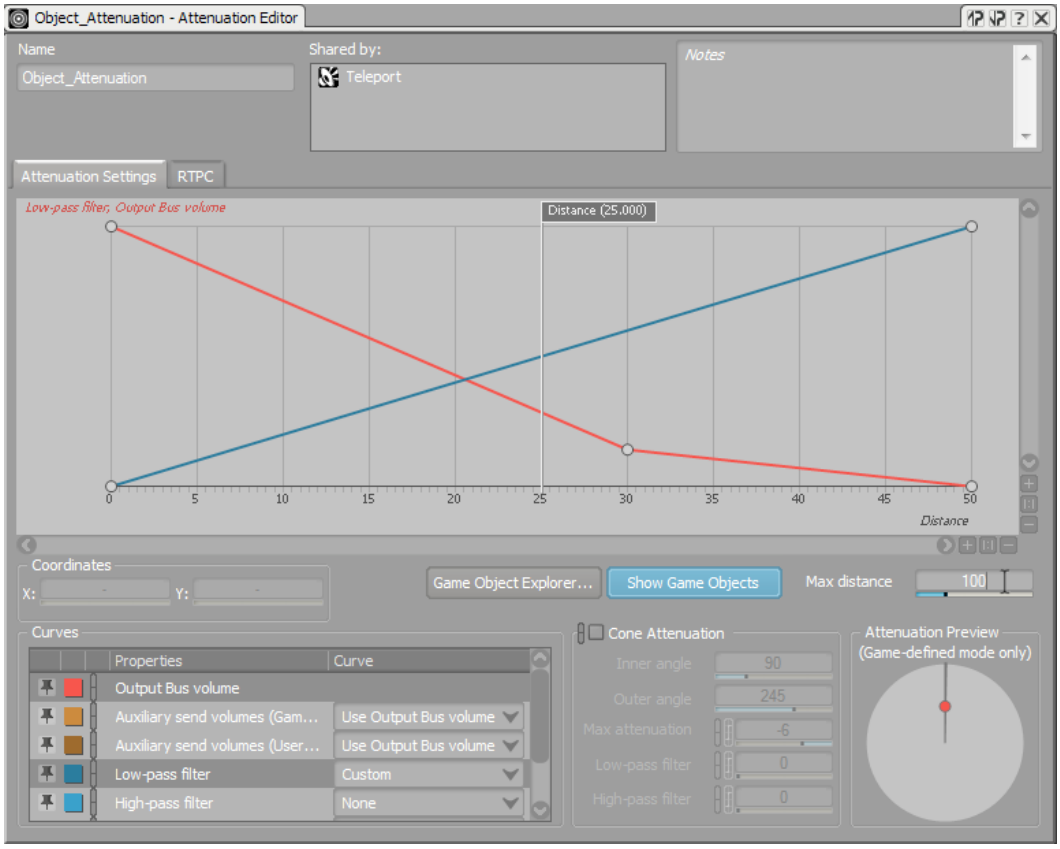


5. **Ctrl**を押したままOutput Bus volume curveを選択し、両方のカーブを表示させ、変更した内容がどのようにサウンドに影響を与えるかテストします。



場合によっては、カーブの形状が決まった後で、カーブ全体のスケールを調整したいと思うかもしれません。カーブを作り直さずに容易に変更が可能です。

6. Max distance値を100に戻します。



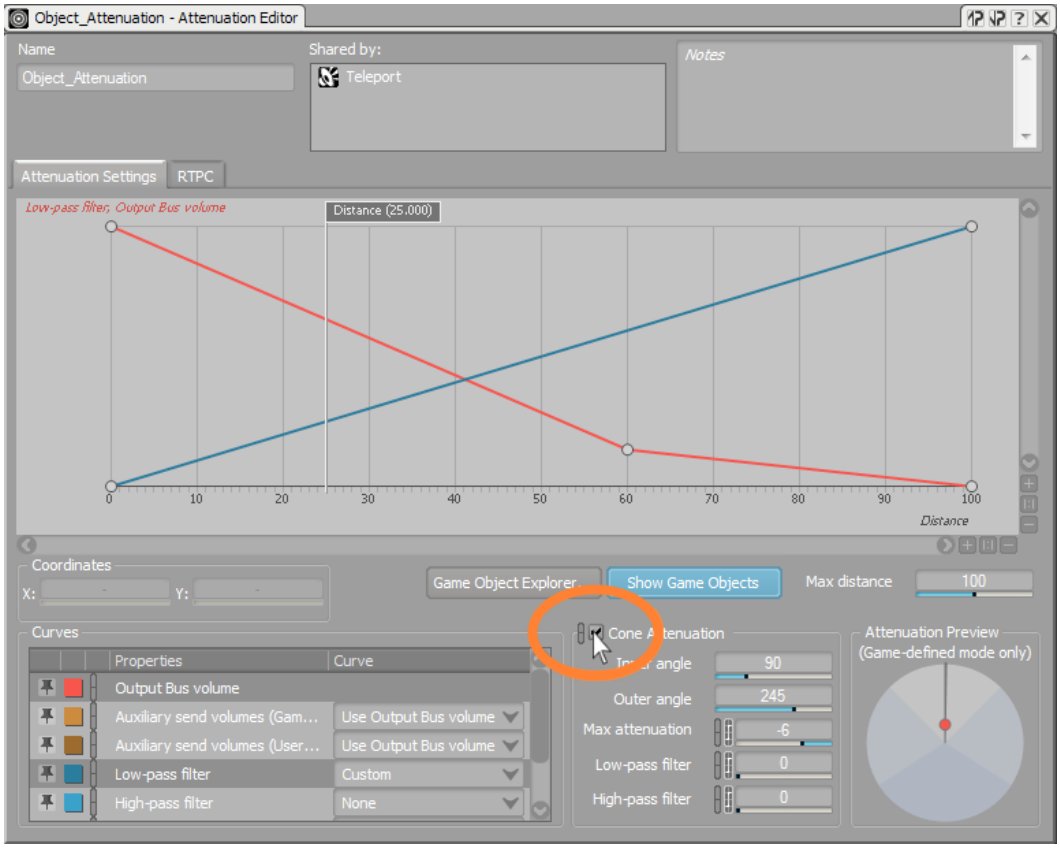
全体のカーブが新しいdistance範囲に対応して自動的にスケーリングされ、リスナーがテレポーターから60mのところでサウンドが低減するのを確認して下さい。

コーン減衰を追加する

距離に対してサウンドがどのように変化するかを考慮することは重要ですが、音源が移動せずに、向きが変わる場合はサウンドはどう変化するでしょうか。例えば、あなたがトランペットのサウンドを聴いていて、そのトランペットプレイヤーは3mの距離であなたの方を向いているのを想像して下さい。もし、そのトランペットプレイヤーが180回転し、背を向けたら、同じように聞こえるでしょうか？大半のトランペットのエネルギーは前方に放出されるため、リスナーが後ろにいる場合にはサウンドはより小さく、よりこもって聞こえるでしょう。音源の向きの変化を考慮するために、Wwiseではコーン減衰を提供しています。

コーン減衰プロパティは減衰エディター(Attenuation Editor)の右下のエリアにあります。減衰プレビューイメージでエミッターとリスナーの関係を表示できるようになっています。

1. コーン減衰（Cone Attenuation）チェックボックスを選びます。



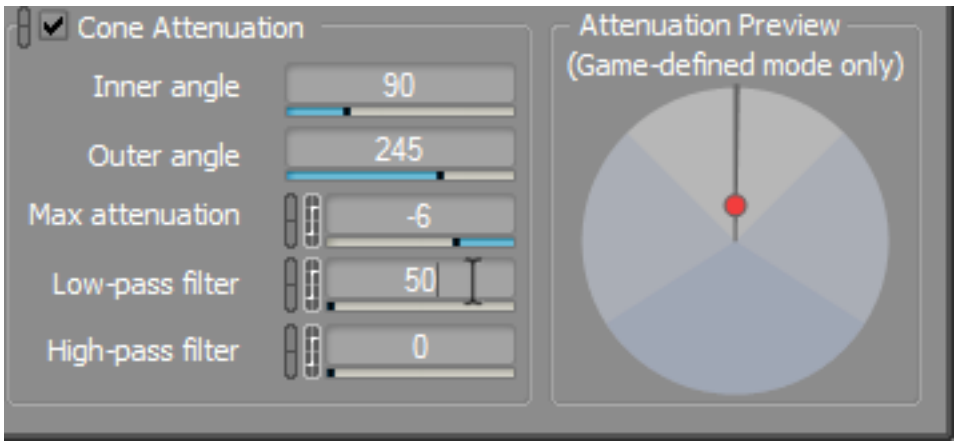
これでコーン減衰プロパティと減衰プレビューエリアが使用できるようになりました。

減衰プレビューでは、ダイアグラム表示のみならず、エミッターとリスナーの関係をシミュレーションできるように設計されています。コーン減数のダイアグラム表示において、リスナーを円の中心に、赤い点がエミッターを表すと考えるのが自然ではありますが、実際には逆になります。ダイアグラムでは、ゲームにおいてサウンドを発するオブジェクトが、円の中心に位置する音源からワールドおいて、サウンドがどのように発せされるかを示します。リスナーの位置は赤い点で示されます。

ライトグレイのエリアはリスナーがフォワード（正面向き）ゾーン内にいるかどうかを示し、サウンドエミッターのプロパティには追加の変更は行なわれません。ダークブルーのエリアはリスナーがエミッターの後部にある場合は、エミッタ—のオブジェクトプロパティが、減衰プレビュー（Attenuation Preview）の左側にあるコーン減衰エリアで設定される追加のコーン減衰プロパティを使用して変化することを示しています。ライトブルーのエリアは遷移エリアを示し、ライトグレイエリアからダークブルーエリアまで徐々にプロパティ値が変化します。

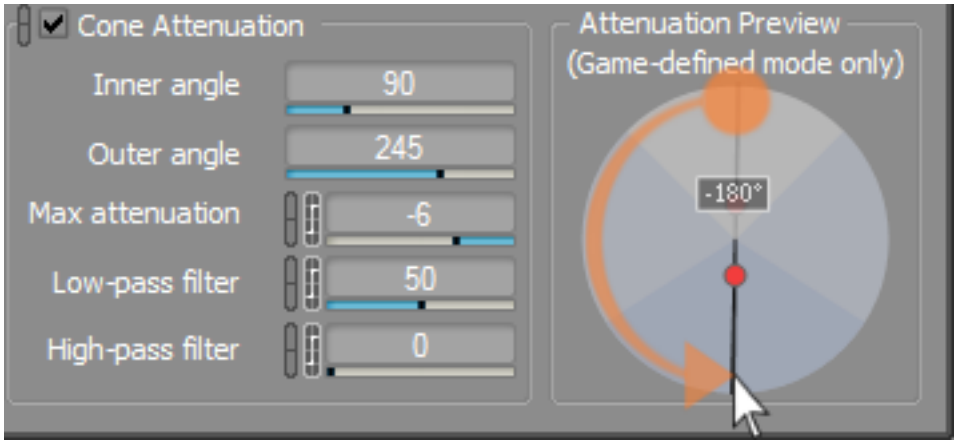
設定を色々と変更して試してみることで、減衰プレビューのビジュアル表示を解釈し、リスナーの位置と相対的なサウンドエミッターの距離と角度の変化を減衰プレビューを使用してシミュレーションを行なうことが容易にできます。

2. ローパスフィルターの値を50に変更して**Enter**を押します。



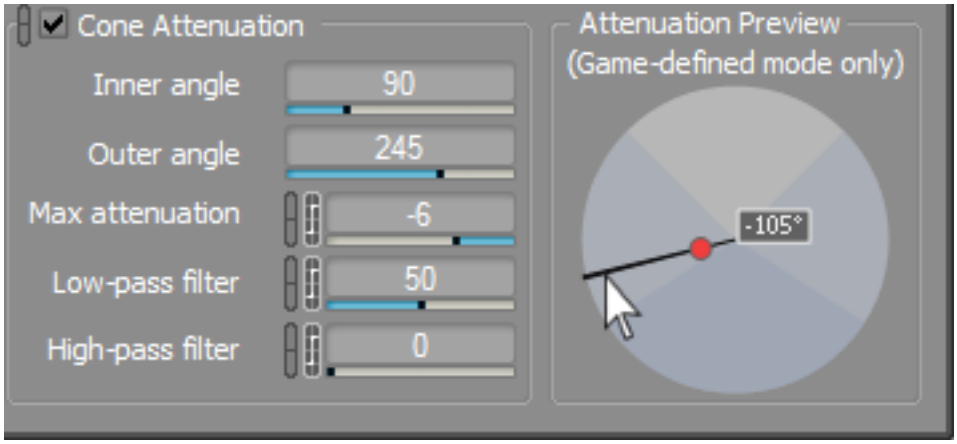
ローパスフィルターに変更を加えましたが、変化を聞くことはできません。なぜならこの値の影響はリスナーがエミッターの後方を向いている際にのみ聞こえるものだからです。従って、リスナーをエミッターの後方に移動させなければなりません。

3. 減衰プレビュー内で、黒い線を反対側へドラッグします。



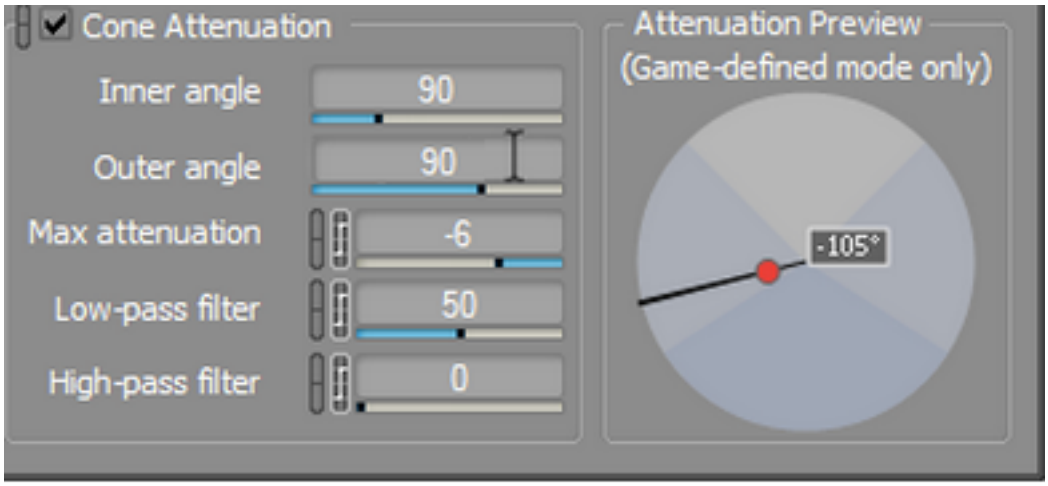
これでローパスフィルターの効果がはっきりと聞こえると思います。また、点をエミッターの後方に振り、ライトブルーのエリアを通過する際に、フィルター効果の増加があったことに気づかれるかもしれません。

4. 黒い線を-105° にドラッグします。

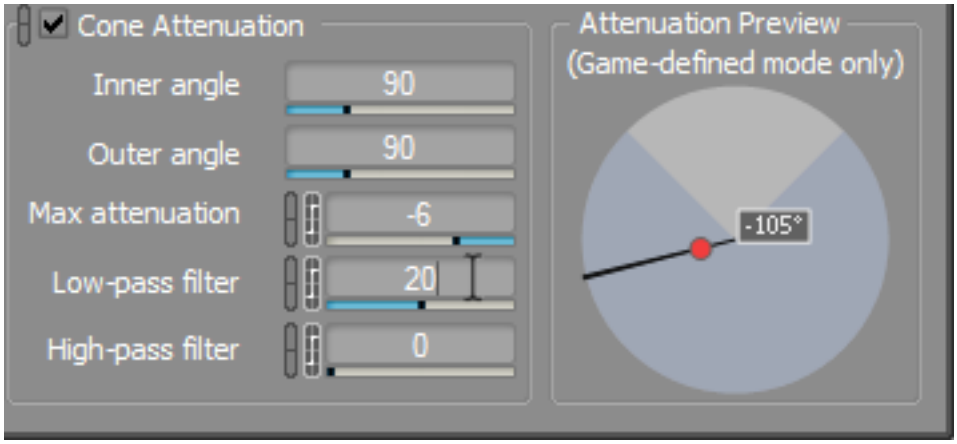


これでリスナーは遷移ゾーンにあります。遷移ゾーンのサイズは、Inner もしくはOuter angle プロパティを変更することでカスタマイズすることができます。

5. Outer angle 値を90° に変更し**Enter**を押します。

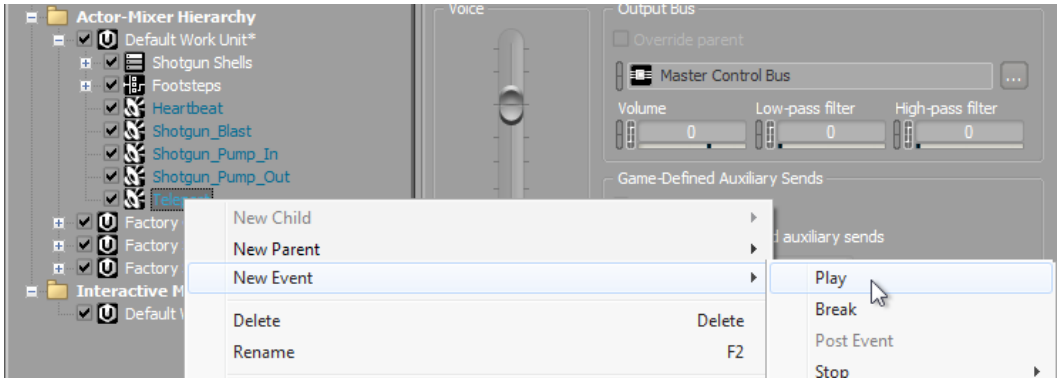


6. ローパスフィルター値を 20に、もしくはお好みの値に設定します。

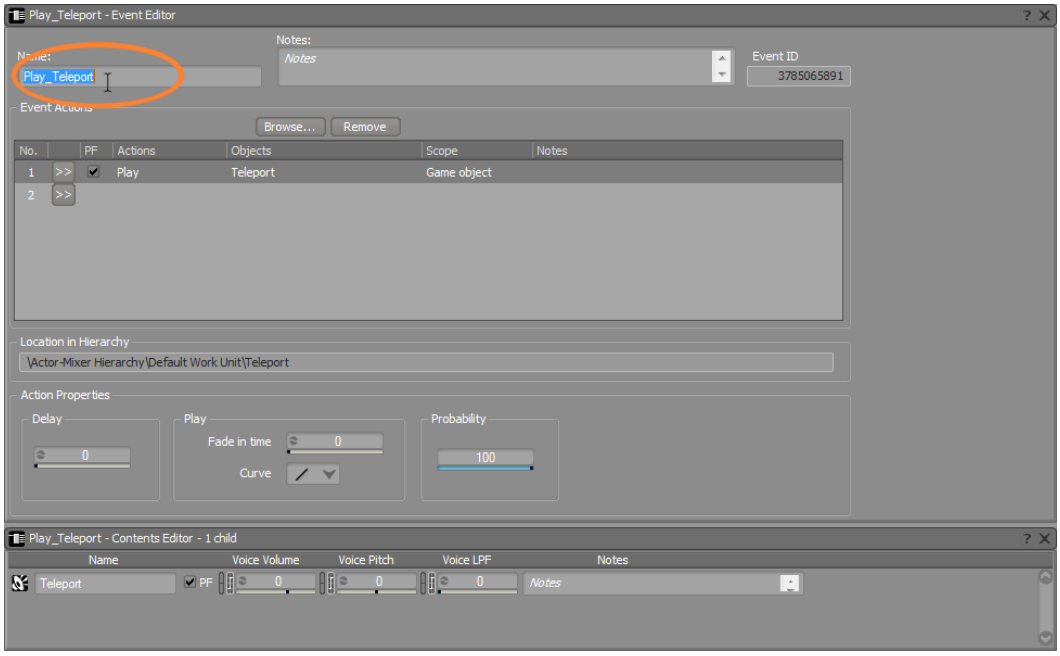


SFXオブジェクトの設定は完了しているので、次にゲームでそれが聞こえるようにイベントが必要です。

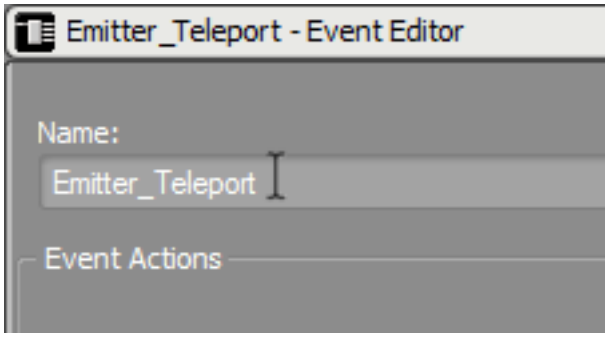
7. 減衰エディターを閉じ、テレポートSFXオブジェクトを右クリックし、**New Event > Play**を選択します。



イベントのNameフィールドが既にハイライト表示されていますので、そのまま入力ができるようになっています。



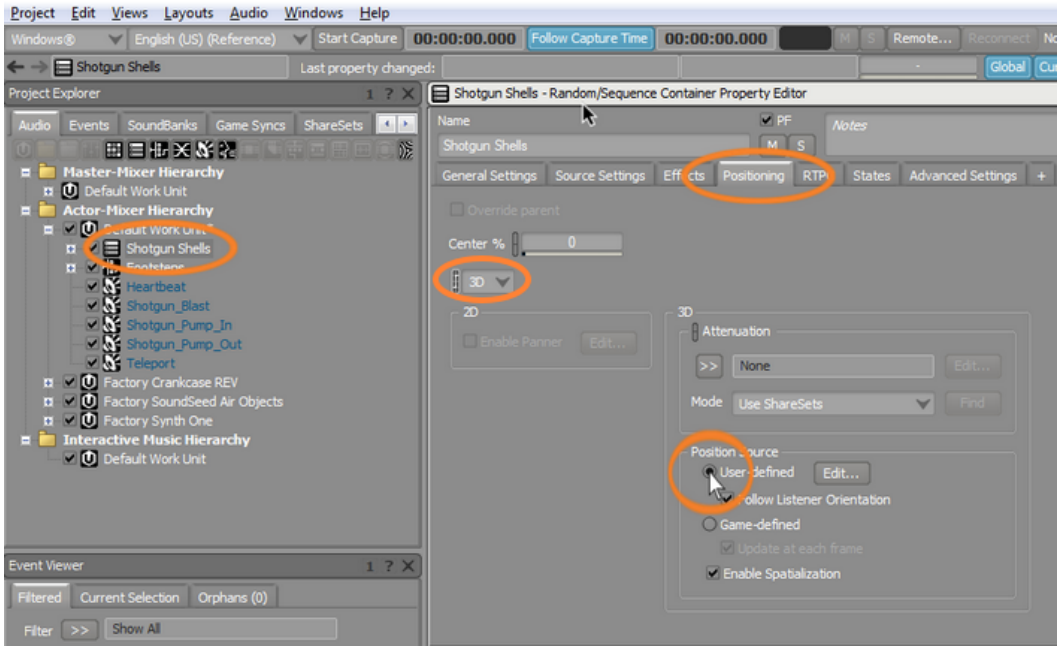
8. このイベントを Emitter_Teleport とし**Enter**を押します。



3D User Definedを理解する

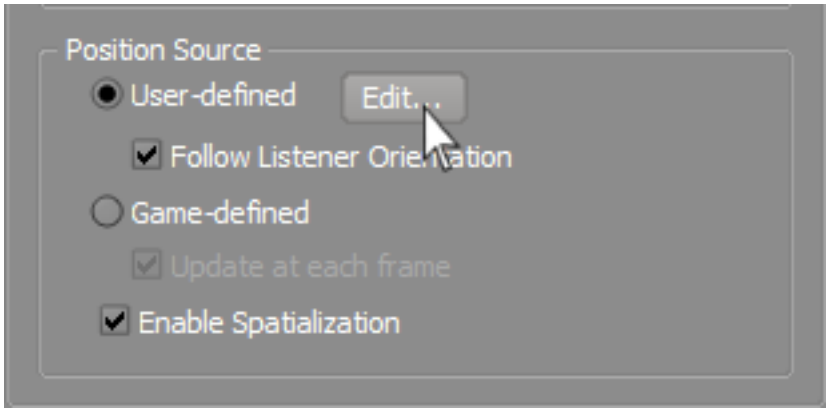
レッスン2であなたが追加したショットガンのシェルのように実際のゲームオブジェクトとは関連付けがされていないサウンドではどうなるでしょうか?ショットガンがシェルを排出する際に、右側へ落ち、地面で跳ねて、最終的にショットガンから数メートル先にいったとします。ここでリアリズムを生むために、あなたはゲームプレイ中にシェル音を右側からリスナーに聞かせたいと考えますが、ショットガンのシェルに対するゲームオブジェクトが存在しないので、前レッスンで学んだような3Dワールドにおいてその音をWwiseがポジショニングするための座標情報がありません。幸いなことに、Wwiseでは対応するゲームオブジェクトを持たないサウンドに対しても3Dワールド内でサウンドオブジェクトのポジショニングをシミュレーションすることが可能です。これをユーザー定義の3D音源（User-defined 3D sound source)と呼んでいます。

1. プロジェクトエクスプローラーのAudioタブで、ショットガンのシェルを選び、Property Editorで、Positioningタブを選びます。2Dから3Dへ変更するには、Position SourceエリアのUser-definedラジオボタンを選びます。



User-definedを選ぶことはこのプロセスの第一歩です。 次に、Wwise Position Editorを使用して、3Dワールド内でリスナーの位置に相対的にどこにサウンドを配置するかの具体的な情報を入力します。

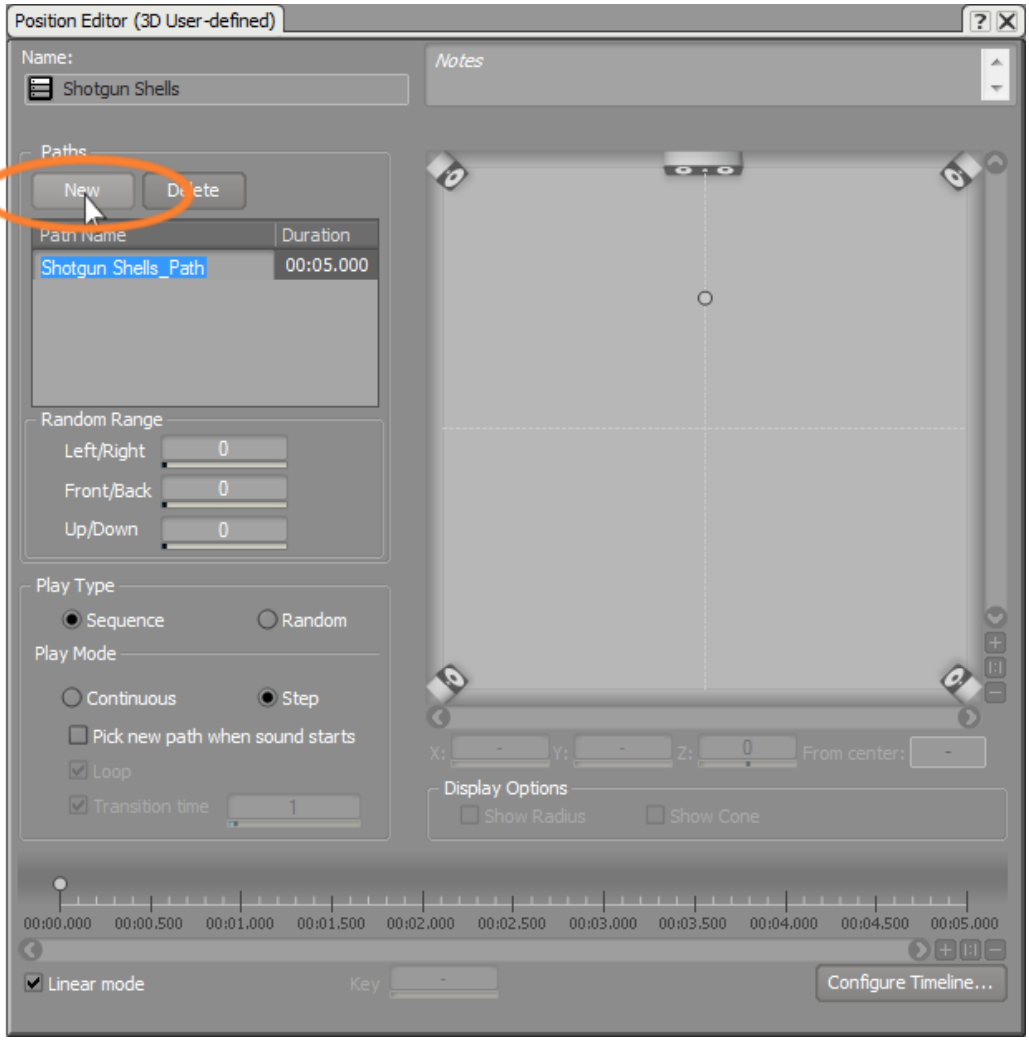
2. 今、選択したUser-definedラジオボタンの隣にある**Edit** をクリックします。



Position Editorウインドウが開きます。このエディタは非常にパワフルかつ機能豊富なツールで、ゲームプログラマーの介在無くとも、3D空間内に高度にカスタマイズ可能で、ダイナミックなサウンドの動きを作成することができます。頭の上を鳥たちが飛んだり、古い建物が軋んだりゴトゴト音を立てるような環境空間を作成するのには素晴らしいツールです。このPosition Editorを使用して、ショットガンのシェルのサウンドに床面を跳ねて移動する位置的な変化を追加することで、リアリティの高いものにします。

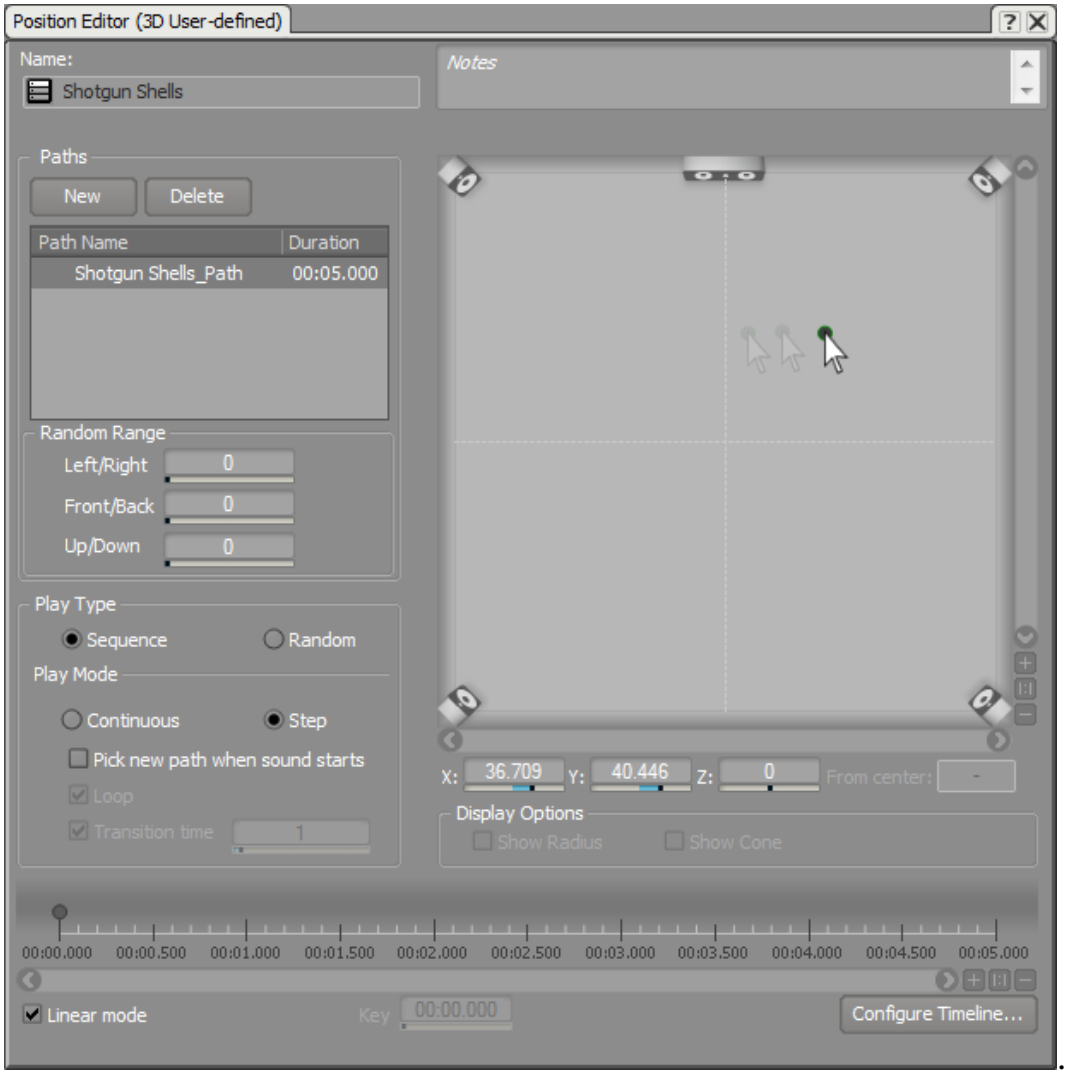
Position Editorはサウンドオブジェクトが追随するユーザー定義の軌道であるpathsを使用して音源のダイナミックな動きを実現します。

3. Pathsエリアで **New**をクリックします。



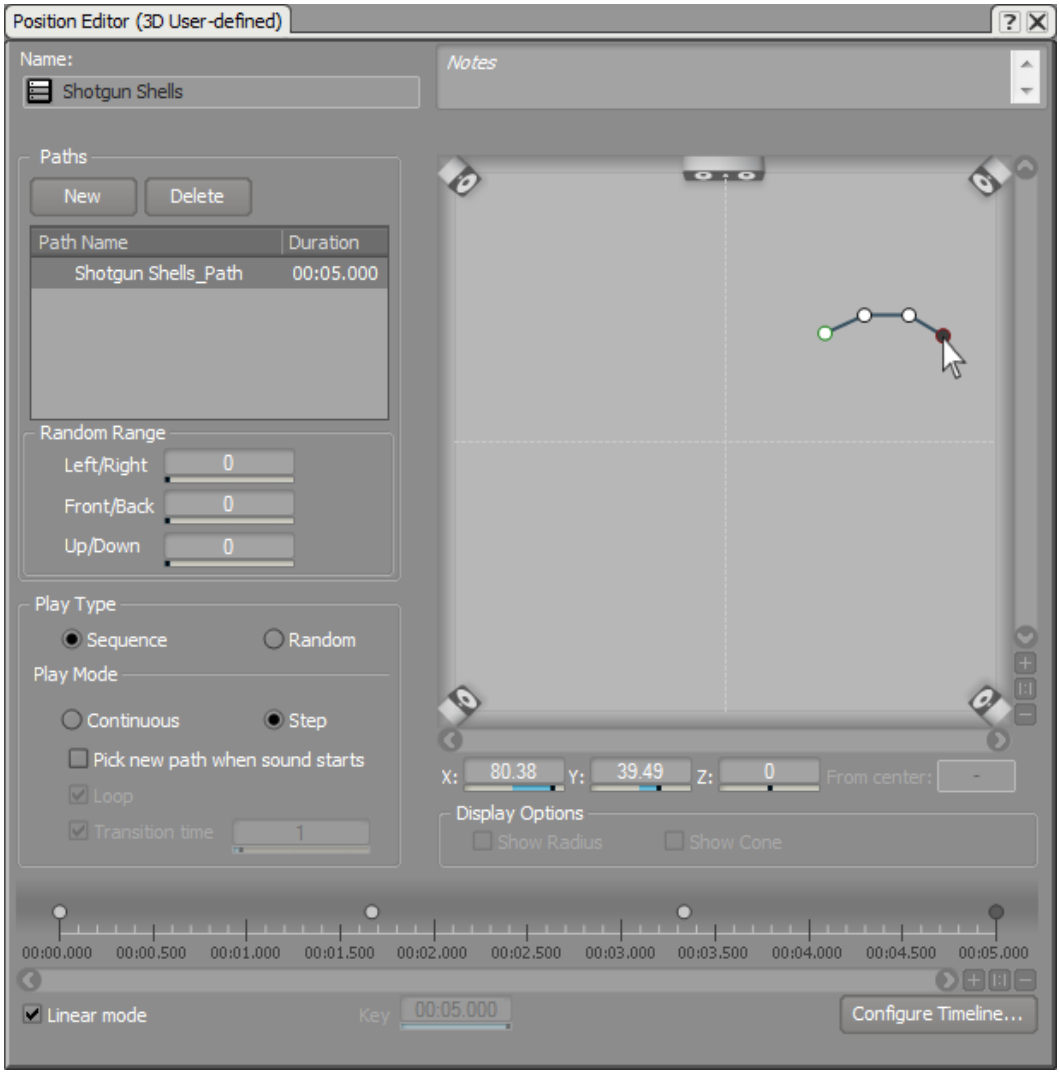
ディスプレイ内に点が現れ、四角の中央にあるリスナーのポジションに関係し、サウンドが発せられる場所を示します。シェルはショットガンの右側から排出されるので、この位置を右側へと移動します。

4. コントロールポイントを右にドラッグします。



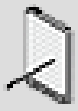
シェルは地面で跳ね続けるので、ビューをダブルクリックし、シェルがたどるパスを作成するための追加ポイントを加えます。

5. エリアの右側に追加のコントロールポイントを作成します。



点群は、ラインで接続され、3D空間でたどるパスを示します。

また、Position Editorの下部に、タイムラインがあり、ディスプレイ内に点を追加する度に、対応する点がタイムライン上にも生成されていることにお気づきかもしれません。このタイムラインはゲーム内のヴァーチャルオブジェクトがある点からある点まで移動するのにかかる時間を示しています。

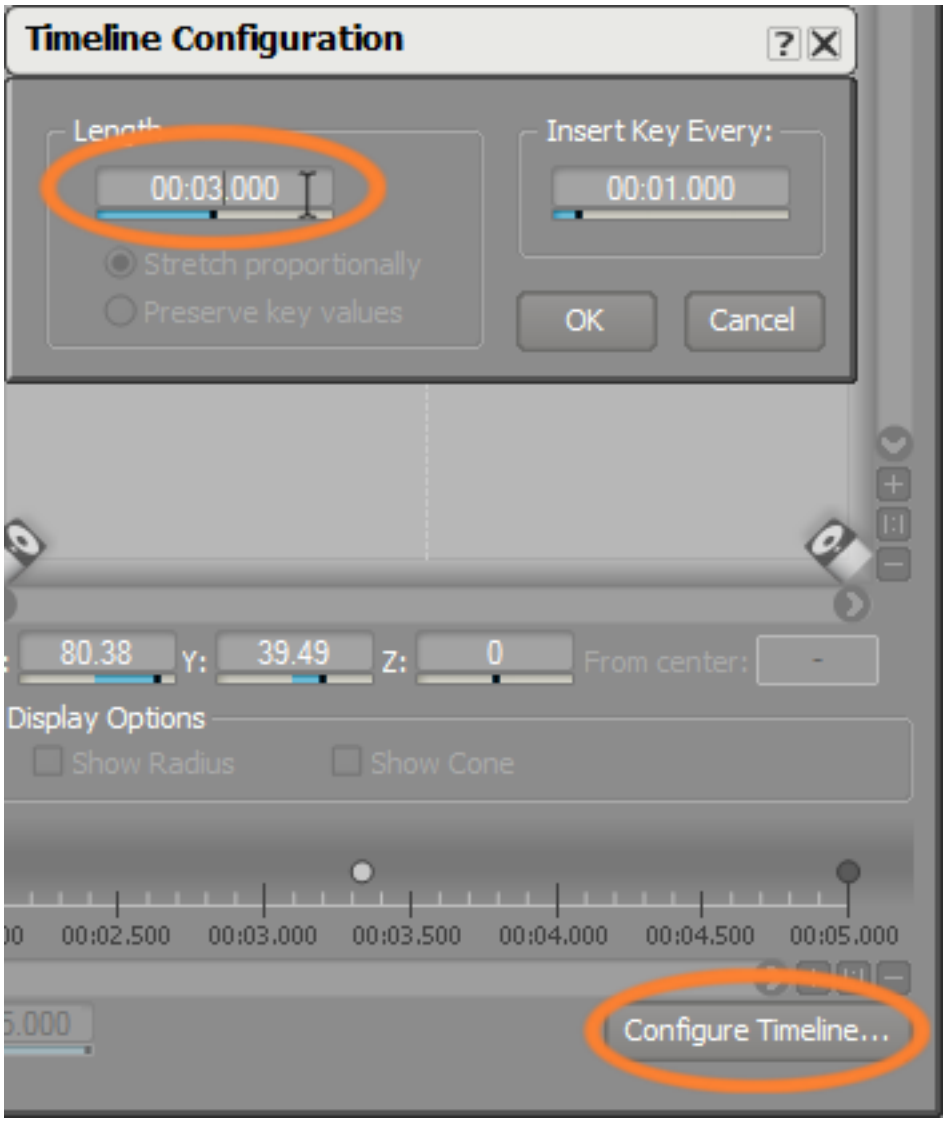


注記

Wwiseの現行のMac版では、タイムラインが表示できないという制限があります。これはWwiseの今後のバージョンで修正されます。タイムラインが表示されなくても、レッスンの手順に影響しません。

跳ねるシェルの音は、落下時に数秒程度しか続かないので、実際にシェルの音が聞こえる長さにタイムラインの長さを合せます。

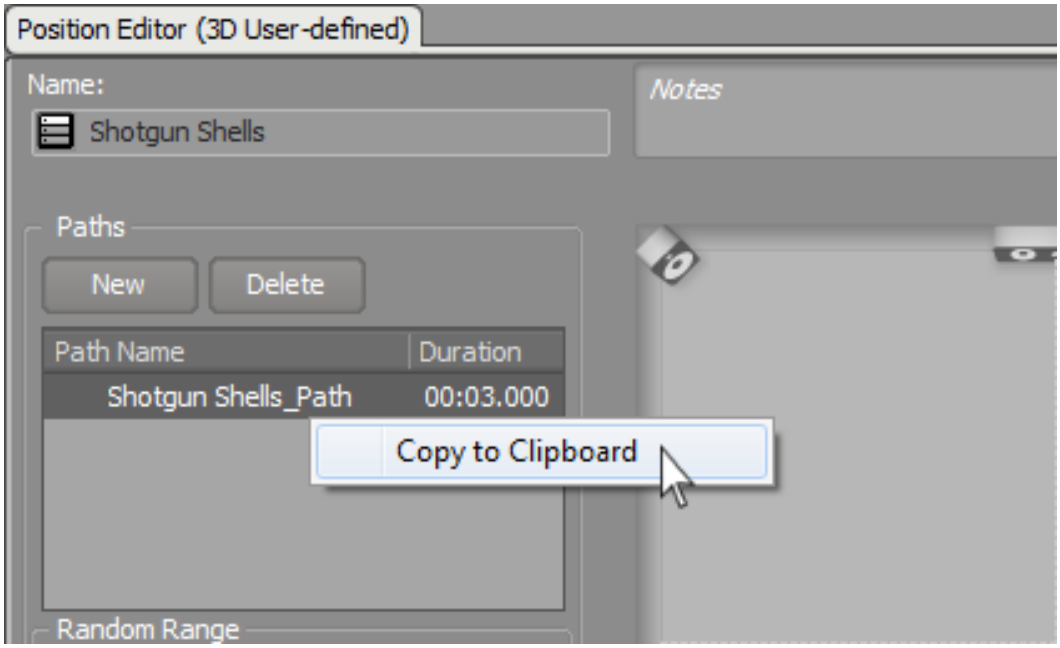
6. **Configure Timeline** をクリックしLengthプロパティを3秒に変更し**OK**をクリックします。



現実的には、地面に落ちるシェルが跳ねる際に、常に同じ物理的な軌道に沿って移動するわけではありませんので、リアリティを増すために、それぞれのシェルが異なるパスを使って跳ねるように、複数のパスを作成します。

新規パスをはじめから作成するのではなく、既存のパスを複数回コピーし、これらのパスに変更を加えることで、シェルが跳ねる様々な軌道を作成します。

7. Shotgun Shells_Pathを選択した状態で、右クリックし Copy to Clipboardを選択します。

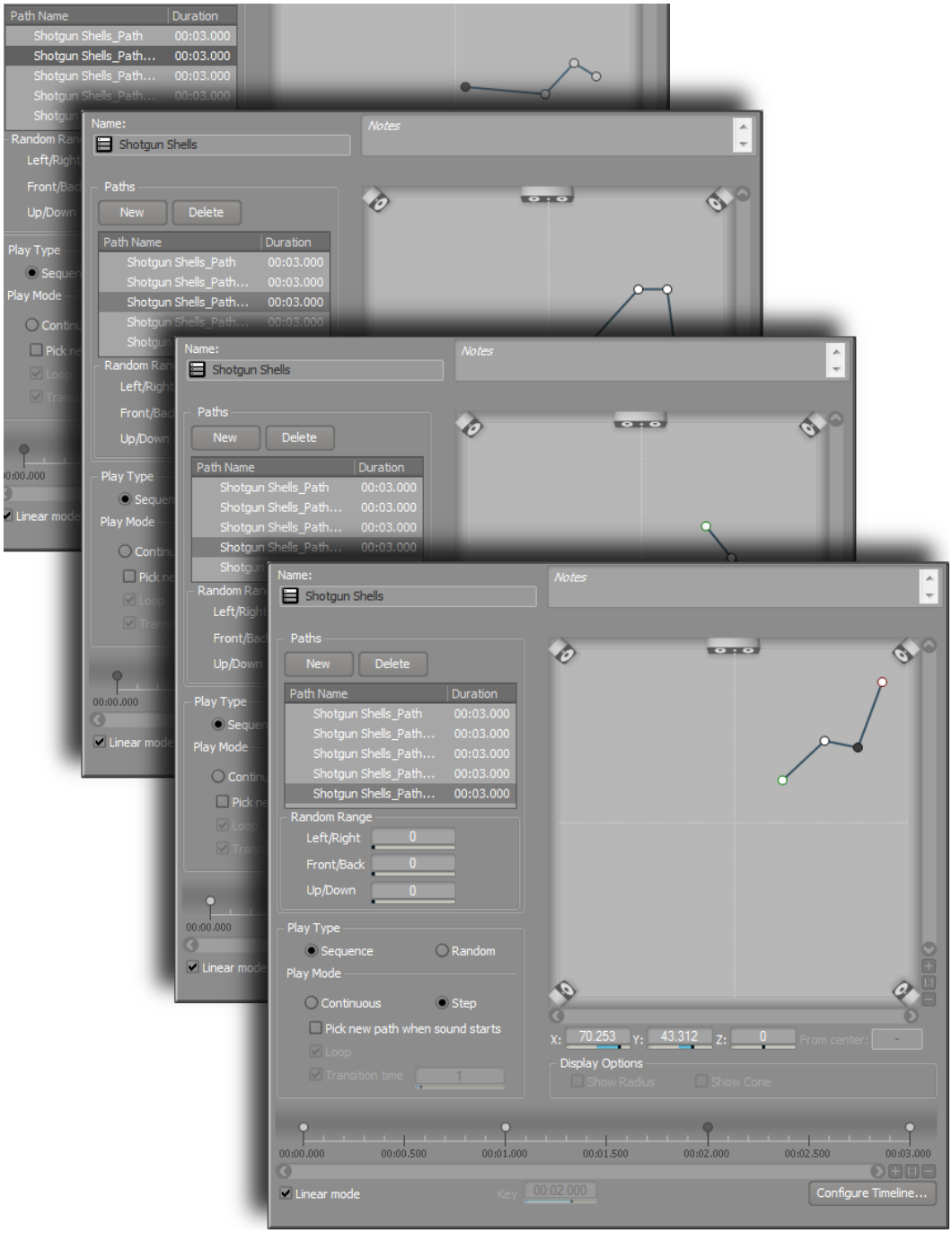


ティップ

Pathsリストからパスを選択し、CTRL-Cでコピーすることができます。

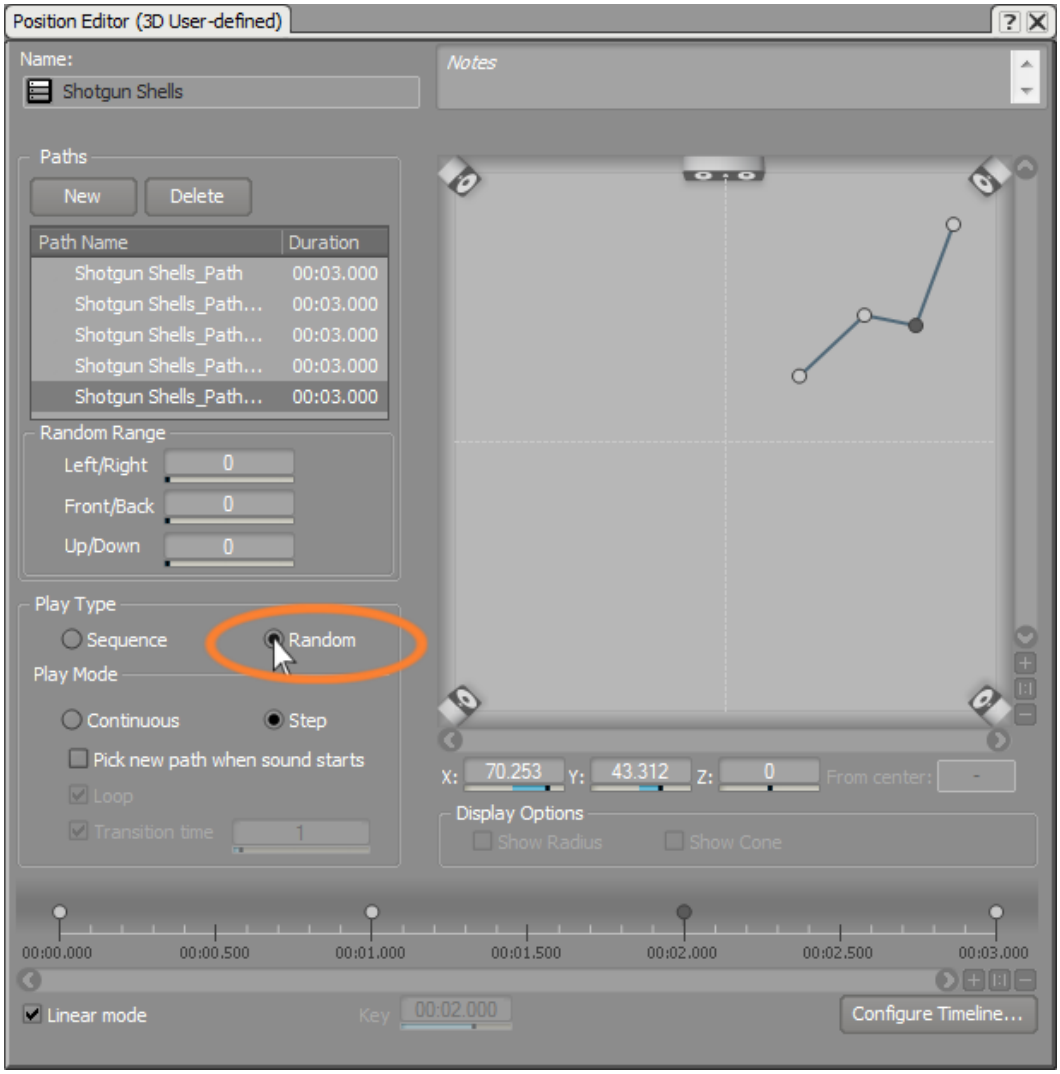
現在のパスがクリップボードへコピーされます。

8. **Ctrl** +Vを数回押して、６つの新規パスを作成します。
9. パスを選択し、点の位置を変更します。同一なパスが無くなるまで繰り返します。



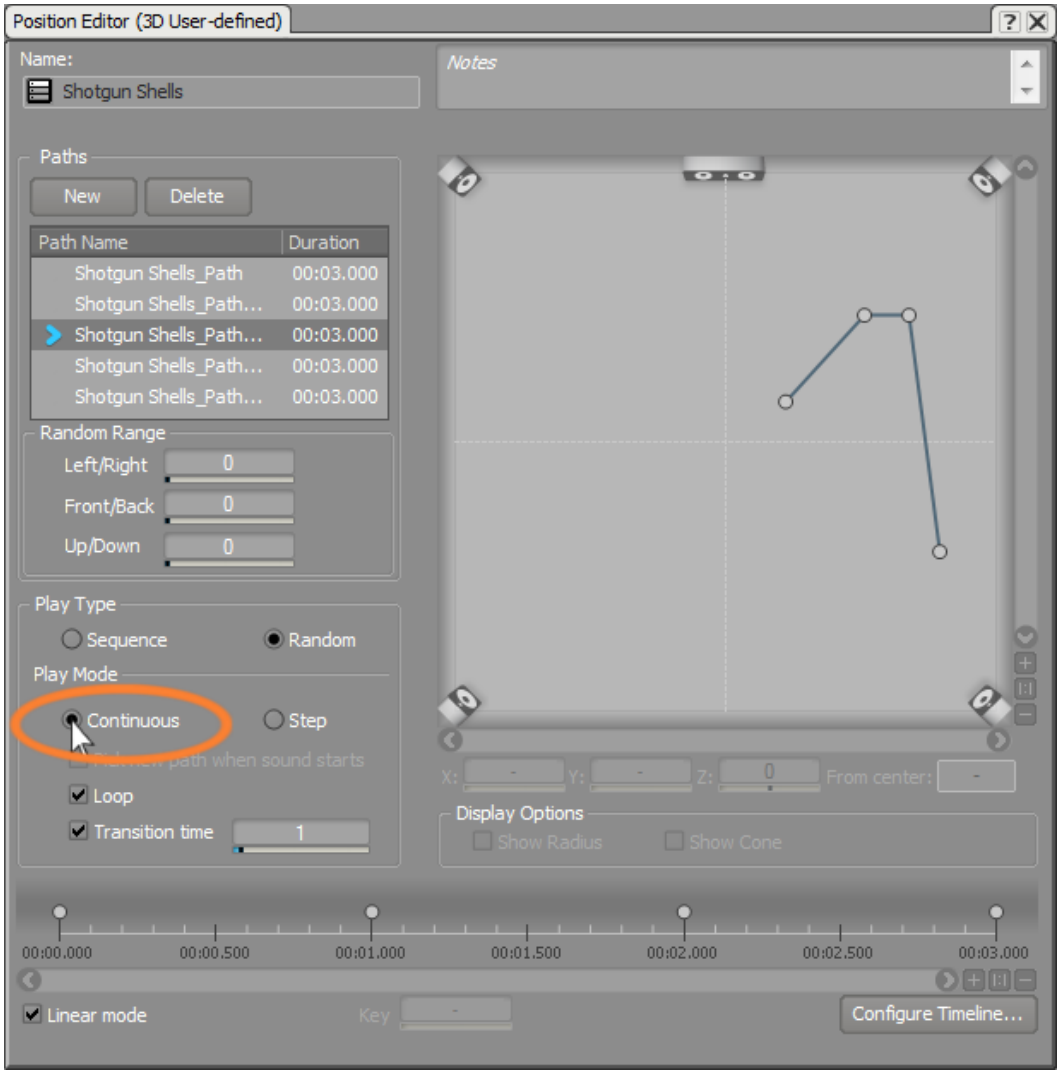
もうすぐで準備完了です。ここでいくつかの調整を行ないます。まずシェルが地面に落ちる度に、任意のパスがランダムに選択されるように指定します。

10Randomラジオボタンをクリックします。



デフォルトでユーザー定義のパスが設定されたオブジェクトがプレイされる場合、指定されたパスの最初のポジションからプレイされます。次回プレイされた際には、そのパスの次のポジションへ移動し、このように続きます。シェルに対しては、シェルオブジェクトがプレイされる際にオブジェクトの動きがパスに沿って継続して移動するようにし、複数のシェルが地面に落ちる継続的な動きを効果的に作成します。

11Playモードエリアで、 **Continuous**を選択します。



12Position Editorを閉じて、ショットガンシェルをプレイし、作業の結果を聞いてみましょう。

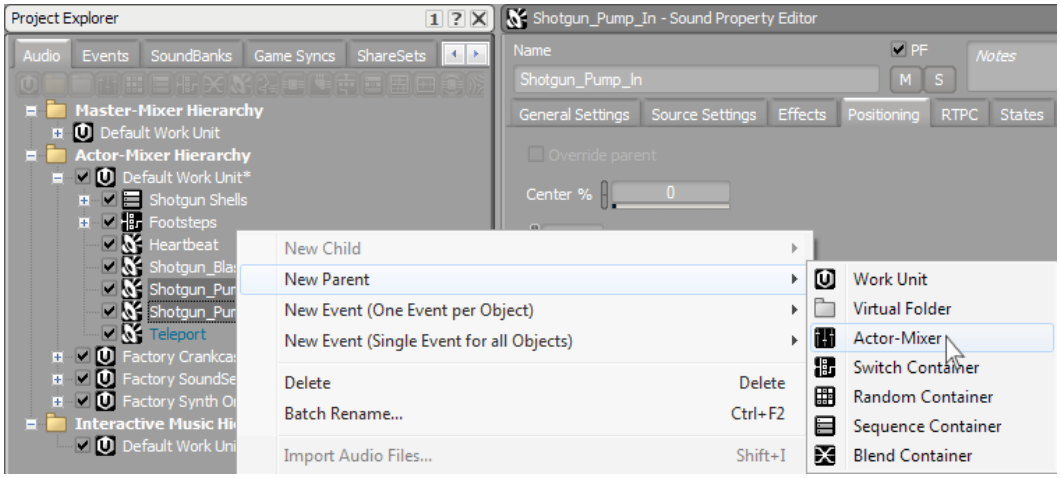
2Dパンニングの使用

サウンドのポジショニングにおいて3Dでの扱いが不要な場合があります。 これは実際のゲームオブジェクトとして表されない要素として、音楽やナレーションなどがまさにこれに当てはまります。例えば、ゲームのサウンドトラックを追加したいと思うかもしれませんが、キューブデモの3Dワールド内でロックバンドとして登場させるのはありそうもないことです。しかしながら、どのスピーカーから音が出力されるかの制御は、より一般的なアプローチとして出力先のスピーカーに対してサウンドのパンニングで行なうことが出来ます。例として、ナレーションをサラウンドスピーカーのLとRに配置することでリスナーの後方から聞こえるようにしたり、音楽をフロントスピーカーのLとRのみから聞こえるようにしたりします。エミッターの3Dワールドにおける配置を考慮しない、従来の手法を使ったサウンドのポジショニングは、2Dパンニングと呼ばれています。

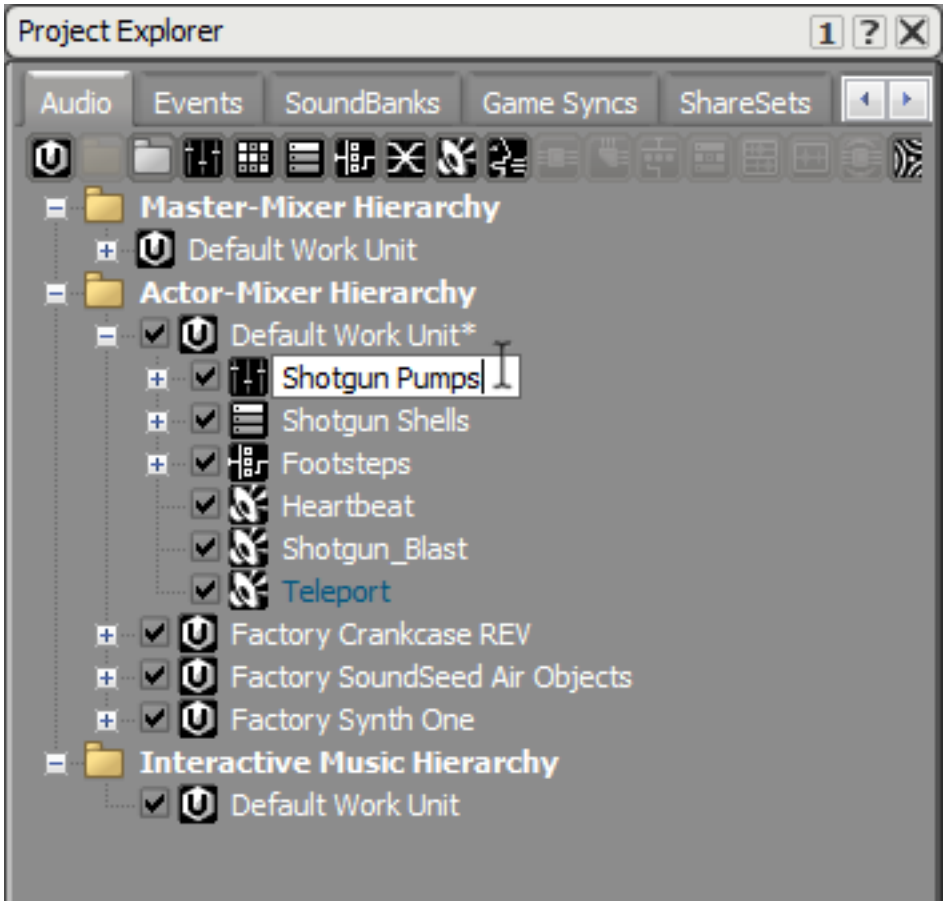
2Dポジショニングは、ゲーム内でヴィジュアルオブジェクトに付随するサウンドに使用することができますが、通常この手法はこれらオブジェクトがプレイヤーの視点に対して同じ位置に見える際に使用されます。 例えば、キューブデモにおいてプレイヤーがショットガンをポンプアクションで装填する度に、そのアニメーションはプレイヤーの目の前で、かつ多少右寄りになっています。2Dパンニングを使用して、ショットガンのポンプ装填サウンドにワールド内において多少なりとも独自のスペース感を表現できます。それ自体はキューブデモの中では実際のゲームオブジェクトとして存在するわけではないのですが。

実際には2種類のショットガンのポンプアクションサウンド、pump_in と pump_outがあるのですが、これらを一つのActor-Mixerでグループとして扱います。Actor-Mixerオブジェクトはあなたのプロジェクトを整理するフォルダとして機能しますが、それに包含される全てのオブジェクトに対してプロパティ設定を素早く反映させる機能も利点として備えています。ショットガンポンプアクションの場合では、一般にボリュームやパンニングなどを均一に調整したいと考えるでしょう。これらをActor-Mixer内に格納することで同じ調整作業を繰り返す必要がなくなります。

1. プロジェクトエクスプローラー内で、Shotgun_Pump_In と the Shotgun_Pump_Out オブジェクトを選択し、右クリックし**Actor-Mixer**を選択します。

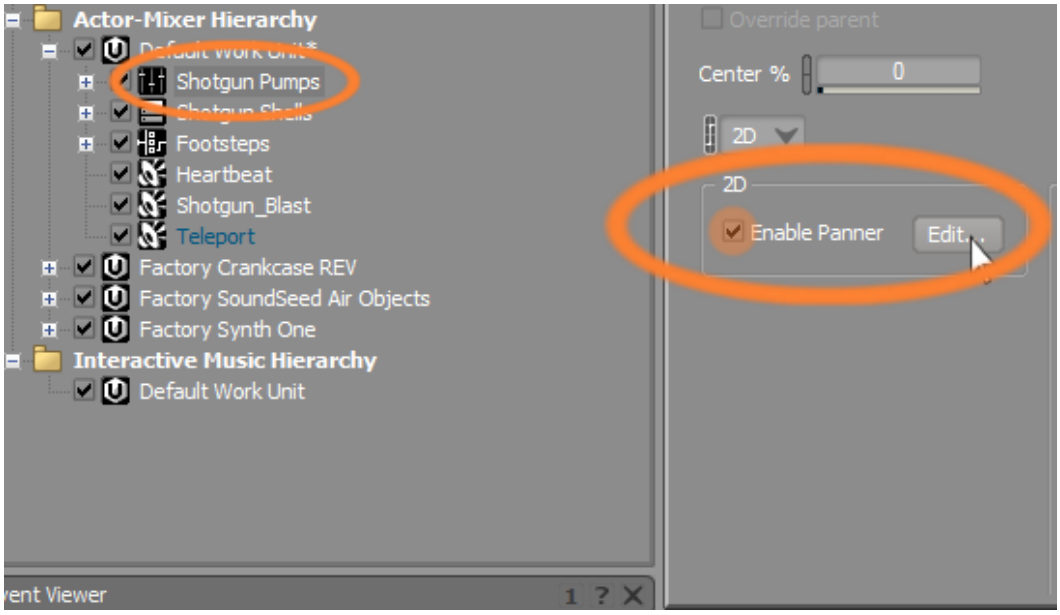


2. このActor-MixerをShotgun Pumpsと名付けます。

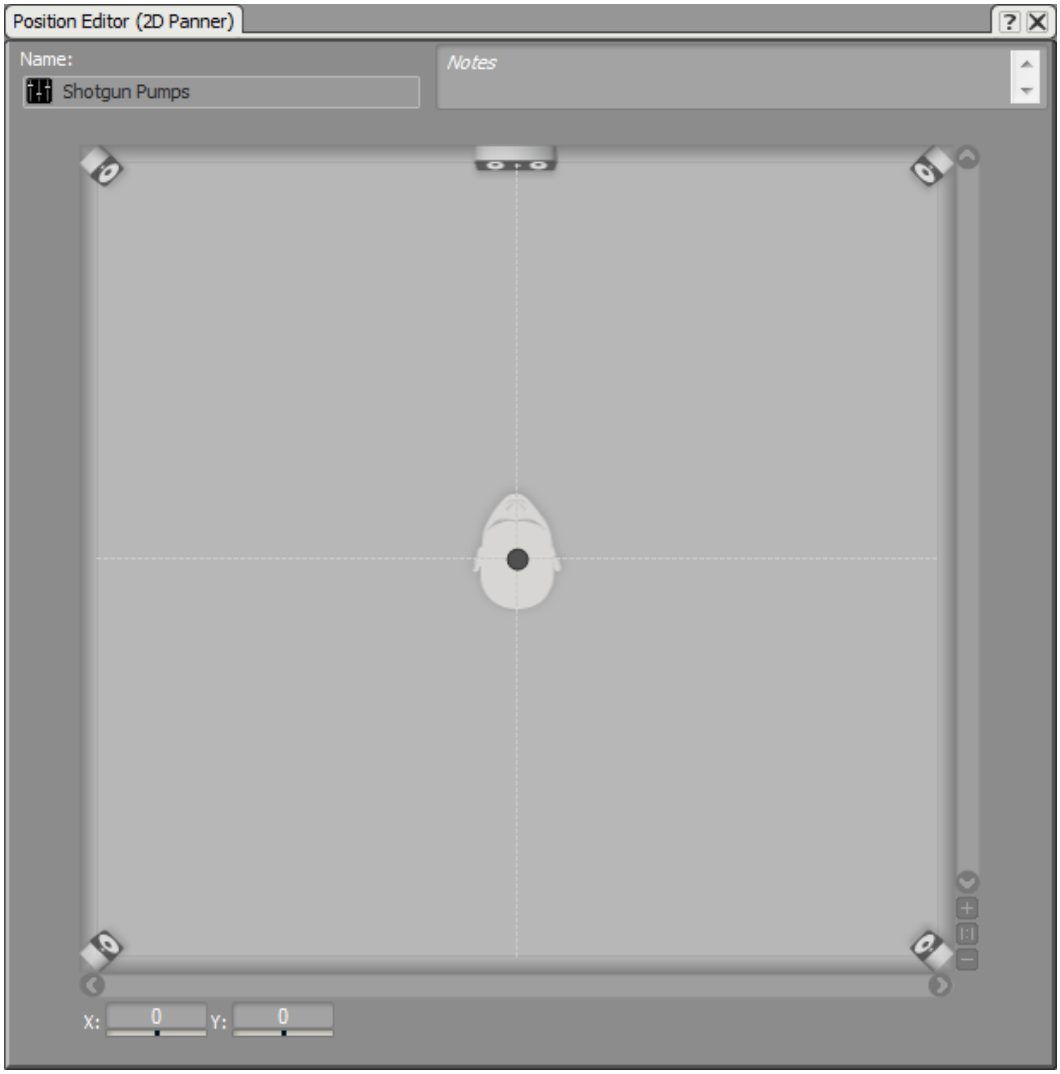


デフォルトで、2Dサウンドのソースチャンネルは共にリンクされ、リスナーやゲームオブジェクトの位置や向きに拘わらず、フロントのLとRのスピーカーから再生されます。更に柔軟性を持たせるために、Wwiseでは各チャンネルのボリュームのバランスをとるために 2Dパンナーを提供しており、サウンドがサラウンド再生環境においてフロントとリアのスピーカーから聞こえるようにすることができます。

3. Shotgun Pumpsを選択した状態で、2Dエリア内のEnable Pannerボックスを選択し、Editボタンをクリックします。



2D Position Editorが開き、一般的な5.1chサラウンドサウンド環境の部屋の中央に人間を上からみた図が表示されます。

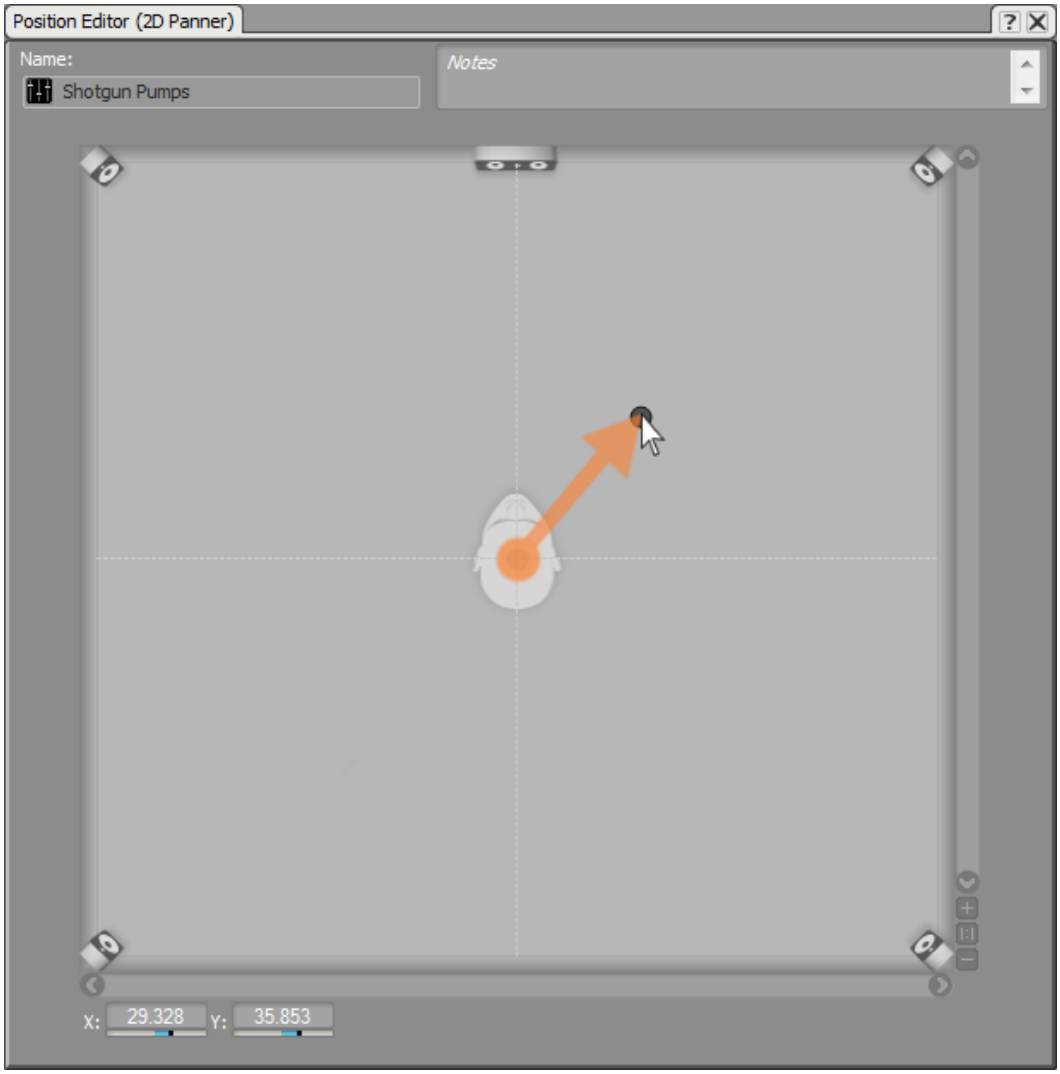


プレイヤーの頭の真上にコントロールポイントがあり、サウンドはサラウンドシステムの各スピーカーに分配されます。

ティップ

あなたのゲームで、ゲームプレイにおいて重要な音については、ゲームプレイヤーにはっきりと聞こえるようにしたいと考えるでしょう。これを実現するのに、2D, 3Dサウンドやミュージックオブジェクトをセンタースピーカーへ好きな%だけ送ることが出来ます。例えば、ドライバーの声や、レース場でのアナウンサーの声は完全にセンタースピーカーへ送られることで、ゲーム中の他の音がどんなに大きくても完璧に聞こえるようにすることができます。

4. コントロールポイントを前に、そして右にドラッグします。

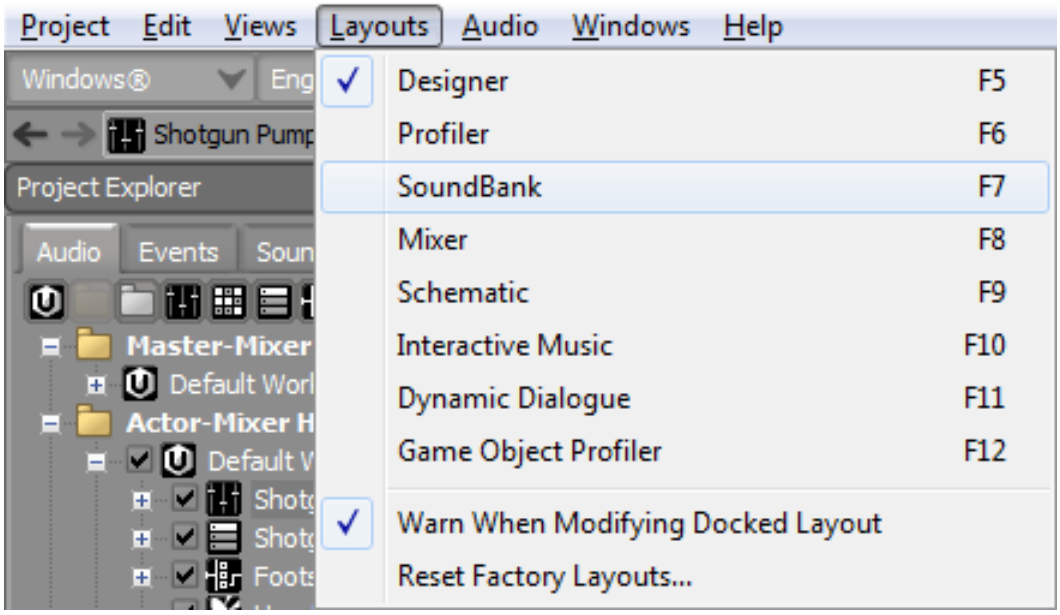


これで、ショットガンが発砲される度に、ショットガンのポンプアクションのサウンドがプレイヤー視点から見て前方右寄りに配置されます。

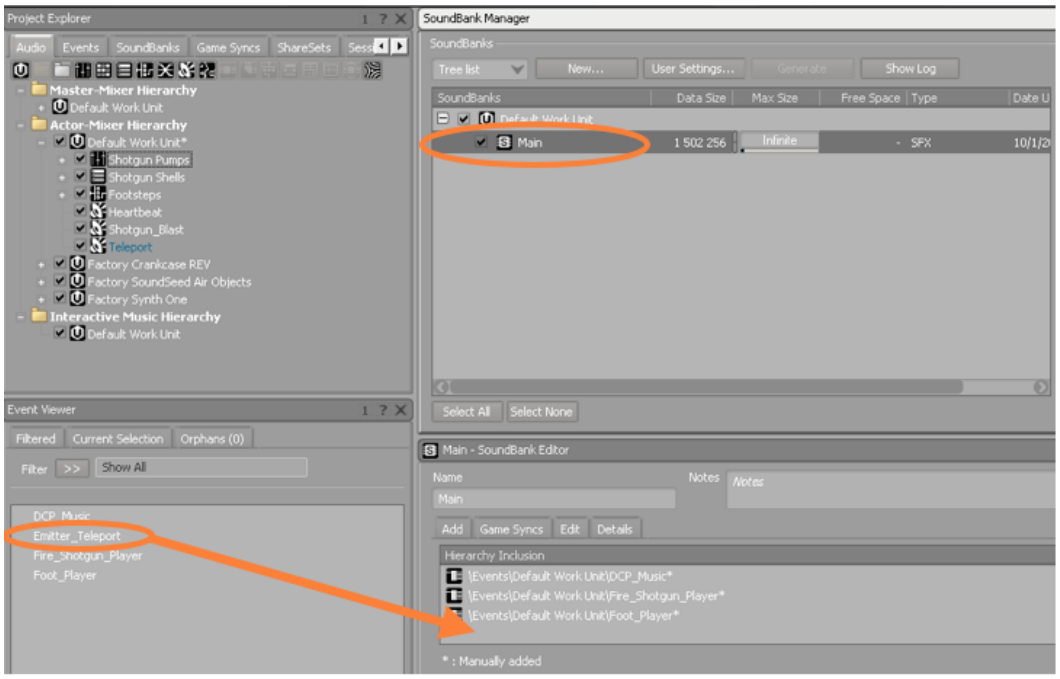
5. 2Dパンナーを閉じます。

ゲームをプレイする

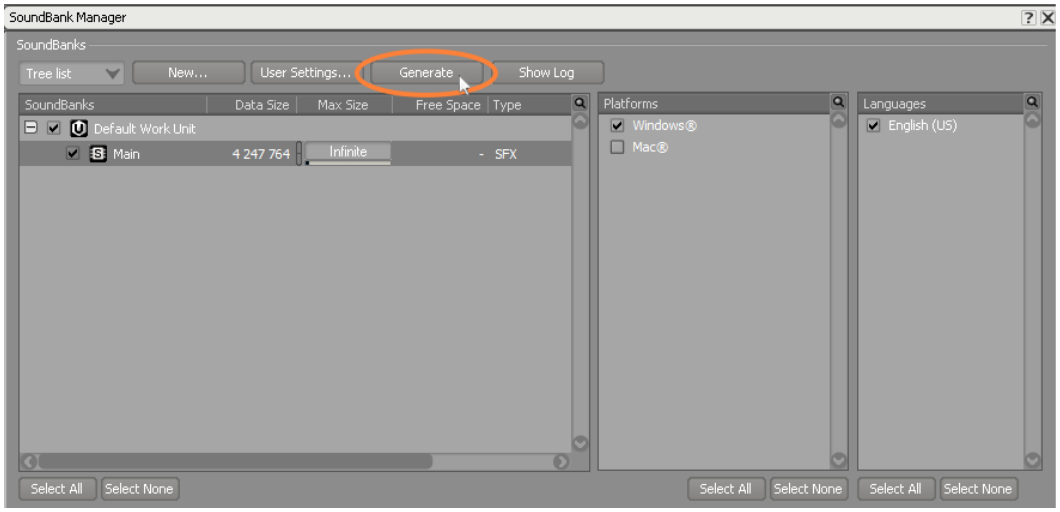
1. **Layouts** > **SoundBank**を選択、もしくは**F7**を押します。



2. SoundBanksビュー内でMain SoundBank を選択し、次にEmitter_Teleportイベントを Event ViewerからAddタブ内にあるSoundBank Editor Hierarchy Inclusionエリアにドラッグします。



3. **Generate**をクリックしサウンドバンクを生成します。



4. 新しく追加したサウンドと共にCubeデモを起動し、プレイし、テレポーターに近づくにつれあなたが設定した減衰カーブ設定がどのように機能しているか確認して下さい。

あなたが構築した設定がどのようにゲームプレイ中のサウンドに影響しているか聞いてみて下さい。テレポーターに近づいたり遠ざかったりして、距離減衰がサウンドをどのように変化させるか、ショットガンを発砲し、シェルの位置について耳で確認してみて下さい。Wwise側に戻り、更なる調整をし、再度サウンドバンクを生成し、理想の効果が実現できるまで繰り返して下さい。

これで本レッスンで行なった変更について更に深く学ぶ準備が出来ました。

レッスン 5：オーディオシグナルフローを理解する

アクターミキサーの構築 179
マスタミキサー階層の使用 184
各種エフェクトの使用 198
スケマティックビューの使用 212

完成したゲームには数百、数千ものサウンドが用意され、プレイヤーが耳を澄ませるスピーカーへと送られます。これまでに学んできたように、Wwise内のサウンドはオーディオファイルや、シグナルジェネレータの様な音源を元にしていますが、各信号が起点から、Wwiseのミックスシステムで処理され、コンピュータのスピーカーへ到達するまでの経路は様々です。デジタルオーディオワークステーション（DAW）や従来のミックスコンソールの様に、ソース音源が便利に整理され、信号フローを制御出来るようにバスへまとめられます。信号の一部はAux sendを介してエフェクトを適用することが出来ます。本レッスンではこれらの信号経路の構築方法、さらにその重要性について学びます。

アクターミキサーの構築

プロジェクトのオーディオ信号フローを構築する最初の手順は、まずアクターミキサー階層内に各オブジェクトを構成します。これまでのレッスンで、様々なサウンドオブジェクトをコンテナに格納し、同様なサウンドをまとめて整理したり、コンテナのプロパティを使用してそのコンテナに内包されている全てのオブジェクトに対してピッチのランダム化を素早く適用しました。同様なタイプのオブジェクトを一緒にまとめておくのは、ゲーム用にインテグレートする数百、数千ものサウンドを管理する上で有効な手段です。

ここではヒーロー役のプレイヤーがキューブデモの様々なレベルで発見する色々なアイテム、テレポーター、アーマーなどと関連付けられたサウンドの全てを一つのActor-Mixerオブジェクトに配置することから始めます。

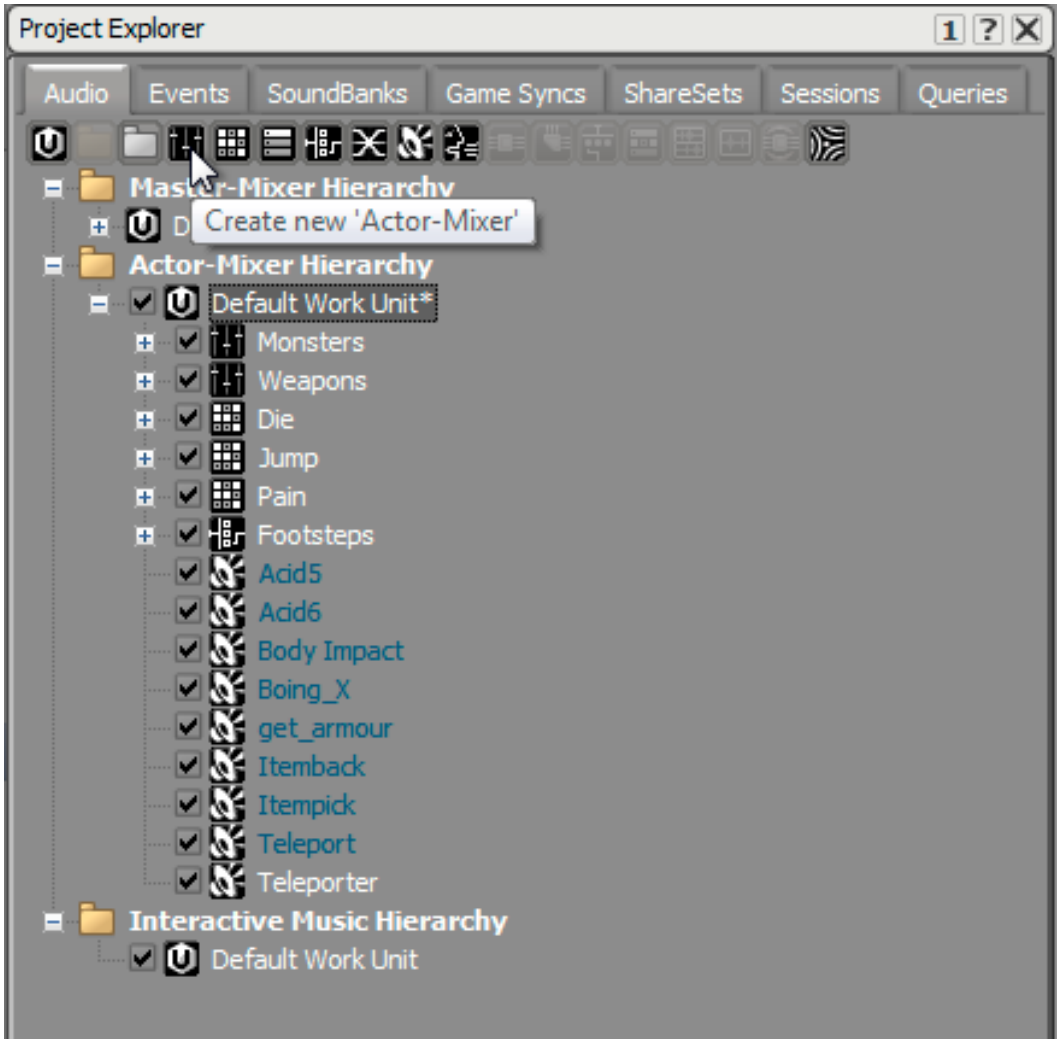
1. レッスン5のプロジェクトを開きます。

   レッスン5のプロジェクトには様々な新規サウンドSFXオブジェクトやイベントが既に用意されており、どのようにして更に開発の進んだプロジェクトを整理していくのかを学べるようになっています。

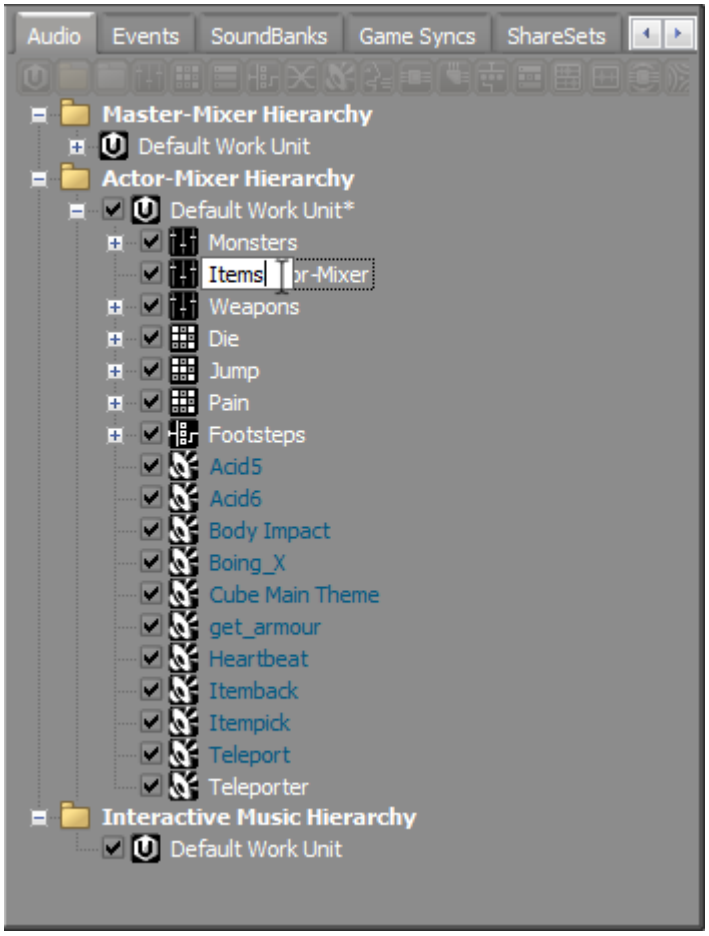
ティップ

後で時間をとってプロジェクトの詳細を確認し、どのように提供されたコンテンツが組み立てられているかを理解します。一般的な構成と手法についてはもう理解されていると思います。

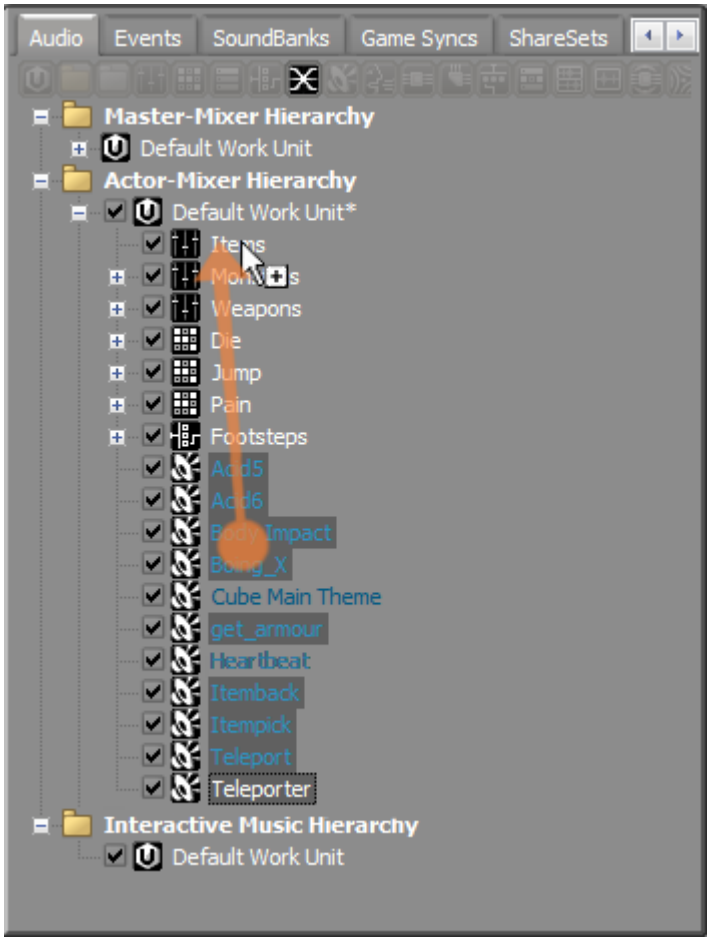
2. プロジェクトエクスプローラーのAudioタブで**Default Work Unit**を選択し**Create new Actor-Mixer** ボタンをクリックします。



3. 新規に作成したActor-Mixerを**アイテム**と命名します。



4. Cube Main Theme と Heartbeatを除いてFootsteps以下の全てを選択します。選択したものを**アイテム**アクターミキサーへドラッグします。



5. プロジェクトエクスプローラー内で、新規作成した**アイテム**アクターミキサーを選択します。

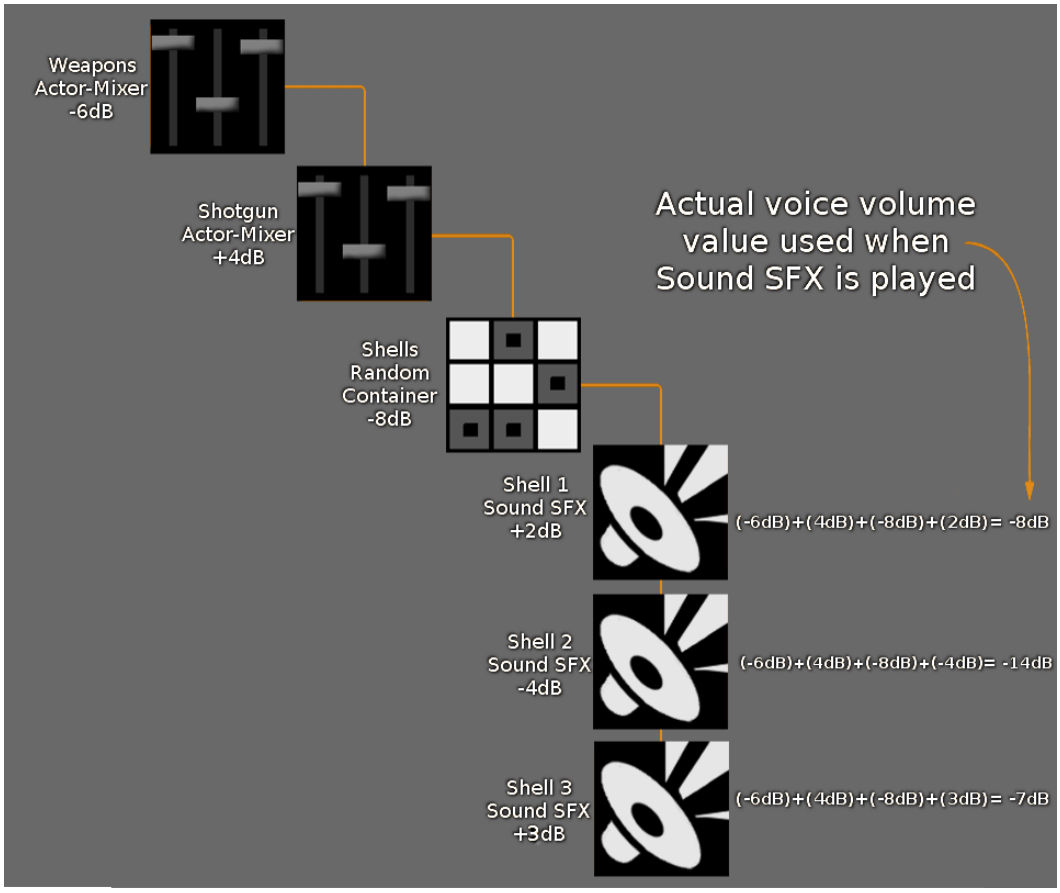
   さて、あなたのアイテムはアイテムアActor-Mixer内に格納されました。ここであなたはActor-Mixerが何なのか、もっと重要なのはActor-Mixerは何でないのかを明確に理解する必要があります。

   名称から、様々なオブジェクトから出力されるオーディオ信号の全てがまとめられ、アクターミキサーのGeneral Settingsタブにあるボリュームフェーダーがあたかもこれら統合された信号全体のマスターフェーダーのように機能するのではと思われるかもしれません。実際はそうではありません。名称が隠喩するものとは異なり、アクターミキサーは実際にはなにもミックスすることはせず、アクターミキサー内に格納されたサウンドはミックスされることはありません。

   そうではなく、ボイスボリュームやローパスフィルターのプロパティ値がそれぞれ包含されたアイテムの対応するプロパティのオフセット値になります。例えば、アクターミキサーのボイスボリュームが-3であれば、それに包含された各オブジェクトのボイスボリュームから3dB減算されますが、これら包含されたオブジェクトを直接見てもその値を確認することは出来ません。 このエフェクトは累積的で、あるアクターミキサーがもうひとつのアクターミキサーや、ボ

イスボリュームパラメータを持つランダム、シーケンスコンテナに内包されている場合、オフセット値の全てが加算され、サウンドソースが再生される度にボリューム値が決定されます。

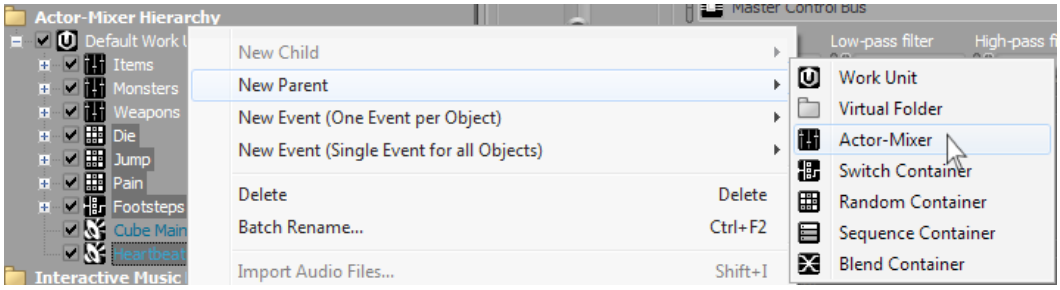
階層の最上位にあるアクターミキサーのボイスボリュームプロパティを完全に落とすと、そこに含まれる全てのオブジェクトのボイスボリュームが同じだけ減少（-96 dB）します。結論として、全てのオブジェクトのボイスボリュームを、それぞれ個別に落とすのと同じことを行ったことになります。ただし、このように劇的にボリュームを減少させたとしても、階層全体の累積ボイスボリュームが充分に大きく、-96 dB をオフセットするだけの効果があれば、オブジェクトが無音にならないことも考えられます。以下のダイアグラムで、サウンドSFXオブジェクトが他のオブジェクトに組み込まれている場合の、ボイスボリュームプロパティに対する累積的な効果を確認して下さい。



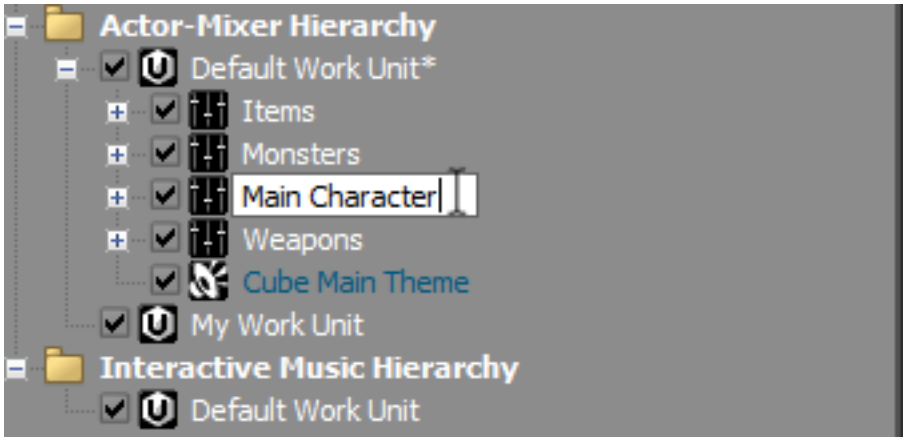
絶対値設定がされており、相対オフセットが適用されないプロパティについては動作が異なります。このような絶対値の例の一つとして、オブジェクトの出力バス設定があり、これについては次の演習で学びます。

アクターミキサーについて理解できましたので、次にメインキャラクターに直接関連するサウンドの全てを、独自のアクターミキサーに配置しますが、新しい手法を使って行ないます。

6. Die, Jump, Pain, Footsteps, 及び Heartbeat オブジェクトを選択し、これら選択されたオブジェクトの一つを右クリックし、**New Parent >Actor-Mixer**を選択します。



7. 新規に作成したアクターミキサーを**Main Character**と命名します。



マスタミキサー階層の使用

ゲームプレイにおいて、50以上のサウンドが同時に再生されることは一般的ではありません。あなたのコンピューターのオーディオ出力がLとRの2チャンネル出力しかないことを考慮すれば、Wwiseはこれらの信号があなたのコンピューターのオーディオインターフェイスに到達する前に、内部的にミックスを行なわなければなりません。

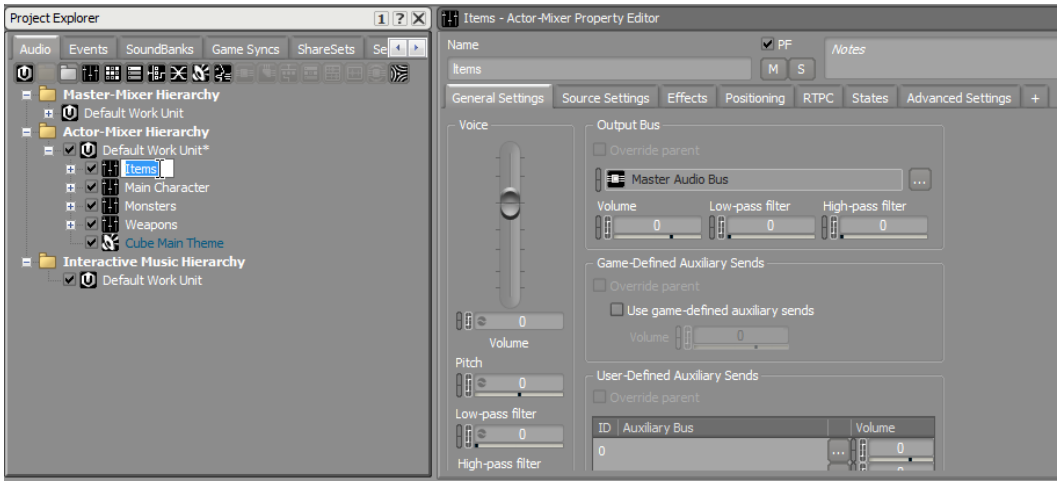
アクターミキサー階層ではオーディオの実際のミックス処理は行なわれないことを学びましたが、それではこれら様々なサウンドのミックスがどこで起こるのか疑問に思われるでしょう。これは、プロジェクトエクスプローラーのマスターミキサー階層で行なわれ、これについてはまだ学んでいませんが、既に使っていたことをまもなく理解されることと思います。

マスターオーディオバス

Wwiseで、バスというのは一種のオーディオオブジェクトで、ここで複数のオーディオ信号が新たな単一オーディオパスとして実際に合算される処理が行なわれます。バスはアクターミキサー階層には存在しませんが、マスターミキサー階層にはあります。全てのWwiseプロジェクトには、マスターオーディオバスと呼ばれる、少なくとも一つのオーディオバスが、マスターミキサー階層に存在ます。ゲームで

聞こえる全てのサウンドは最終的にはこのマスターオーディオバスを通過しますこの非常に重要なバスについて見てみましょう。

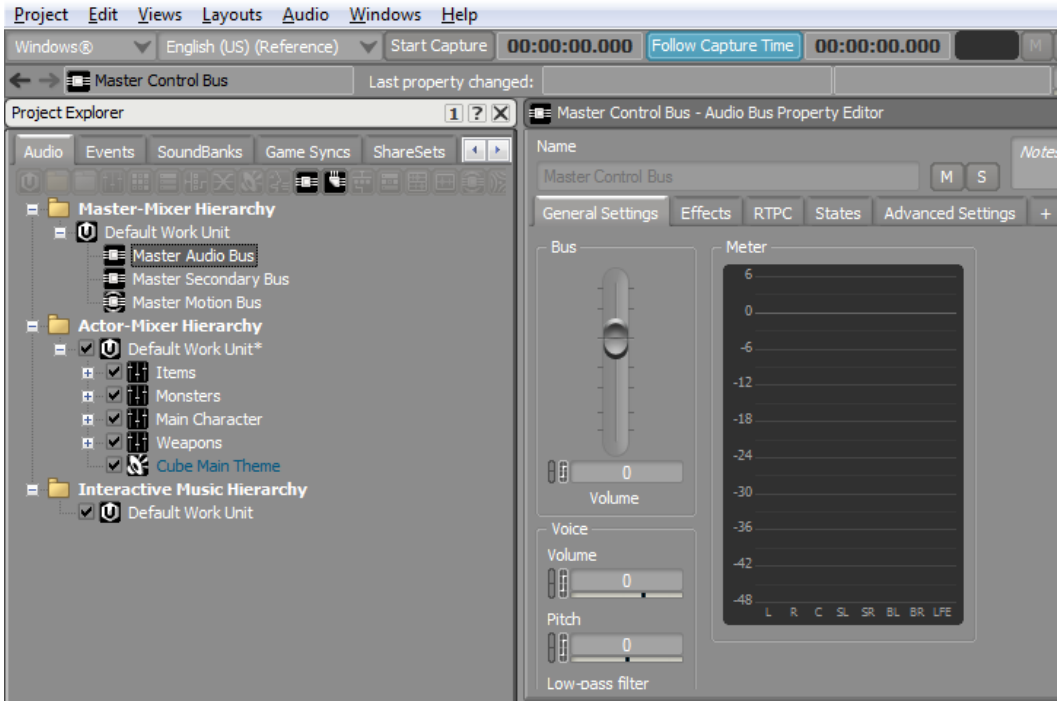
1. Default WorkUnit内の**Items**を選択します。



どのようにアクターミキサー出力バスが、マスターオーディオバスに設定されているか確認します。こちらはアクターミキサーに設定されているので、内包されたオブジェクトは全て出力先をマスターオーディオバスへ設定されています。言い換えれば、それぞれのサウンドオブジェクトは 1 つのオーディオ機器とし、それらがオーディオケーブルでマスターオーディオバスと呼ばれるオーディオミキシングコンソールへ直接接続されていると考えることができます。物理的な入力数に制限がある、従来のオーディオミキシングコンソールとは異なり、バスに接続できる個別のオーディオオブジェクト数に実質上制限はありません。

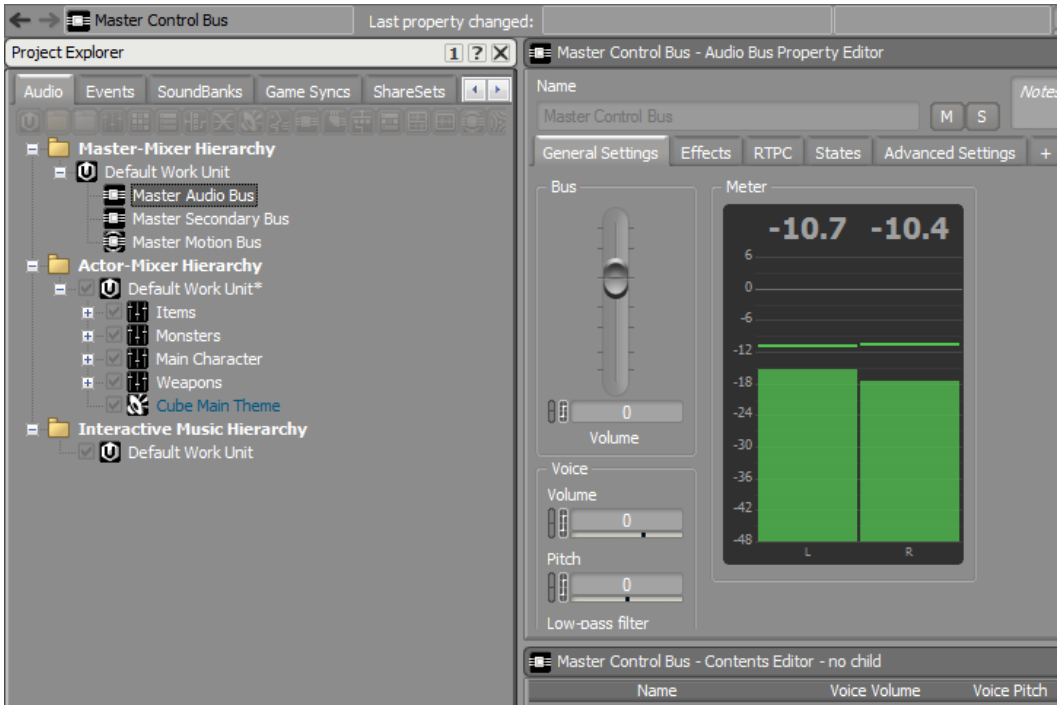
ここで、マスターミキサー階層中のマスターオーディオバスを確認します。

2. マスターミキサー階層内で、デフォルトワークユニットをエクスパンドし、**マスターオーディオバス**を選択します。

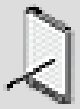


オーディオバスプロパティエディター内で、オーディオメーターを確認します。メーターは、オーディオが実際に通過しないアクターミキサーには存在しません。実際のオーディオレベルのメーターリングは、オーディオバスに存在し、この事実からオーディオがオブジェクトを介して渡されることが確認できます。

3. アクターミキサー階層内の任意のサウンドSFXオブジェクトを選択し、その**マスターオーディオバス** を再度選択し、オブジェクトを再生します。



メーターは対象のバスに送られる合計のサウンドレベルを表示します。



注記

Wwiseをサラウンドサウンド再生の設定にしている場合には、異なるレイアウトになります。

２つのボリュームコントロールがあることにお気づきかもしれません。ボリュームフェーダーを含むバスボリュームコントロールがメインマスターミックス出力になります。このフェーダーを調整することで、このバスに送られる、全てのソース音源の合計信号を増加、減少させます。ボイス内にあるこのボリュームプロパティ及びピッチとローパスフィルターは、合計されたオーディオ信号に直接適用されるのではなく、このバスに送られるオブジェクトに対するオフセット制御として適用されます。全てのオブジェクトは最終的にマスターオーディオバスへ送られるので、これらのプロパティを調整することは、プロジェクト内の全てのオブジェクトに対する調整を行なうことと同じなので、気をつけて下さい。

追加オーディオバスを使用したサブミキシング

アクターミキサー階層内で、オブジェクト内にオブジェクトを入れ子にできるように、マスターオーディオバスを含め、オーディオバス内でオーディオバスを入れ子にすることができます。追加のバスを使用することは、サウンドの特定のカテゴリーの何かについてすばやく調整するには便利です。例えば、多くのゲームでは、ゲーム内の他の音とは独立した形で音楽のボリュームを設定することができるようになっています。全ての音楽を一つのミュージックバスで扱うことで、ユーザー入

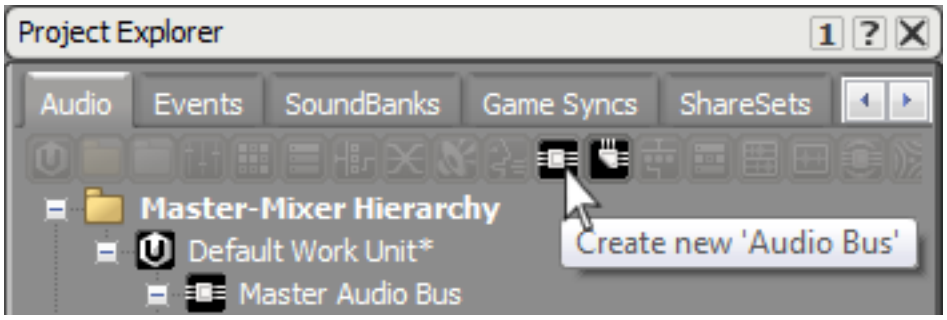
力をミュージックバスボリュームプロパティへマッピングすることができます。マルチバス手法のもう一つの応用は、ディレイやリバーブなどのエフェクトをサウンドの特定のカテゴリー、例えば、ゲームの仮想環境内で聞こえるサウンドエフェクトやダイアログに対して適用することです。このマルチバス手法をキューブデモの実装でも使用します。

注記

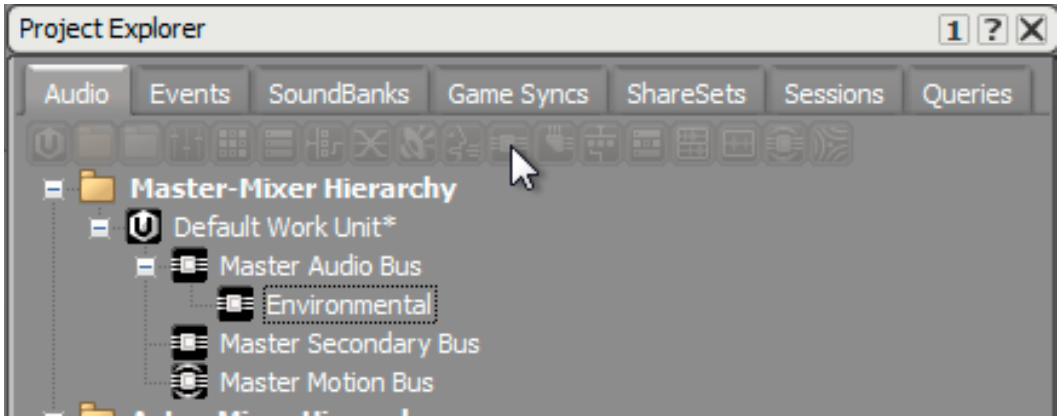
マスターセカンダリーバスはビルドインスピーカー搭載のSony PS4のコントローラーなど、セカンダリーオーディオ出力パスを持つデバイスにおいて使用します。マスターモーションバスは、特定のゲームコントローラーにあるランブル、振動等の機能と共に使用します。

まずは環境バスを作成することから始め、後に多くのサウンドエフェクトをこちらにアサインしてきます。

1. マスターオーディオバスを選択し、**Create new Audio Bus**をクリックします。

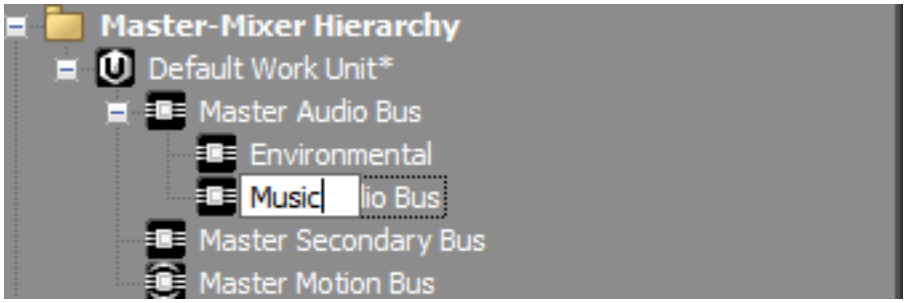


2. 新しいオーディオバスを**Environmental**と命名します。



次にゲームで使用する音楽用のバスを作成します。

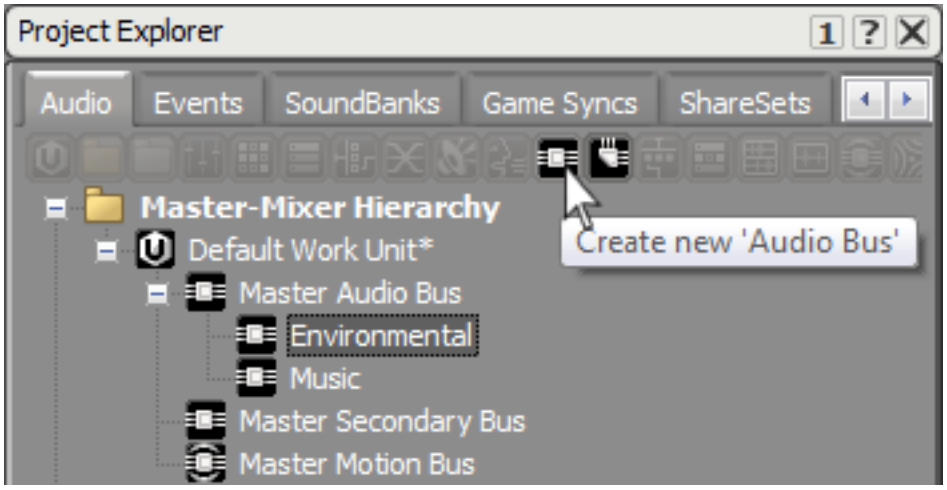
3. マスターオーディオバス内で新たにオーディオバスを作成し、これを**Music**と命名します。



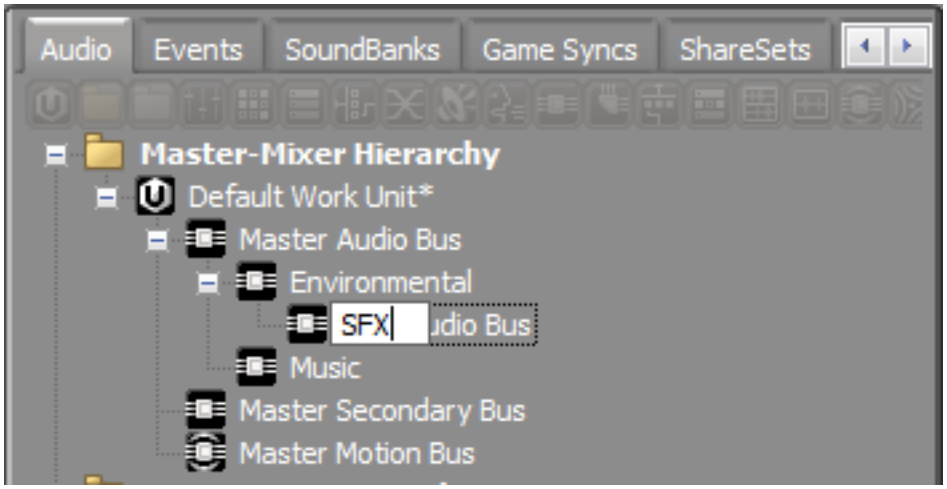
これでディスクリートミュージックと環境バスができました。こうすることで、後々、環境バスへ適用する可能性がある環境エフェクトが、音楽に影響を与えないようにできます。

しかし、もしあなたがゲームのプレイヤーに、環境バスにアサインされるかもしれない、ゲーム進行に重要なダイアログを残したまま、サウンドエフェクトのみオフにするオプションを提供したい場合はどうしましょう。これを実現するには、単純にもう一つのバスを用意し、これを既に作成した環境バスの中にサウンドエフェクト用のバスとして追加すれば良いのです。

4. 環境バスを選択し、**Create new Audio Bus**をクリックします。



5. 新規作成したバスを**SFX.**と命名します。

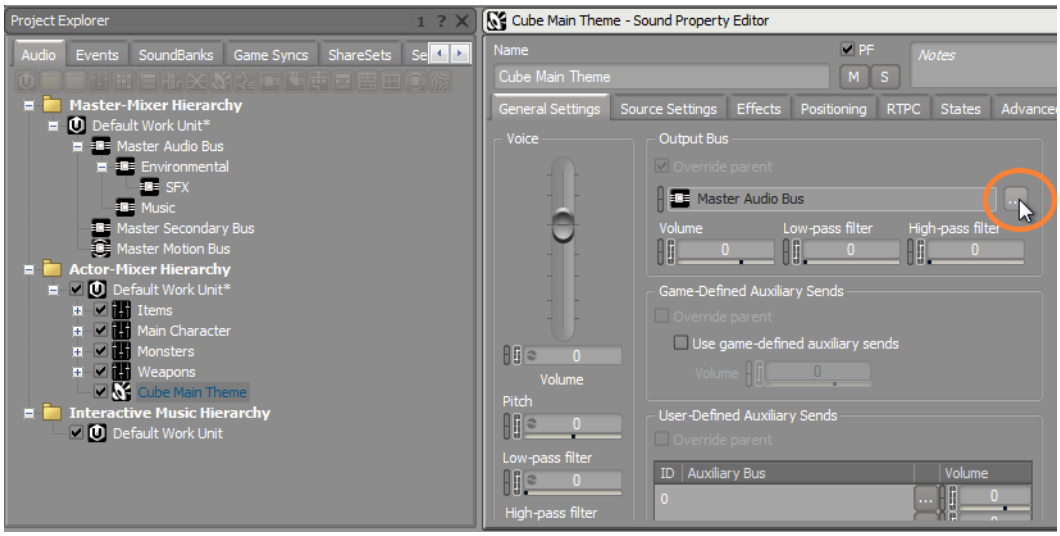


これであなたのゲームオーディオのバス構成が準備できました。

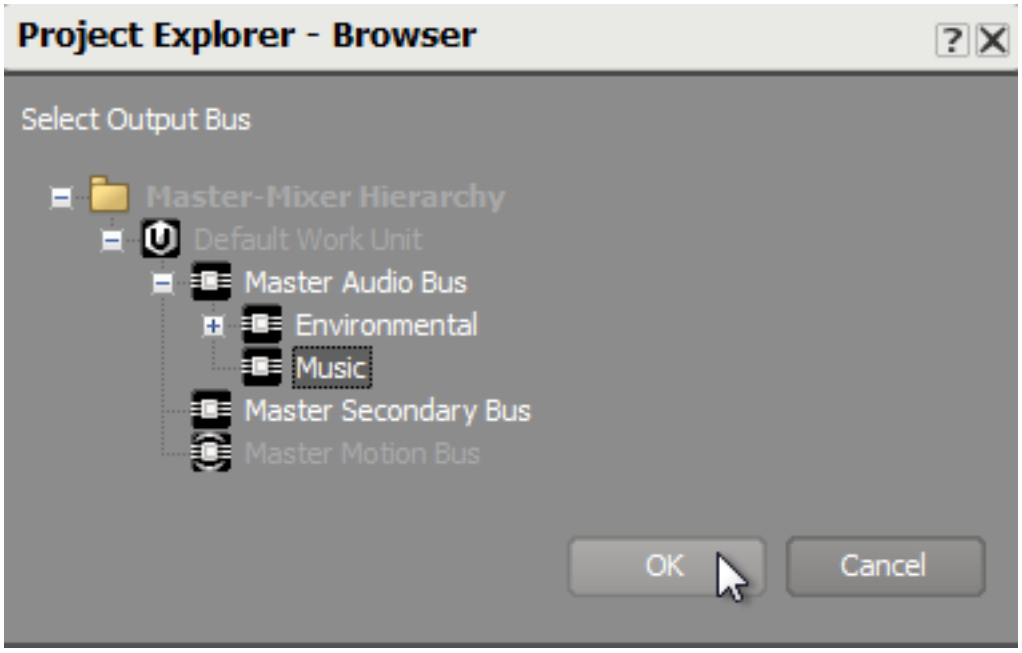
オーディオバスのアサイン

新しいバスを作成しましたが、現状アクターミキサー階層内のオブジェクトはマスターオーディオバスへ直接アサインされています。したがって、適切な形で改めてアサインする必要があります。まず、ミュージックサウンドSFXオブジェクトをミュージックバスへアサインします。

1. Cube Main Themeオブジェクトを選択してから、サウンドプロパティエディターのOutput Bus グループで、Master Audio Busがアサインされている右側のブラウズボタン **[...]** をクリックします。



2. デフォルトワークユニット（Default Work Unit） と マスターオーディオバス（Master Audio Bus）をエクスパンドし、ミュージックオーディオバス（Music Audio Bus） を選択し **OK**をクリックします。

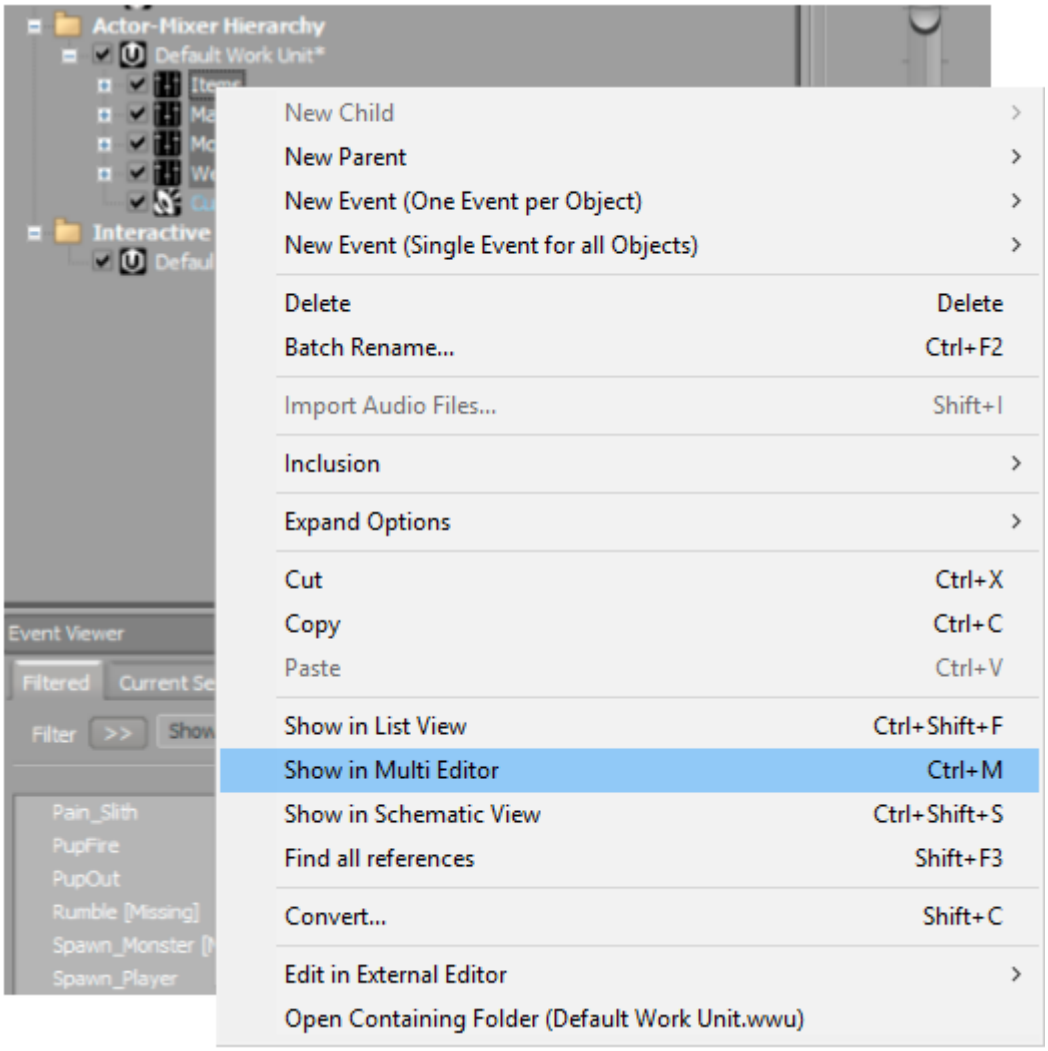


その他のサウンドはプレイヤー環境のサウンドのカテゴリーに入れます。環境のサブカテゴリー用の、２つのアクターミキサーがありますので、トップレベルのアクターミキサーのアウトプットバス指定を再アサインすることで、これらの子オブジェクトのバス出力の全てを素早く再アサインすることができます。この作業を更に早く実行するには、Wwiseの強力なマルチエディット機能を使います。この機能では複数のオブジェクトのプロパティを、それらが同じ階層に存在しない場合でも、一度に調整することが出来ます。

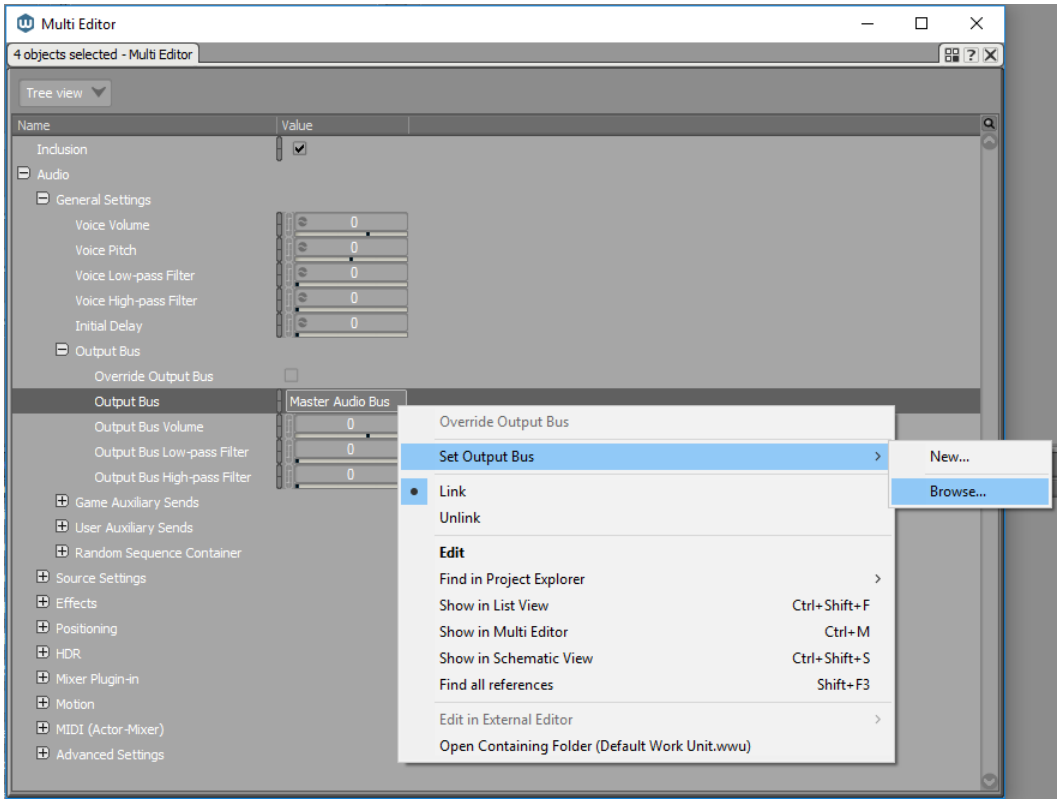
ティップ

また、プロジェクトエクスプローラーからProperty Editorの一般設定（General Settings）タブのOutput Busボックスへバスオブジェクトをドラッグ・アンド・ドロップすることができます。

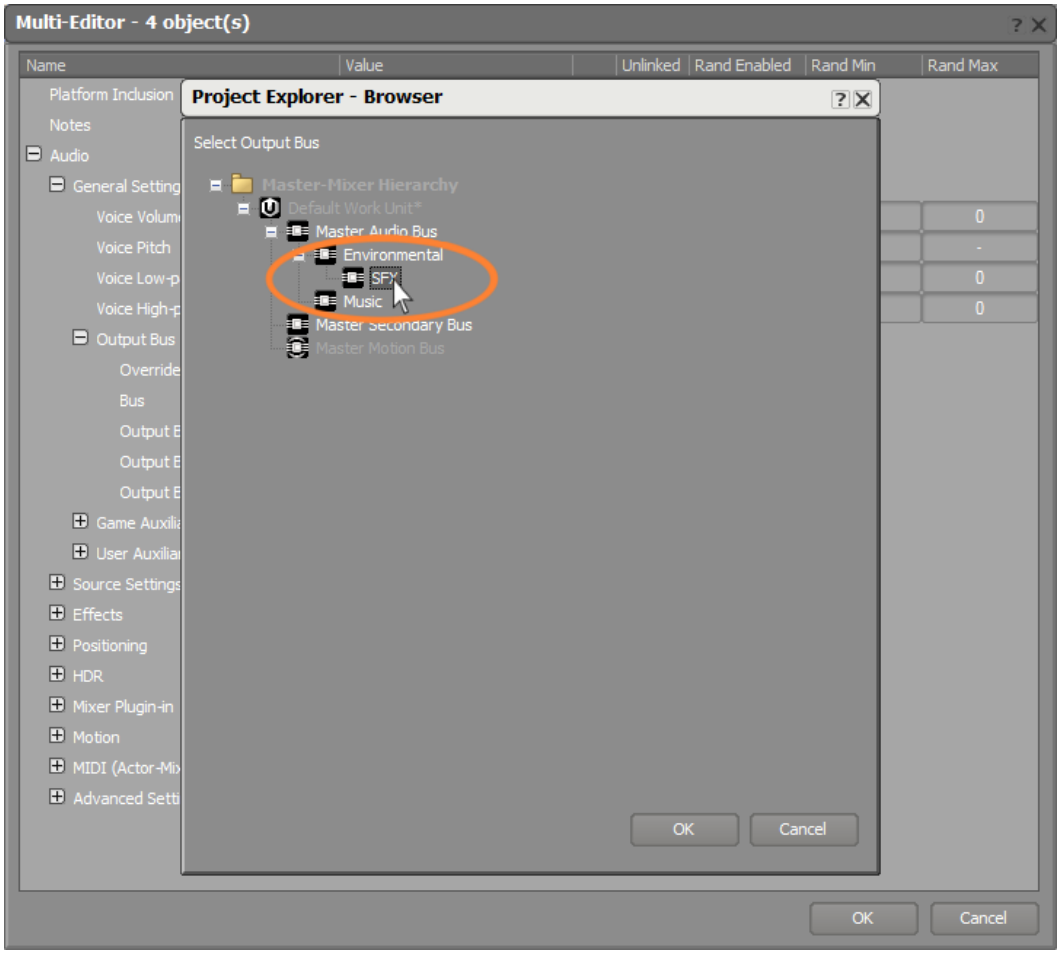
3. **Shift**を押しながら他のアクターミキサーを選択し、選択されたオブジェクトの一つを右クリックし、**Multi-edit** を選ぶ、もしくは**Ctrl+M**を押します。



4. Audio, General Settings, Output Bus プロパティエリアをエクスパンドし、バスプロパティ(Master Audio Bus)の右にある [...] **Browse**をクリックします。



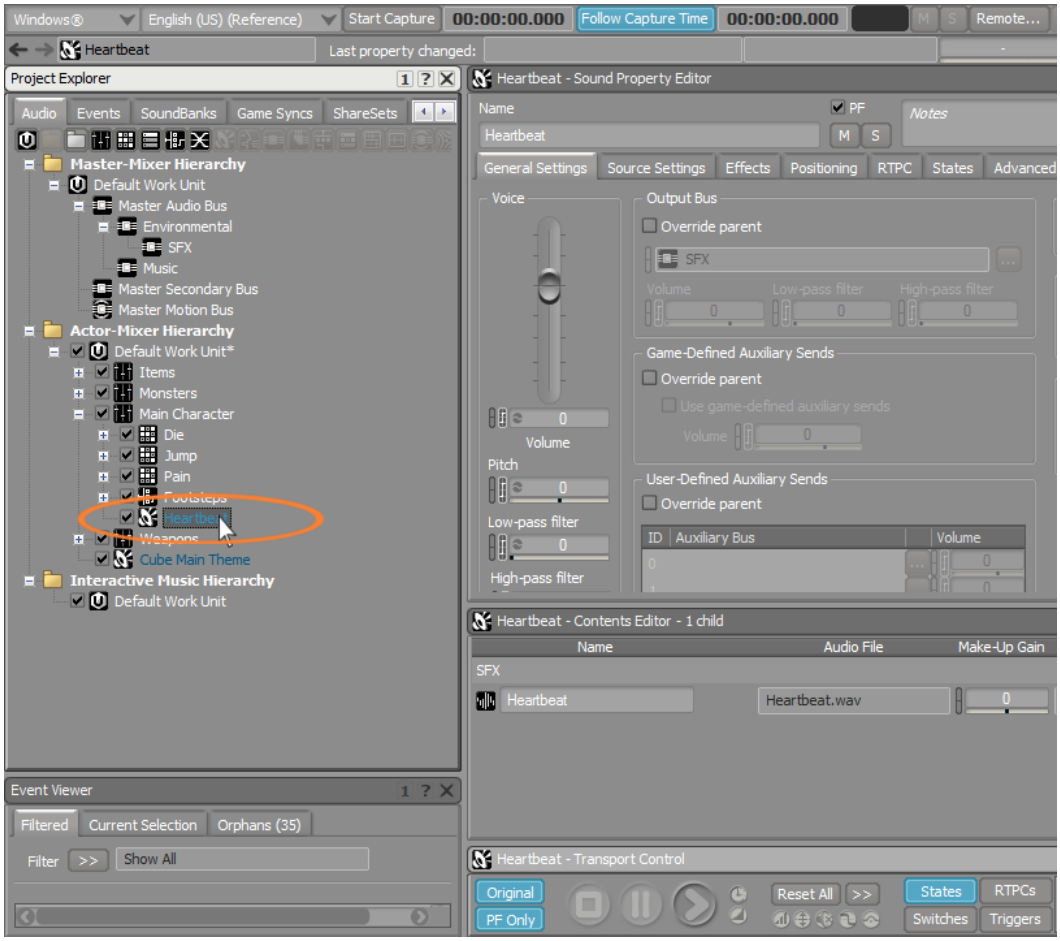
5. ブラウザーであるプロジェクトエクスプローラーでデフォルトワークユニット（Default Work Unit），マスターオーディオバス（Master Audio Bus）及び環境バス（Environmental Bus）をエクスパンドし、**SFX**を選択し、**OK**をクリックします。



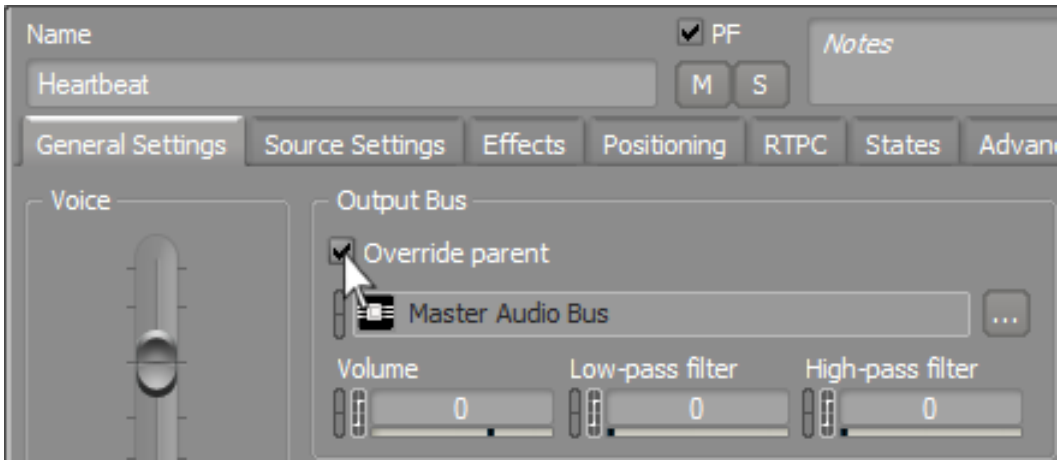
これで残りのオブジェクトの全てがSFXバスにアサインできました。

次の演習では、環境内の全てのオブジェクトに対してリバーブ効果を適用しますが、心拍音はその空間においては十分大きな音ではないのでリバーブ効果を聞くことが出来ません。このため、心拍音は別ルートを通り、環境バスを完全に回避し、マスターオーディオバスへ直接送られます。しかしながら、分類整理の関係上、心拍音（Heartbeat)をメインキャラクターアクターミキサーに格納するのが理にかなっています。これを両方とも実現するには、Heartbeat Sound SFXオブジェクト が親バスのアサインをオーバーライドするように指定することで可能です。

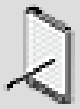
6. メインキャラクターアクターミキサーで、 **Heartbeat** Sound SFX オブジェクトを選択します。



7. 心拍音（Heartbeat）のサウンドプロパティエディターで **Override parent** チェックボックスをクリックします。



この心拍音（Heartbeat）オブジェクトはデフォルトマスターオーディオバスのアサイン設定が反映されるようになります。



注記

バスアサインの右にある**Output Bus**をクリックすると、マスターミキサー階層にある任意のバスを選択できます。

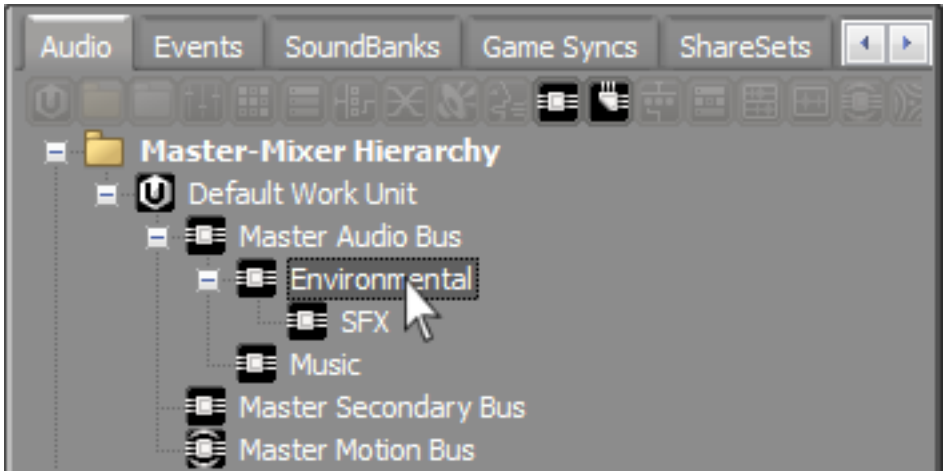
ダッキング

異なるサウンドタイプ用に特別なオーディオバスを作成できる利点の一つは、これら複数のバス間にダイナミックな関係を作成できることです。この一般的な例はダッキングと呼ばれる手法です。ダッキングは、あるオーディオ信号パスのボリュームを、別のオーディオ信号パスのボリュームによって操作する方法の一つです。この典型的な例はラジオDJです。DJがマイクへ話している際には、マイクの信号の有無が音楽のボリュームを自動的に抑えるのに使われ、DJの声が聞きやすいようにしています。DJの話が終わると、音楽は再度ボリュームが自動的に元のレベルに戻ります。

ゲームオーディオにおいてダッキングはとても強力な機能で、この機能がオーディオバスに対して提供されていることで、設定も非常に容易になっています。この例では、ゲーム内でアクションがあり、あなたが一生懸命実装したそのサウンドをプレイヤーに確実に聴かせたい時に、音楽のボリュームをダッキングさせます。しかしながら、プレイヤーが物陰に隠れ、じっとしており、なんの環境音も聞こえないような状況もあります。こうした場合には、音楽のボリュームを少しだけ上げるようにします。

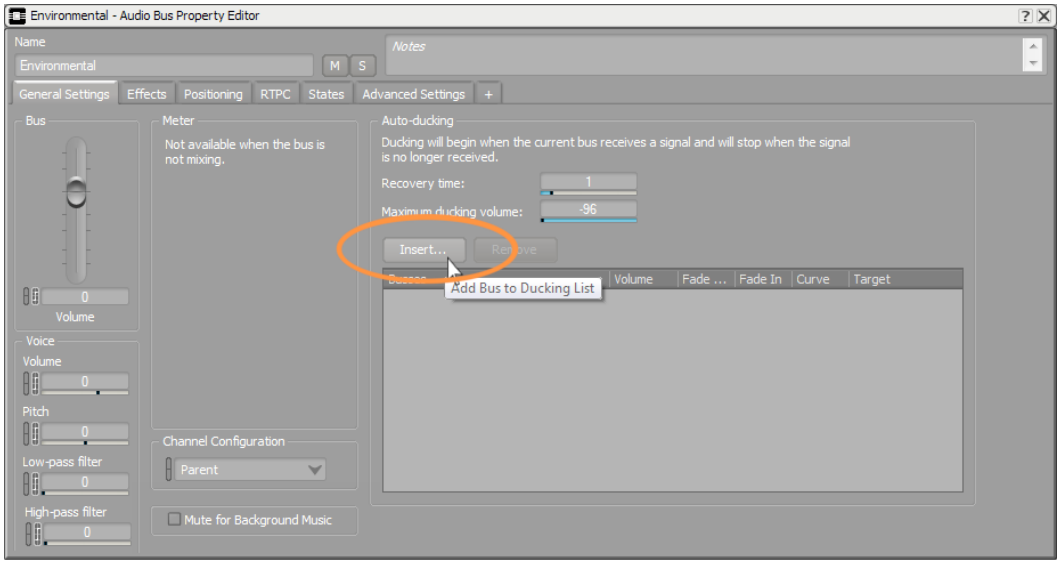
別のバスのオーディオボリュームを制御するのに使うバスを選択することから始めます。

1. マスターミキサー階層で、環境バス（**Environmental Bus**）を選びます。

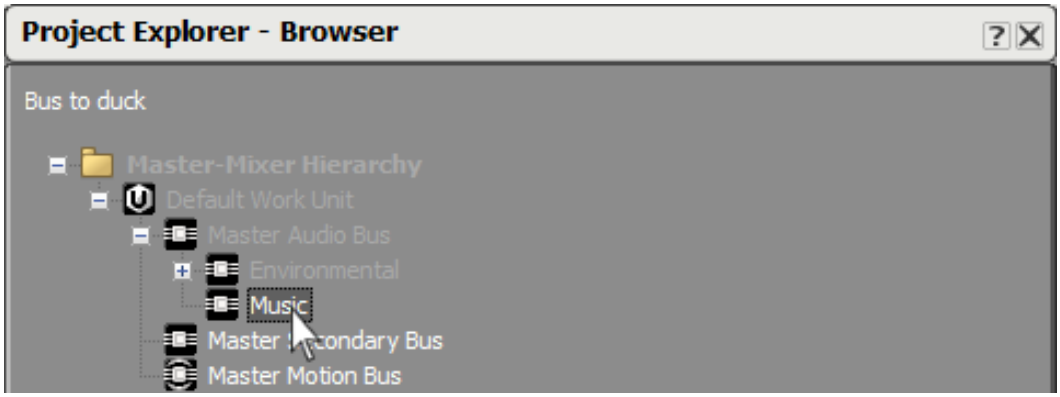


ここで、この環境バスでどのバスを制御するかを指定する必要があります。

2. オートダッキング（Auto-ducking）エリアで、**Insert**をクリックします。



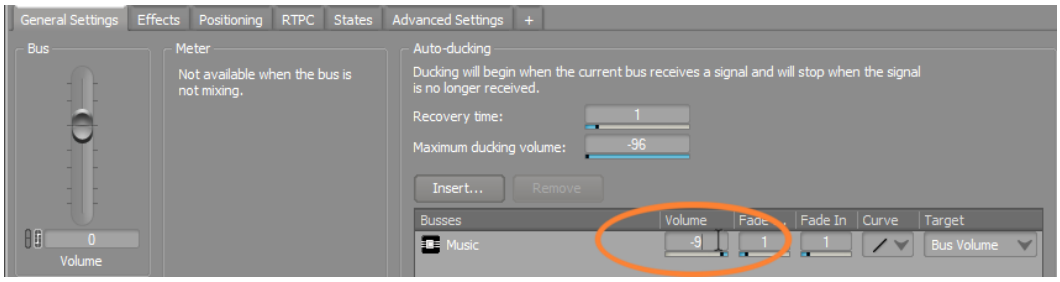
ブラウザーであるプロジェクトエクスプローラーが開きます。



3. マスターオーディオバスをエクスパンドし、**Music**を選択し、**OK**をクリックします。

   ここで、環境バスにオーディオ信号がある場合に、ミュージックバスのボリュームがどのように影響を受けるかを指定する必要があります。デフォルトのボリュームの変化は -6dBで、これはボリューム変更がようやく気がつきはじめる程度です。ボリューム変更をもう少し大げさな設定にしてみます。

4. ボリュームプロパティを-9に設定します。



バスに信号が検知され、ターゲットバスの変更処理が起こるまでの経過時間を指定するフェードイン、フェードアウトなど、他のプロパティがあります。ターゲット列は、ボリューム変更がバス全体のレベルで反映されるようになっていますが、ボイスボリュームに変更することも可能です。ボイスボリュームを選択すると、ミュージックバスに送られる各オブジェクトのボイスボリュームプロパティから、そのボリュームが減算されます。

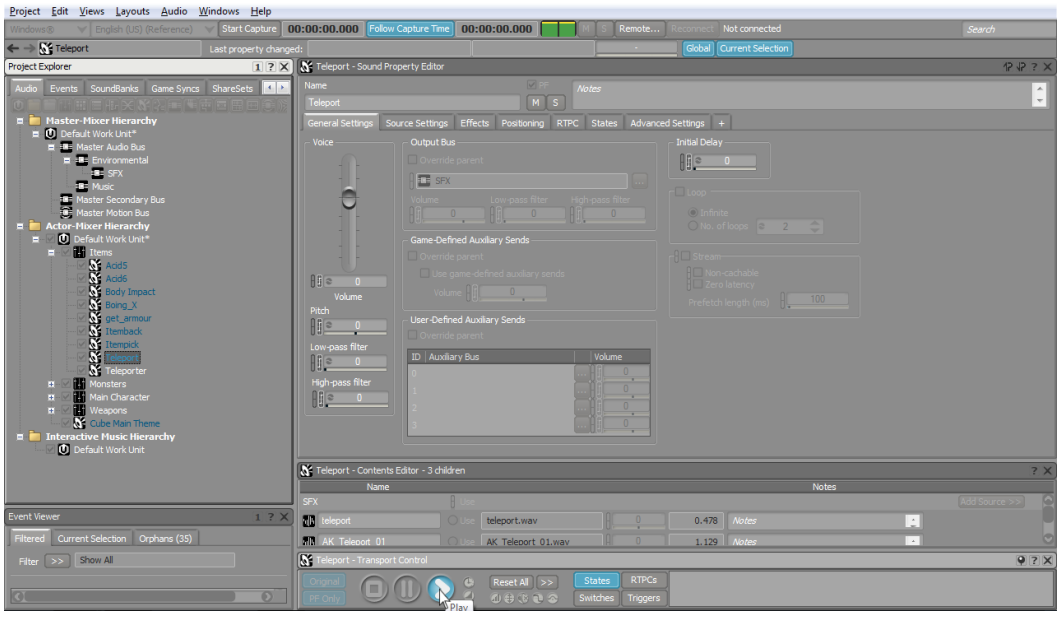
各種エフェクトの使用

ここまででWwseにおけるオーディオ信号のフロー構造について理解したので、次にオーディオ信号を信号フローにおける様々なポイントで変更を加える色々な方法を学んでいきます。既にあなたはピッチやフィルタープロパティを使用して、サウンドの聞こえ方を変える方法を学んできました。さらにエフェクトプラグインを使用することでサウンドに変化を与えるもっと多くの方法があります。エフェクトプラグインは、リバーブ、ダイナミクスプロセッサ#、イコライゼーション、ディレイ、など、数多くのサウンドツールを提供します。

まず、Wwiseにインポートした後に、どのようにプラグインを使用してサウンドをあなたの好みに仕上げることができるかを学びます。

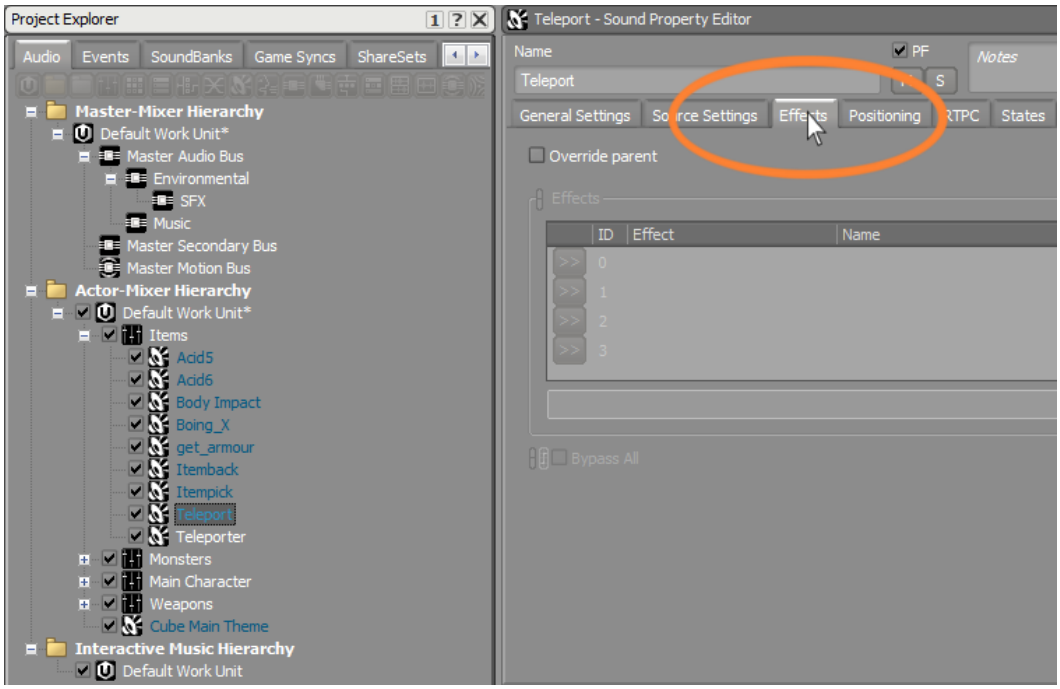
エフェクトの挿入

1. プロジェクトエクスプローラーで、**アイテム**アクターミキサーを開き、**Teleport** オブジェクトを選択し、再生します。



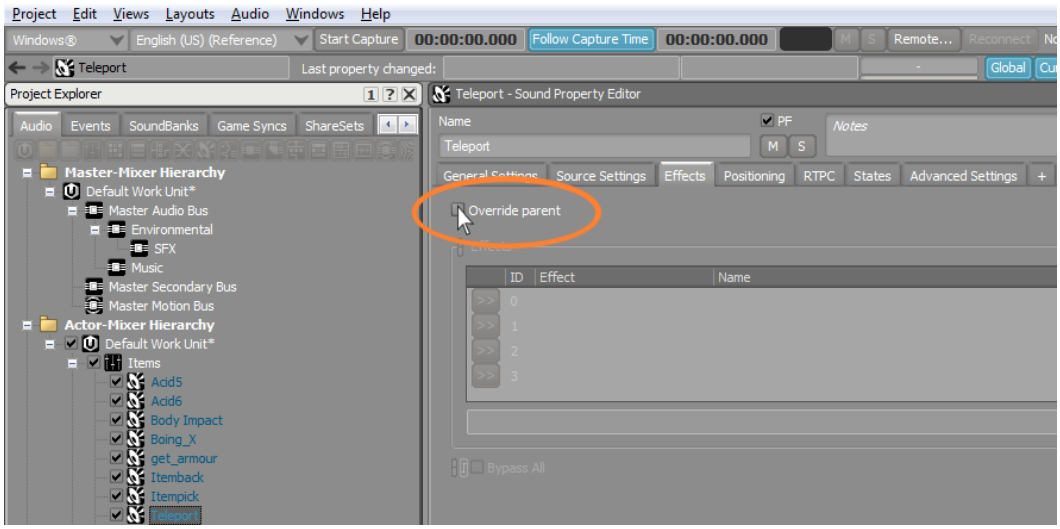
テレポートのサウンドは、プレイヤーがテレポーターを通過する際に再生され、プレイヤーを一瞬にしてマップ上の別の物理的なロケーションに移動させます。ここで少し問題があり、テレポートのサウンドが少々短いようです。サウンドをもう少し長くし、スペシャルエフェクトっぽさを追加するために、テレポートサウンドにディレイと、エコーを追加します。

2. テレポートサウンドのプロパティエディターで、Effectsタブをクリックします。

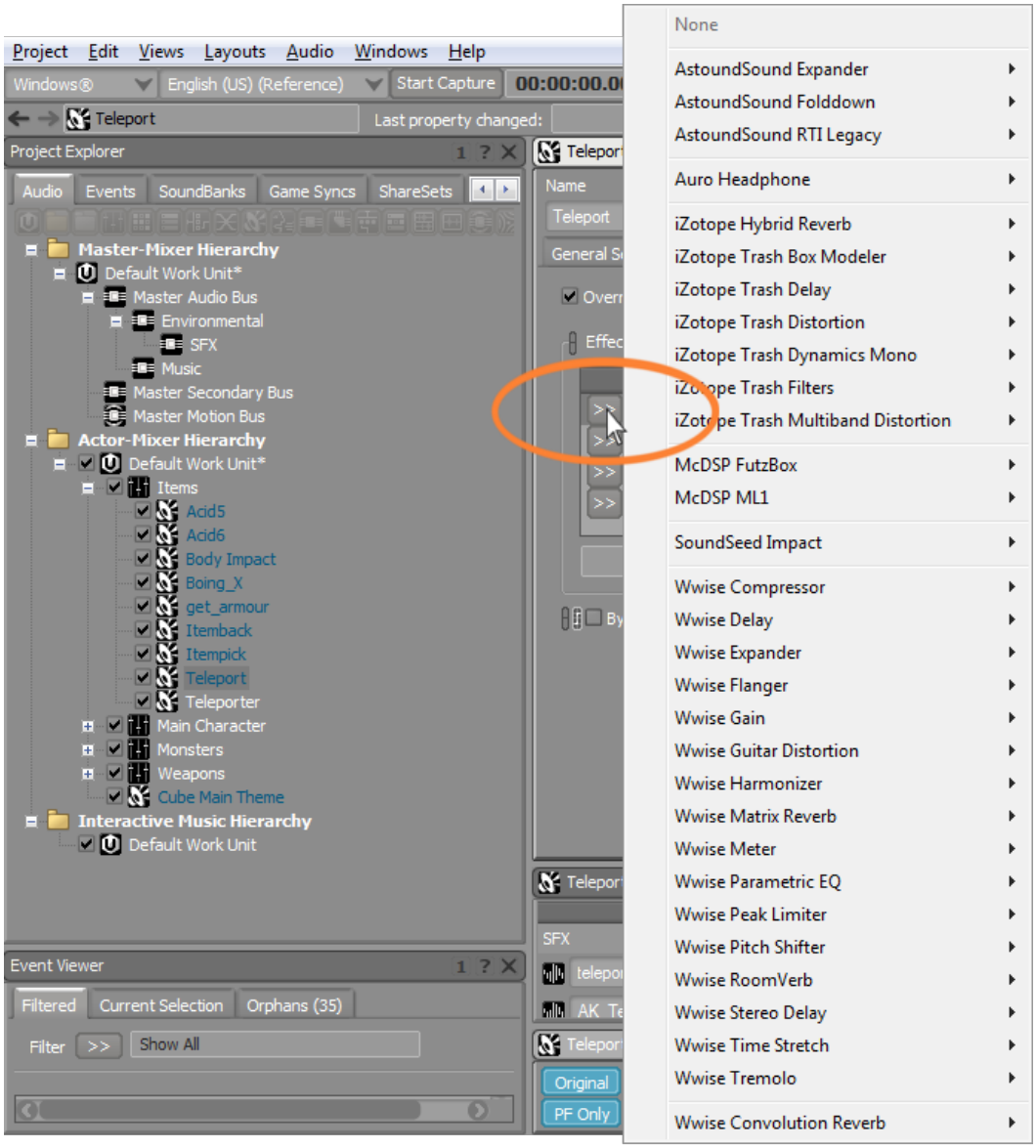


エフェクトエリアには４列から構成され、それぞれが使用可能なエフェクトプラグインを設定することができます。しかしながら、使用できるプラグインが見つかりません、それはテレポートオブジェクトはアイテムアクターミキサー内にあるので、デフォルトでテレポートはアイテムアクターミキサーのエフェクト構成を継承しているからです。テレポートオブジェクトのエフェクト構成を変更するには、親オブジェクト構成を無視する設定をする必要があります。

3. **親をオーバーライド（Override parent）** チェックボックスを選択します。



4. **Effect ID 0**のセレクターボタンをクリックします。



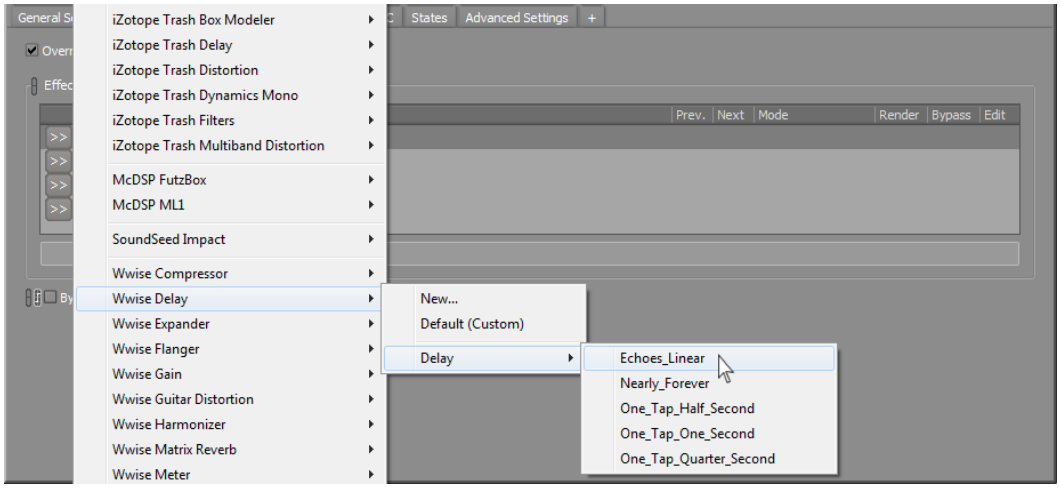
メニューには選択可能なエフェクトプラグインの広範なリストが表示されます。

注記

リストからどれでも選択可能ですが、Wwise コンボルーションリバーブの他、その他のWwiseで名前が始まらないプラグインは、ゲームの商用展開には別途追加のライセンス費用が発生するサードパーティプラグインになります。

一つのエフェクトを選択すると、特定のプラグインのプリセットを表す追加のサブセレクションが用意されていることがあります。

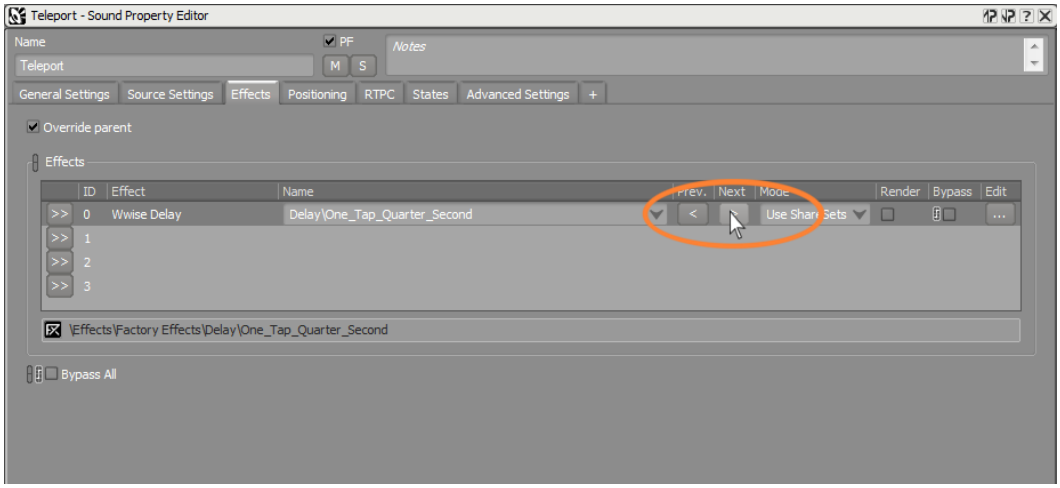
5. ここでは**Wwise Delay** > **Delay** > **Echoes_Linear**を選びます。



このWwise ディレイプラグインは、Echoes_Linearプリセットと共に、テレポートサウンドオブジェクトに直接挿入されます。

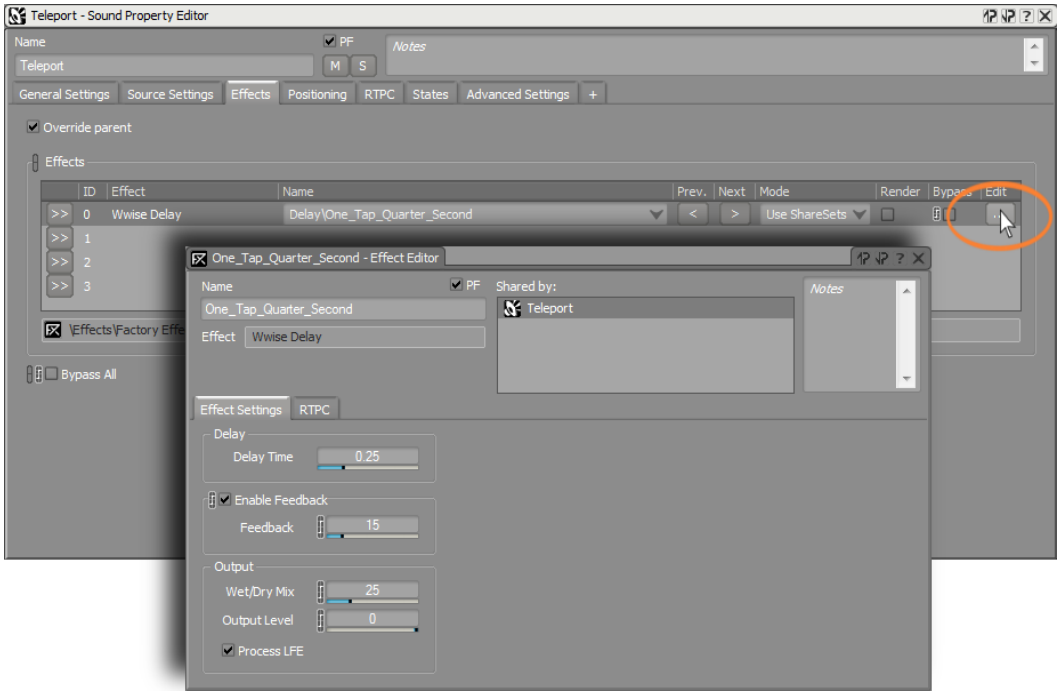
Prev及びNextの矢印で、使用可能なエフェクトプリセットを順番に確認することが出来ます。

6. **Next** を押しながら、**Teleport** オブジェクトを再生することで、異なるディレイ設定をオーディション試聴することができ、最終的に Delay \One_Tap_Quarter_Secondを選択します。



プリセットを使うことで、求めるサウンドにある程度近いところまで持って行くことが出来ますが、この場合エコーが少々長いようです。エフェクトプロパティを調整することで、あなたが求めるサウンドにすることができます。

7. **Edit** をクリックしてエフェクトエディターを開きます。

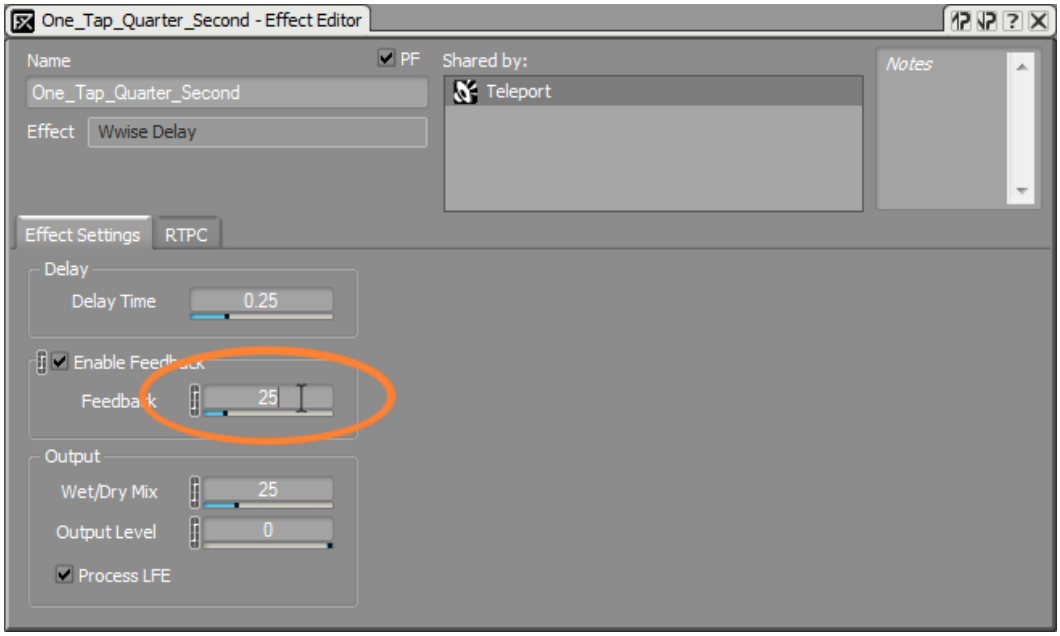


Wwise ディレイプロパティが表示されます。使用可能なプロパティの種類は、エフェクトプラグインの複雑さによって大きく異なります。

ティップ

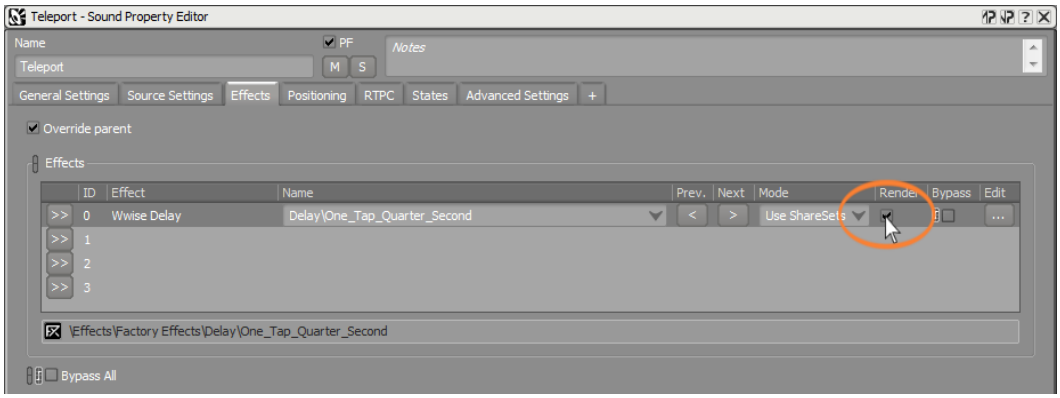
多くのエフェクトにあるパラメータにはウェット/ドライミックスがあります。この値を0%に設定するとエフェクトが全くかからず、オリジナルのドライな信号のみが聞こえます。100%はエフェクトのみが聞こえることになります。ウエット/ドライ比率を調整し、あなたの求める最適なブレンドを探して下さい。

8. **Feedback**プロパティを25にし、エフェクトエディターウインドウを閉じます。



このエフェクトはゲームのプレイ中にリアルタイムで生成されます。様々なリアルタイムエフェクトを多くのオブジェクトで実行している際には、ゲームシステムのCPU処理能力を使い切ってしまう可能性があります。CPU負荷を削減するには、サウンドバンクを再生成する際に、新規オーディオファイルとしてエフェクトをレンダリングしてしまう方法もあります。

9. **Render** チェックボックスを選択します。



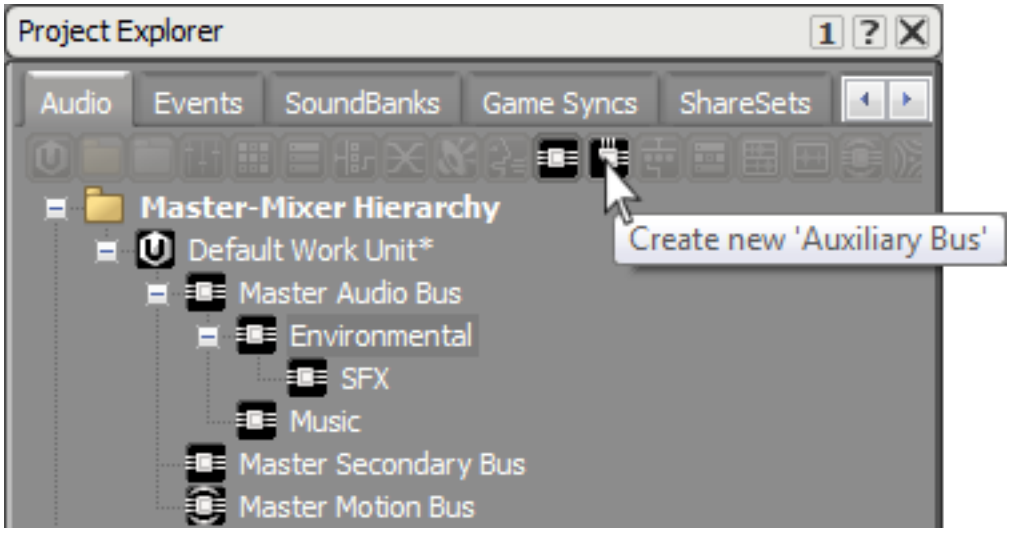
Auxセンド（Auxiliary Send）を使う

Auxセンドは、オブジェクトがAUXバスと呼ばれる特別なバスへ個別に制御されたボリュームを送るのに使うことができる追加出力のことです。 AUXバスの最も一般的な応用は、数多くのソースに対してリバーブやエコーなどのスペシャルエフェクトを適用することです。AUXバスに送られる信号が、影響を受けていないドライな信号に加算されるエフェクトをトリガーします。

キューブデモでは、アンビエント空間の種類を表現する定義済みのゾーンで構成されたワールド内をプレイヤーが移動します。プレイヤーがこれらゾーンの一つに侵入した際に、ゲームは特別なWwiseコールを送り、巨大なホールにおけるリバー

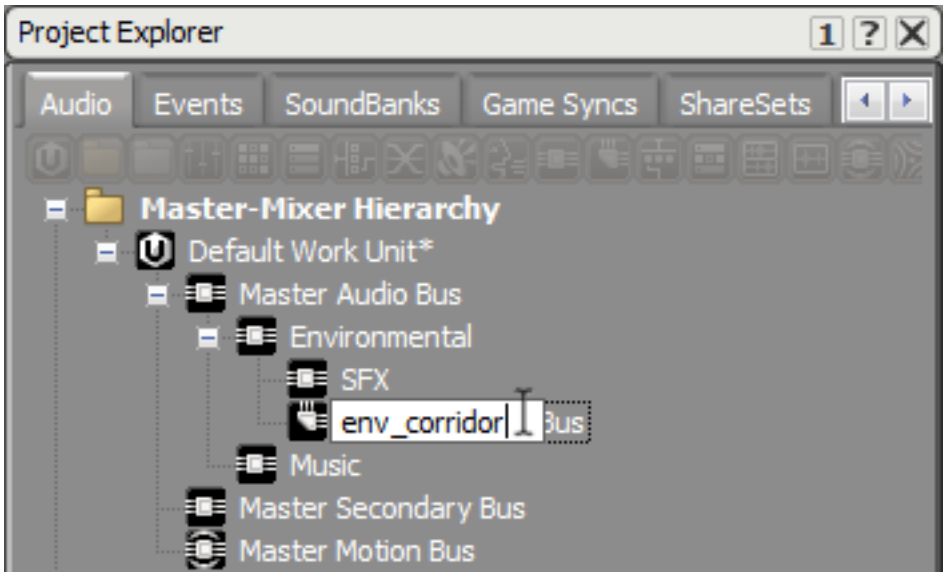
ブ効果など、対応するアンビエントエフェクトが設定されたAUXバスをアクティベートすることができます。このような動作を実現するために、リバーブエフェクトをWwiseで設定します。まず、リバーブエフェクトが格納されるマスターミキサー階層内に、AUXバスを作成します。リバーブは環境の一部であり、あなたはこの新規バスを先ほど作成した環境バス内に作成します。

1. 環境バスオブジェクトを選択し、**Create new Auxiliary Bus**をクリックします。



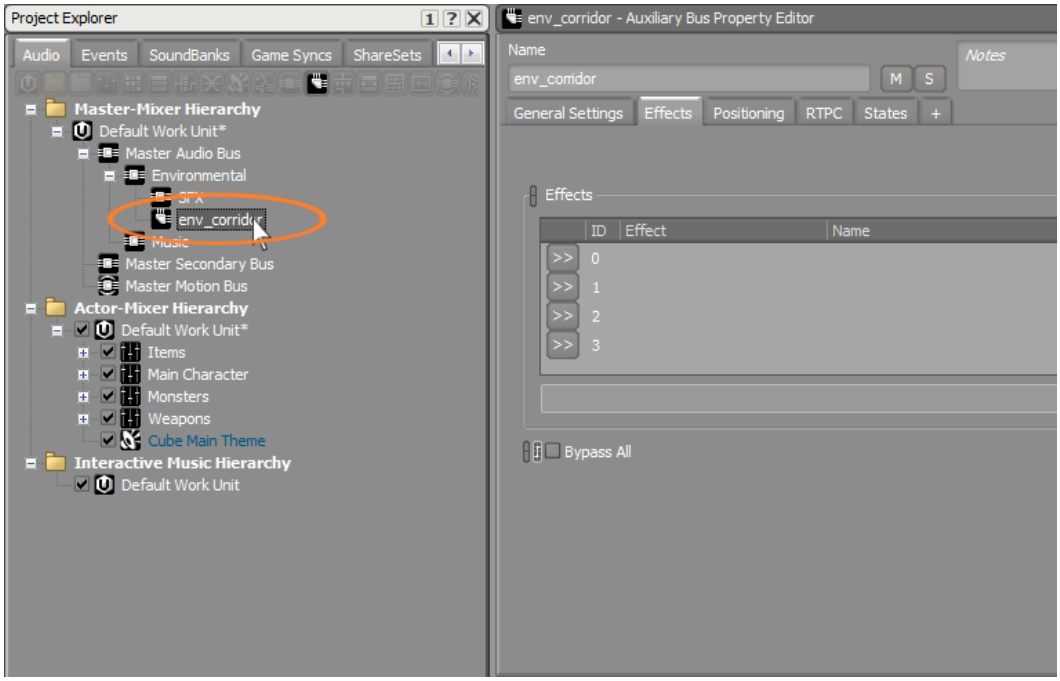
ゲームプレイ中では、アンビエントリバーブエフェクトは、ゲームマップ中の様々な通路をウォークスルーされる際に特にその効果がはっきりとわかると思います。

2. 新規バスを**env_corridor**と命名します。



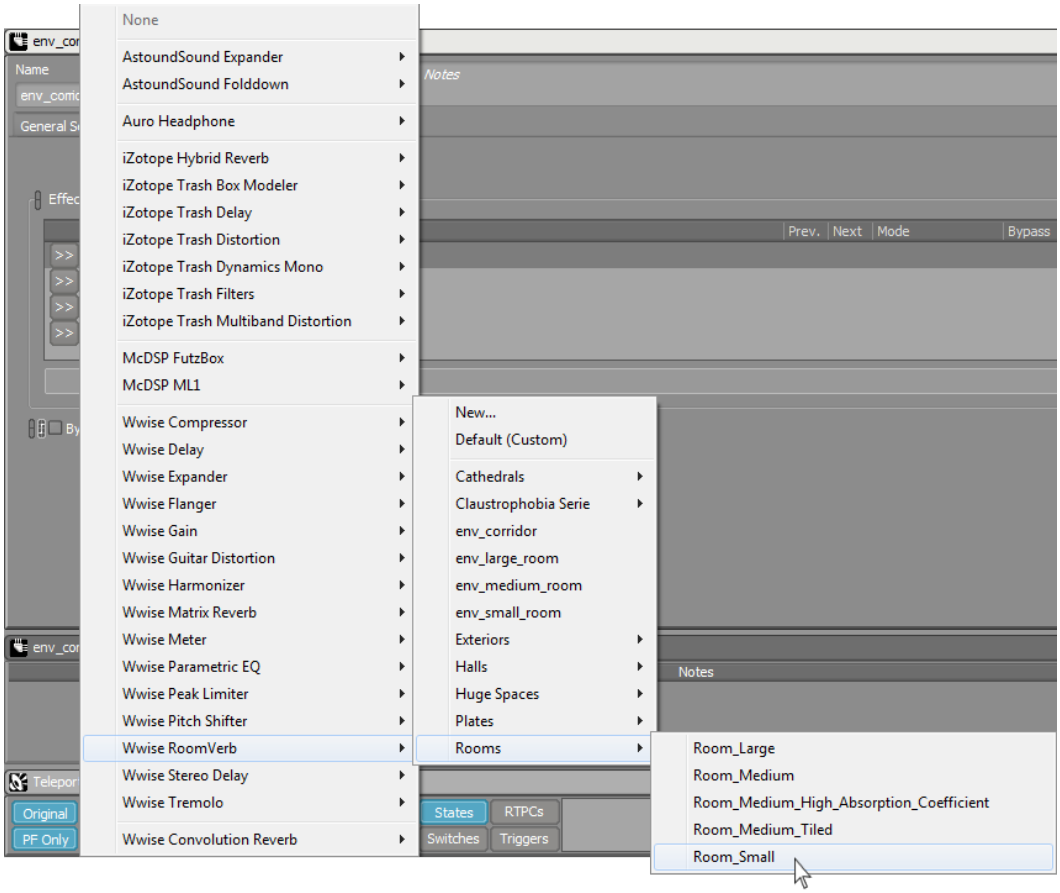
これでAUXバスができましたので、リバーブを生成するのに使う、エフェクトプラグインを追加する必要があります。

3. 新規AUXバスを選択し、プロパティエディターのEffectsタブを選択します。

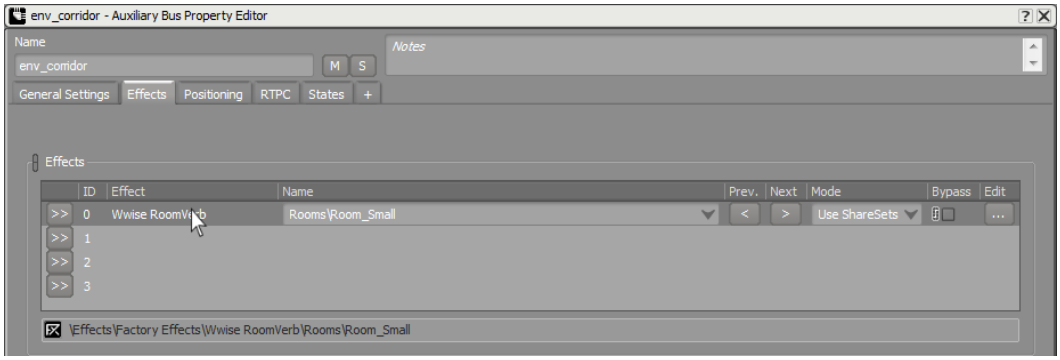


AUSバスプロパティエディター内でエフェクトプラグインがインストール可能な４つのポジションがあるエフェクトセクションがあるのが確認できます。Wwiseにはたくさんの種類のエフェクトプラグインが提供されています。次にベーシックなリバーブをインストールし、環境内で聞こえる全てのサウンドに空間感覚を追加します。

4. Effectsエリアで、最初のエフェクト行のセレクターボタンをクリックし、**Wwise RoomVerb** > **Rooms** > **Room_Small**を選択します。

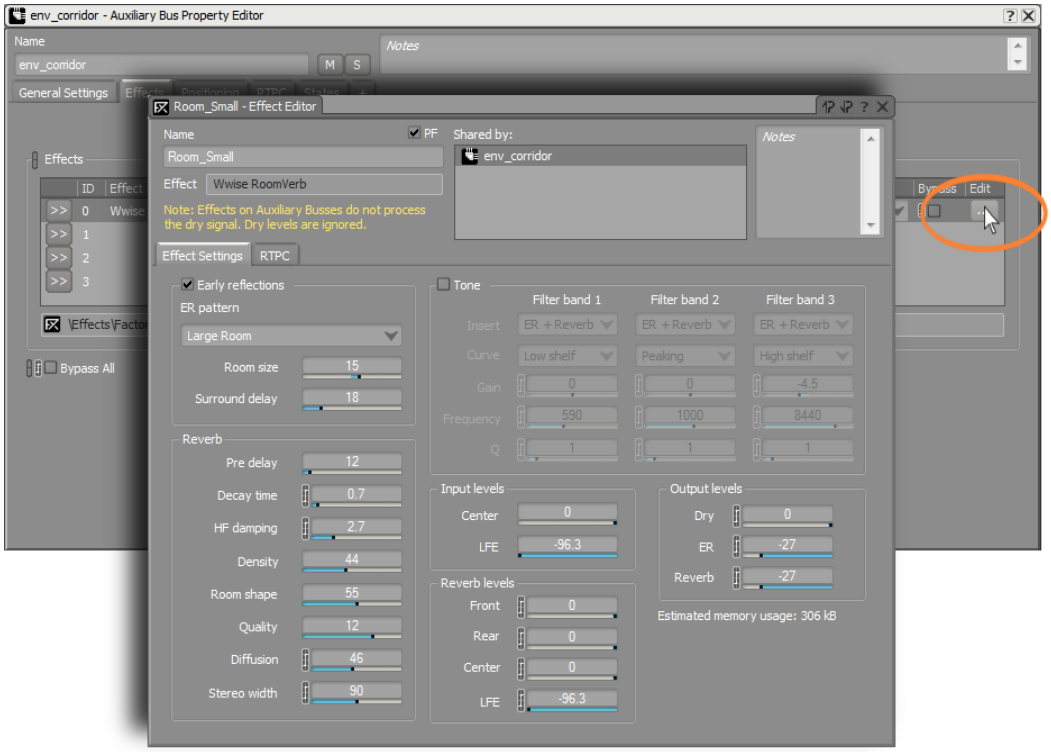


このエフェクトが最初の行にインストールできました。



エフェクトプラグインには様々なパラメータがあり、サウンドをカスタマイズするのに使用します。

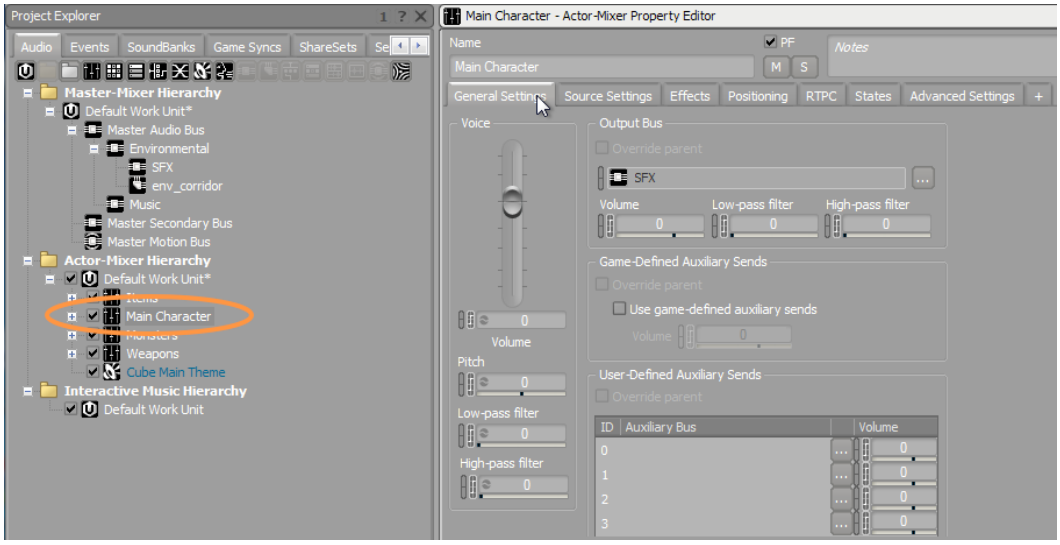
5. Effect行の右にある**Edit** をクリックします。



Wwise RoomVerb プラグインに関連する全てのパラメータが確認できます。後に、ゲームをプレイした際にリバーブを確認し、またここに戻ってディケイ時間や周波数ダンピングなど、お好みでルーム空間サウンドをカスタマイズし、調整することが出来ます。

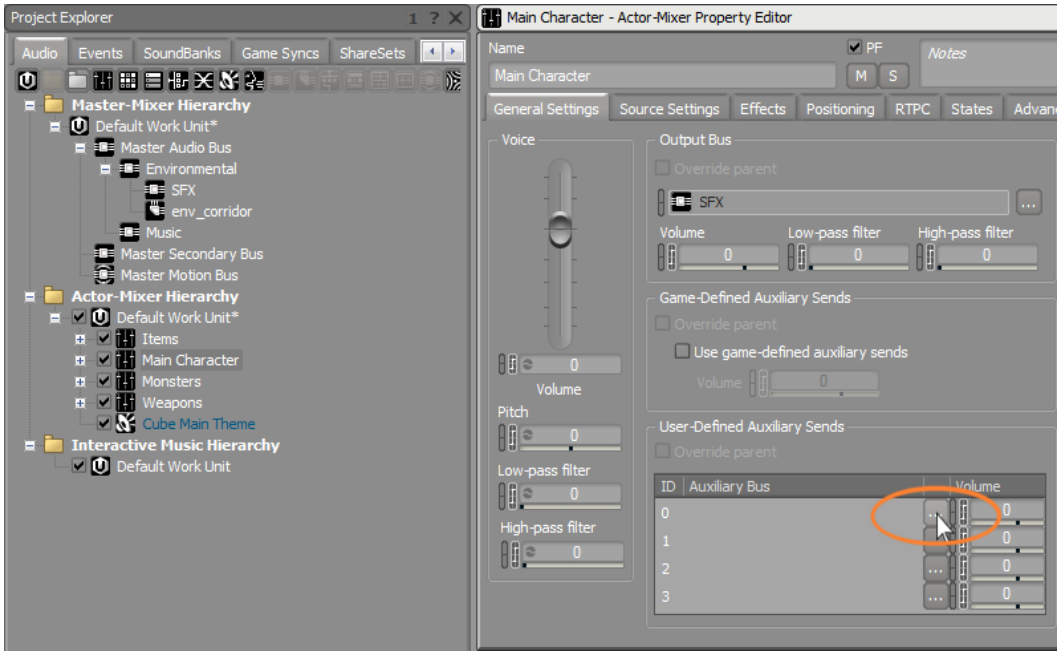
ここで、アクターミキサー階層のどのサウンドSFXオブジェクトが、いま構成したAUXバス内のスモールルームリバーブへ信号を送るのかを選択します。

6. エフェクトエディターを閉じます。
7. アクターミキサー階層内で、**Main Character**アクターミキサーを選択し、プロパティエディターのGeneral Settingsタブを表示します。

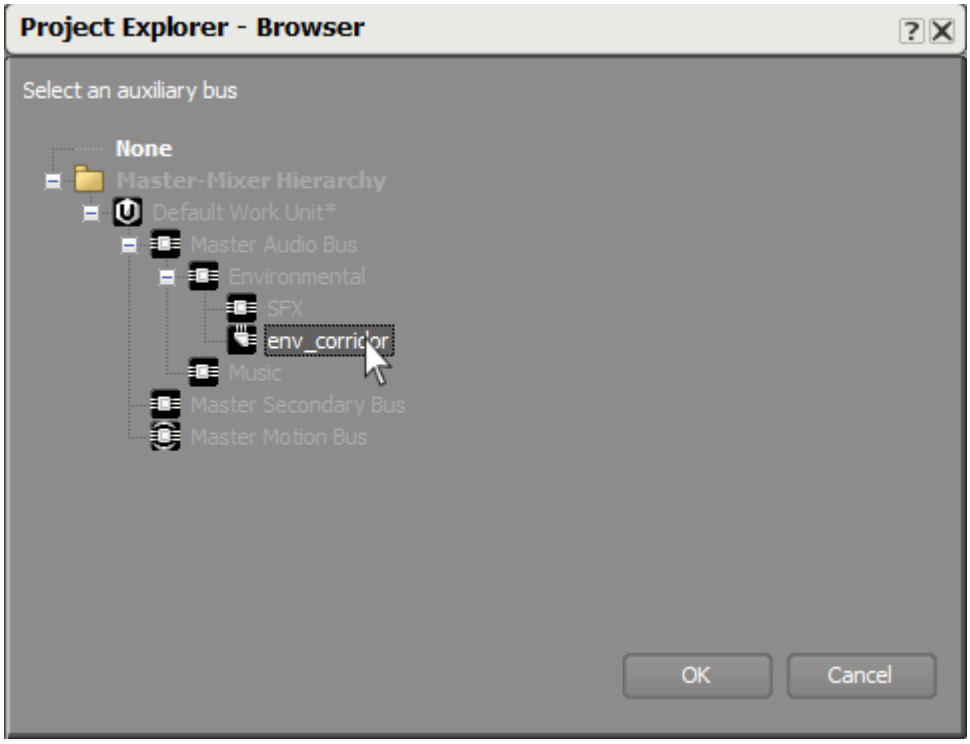


次に、信号を送るAUXバスをアサインします。これには2つの方法があります。最初のオプションは、Wwise内でマニュアルでこれをアサインする方法です。

8. 最初のAUXバス行で、**Browse**をクリックします。



プロジェクトエクスプローラーが開き、マスターミキサー階層と選択可能なAUXバスオブジェクトが表示されます。



9. **env_corridor**AUXバスを選択し、**OK**をクリックします。

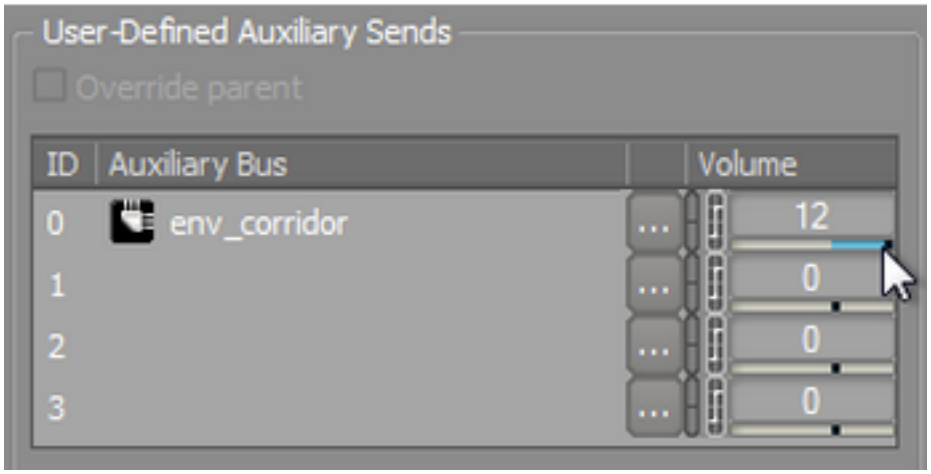
ティップ

AUXバスオブジェクトを、プロジェクトエクスプローラー から User-defined Auxiliary Sends listにドラッグ・アンド・ドロップすることで、素早くセンドを追加することが出来ます。

これで、メインキャラクターアクターミキサーに包含されているオブジェクトを再生する際に、あなたが設定したルームリバーブエフェクトが聞こえるようになりました。

10. **メインキャラクターアクターミキサー**内のオブジェクトを選択し、再生してリバーブエフェクトを聞いて確認してみて下さい。

リバーブエフェクトを増やしたり減らしたりするには、env_corridorアサインの右にあるボリュームコントロールをUp/Downして調整します。



11env_corridor volumeを上げて、 同じ音でリバーブを聞いてみて下さい。

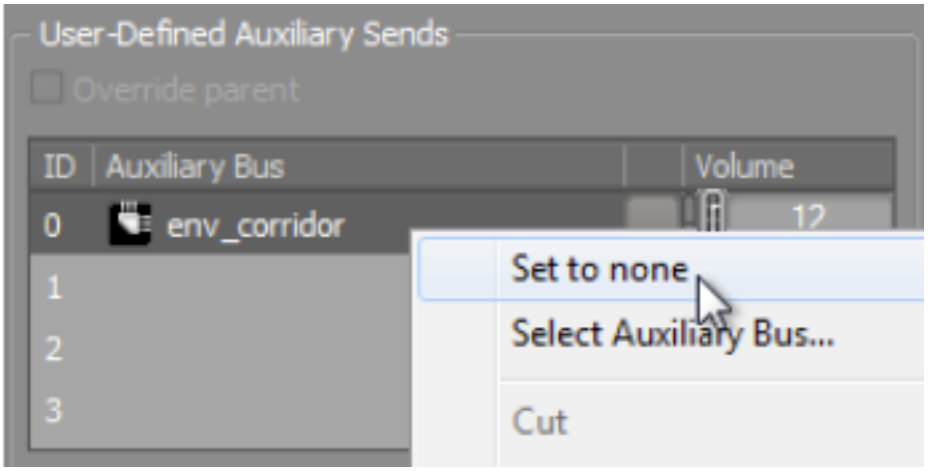
注記

これでリバーブが聞こえるようになりましたので、手順4で確認したリバーブプラグイン設定に戻り、お好みの設定に調整します。

オブジェクトがAUXバスへシグナルを送る2つめの方法は、ゲーム自体がWwiseへメッセージを送り、サウンドが、どのAUXバスのどのレベルに送られるべきかを指定します。このようにCubeデモが動作しています。 As you walk through the maps in Cubeデモのマップをウォークスルーする際に、レベルエディターにはゾーンが設定されており、プレイヤーが特定のゾーンに入った時に、特定のAUXバスへ送られるAUXセンドボリュームが増加し、一方でプレイヤーがそのゾーンを出たときには減少します。こうして、プレイヤーの物理的な位置情報にもとづいて、リバーブをダイナミックに変化させることが出来ます。

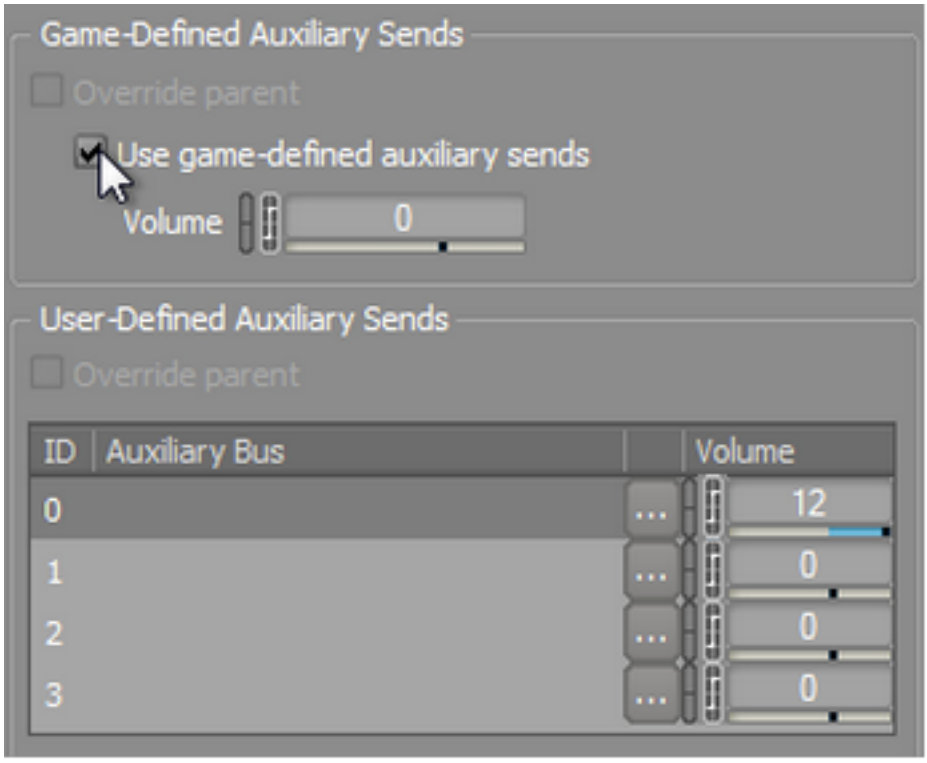
この手法を使用するには、まずUser-Defined Auxiliary Sendsセクション内のenv_corridor のアサインを外す必要があります。

12env_corridorアサインを右クリックし、**Set to none**を選択します。



Game-Defined Auxiliary Sendsを使用するには、チェックボックスでアクティベートするだけです。

13全てのアクターミキサーに対して**Use game-defined auxiliary sends**を選択します。



これで、ゲームをプレイしている際に、メインキャラクターアクターミキサー内に包含されているオブジェクトはenv_corridorのAUXセンドをUp/Downするタイミングを知らせるWwiseゲームコールを認識することができます。 本レッスンの最後にCubeデモをプレイしてもらいますが、マップの異なるエリアを移動する際にリバーブエフェクトがかかったり、外れたりするのを聞いて確認できます。

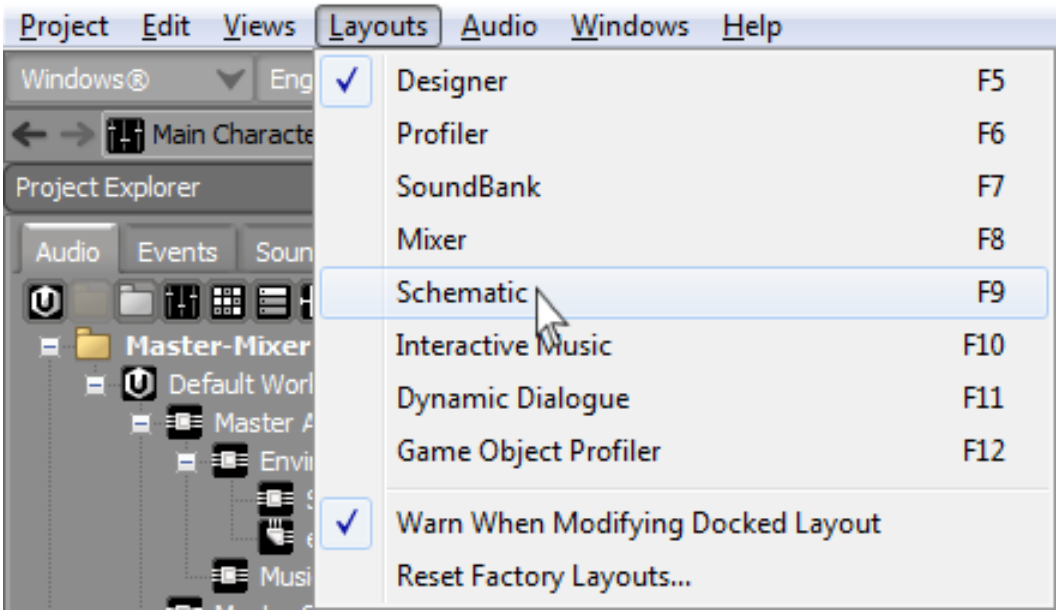
ティップ

Use game-defined auxiliary sendsチェックボックスの下のボリュームコントロールは、ゲームエンジンから送られる値にオフセットを加えるのに使用し、プログラマに調整をお願いすることなく、あなた自身がお好きなように調整できるようになっています。

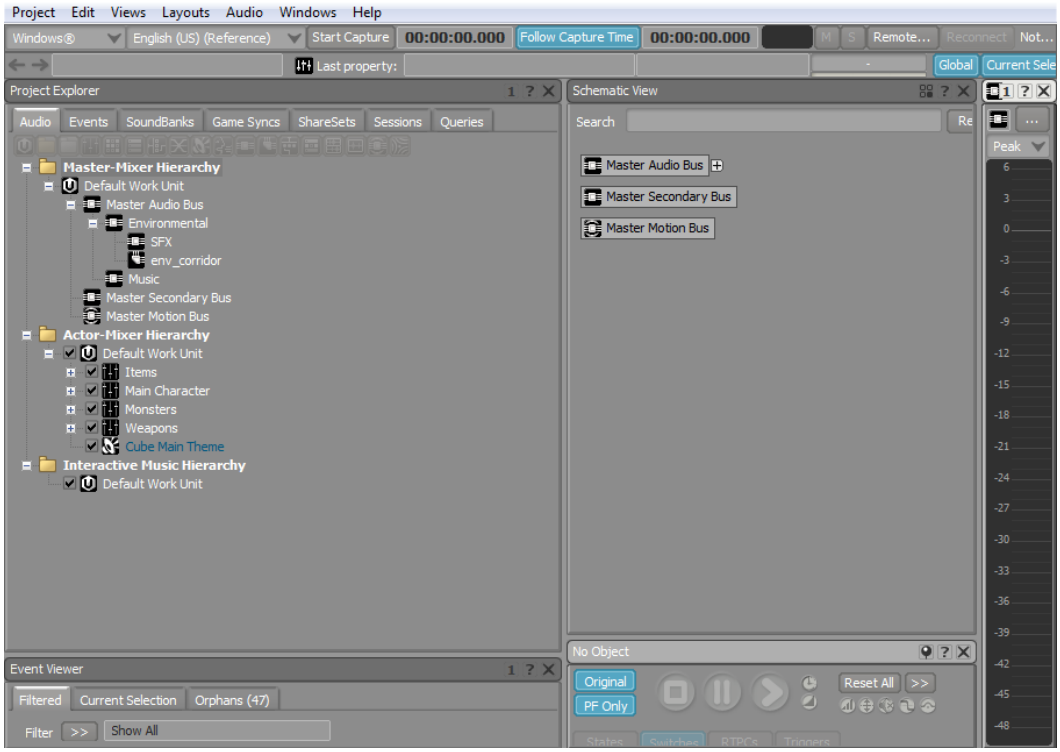
スケマティックビューの使用

本レッスンでは、アクターミキサー階層とマスタミキサー階層がどのようにしてアウトプットバスアサインを使用して、ブリッジできるかを確認しました。これらの階層はプロジェクトエクスプローラー内でお互いの上にスタックされているので、マスターミキサー階層のバスオブジェクトが、どのようにアクターミキサー階層のオブジェクトと関係づけられるのか、はっきりしないことがあります。この関係を見やすくするために、Wwiseではスケマティックビューを用意しています。

1. メインメニューバーから**Layouts** > **Schematic**を選択、もしくは **F9**を押します。



デフォルトで、スケマティックビューでは、Wwiseシグナルフロー構造の最上位である、マスターオーディオバスが表示されます。

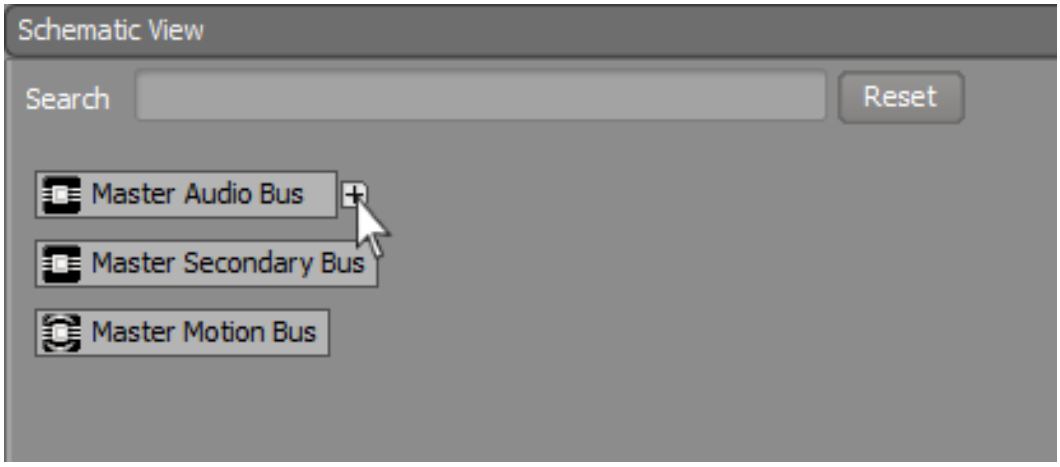


注記

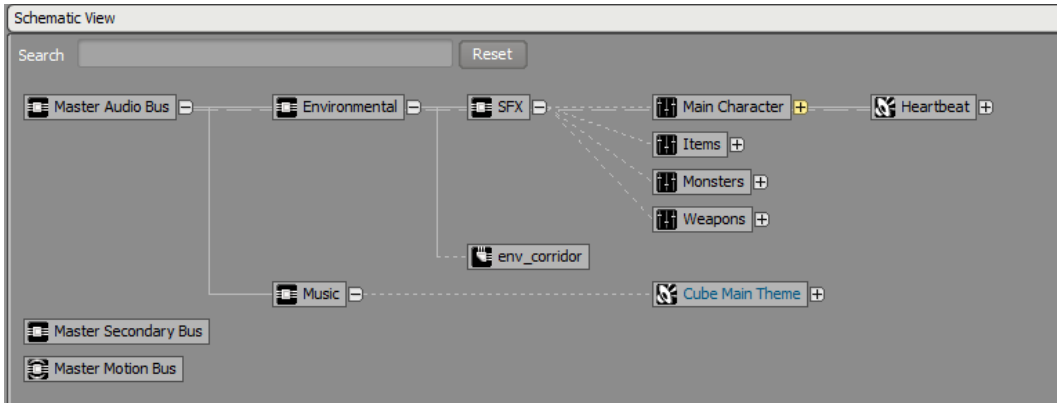
マスターセカンダリーバスは、ゲームコントローラーに実装されているようなセカンダリーオーディオ出力を持つゲームシステムで使用します。同様に、マスターモーションバスは、ゲームコントローラーに実装されたランブル振動機能のように、物理的な動きに対応したデバイスに使用します。

バス階層をエクスパンドすると、いずれのサウンドも元音源までたどることが出来ます。

2. マスターオーディオバスをエクスパンドします。

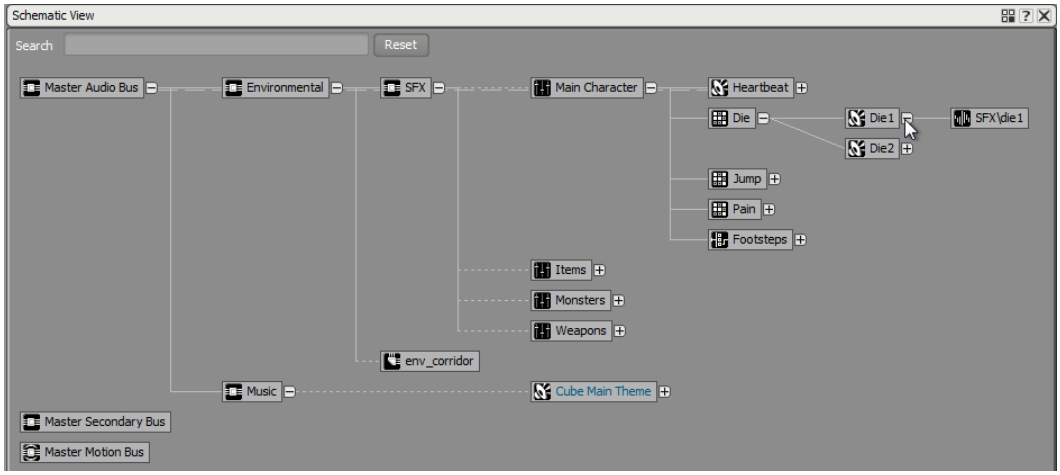


3. Music, Environmental, 及び SFXバスをエクスパンドします。



プロジェクトエクスプローラーで確認できるように、どのようにオブジェクトが相互に関係づけられているのか実線で示されています。小破線で、オブジェクト間のオーディオルーティング関係を示しています。プロジェクトエクスプローラーではわかりにくかった、Cube Main Theme Sound SFX オブジェクトがMusicバスにルーティングされている様子がはっきりと見てとれると思います。心拍音（Heartbeat)から出ているような、大破線で親アウトプットをオーバーライドするアウトプットを持つオブジェクトを示しています。メインキャラクターアクターミキサー内に包含されている心拍音を、直接マスターオーディオバスへアサインしたことを覚えていますか。

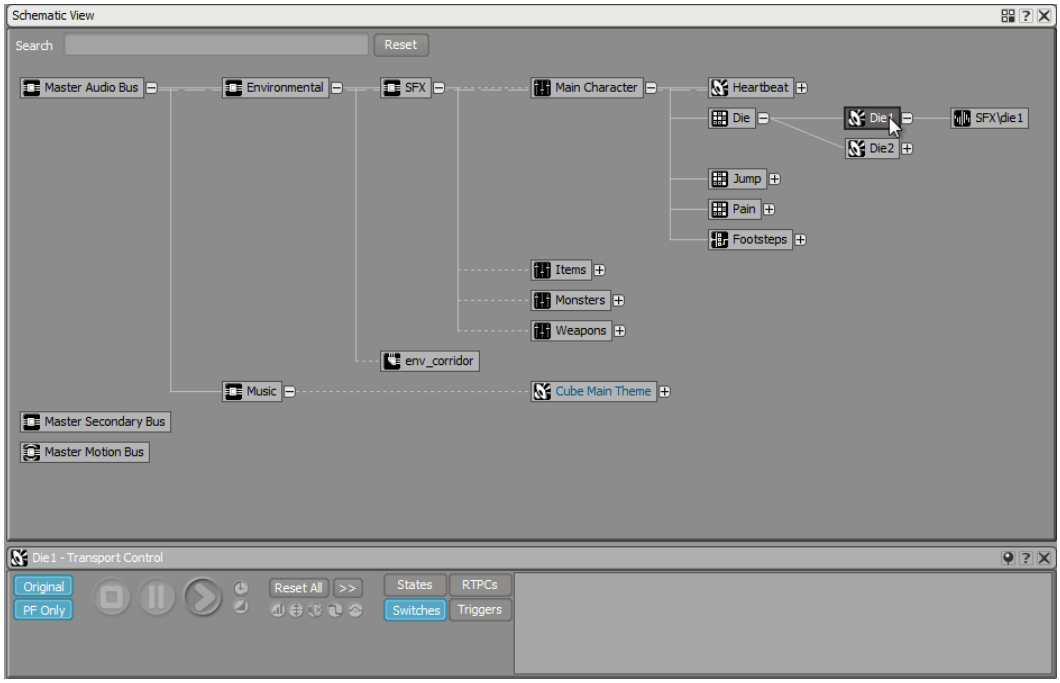
4. メインキャラクターアクターミキサーをエクスパンドし、次にDie ランダムコンテナー、最後にDie 1 Sound SFX オブジェクトをエクスパンドします。



これでSFX\Die 1 オーディオソースファイルが見えるようになり、右から左へ追っていくことでマスターオーディオバスまでのパスが全てトレースできます。

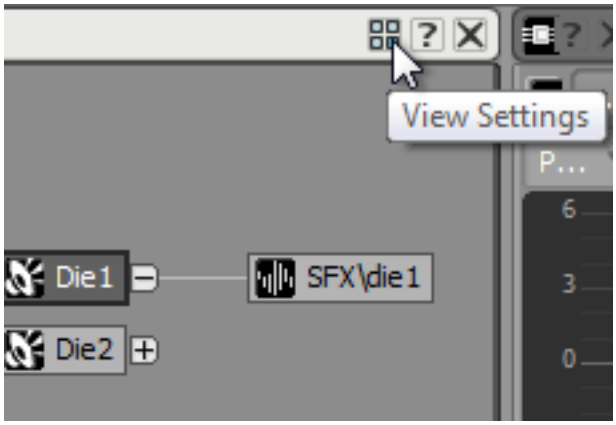
Sound SFXオブジェクトを選択し、再生することで、すぐにオーディション試聴することができます。

5. **Die 1 Sound SFX**オブジェクトを選択し、再生します。

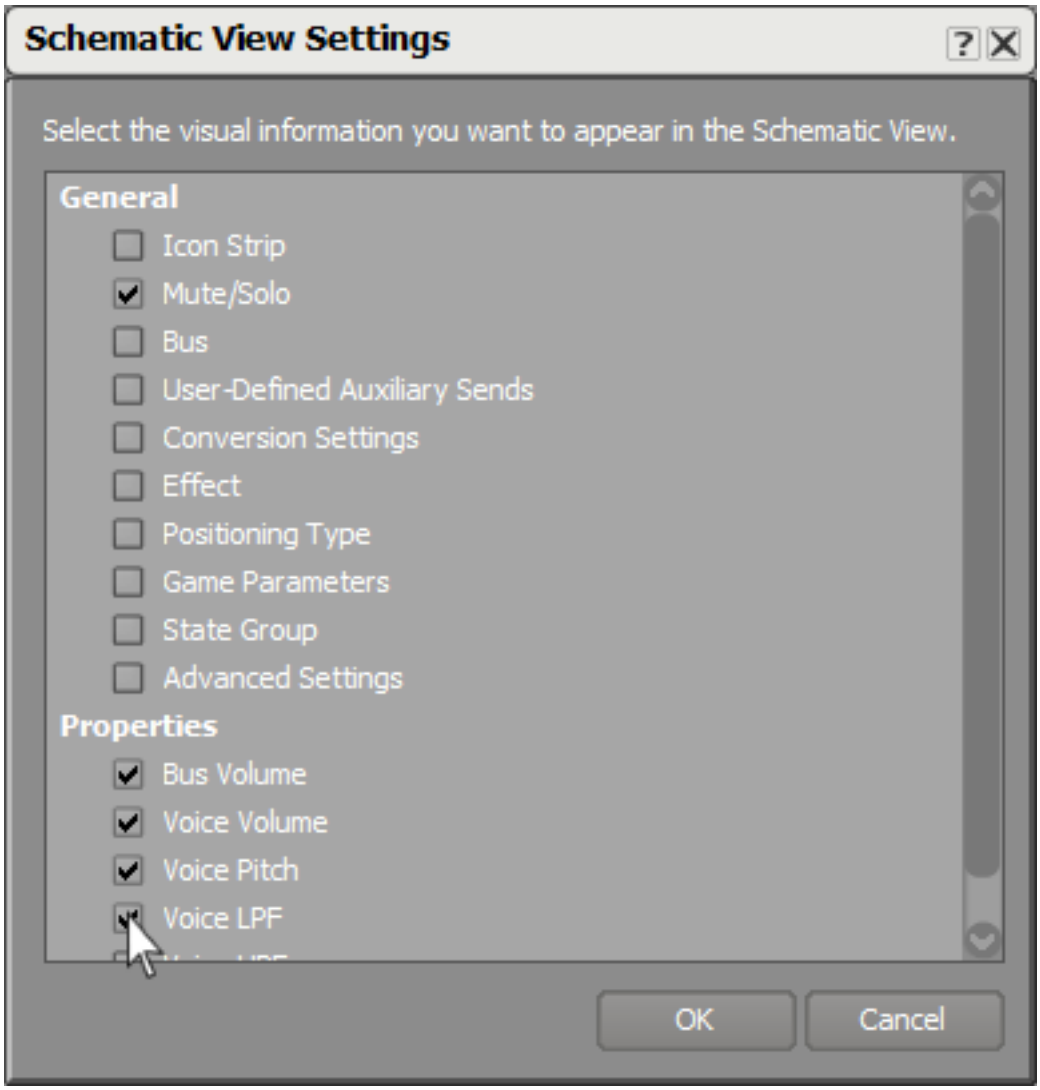


サウンドを試聴する際に、このスケマティックビューから直接、共通オブジェクトプロパティを調整できることが非常に便利であることがおわかり頂けると思います。次に、各オブジェクトに独自のプロパティを追加します。

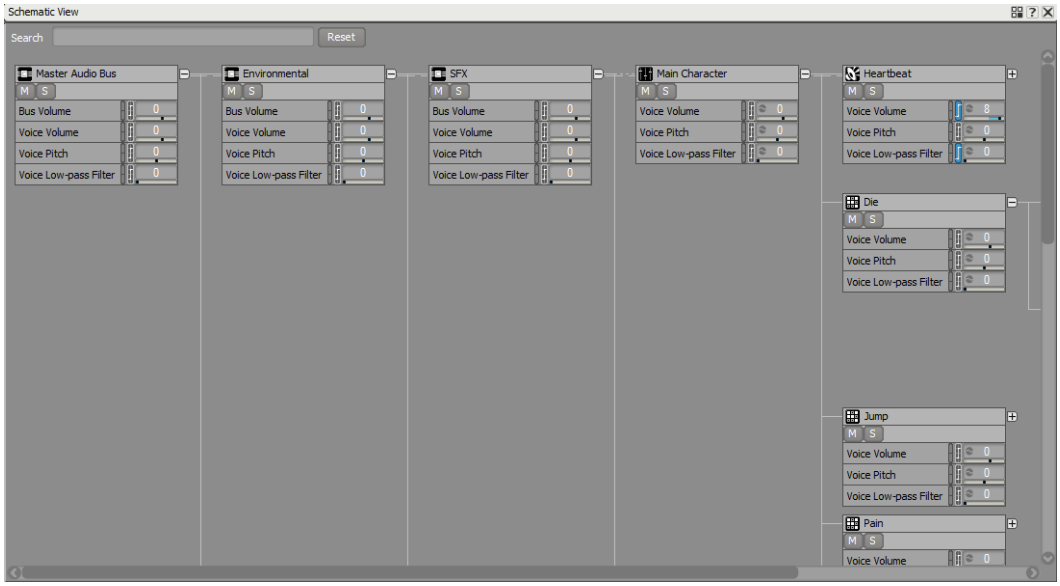
6. スケマティックビューの右上にある **View Settings**をクリックします。



7. Mute/Solo, Bus Volume, Voice Volume, Voice Pitch 及び Voice LPF チェックボックスを選択し **OK**をクリックします。



今、選択したプロパティが、各オブジェクトに追加されました。



これで、スケマティックビュー内で、これら共通設定を素早く調整できるようになりました。

8. 表示されたプロパティについて好きなように変更が可能です。
9. サウンドバンクを生成し、ゲームをプレイします。

あなたが加えた変更点に注意して聞いてみて下さい。マップ内であなたが作成したリバーブがかかったり、消えたりする場所を探してみてください。リバーブがトリガされるように定義された環境は、ゲーム開始時右手の通路にあります。

通路に入ると、右下のコーナーにenv 194525と表示されます。これは環境のID番号で、Wwiseにシグナルを送り、先にあなたが設定したgame defined AUXセンドをUp/Downします。

ティップ

回廊でのリバーブ音は想定通り抑え気味です。回廊でジャンプしてみてください、あなたのプレイするキャラクターの出すうめき声にリバーブがかかっているのが聞こえやすいと思います。もし、エフェクトが聞こえにくいようであれば、Wwiseプロジェクトに戻って、ミュージックを少し落とすことで、聞こえやすくなると思います。

レッスン 6：ミックスの仕上げ

サウンドキャスターの使用 220
ミキシングデスクの構築 229
コントロールサーフェスの使用 242

ゲームオーディオのインテグレーション作業を進めているうちに、ゲーム作品においてあなたが実装した様々なサウンドが相互に、どのように響き合うかという微妙なニュアンスにフォーカスする段階に到達します。個々のサウンドが、注意をそらすものではなく、貢献するようにすることが重要です。これがミックスの美といわれる領域です。

このプロセスと、映画制作においてサウンドがどのようにミックスされるかを比較対照してみてください。映画のオーディオでは、全てのオーディオがまとめられ、大きなミックスコンソールを使用してエンジニアが数百チャネルものサウンドを、映像を支える均一なサウンドスケープとなるように、全てがシームレスにブレンドされるまで調整をします。オーディオコンソールやDAWのミキサービューにあるチャンネルストリップのように、オブジェクトのプロパティをミキサーストリップとして表示できるカスタムバーチャルミキシングコンソールを作成することで、Wwiseでも同じようなワークフローを再現することが出来ます。さらに、これらのプロパティをMIDI対応のコントロールサーフェスの物理ノブやフェーダーにアサインすることで、調整するサウンドを触感的なコネクションを実現することが出来ます。このように、複数のプロパティを同時に操作できるのであなたのワークフローもスピードアップすること出来ます。

映画向けのワークフローとの絶対的な違いは、エンジニアがプロジェクトを再生することができ、全てのサウンドがタイムラインに設定された通りに再生されることです。一方、ビデオゲームでは、事前に決まったタイムラインが無く、サウンドはゲーム中で起こるイベントに従って再生されます。こうした理由で、複数のサウンドが相互にどのように聞こえ方に影響を与えるかをテストするのが多少難しくなります。

サウンドキャスターの使用

これまで、サウンドを再生するには、SFXサウンドオブジェクトなど対応するオブジェクトや、イベントオブジェクトを選択し、トランスポートコントロールビュー (Transport Control view) で再生をクリックしていました。複数のオブジェクトを同時に試聴するには、前のサウンドが聞こえるうちに、追加のサウンドを素早く複数回、繰り返し処理を行なうことが必要でした。Tこれをもっと容易にするために、トランスポートビューの複数インスタンスを事前アレンジし、それぞれのインスタンスが特定のオブジェクトをターゲットとすることで、異なる複数オブジェクトを素早く再生することができます。Wwiseのサウンドキャスター機能でこれを実現できます。

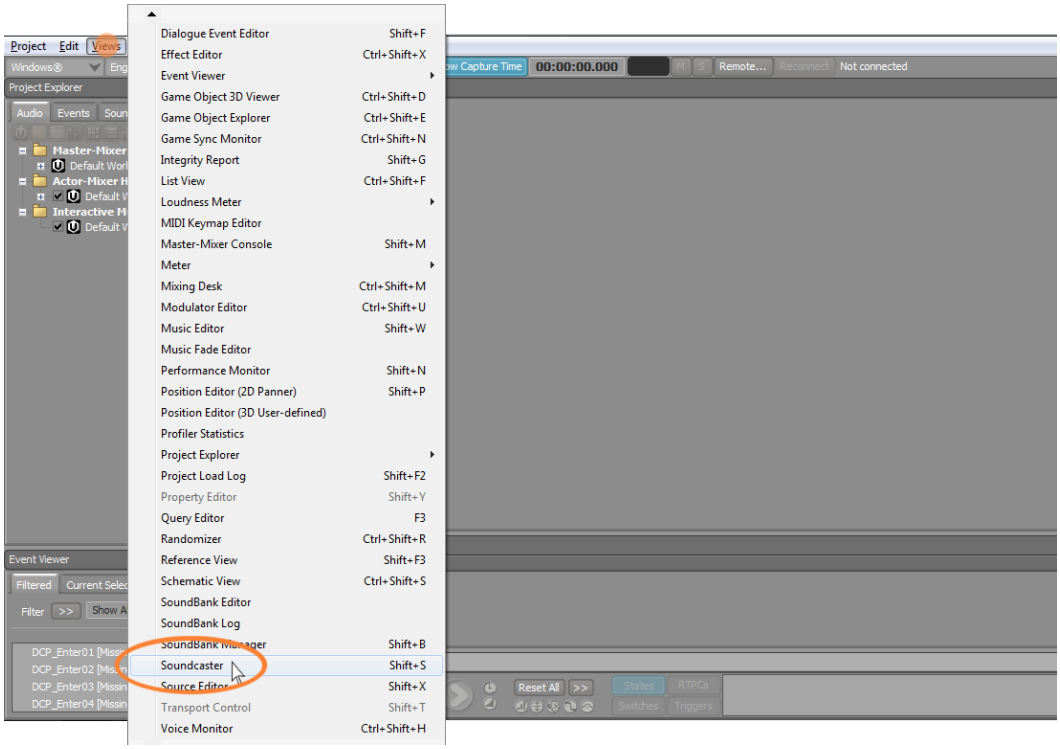
サウンドキャスターセッションの作成

最終的なゲームには数千ものオブジェクトが含まれることがあります。とはいえ、千個の再生ボタンがあるウインドウではあまり便利ではありません。実際に試聴したいものを見つけるのも一苦労でしょう。そうではなく、素早くアクセスしたいオブジェクトのプリセットレイアウトを構築します。このプリセットをサウンドキャスターセッションといいます。好きな数だけサウンドキャスターセッションを用意することができ、お互い関係のある複数サウンドのカスタムトランスポートレイアウトを作成し、再現することができます。

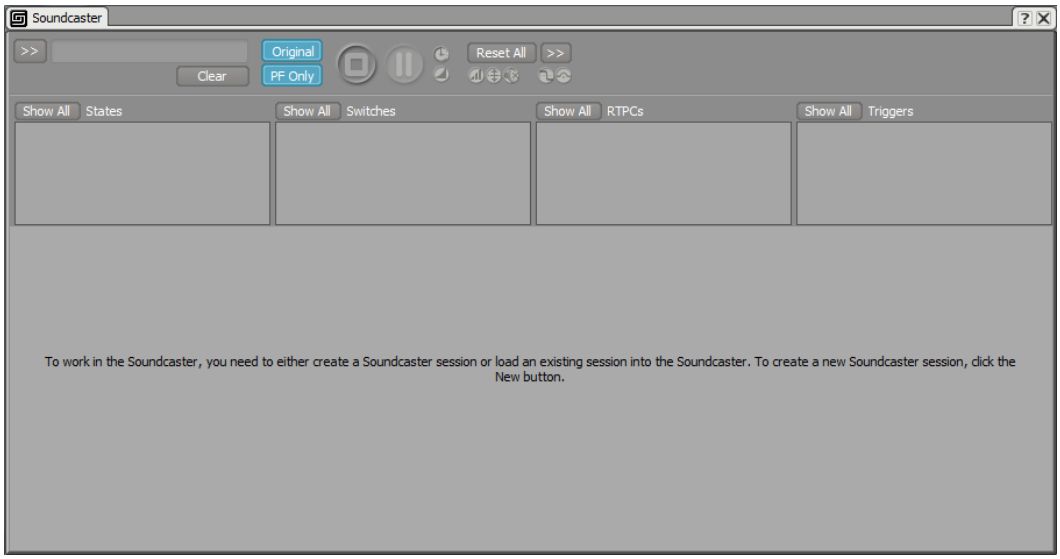
まず、ゲームのヒーローに直接関連付けられたサウンド、例えば足音や、武器のサウンドのサウンドキャスターセッションを作成します。こうして、足音が、ショッ

トガンの発射時に丁度いいボリュームであるかなど、素早く確認することが出来ます。

1. レッスン6プロジェクトの起動
2. メインメニューで、**Views** > **Soundcaster**を選択、もしくは**Shift+S.**を押します。



サウンドキャスターが開きます。

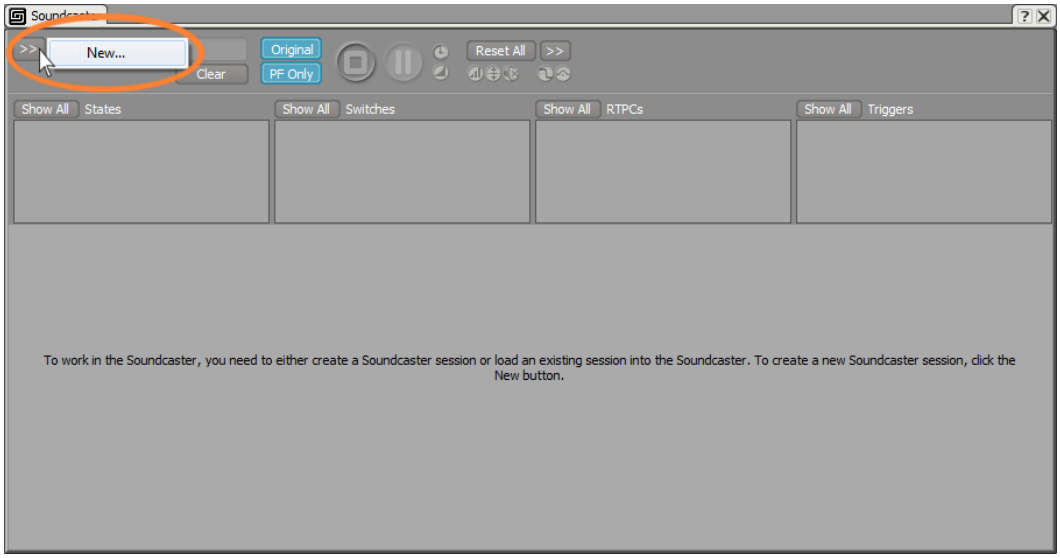


ウインドウの上部に、トランスポートコントロールが確認できます。そのすぐ下に、これまでのレッスンで学んだ、ゲームの全てのステート、スイッチ、RTPCがあります。また、インタラクティブミュージックに使用するTriggersもあります。これらのコントロールを使用することで、これらの値を変更したことが、再生するサウンドにどのように影響を与えるかを、素早く確認することが出来ます。このセクションの下に、新規サウンドキャスターセッションを作成する、空白のエリアがあります。

3. 左上のSelector menu [>>]をクリックし、そして**New**をクリックします。

注記

本ドキュメントでは以後、セレクターを使用してSelector menu [>>] オプションを示します。

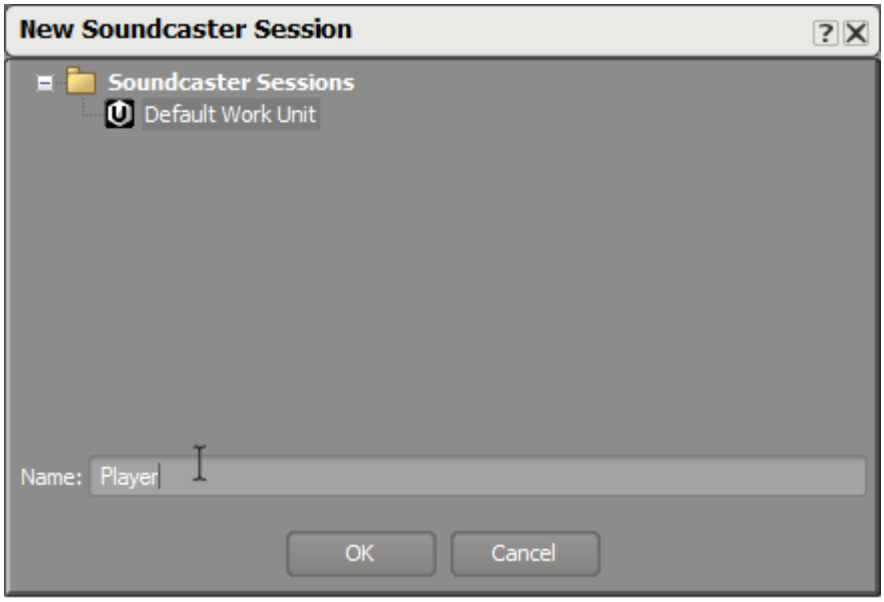


新規サウンドキャスターセッションダイアログが開きます。

サウンドキャスターセッションは、サウンドキャスターセッションフォルダー内のワークユニットに格納されます。

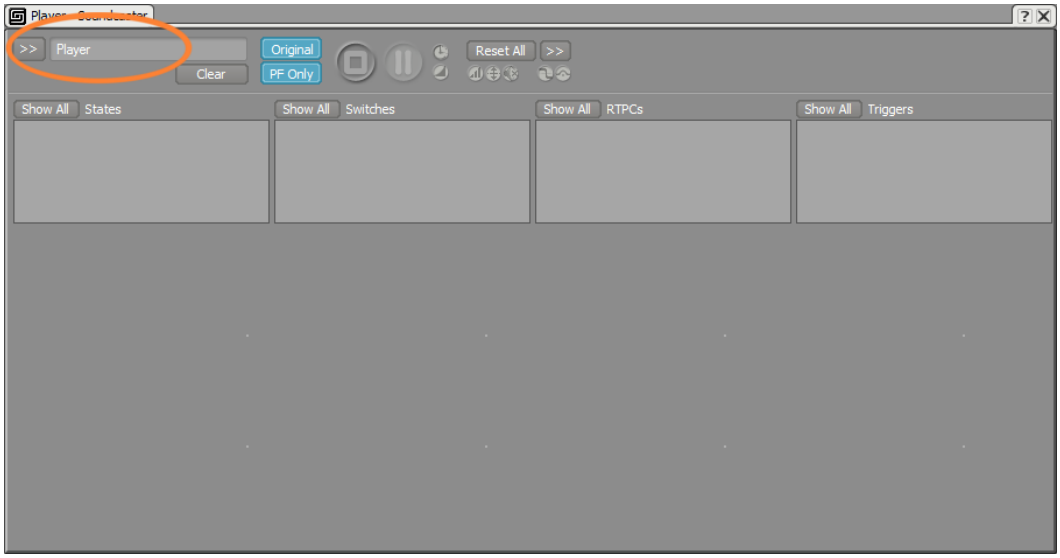
注記

プロジェクトエクスプローラーのセッションタブでサウンドキャスターセッションフォルダを確認できます。



4. デフォルトワークユニットを選択し、**Player**と入力し、**OK**をクリックします。

   新しいサウンドキャスターセッションが作成できました。



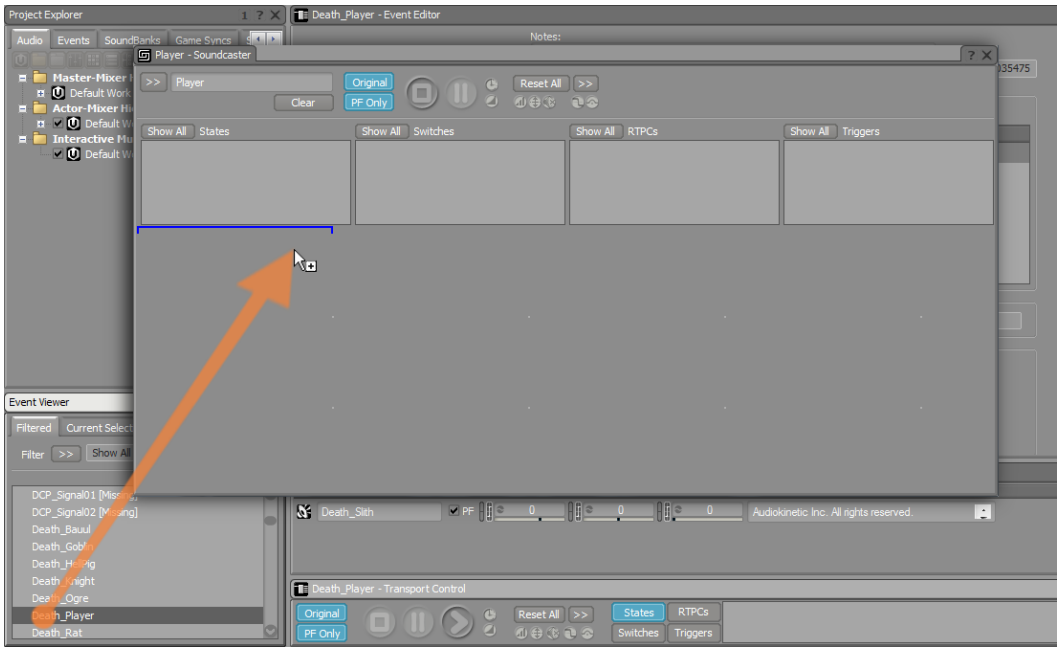
   サウンドキャスターの左上のタブがプレイヤーサウンドキャスターセッションがアクティブであることを示しています。しかし同ウインドウの下のセクションにはなんのオブジェクトトランスポートも表示されていません。よく見ると、このサウンドキャスターセッションで再生するオブジェクトをロードする場所を示すグリッドを構成するドットが見えると思います。

サウンドキャスターセッションにオブジェクトを追加する

サウンドキャスターセッションにオブジェクトを追加するのは簡単です。サウンドキャスターウインドウのグリッドの好きな場所にドラッグします。サウンドキャス

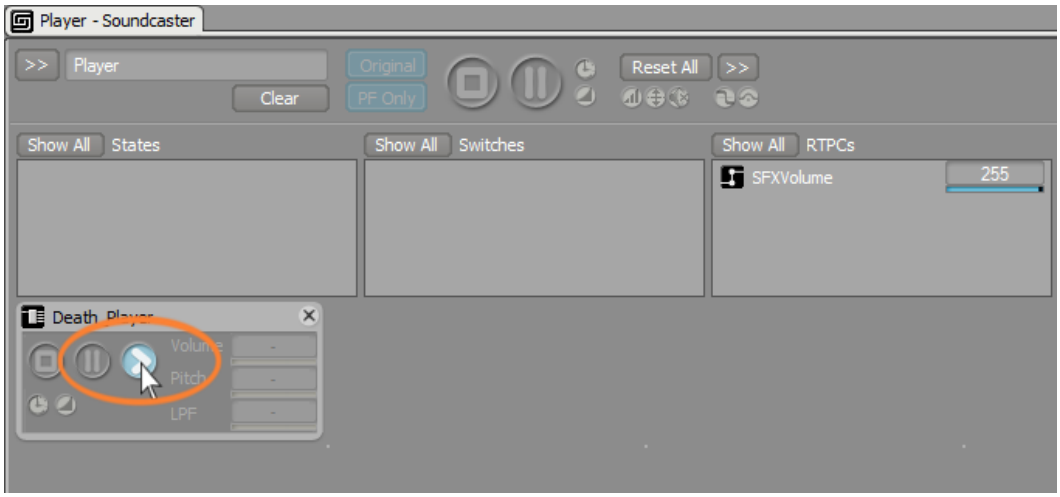
ターグリッドの左上のコーナーへイベントビューアーからDeath_Playerイベントオブジェクトをドラッグします。

1. サウンドキャスターセッションへ、 **Event Viewer**から**Death_Player**イベントをドラッグします。

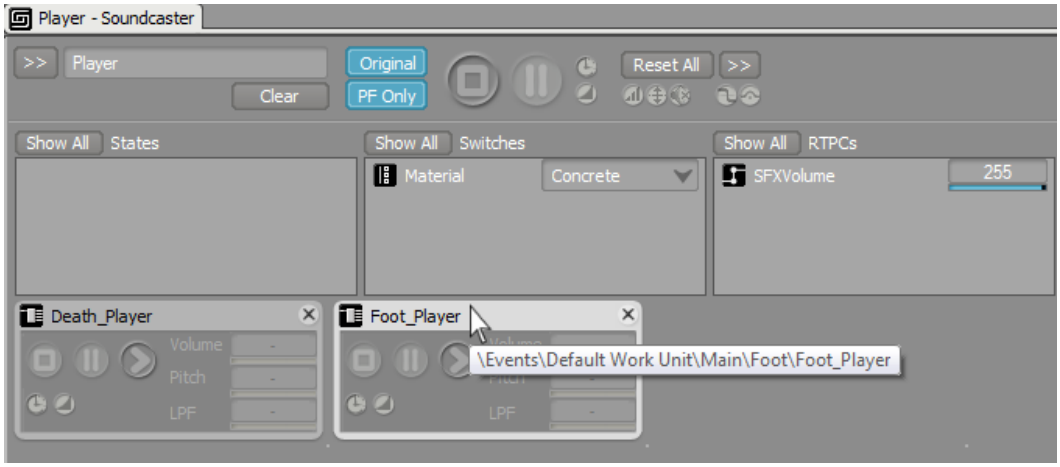


そのオブジェクトは、以前のレッスンで使用したトランスポートコントロールの小型版のようなオブジェクトで表現されています。サウンドキャスターの一番のメリットは、マルチトランスポートコントロールエリアともいえる場所へオブジェクトを継続して追加できることです。

2. サウンドキャスターモジュール内の再生ボタンをクリックし、**Death_Player**イベントを再生します。



3. サウンドキャスターセッションへ**Foot_Player**イベントを追加します。



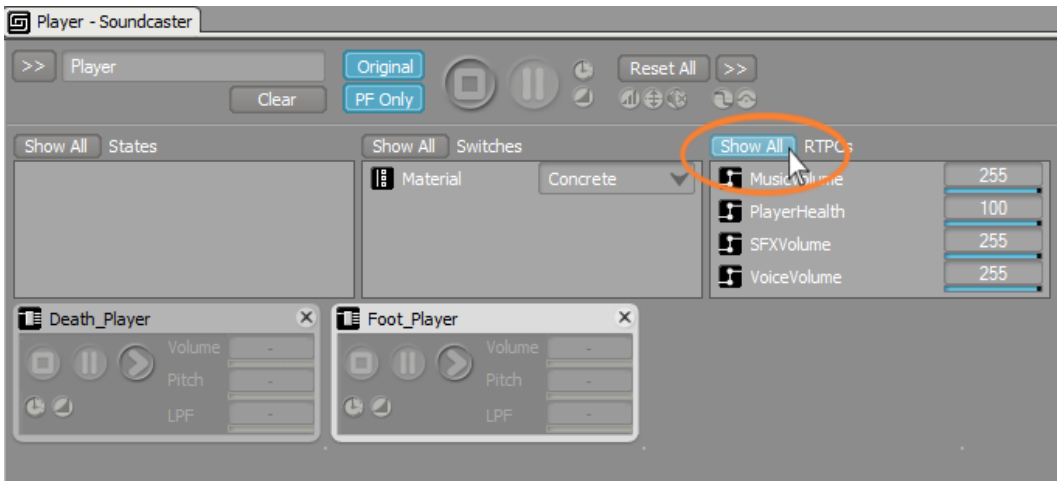
トランスポートコントロールの上に、States, Switches, や RTPCsなど各種ゲームシンクを表示するエリアがあります。Foot_Player イベントを追加した際にMaterialスイッチが現れたのを確認して下さい。これは、このイベントがスイッチコンテナーを含んでいるので、このスイッチ設定を素早く変更できるようになっています。

4. **Foot_Player** イベントを再生して、Materialタイプを変更してみて下さい。

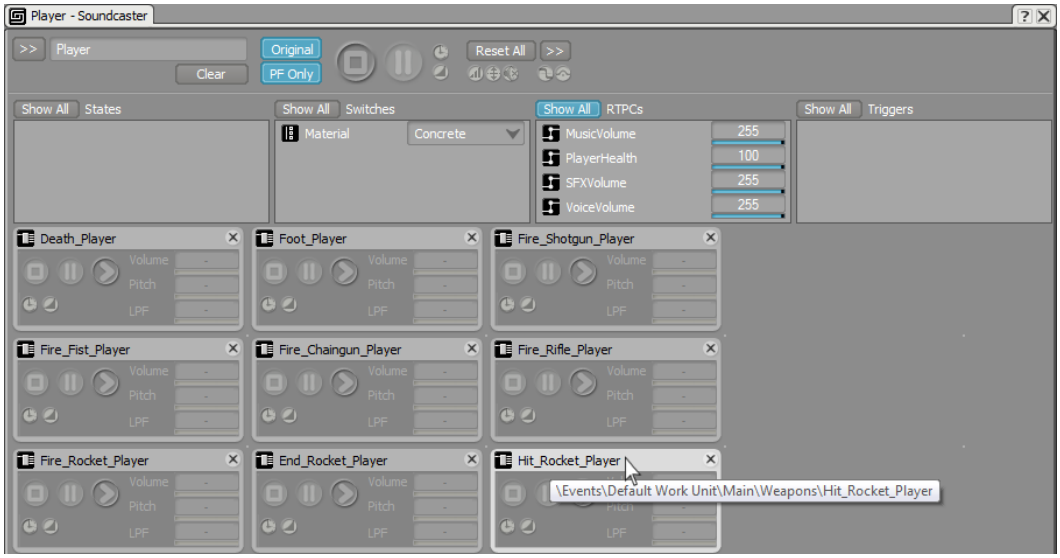
ティップ

デフォルトで、サウンドキャスターはロードしたオブジェクトに対応するゲームシンクのみを表示します。各**Game Sync** ヘッダの隣にある **Show All** ボタンを選択することで、強制的に特定のゲームシンクを表示させることが出来ます。

5. RTPCsエリアの**Show All**をクリックします。



6. 次のイメージのイベントをサウンドキャスターセッションに追加します。

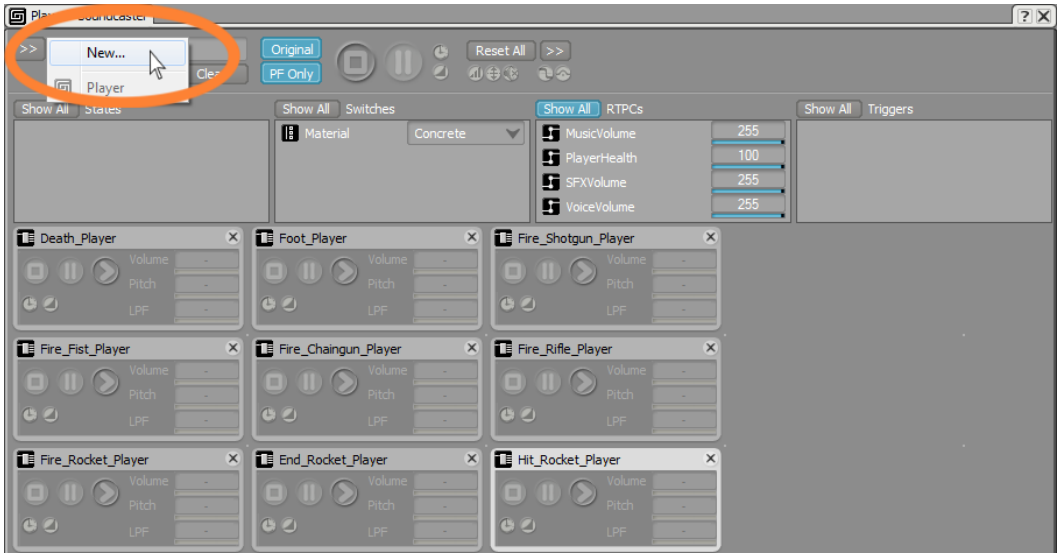


7. 様々なイベントオブジェクトの再生を試してみて下さい。

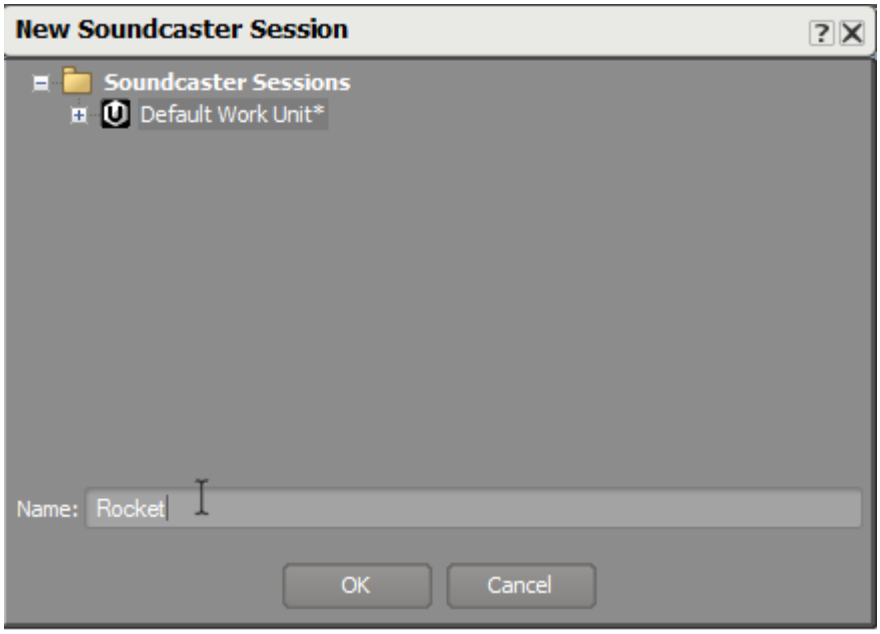
   サウンドキャスターモジュールの各オブジェクトの右側にボリューム、ピッチ、及び LPF (ローパスフィルター)のプロパティがあるのを確認して下さい。イベントオブジェクトにこれらのプロパティが含まれていないので、フィールドに値は表示されていません。これらのプロパティを調整するには、これらイベントオブジェクトでトリガされる様々なオブジェクトを含める必要があります。

   これがどのように動作するかを確認するために、ロケットランチャーと関連付けられたサウンドの微調整をするために設定するもう一つのサウンドキャスターセッションを作成します。

8. セレクターメニューをクリックし、 **New**を選択します。

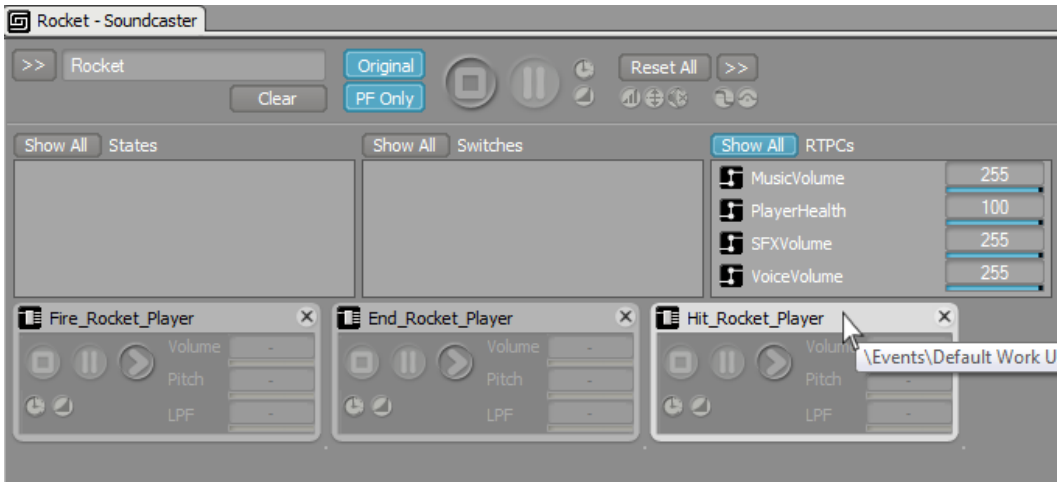


9. 新しいサウンドキャスターセッションを**Rocket**と命名します。



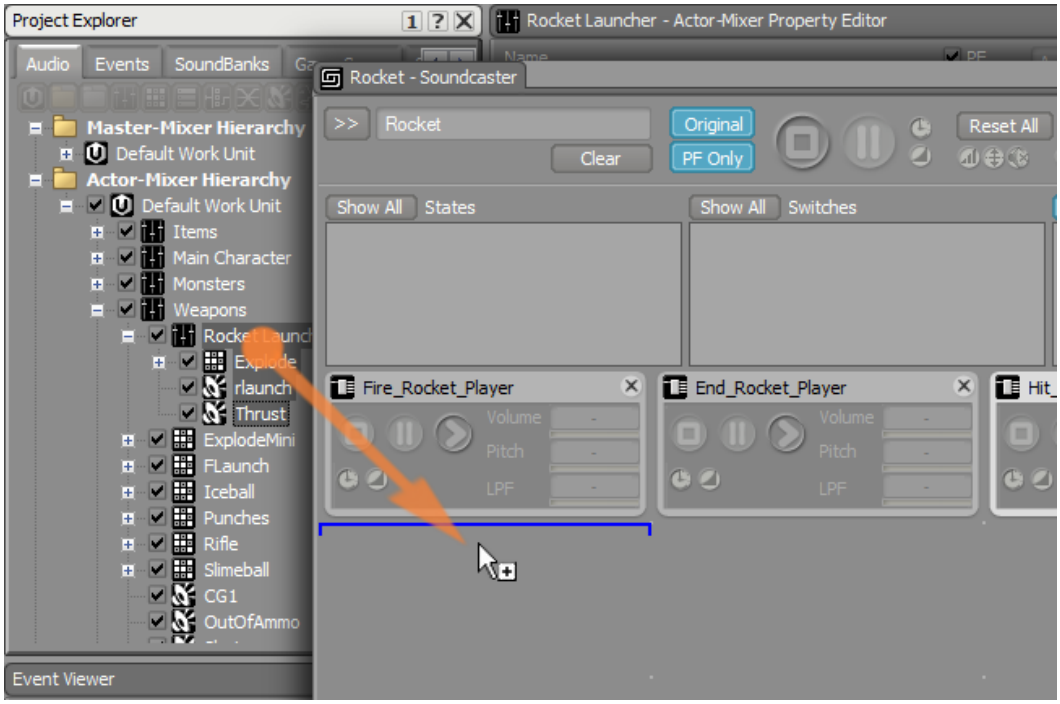
ロケットの使用においては、幾つかのイベントが必要で、一つはロケットを発射し、もう一つはロケットを終了し、最後の一つはモンスターに命中した際にトリガされるものです。

10. **Fire_Rocket_Player**、**End_Rocket_Player** 及び **Hit_Rocket_Player**イベントを、**Rocket** サウンドキャスターセッションに追加します。



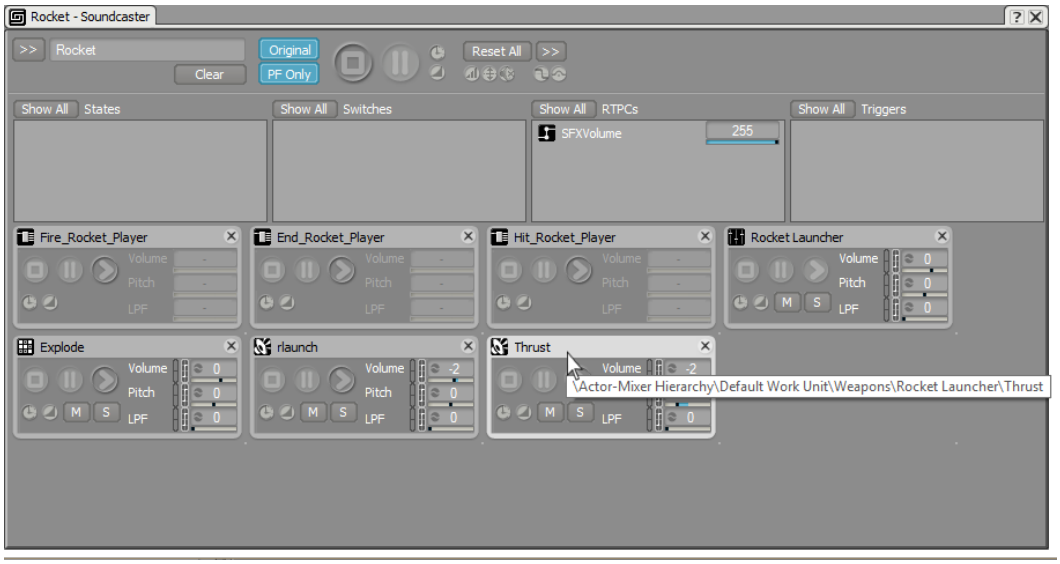
イベントオブジェクトを追加したので、まだボリューム、ピッチ、LPFプロパティを変更するオプションは表示されません。次に、これらイベントによりトリガされる関連オブジェクトを追加し、各プロパティの調整が出来るようにします。

11. プロジェクトエクスプローラー内のアクターミキサー階層から、**Rocket Launcher**、**Explode**、 **rlaunch**、及び**Thrust** オブジェクトをサウンドキャスターの第2行目に追加します。



これでボリューム、ピッチ、LPFプロパティが表示されるようになりましたが、調整をする前にこれらオブジェクトトランスポートグリッド上での配置を再度整理します。

12 **Rocket Launcher** Actor-Mixerオブジェクトを最初の列の最後に移動し、**Explode**、**rlaunch** 及び**Thrust** オブジェクトをグリッドの左側のスペースに移動させ、各イベントに関連付けられたサウンドがそれをトリガするイベントの近くに配置します。



ティップ

サウンドキャスターでは自由にオブジェクトを配置することが出来るので、プロジェクトエクスプローラーで見る階層構造を反映する必要はありません。例えば、ロケットランチャーアクターミキサーには、実際にrlaunch、Explode、及び Thrust オブジェクトが含まれており、サウンドキャスター内でRocketLauncher プロパティの変更を行なうことは、実際には、これら包含されているオブジェクトのプロパティ値にオフセットを加えることになります。

ロケットサウンドの微調整をする準備が出来ました。イベントオブジェクトを左から右へ順番に再生することで、プレイヤーがロケットを発射する際に起きることをシミュレーションすることができます。注意深く聞いてみて、thrust音が、ロケット発射音や、爆発音に対してバランスがとれているか確認します。使用可能なプロパティ値を使用してサウンドへ変更を素早く行なうことが出来ます。

13各種オブジェクトを再生し、異なるボリューム、ピッチ、LPF設定を試してみて下さい。

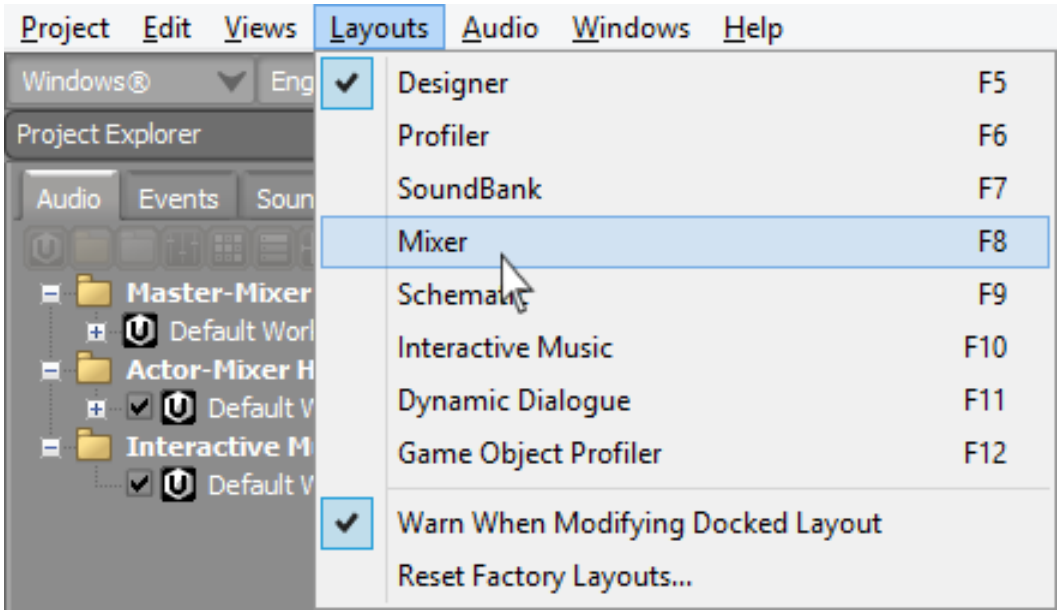
ミキシングデスクの構築

ミキシングデスク(Mixing Desk)は、柔軟かつパワフルなミキシングコンソールで、様々なプロパティを一つのビューにグループ化し、あなたのゲームのオーディオミックスの微調整を可能にします。作業対象の特定オブジェクト群をミックスデスクに配置し、オブジェクトのルーティングを定義し、エフェクトや減衰カーブを適用し、各オブジェクトやバスのプロパティを調整します。まずは最初にCubeデモで使用されるロケットサウンドに注目して、カスタムミキシングデスクを作成します。

新規ミキシングデスクの作成

ミキシングデスクビューは、Wwiseの他のウインドウと同様にビューとして開かれますが、前回のレッスンで作成したような、サウンドキャスターセッションと連携させると特に有用です。こうした理由から、ミキサーと呼ばれるデフォルトレイアウトは、サウンドキャスターびゃミキシングデスクビューの他に、プロジェクトエクスプローラーとイベントビューアーとして、素早くリコールすることができます。

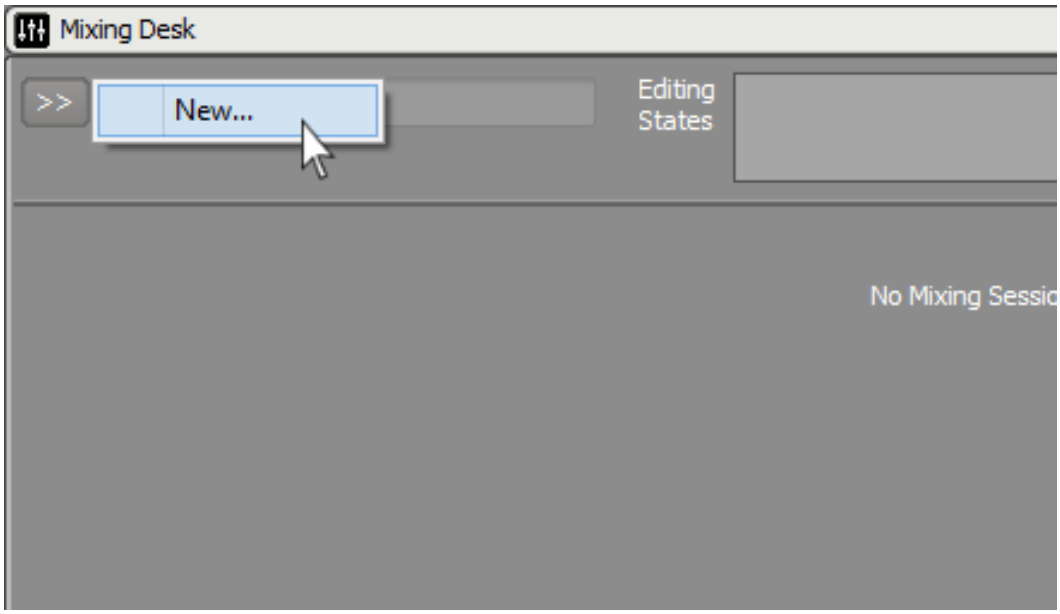
1. メインメニューで、**Layouts** > **Mixer**を選択、もしくは**F8**を押します。



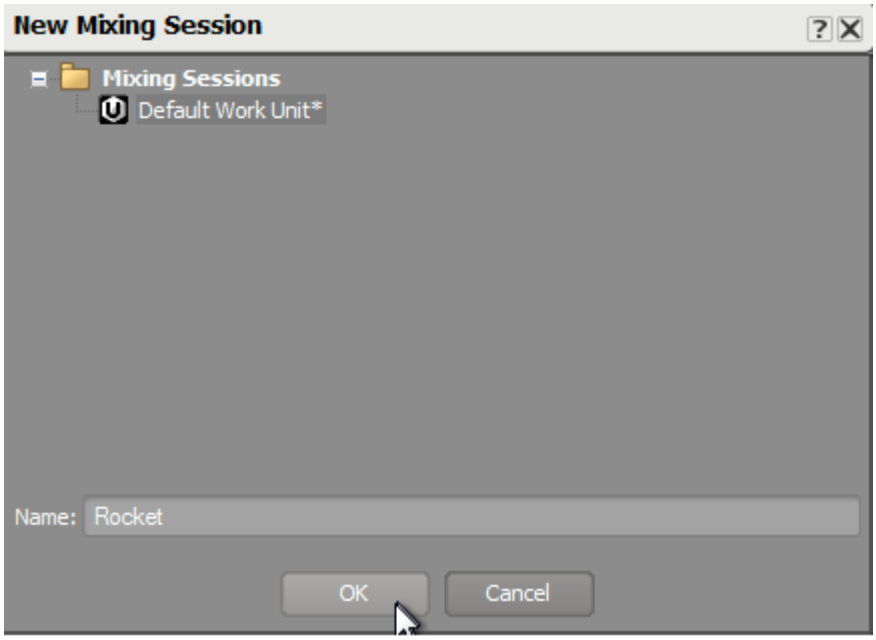
Mixerレイアウトが開きます。 前回の演習で作成したロケットサウンドキャスタービューが、空のミキシングデスクビューの下に確認できます。

サウンドキャスター同様、ミクシングチャンネルを用意したいオブジェクトのプリセットとして作用するミキシングデスクセッションを作成することができます。ロケットサウンドキャスターセッションを補足するミキシングデスクセッションを作成します。

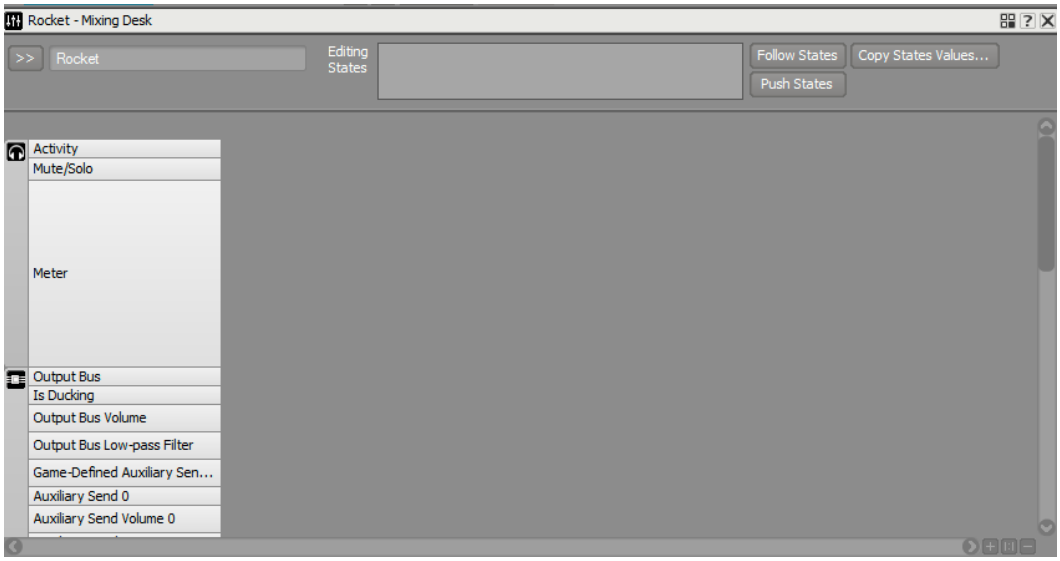
2. セレクターをクリックし、**New**を選びます。



そのミキシングセッションを**Rocket**と命名し、**OK**をクリックします。



新規ミキシングデスクが表示され、現状はミキシングデスクに追加されるオブジェクトで使用可能なプロパティのリストが表示されるのみです。この点は、サウンドキャスターのモジュールで表示される、制限された数のプロパティと比べて、大きな優位点になります。



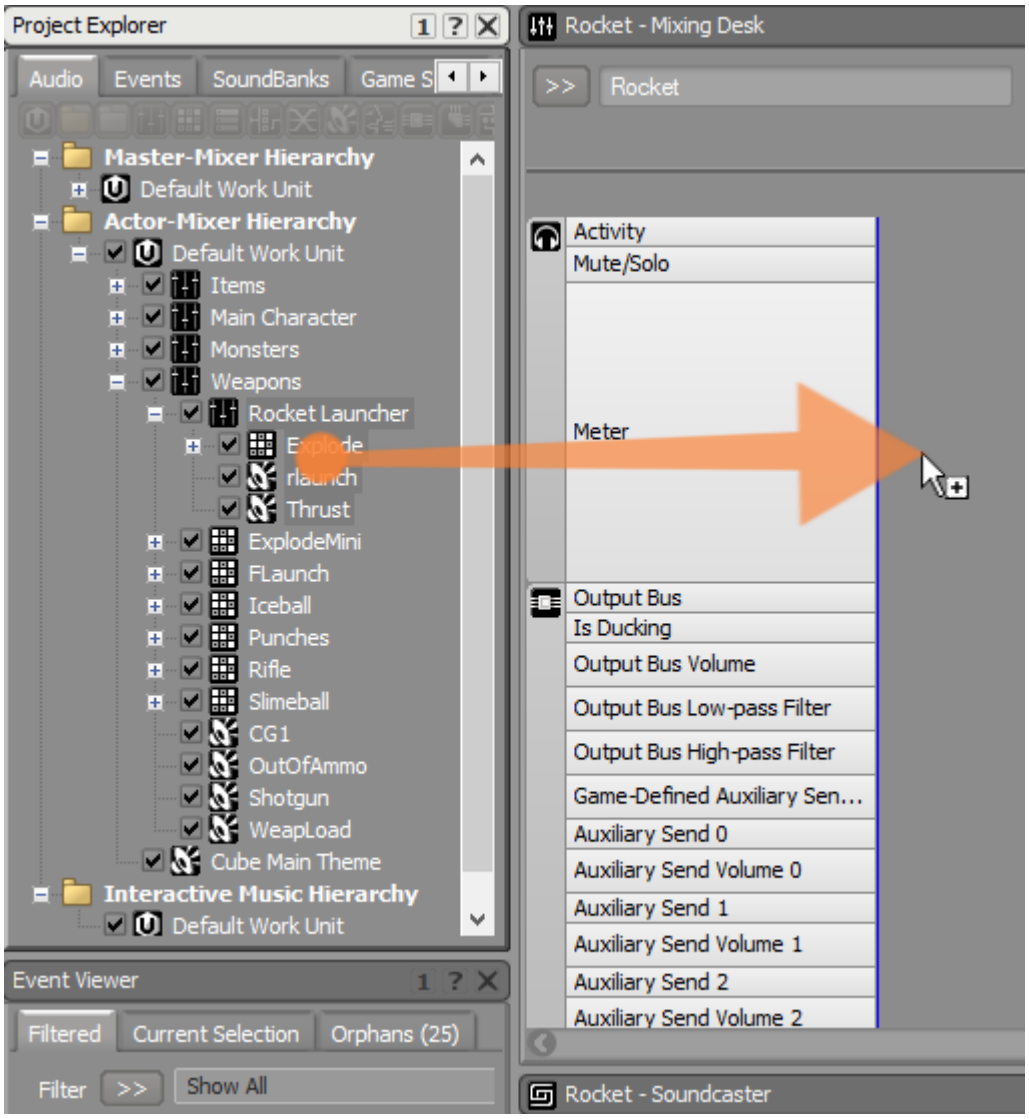
ミキサーへオブジェクトを追加する

オブジェクトはミキシングデスクへ、サウンドキャスターへ追加するのと同じように、プロジェクトエクスプローラーからドラッグ・アンド・ドロップすることで追加することが出来ます。

注記

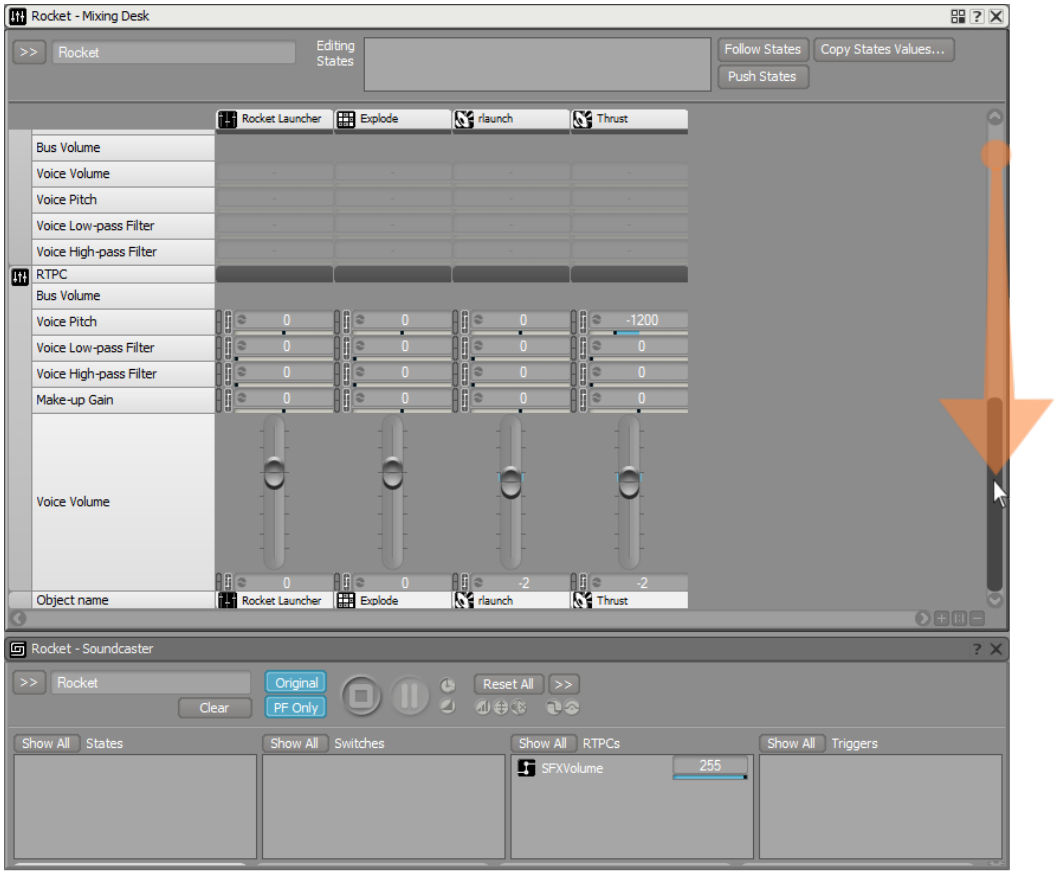
ミキシングデスクではどのようにサウンドが再生されるかを直接的に影響を与えるプロパティのみが表示されており、イベントオブジェクトを追加することは出来ません。

1. プロジェクトエクスプローラーのアクターミキサー階層で、**Rocket Launcher**、**Explode**、**rlaunch**及び**Thrust** オブジェクトを選択し、それらを**Rocket** Mixing Deskへドラッグ・アンド・ドロップしてください。



これらオブジェクトはミキシングデスクへ追加され、一般的なミキシングコンソールのように、縦のミキサーストリップとして表示されます。

2. **Mixing Desk View**を下へスクロールし、使用可能なプロパティを全て確認します。

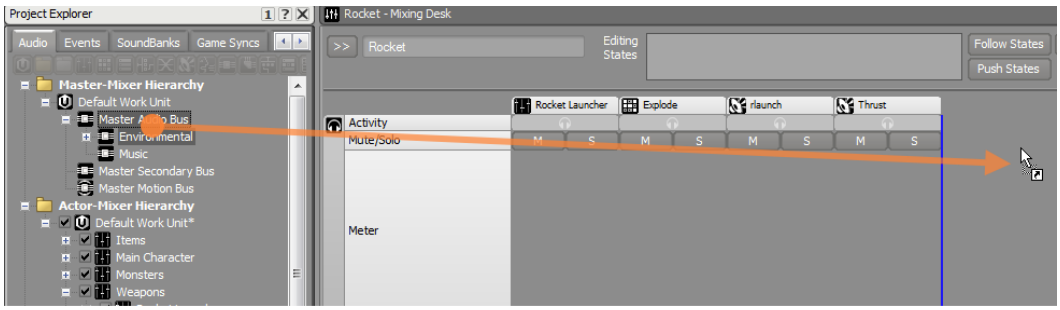


ティップ

表示させたいパラメータを、ウインドウの右上コーナーの **View Settings** ボタンを使用して、カスタマイズすることができます。

サウンドキャスターとは異なり、オーディオバスオブジェクトをミキシングデスクへ追加することで、サウンドソースの信号フローの全ての点について制御することが可能です。

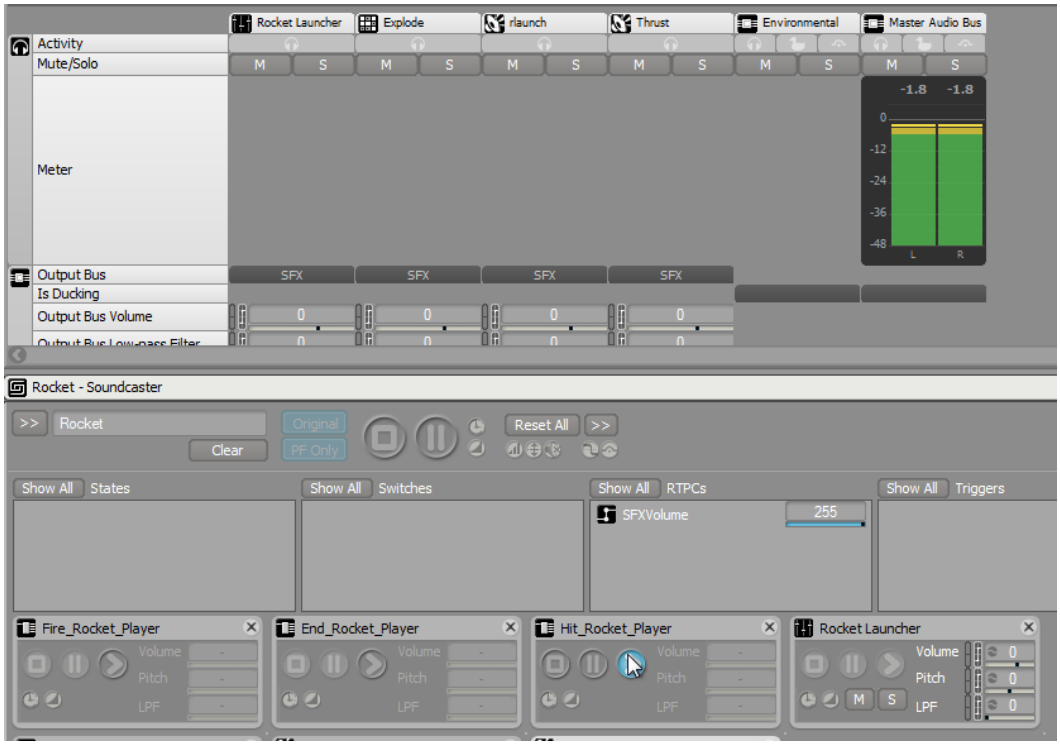
3. プロジェクトエクスプローラーのマスターミキサー階層で、 **Master Audio Bus** と **Environments Bus**を選択し、それらを**Rocket** Mixing Deskへドラッグ・アンド・ドロップします。



典型的には、ミキシングコンソールの最終出力は、ミキシングコンソールの右端に現れます。これは、ミキサーストリップの順番を再配置することで容易に可能です。

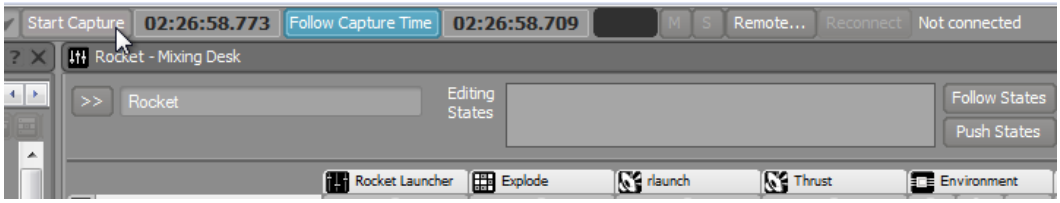
4. **Master Audio Bus**ヘッダを**Environment Bus**の右へドラッグします。

5. サウンドキャスター内のオブジェクトを再生し、マスターオーディオバスのメーターがどのように反応するかを確認します。

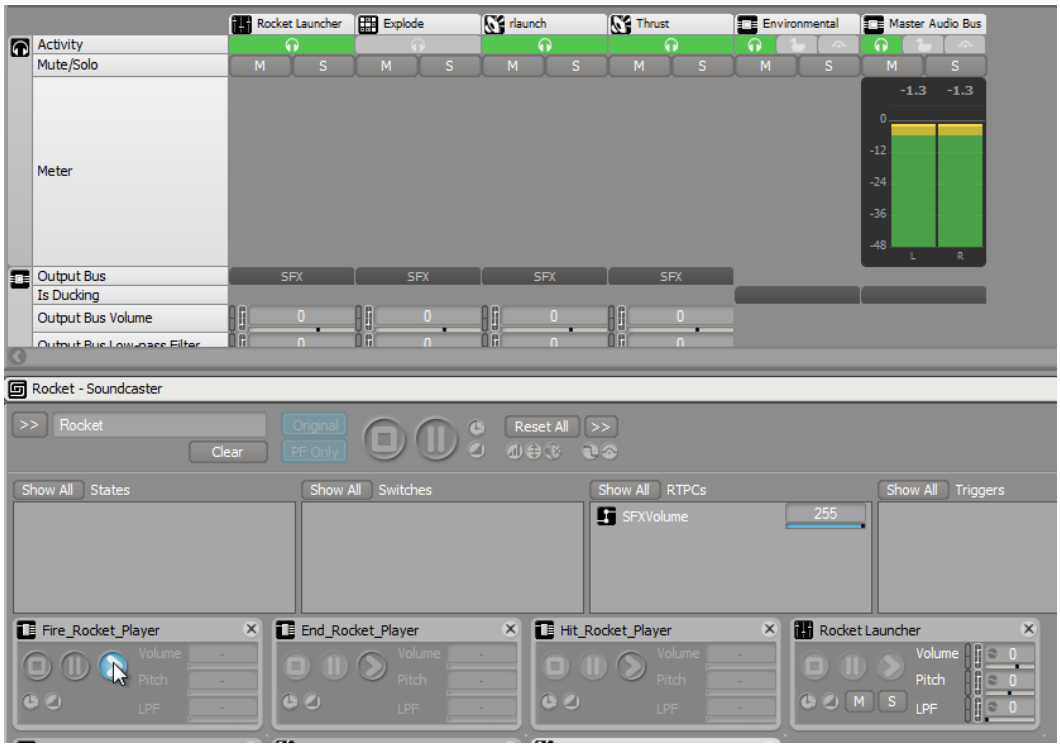


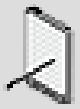
キャプチャーセッションを開始すると、どのオブジェクトが現在アクティブであるかや、どのバスがダッキングされているかなど、より多くのビジュアルなフィードバックが得られます。

6. **Start Capture**をクリックします。



7. **Soundcaster**ウインドウで、**Fire_Rocket_Player**イベントを再生します。





注記

Thrust音は、あなたがEnd_Rocket_Playerイベントを再生するまで、再生が続きます。

オブジェクトが使用されると、アクティビティアイコン(ヘッドフォン) がグリーンに変わるのが確認できます。これはあるサウンドが正しく再生されない理由を検証するうえで、非常に便利です。それはどのオブジェクトがマスターオーディオバスへのパスにおいてサウンドを制御しているか素早く確認できるからです。

8. **ミキシングデスク**に表示されている様々なロケットのプロパティを操作して色々実験してみましょう。

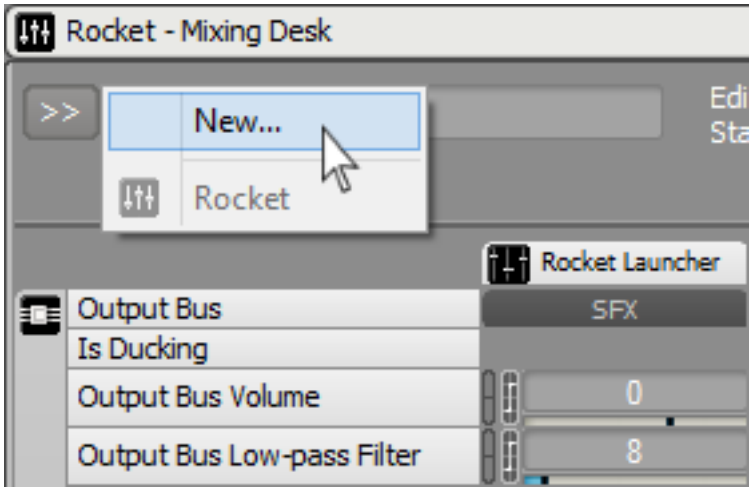
ティップ

複数のオブジェクトが選択されている際に、（スライダー、フェーダーなどの）プロパティを変更することで、選択されたオブジェクト全体に影響が及び、オブジェクトの値がその他全ての選択されたオブジェクトにセットされます。しかしながら、ALTキーを押しながら、スライダーやフェーダーを操作すると、選択されたオブジェクトの値がオフセット扱いとなり、絶対値として設定されないようになります。

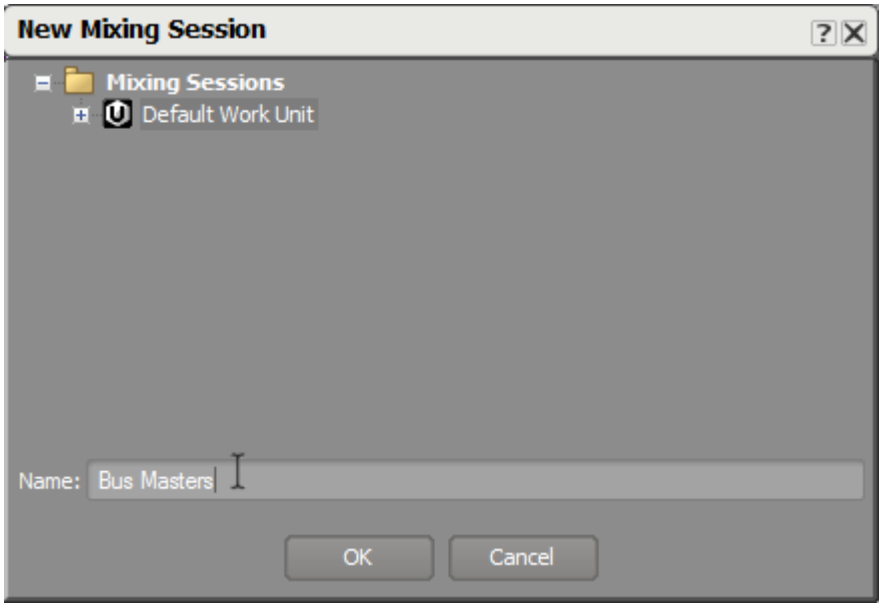
追加のミキシングデスクの作成

複数のミキシングデスクを作成することができ、素早くリコールできるので、あなたのゲームのミックス作業で特定の部分に集中することができます。1つのスクリーンであなたのバスオブジェクトの全てを見られるようにするのは、ゲームの信号フローの最後の段階を表すことなので一般的です。メーターを使用することで、これらのバスに送られる信号の総計レベルが高すぎるかを確認したり、ダッキング時に複数バス間の相互作用を目で確認することが出来ます。

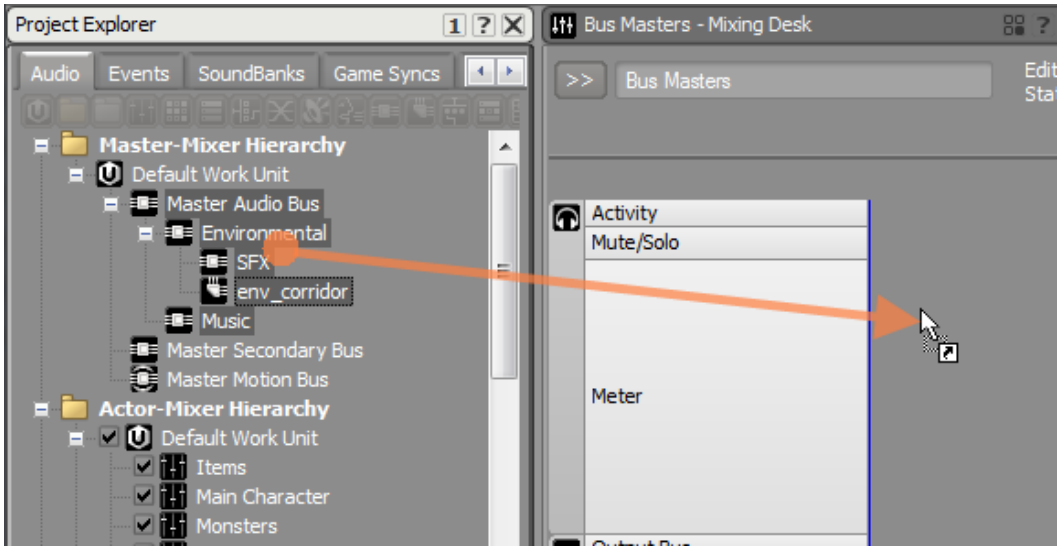
1. **ミキシングデスク**ビューで、セレクターボタンをクリックし、 **New**を選択して下さい。



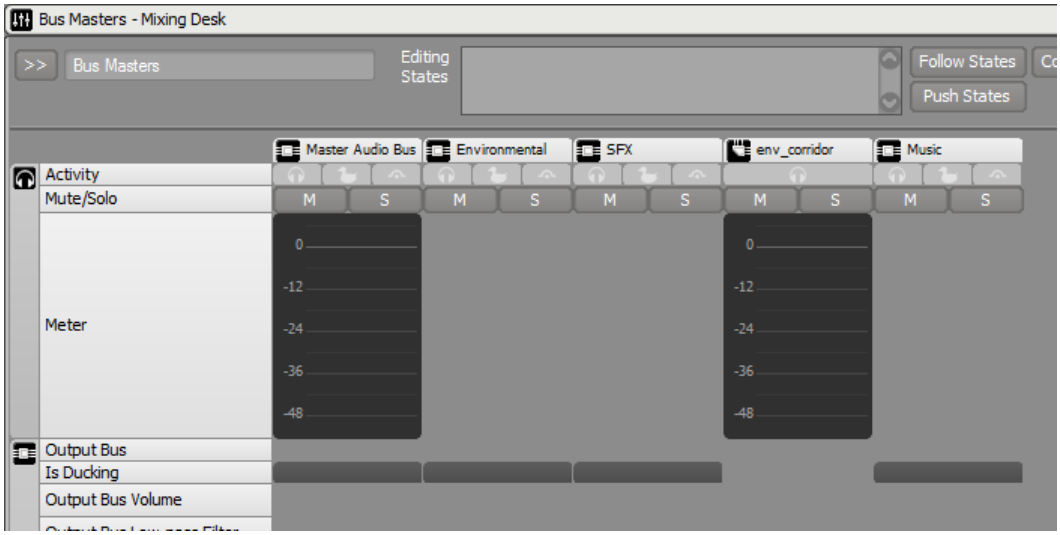
2. 新規ミキシングセッションを**Bus Masters**と命名し、**OK**をクリックします。



3. プロジェクトエクスプローラーのマスターミキサー階層から、**Master Audio Bus**、**Environment**、**SFX**、**env_corridor,** 及び**Music busses** を<e6>Bus Masters</e6> Mixing Deskへ追加します。



これで、バスマスターでの作業用のミキシングデスクが用意できました。こちらは次の演習で使用します。



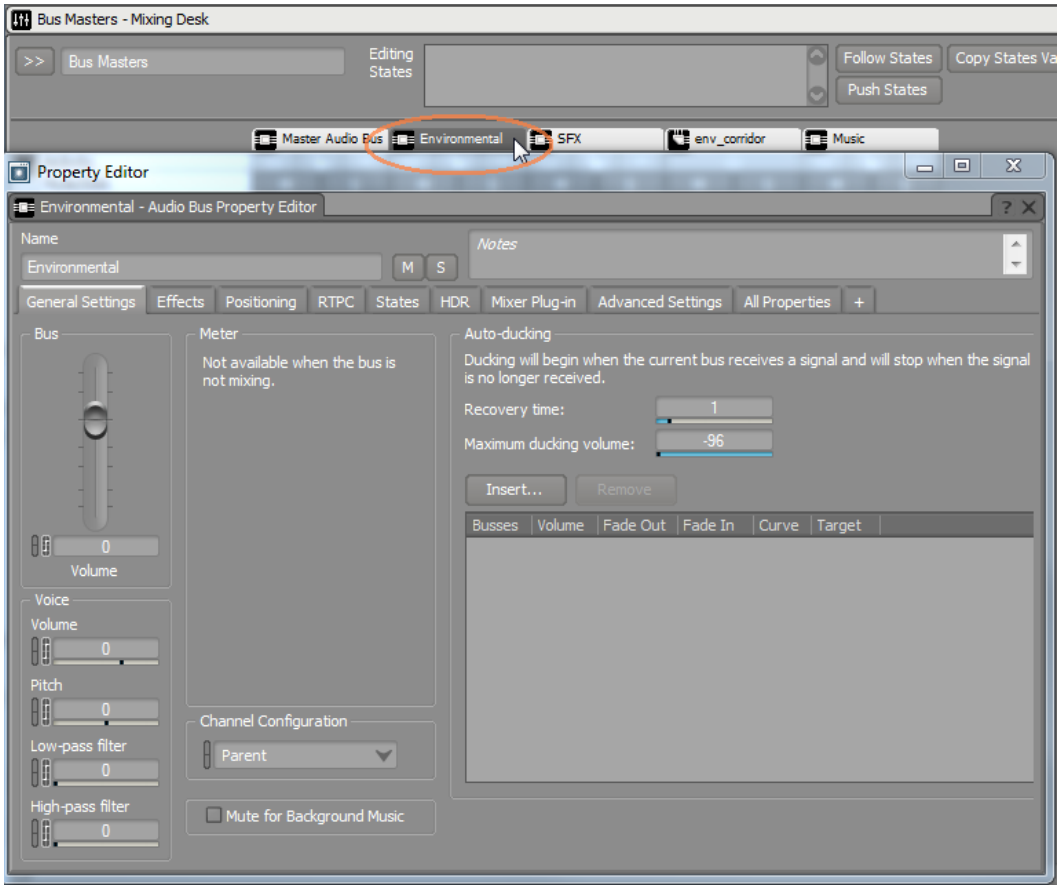
各種ステートの使用

ミキシングデスクの便利な機能の一つとして、ゲームステートの状況に応じて、異なるミックスを素早く作成することができます。レッスン3で、あなたは、どのようにしてオブジェクトレベルのゲームステートを使用してプレイヤーの心拍音へ影響させるかを学びました。ここでは、ステートを使用してどのようにオーディオバスオブジェクトが変更できるか、またどのようにしてステートの構築、テストがミキシングデスクビューから全て行なうことが出来るかを学びます。

目標は、プレイヤーが倒れた際に、環境音が静かになり、存在感が薄れ、その一方で音楽には影響が無いようなミックスを仕上げることです。

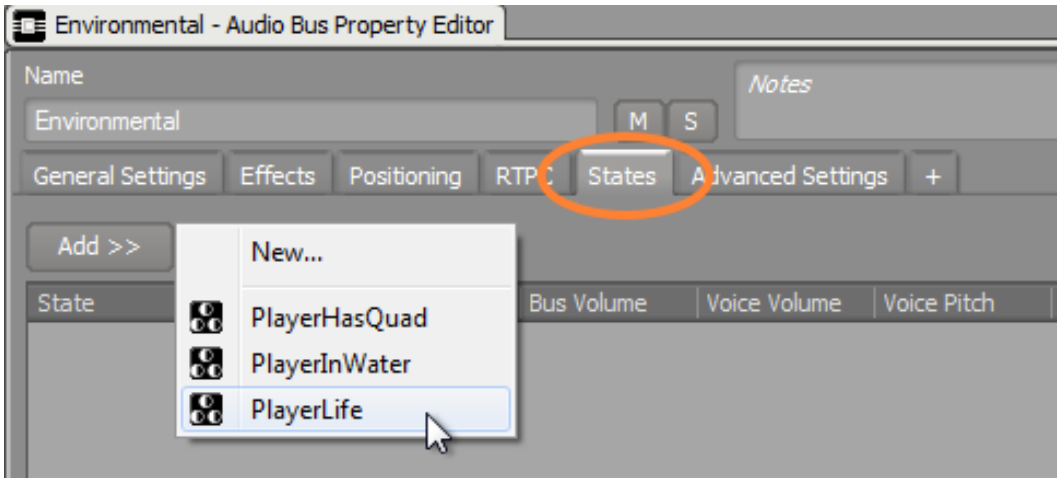
作業を始める前に、ミキサーの上部にあるステート編集(Editing States)エリアがありますが、対応するボックスには何もないことを確認して下さい。これは、ミキサー内のオブジェクトがいずれも、ステート値によって現在制御されていないからです。

1. **Environmental** ミキサーストリップのヘッダをダブルクリックします。

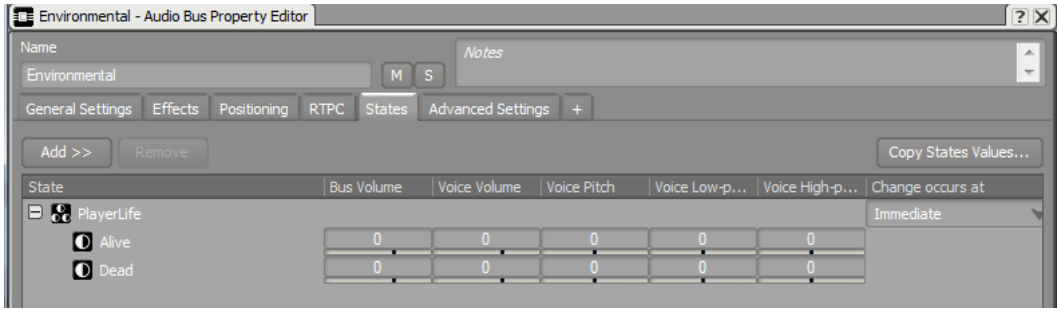


移動可能な**Environmental Bus Property Editor**ウインドウが開きます。

2. States タブをクリックし、**PlayerLife** State Groupを追加します。

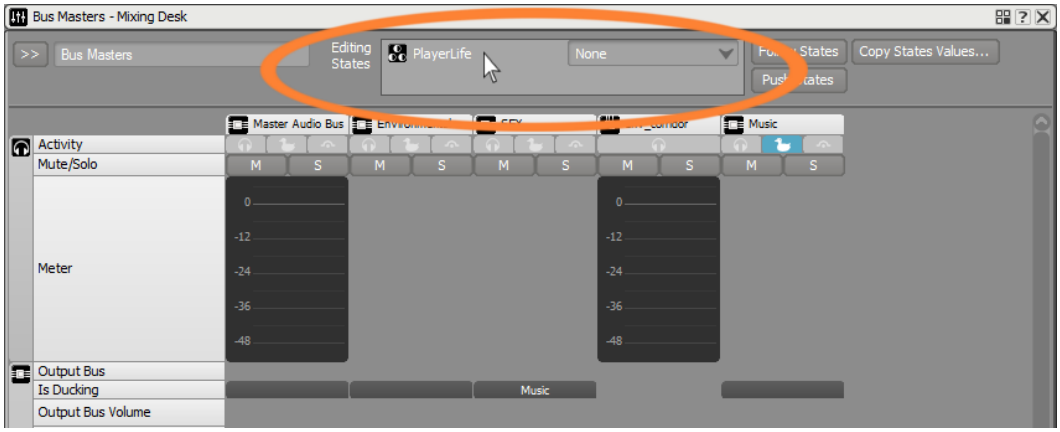


ステートグループ（およびその複数ステート）が環境バスに追加され、あなたがレッスン3で行なったようにプロパティ値が設定できるようになります。しかしながら、今回はプロパティエディターで数値を調整するのでは無く、ミキシングデスクから行ないます。



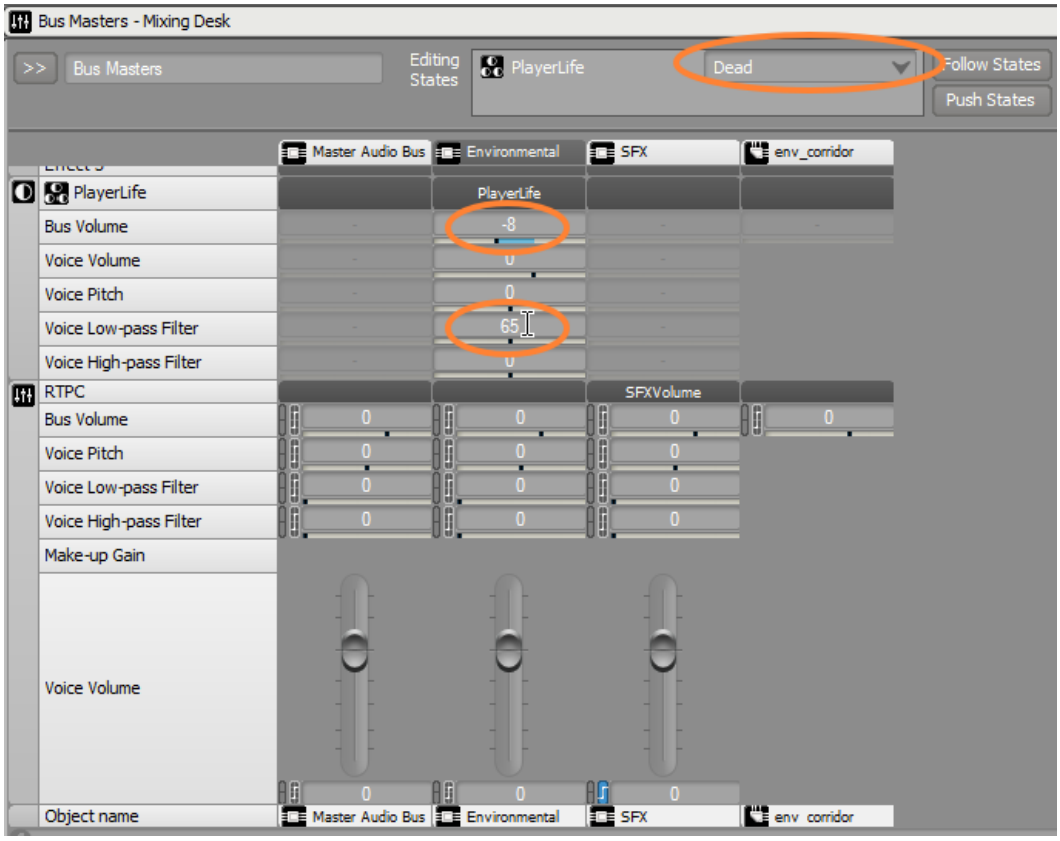
3. Property Editorウインドウを閉じます。

   ステート編集エリアが現在のステート値を表示するプルダウンメニューがあるPlayerLifeを表示しているのを確認して下さい。



   さらに、ステートのオフセット値が、ミキサーストリップに加算されていますが、おそらくそれを見ることはできないでしょう。

4. PlayerLifeのステートをDeadに変更し、下にスクロールし、 バスボリュームを-8に、ローパスフィルター値を65に調整します。



いま設定したステートオフセット値は、プレイヤーが倒れた際にのみ反映されます。プレイヤーが生存している際に使用される値を確認するには、ステートプルダウンメニューをAliveに戻します。

5. **Playerlife**ステートを**Alive**に変更します。

バスボリュームとボイスローパスフィルター設定が、Aliveステートに関連付けられたものに戻っていることを確認してください。

ティップ

Follow Statesボタンを使用して、ゲームを接続時に、キャプチャセッションを開始し、プレイ中にゲームの現在のステート値に従ってこのステートを自動的に変更するようにすることができます。 **Push States**ボタンをアクティベートして、Wwiseがゲームに対してどのステートに移行するか、例えばプレイヤーがゲーム中で倒れていなくてもあたかも倒れたようにゲームが振る舞うように強制することが出来ます。これは、ゲーム内で特定のステートになるようにゲームをプレイする必要なく、ミックスをテストできるので非常に便利です。

コントロールサーフェスの使用

学んできたとおり、サウンドキャスターとミキシングデスクビューは、サウンドを素早く再生し、サウンドに影響を与えるプロパティを変更するのに便利です。唯一の制限は、ミキシングデスクチャネルの数や、サウンドキャスタートランスポートの数に拘わらず、コンピューターで扱う限り、一度にWwiseで操作ができるパラメータは 1 つしかありません。この制限を超えるために、別の入力デバイスとしてMIDIコントローラーを使用することで、劇的に生産性を高めることが出来ます。Wwiseでは、複数のオブジェクトプロパティや、再生などのWwiseコマンドをノブ、フェーダー、ボタンなどのMIDIコントロールデバイスにある複数の物理コントロールにマッピングすることができます。コンピューターに接続された外部MIDコントローラーをお持ちであれば、本レッスンの最後の演習として、これがどのように実現できるかを学びます。

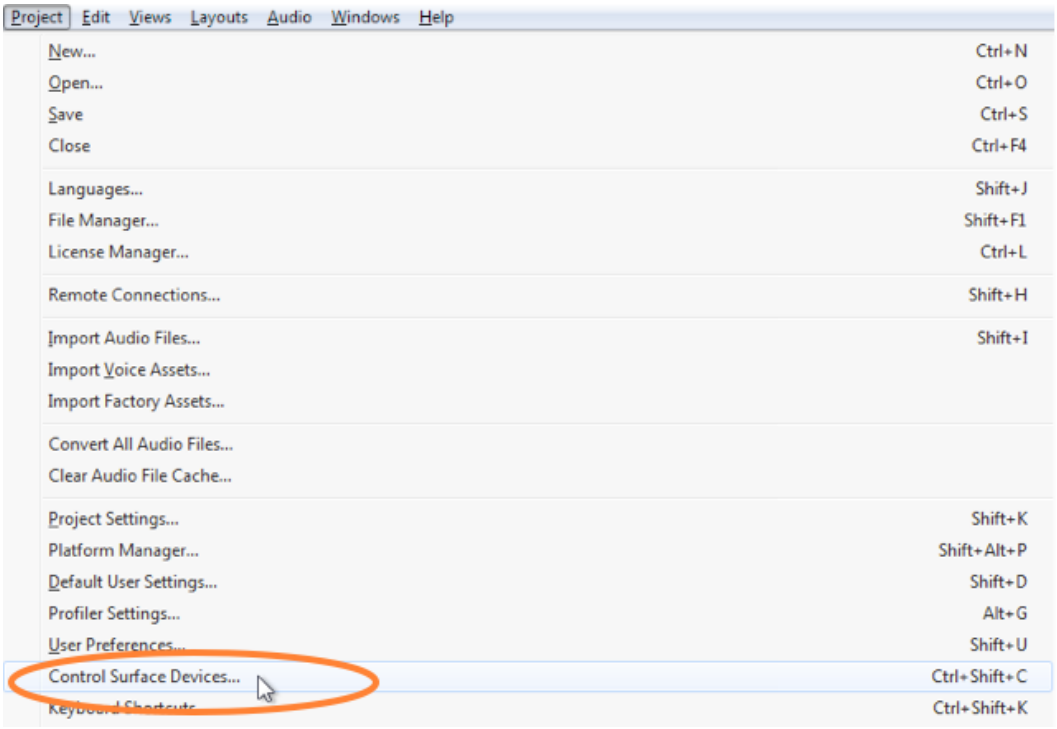
注記

ここでWwiseプロジェクトをセーブし、MIDIコントローラーをコンピューターに接続後、再起動します。本演習ではKorg Nano Kontrol surfaceを使用しますが、名の知られているMIDIやMacieコントロール対応のコントローラであれば動作します。

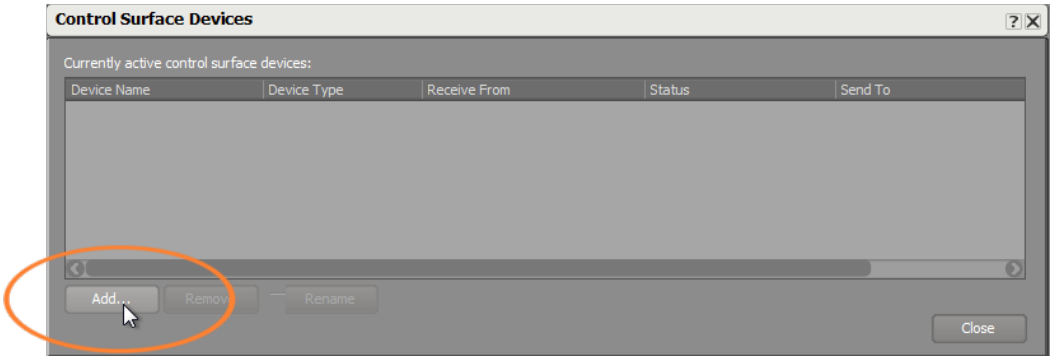
コントローラーの設定

まず、Wwiseにどの外部MIDIデバイスをコントロールサーフェスとして使用するか設定する必要があります。

1. メインメニューから**Project** > **Control Surface Devices**を選択します。

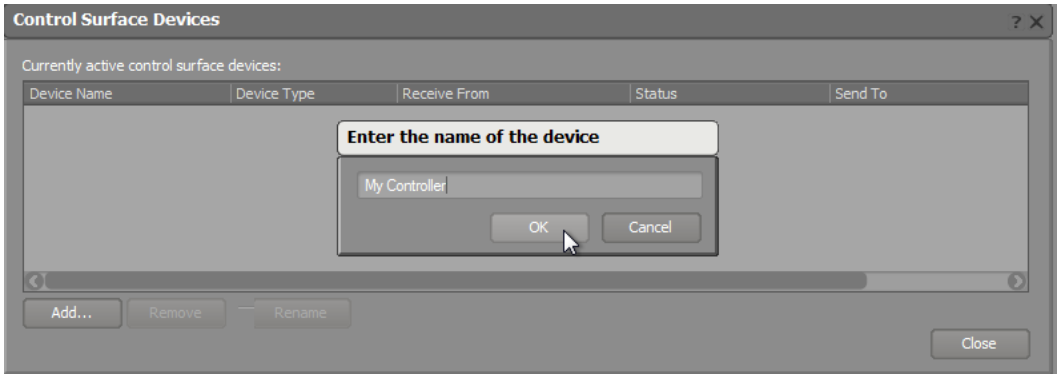


2. **Add**をクリックして新規デバイスを追加します。



使用するコントローラーに名称を付けることが出来ます。

3. **My Controller**と命名し、**OK**をクリックします。

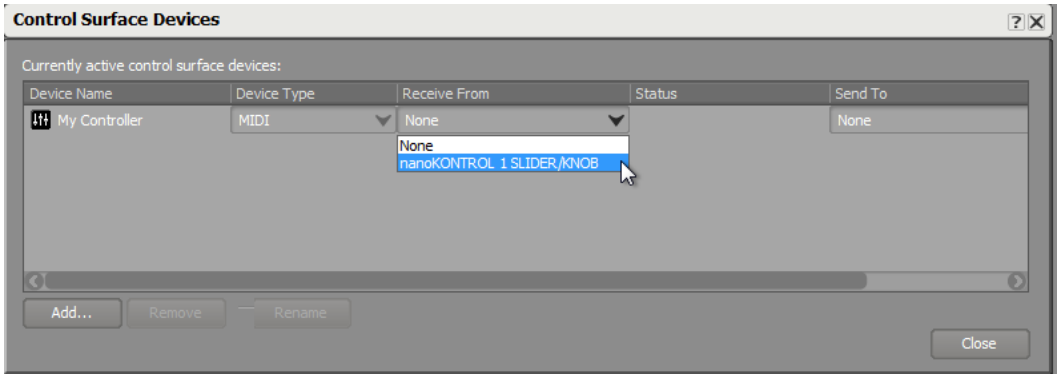


新規エントリーが作成できました。デフォルトで、デバイスタイプはMIDIに設定されていますが、これはデバイスが汎用MIDIメッセージで通信することを意味し、これが基本的なパラメータ制御のためにあなたが求めていることです。次に実際のハードウエアデバイスを、先ほど名前を付けたエントリーに関連付けます。MIDI情報をコントローラーからWwiseへ送ることのみなので、Receive From行のプルダウンメニューを使用してデバイスを設定します。

4. **Receive From**フィールドのプルダウンメニューで、お使いのコントローラーを選択します。

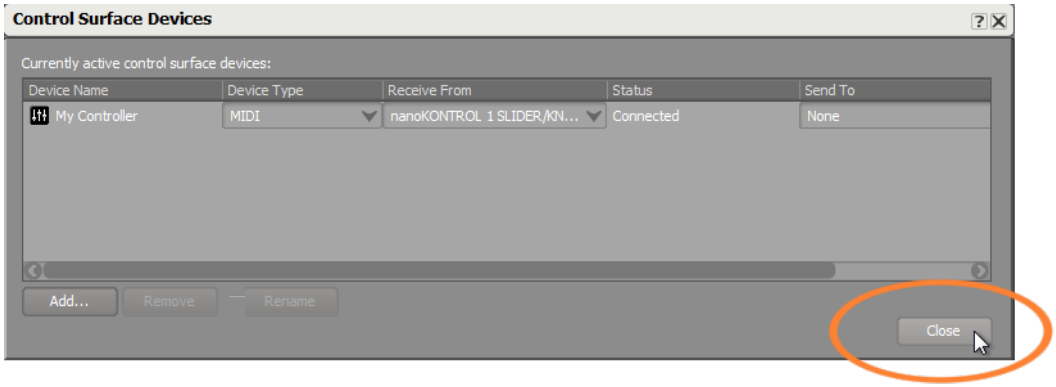
注記

ここでは、デモ目的でKorg nanoKontrolハードウエアを使用しています。あなたの使用するハードウエアにあわせて選択します。コントローラーの選択肢が現れなければ、あなたのコンピュータと、もしくはWwiseがコントローラーが接続されたことを認識できていないからです。



選択が完了したら、**Status**行にデバイスが接続されたことが表示され、次に進めることが出来ます。

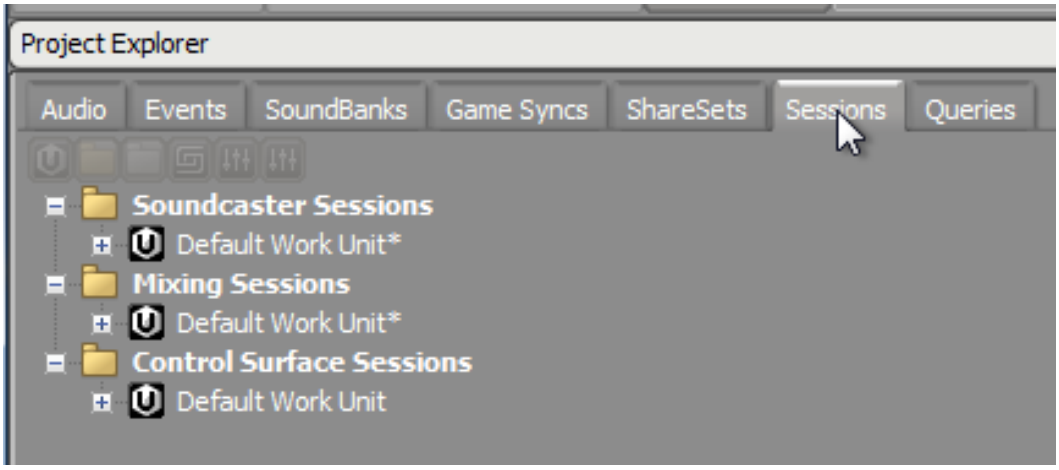
5. **Close**をクリックします。



ハードウエアコントロールをプロパティとコマンドにマッピングする

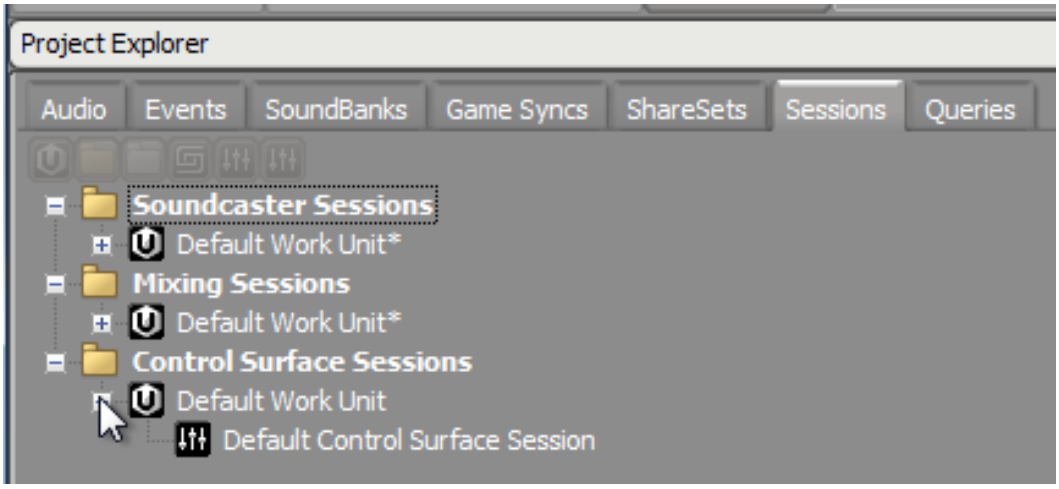
Control Surface Sessionを使用して、特定の外部MIDIインプットソースで使用するWwise機能をマッピングします。

1. プロジェクトエクスプローラー（Project Explorer）エクスプローラーでSessionsタブを選択します。



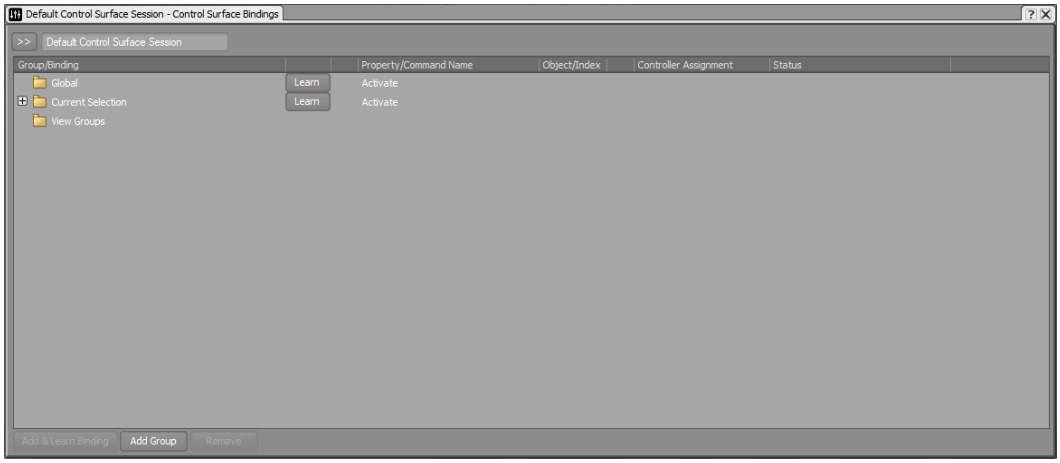
コントロールサーフェスセッションフォルダには、ワークユニット内のコントロールサーフェス構成設定が格納されます

2. **コントロールサーフェスセッション**フォルダ内の**Default Work Unit**をエクスパンドします。



デフォルトコントロールサーフェスセッションと呼ばれるコントロールサーフェスセッションが表示されます。コントロールサーフェスセッションは、どのようにコントロールサーフェスがWwise内の機能に関連付けられるかをマッピングする場所になります。

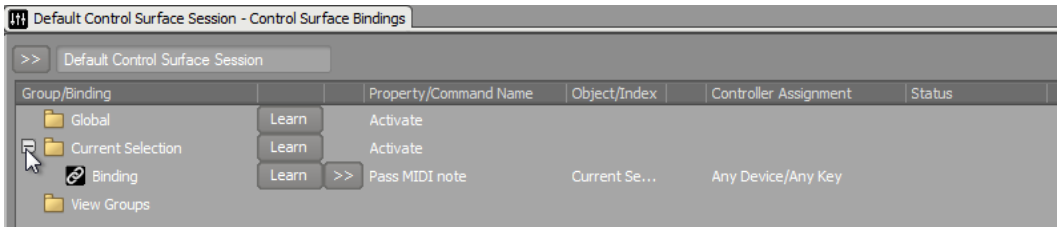
3. **デフォルトコントロールサーフェスセッション(Default Control Surface Session)**オブジェクトをダブルクリックします。



コントロールサーフェスバインディングウインドウが開きます。バインディングスでWwise内の特定の機能を、お使いのコントロールサーフェス上の特定のMIDIコントロールにリンクします。バインディングスは実際にはフォルダ内にあり、提供されるパラメータのスコープに従って別のフォルダに格納されます。例えば、グローバルフォルダ内には、特定のMIDIコントロールを、ユーザーインターフェイス側でどのオブジェクトが選択されていても、特定のプロパティにバインドすることが出来ます。これは、あなたがどのオブジェクトを現在選択しているにしろ、プレイヤーのガンショット音のボリュームをフェーダーで常に制御したい場合などに便利です。

その場合、MIDIコントローラーのフェーダーを選択した任意のオブジェクトのボイスボリュームを素早く制御できるように設定します。これは、カレントセレクションフォルダ(Current Selection folder)内で実現できます。

4. **カレントセレクション**フォルダをエクスパンドします。

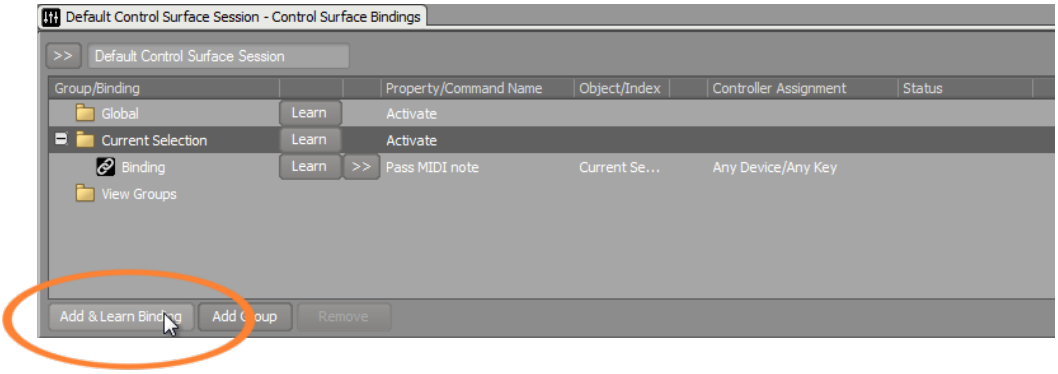


カレントセレクションフォルダ内に、受け取ったMIDIノートを現在選択されたオブジェクトに渡すことのできるデフォルトバインディングがあるのが確認できます。これは、楽器として演奏可能なSynthe Oneシンセサイザーオブジェクトと共に使用するのが特に便利です。また、MIDIコントロールを、オブジェクトプロパティや、再生や停止のようなオブジェクトコマンドにバインドすることもできます。その場合、コントロールサーフェスのフェーダーを現在選択されているオブジェクトのボイスボリュームプロパティにバインドします。

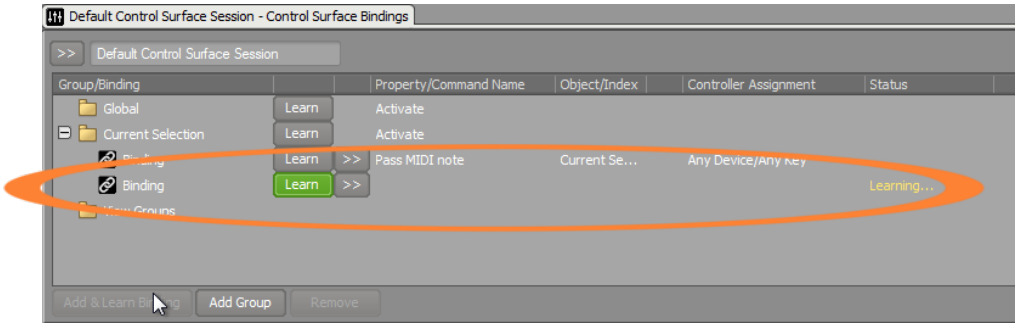
注記

お持ちのコントロールサーフェスにフェーダーがなければ、ノブやモジュレーションホイールなども同様に動作します。

5. Select the **カレントセレクション**フォルダを選択し、 左下の **バインディングを追加し学習（Add & Learn Binding）** ボタンを押します。

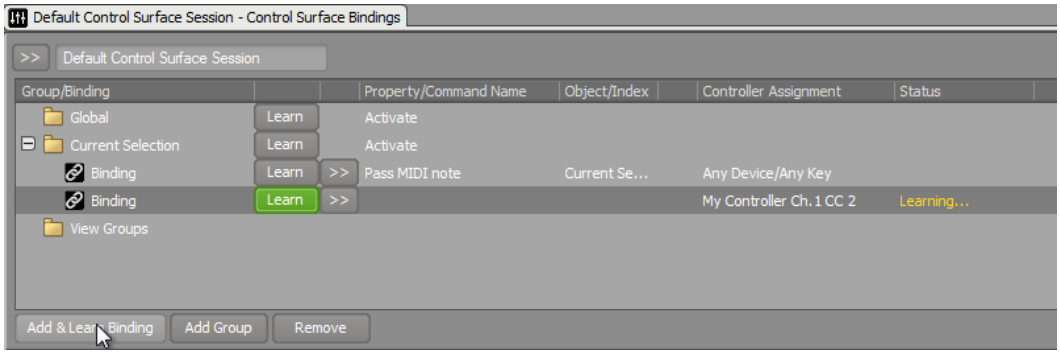


カレントセレクションフォルダ内に新規バインディングが表示されます。



このバインディングについて、コントローラーアサイメント(Controller Assignment)行の下のフィールドが空白であることを確認して下さい。それはあなたがまだWwiseにどのコントローラを使用するかを伝えていないからです。グリーンの学習(Learn)ボタンは、MIDIコントローラーのどの物理コンポーネントを、Wwise内の特定機能にバインドするか、選択待ち受け状態であることを示しています。

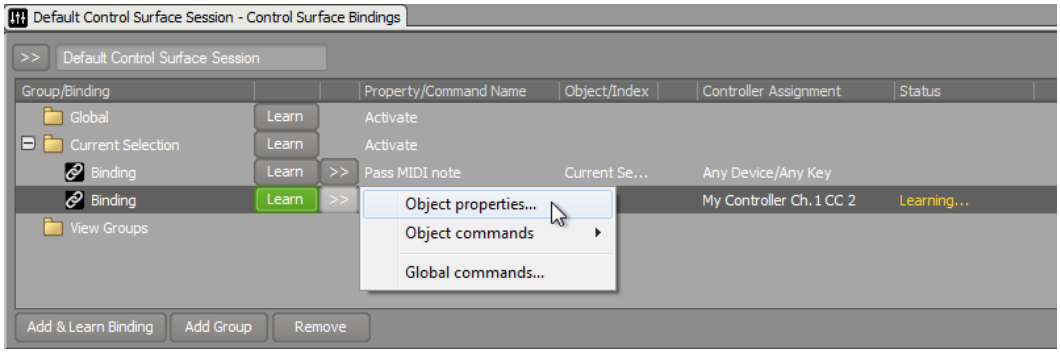
6. お使いのコントローラーのフェーダーを動かしてみて下さい。



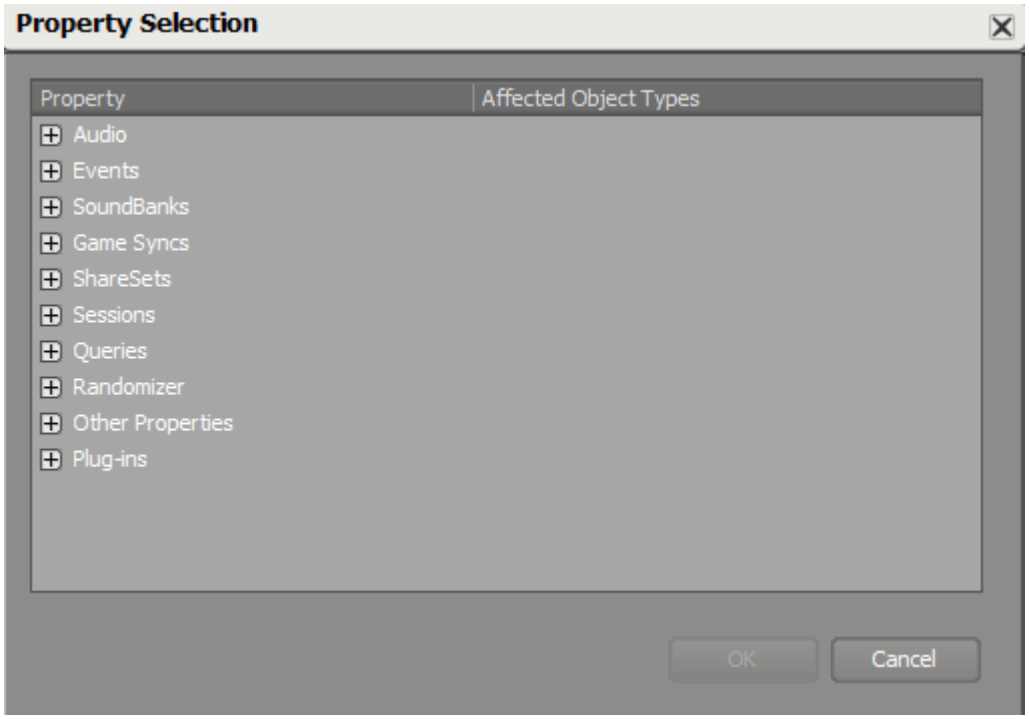
コントローラーアサイメント行に表示されたバインディングは、特定のMIDIチャンネルと、そのバインディングが反応するMIDI連続コントローラー番号(MIDI continuous controller number)を表しています。

ここで、このフェーダーを動かした際に、Wwiseのどの機能を応答させたいかを指定します。

7. グリーンの**Learn(学習)**ボタンの右にある、セレクションメニューボタンを選択し、次に**Object(オブジェクト)**プロパティを選択します。

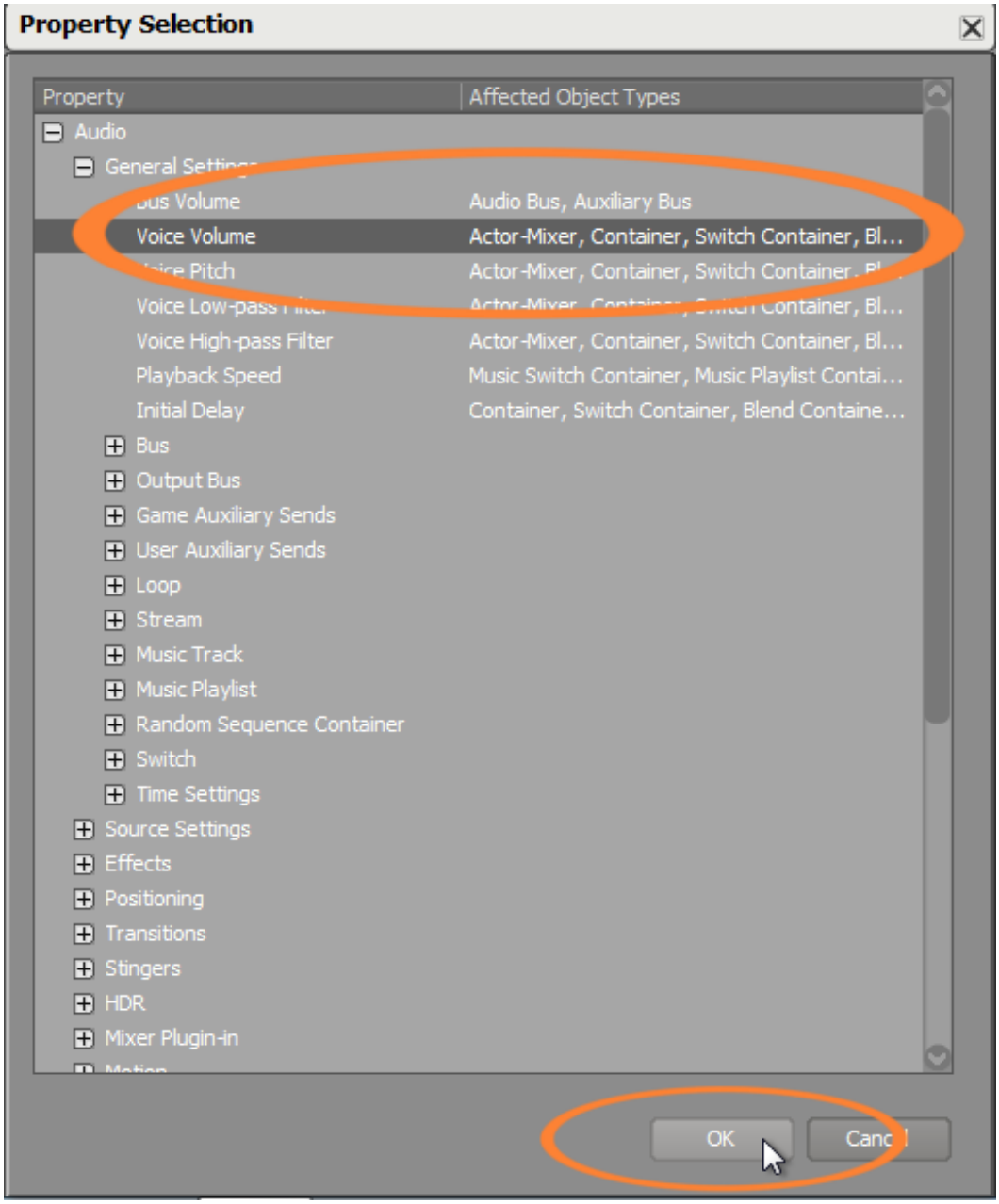


プロパティセレクションダイアログボックスが開きます。



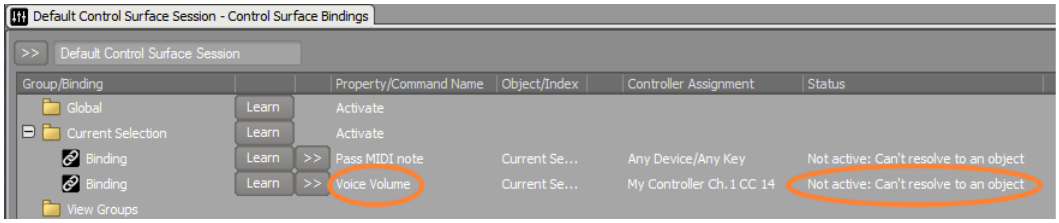
Wwiseには制御出来る様々な機能があり、それらは様々なカテゴリー、サブカテゴリーに分類されています。

8. オーディオプロパティをエクスパンドし、次に一般設定(General Settings)プロパティ、そして**ボイスボリューム(Voice Volume)**を選択し、**OK**をクリックします。



これで何を制御したいか定義できました。次に、どのMIDIコントロールを使用して、オブジェクトのボイスボリュームを制御したいかを設定する必要があります。

9. **プロパティセレクション(Property Selection)**ウインドウを閉じます。



コントロールサーフェスバインディングウインドウに、ボイスボリュームが、コントローラーからの適切なMIDIメッセージを受信した際に、現在選択されたオブジェクトで制御されるプロパティであることが確認できます。Status行がこのバインディングがアクティブで無いことを示しています。 これは、あなたが最後に選択したオブジェクトがデフォルトコントロールサーフェスセッションで、それにはボイスボリュームのプロパティが含まれていないからです。

10. **コントロールサーフェスバインディングズ** ウインドウを閉じ、 プロジェクト内の**Shotgun_Blast SFX オブジェクト**を選択し、あなたのコントローラーのフェーダーを使用してボリュームプロパティを制御します。

    ゲームに接続された際に、WwiseはWwiseインターフェイス内でリアルタイムミキシングを可能にします。これはまた、Wwise内のプロパティがコントロールサーフェスへマッピングされている場合、一般のオーディオコンソールでの作業と同様にミックスをすることができるということです。

11. まず、ゲームのサウンドバンクを生成し、**Cube**デモを起動し、Wwiseからゲームに接続します。

12. コントロールサーフェスを使って、ゲームをプレイしながらショットガンのボリュームを調整してみて下さい。また、プレイヤーが倒れた際の、環境バスにおけるボリュームとフィルターの変化など、あなたが先に行なった変更の結果を聞いて確認してみて下さい。

ティップ

最も一般的に調整に使われるピッチや、ローパスフィルターなどのパラメータをあなたのサーフェスコントローラーが対応している限り多くのノブやフェーダーにマッピングすることで、サウンドデザインに集中でき、パラメータの調整作業を素早く行なえるようになります。また、あなたのサーフェスコントローラーのフェーダーをミキシングセッションの様々なフェーダーにアサインすることで、リアルタイムで、直感的なミックスが可能になります。

レッスン 7：ゲームの最適化

メモリの管理 253
プロセッサーにおける最適化 273
プロファイラーを使用したリアルタイムモニタリング 277

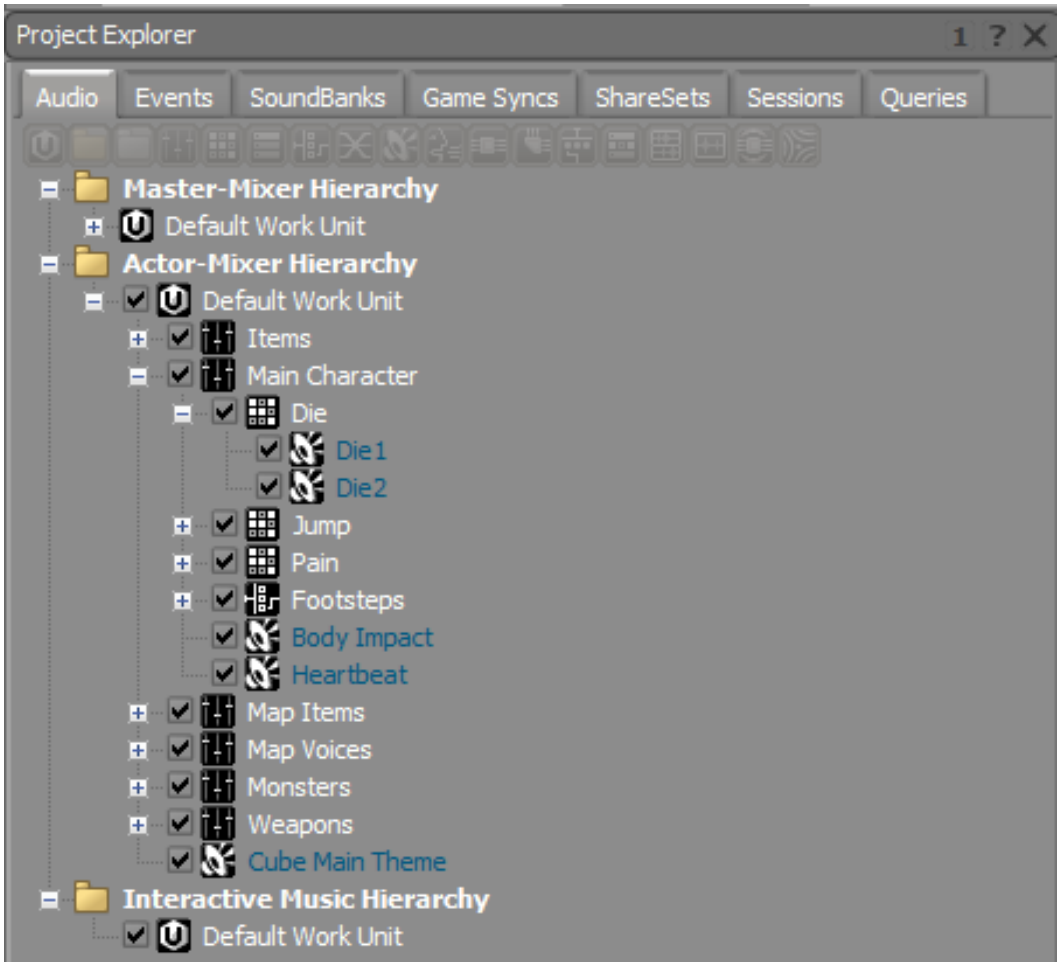
あなたはゲーム用に驚嘆するような優れたサウンドスケープをデザインすることが出来ますが、それがエンドユーザーのゲームシステムで使用できるリソースで実行できなれば意味がありません。こうした理由から、ゲームオーディオのインテグレーションは、少ない処理で多くのことを実現する技術でもあります。

Wwiseでは、クリエイティブな音響クオリティを犠牲にすること無く、ゲームへサウンドを組み込む際に、作業効率を最大限にする各種パワフルな機能を提供しています。 本レッスンではオーディオアセットのサイズを最小化し、RAMやCPUの使用量を節減する方法を学びます。Wwiseの最も強力であるポイントの一つでもあるのですが、あなたはこれまでの演習においてクリエイティブな面のみに集中することが出来、サウンドエンジンのパフォーマンス面の要因を気にすることはありませんでした。ここで、どのようにしてプロジェクトにスケーラビリティを与え、メモリやパフォーマンス予算内で動作するようにするかを学びます。

メモリの管理

本レッスンでは完全にビルドされてありますが、最適化されていないCubeデモを使用します。

1. レッスン7のプロジェクトを開き、プロジェクトエクスプローラーでオブジェクトをブラウズし、ゲーム内で使用されているオブジェクトを把握します。

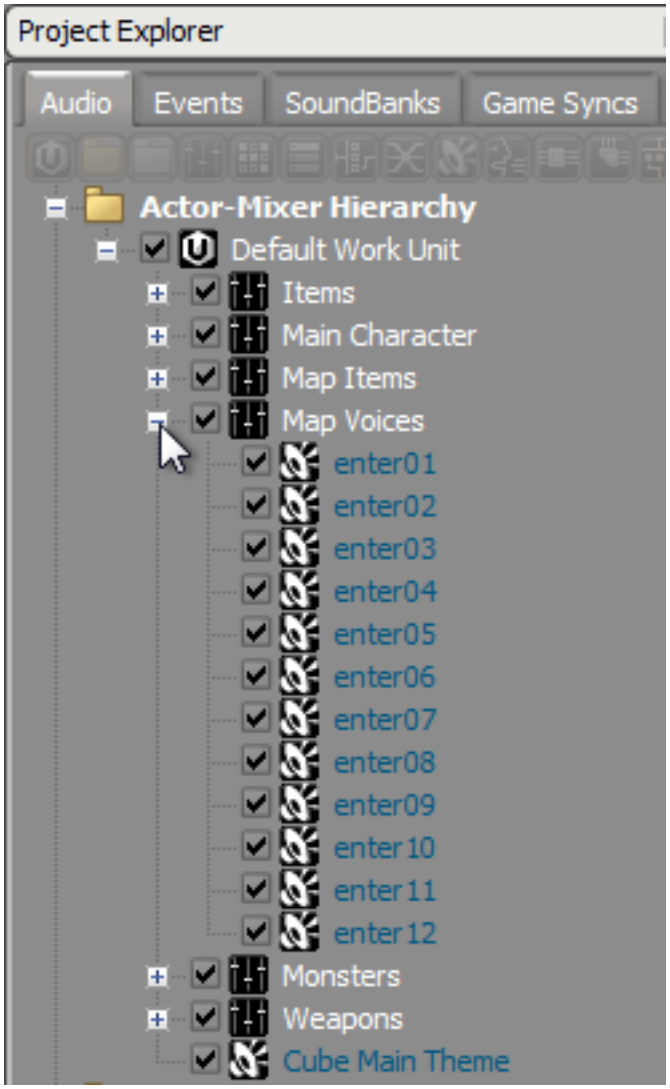


複数のサウンドバンクの活用

多くのビデオゲームはレベル毎に構成されており、特定のレベルのみに関連付けられたサウンドがあります。例えば、最後のゲームレベルが完了した際に、あなたのキャラクターの勝利のお祝いのシーンでのみ聞こえる花火のサウンドなどです。これに関連したサウンドは、ゲームの特定の箇所に到達していなければ、メモリにロードする必要はありません。

本ビルドのCubeデモでは、dcp_the_coreというCubeデモの特定レベルでのみ聞こえるセリフオブジェクトを含むMap Voicesと呼ばれるアクターミキサーオブジェクトがあります。

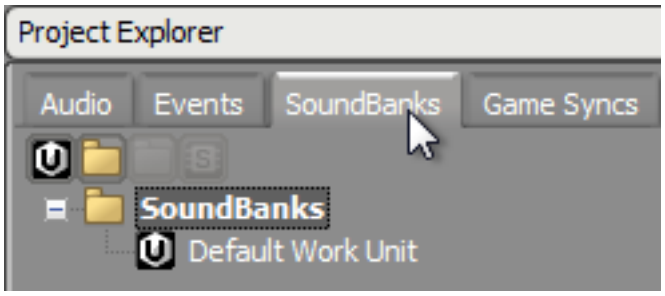
1. **マップボイス(Map Voices)** アクターミキサーをエクスパンドし、その中に包含されているSFXオブジェクトのいずれかをオーディション試聴してみてください。



マップボイスアクターミキサーの全ては、ゲーム内の特定の1レベルでのみ使用されるので、dcp_the_core レベルに達した際にロードされる独立したサウンド

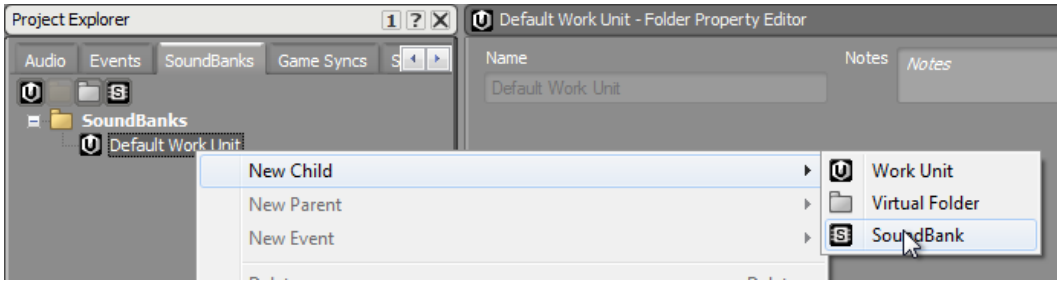
バンクにこれらサウンドを追加することで、より優れたメモリ管理が実現できます。

2. プロジェクトエクスプローラー内で、SoundBanksタブを選びます。

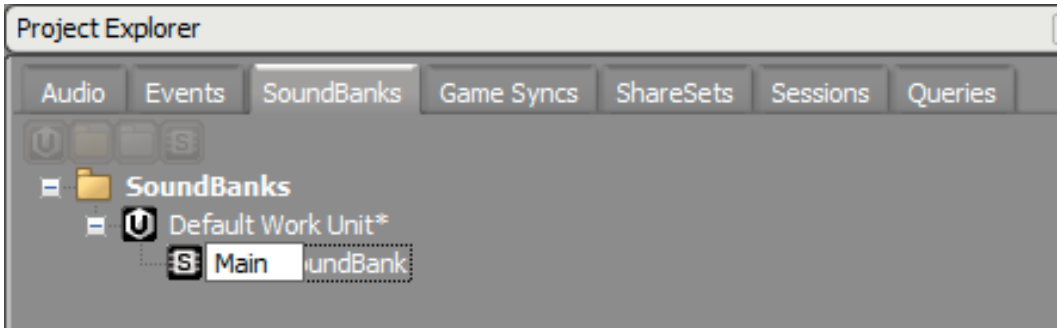


これまでのレッスンで使用してきたプロジェクトと違い、本プロジェクトにはサウンドバンクが 1 つも含まれていません。ゲームにはそれぞれ最低一つのサウンドバンクが必要で、それはサウンドバンクにはオーディオアセットの他に、そのオーディオがどのように再生されるかの指示が含まれているからです。これまでのプロジェクトでは 1 つのサウンドバンクを使用してきました。しかしながら、本プロジェクトでは2つ使用します、1 つはゲーム全体で使用するメインサウンド用、もう一つはdcp_the_coreと呼ばれるゲームレベルで使用されるものです。

3. デフォルトワークユニットを右クリックし、**New Child >SoundBank**を選択します。



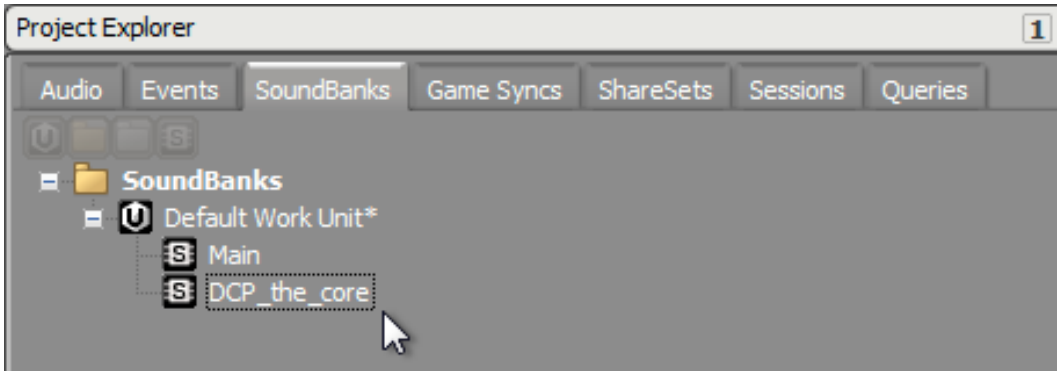
4. このサウンドバンクを**Main**と命名します。



注記

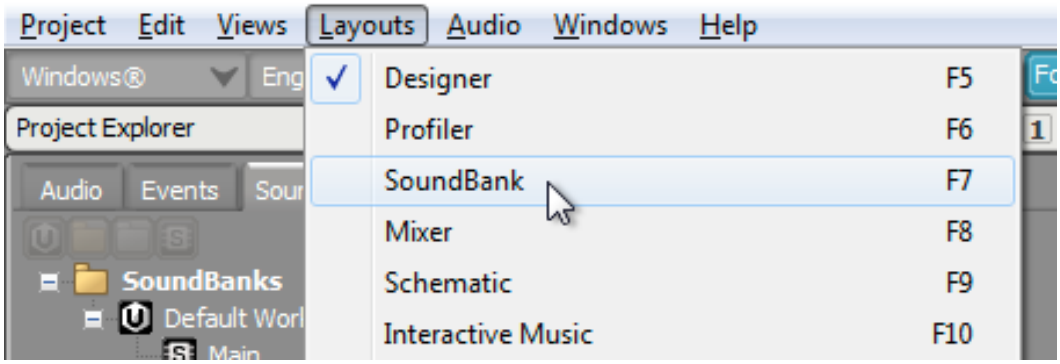
ゲームプログラマーはサウンドバンクを名前を指定してプログラム側でロードする必要があるので、サウンドバンクの名前が正しいことが重要です。

5. 同じ**デフォルトワークユニット**内に**DCP_the_core**という名前の別のサウンドバンクを作成します。

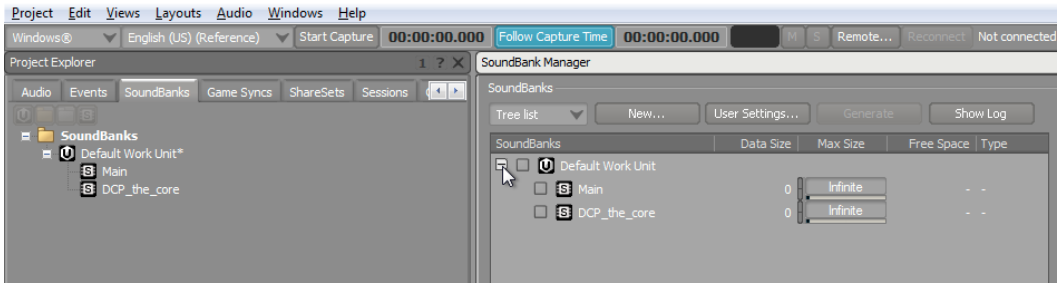


これで、どのサウンドがどちらのサウンドバンクにロードされるかを指定する準備が整いました。

6. メインメニューで**Layouts** > **SoundBank**を選択、もしくは **F7**を押します。



7. サウンドバンクマネジャー内で**Default Work Unit**をエクスパンドします。

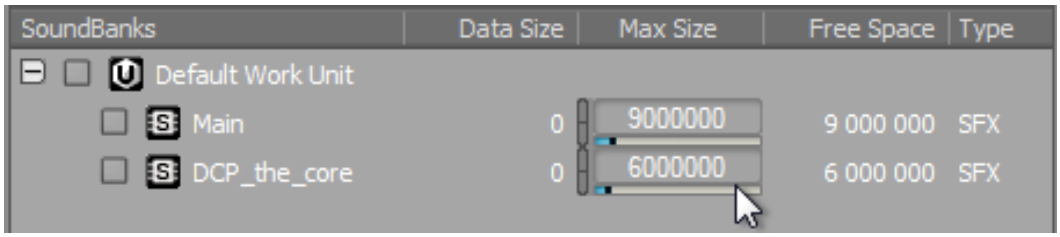


作成した２つのサウンドバンクが確認できます。

各サウンドバンクには、データサイズに関する列があります。現在、両方のサウンドバンクの値は0になています、これは何もサウンドバンクにアサインされていないからです。また、最大サイズ列があり、対象のサウンドバンクに対して想定するメモリ予算をbyte単位で指定できます。メモリ予算は、しばしばゲーム開発プロセスの初期に決定されますが、開発期間中に変更されることもあります。メモリ予算を理解し、それに収めることが重要です。それはあなたが使用するメモリリソースは、ゲームの他の処理、例えばグラフィクスには使用できないという前提だからです。

最大予算値を設定すること自体は、技術的にその値に制限する訳ではありませんが、予算を超えた際に見える形での警告を発することになります。

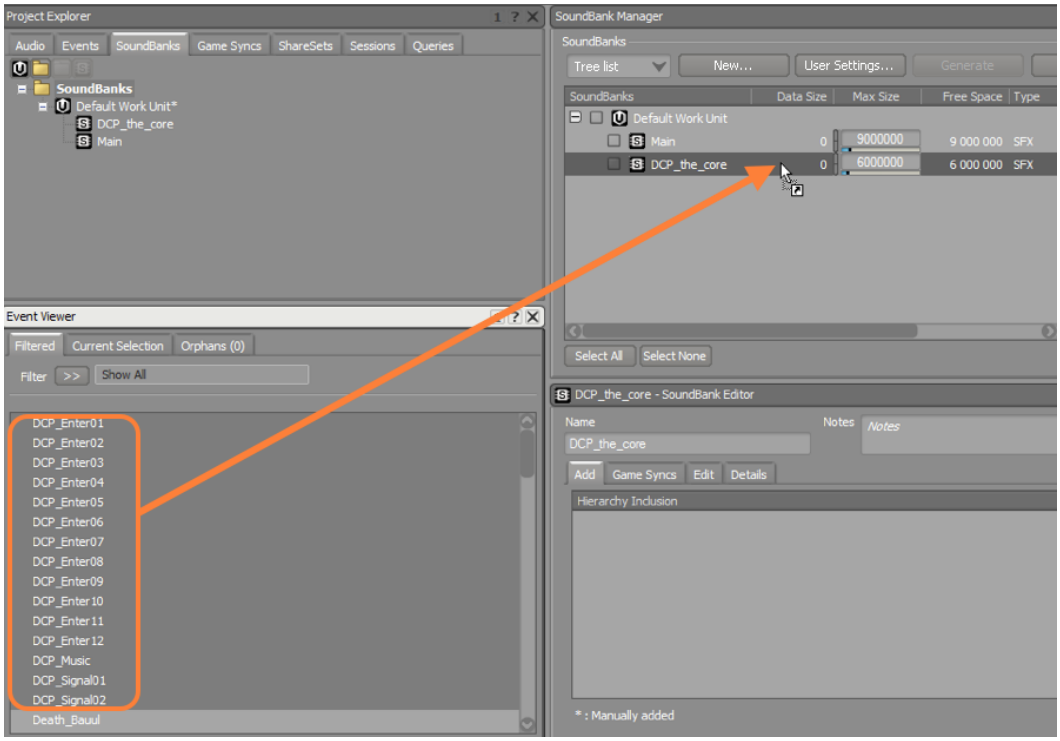
8. メインサウンドバンクの最大サイズ値を 9,000,000 に、DCP_the_core size を 6,000,000に設定します。



ここで、サウンドバンクへオブジェクトをアサインする必要があります。これまでのレッスンでは、イベントやオブジェクトをサウンドバンクへドラッグして、必要なオーディオアセットが自動的にサウンドバンクの一部に加えてきました。

dcp_the_core レベルのみでトリガーされるイベントを特定する必要があります。幸い、このレベル専用のイベントは、この特定のレベルで使用されることを示す独自プリフィックスが付いているので容易になっています。

9. DCP プリフィックスが付いたイベントを全て選択し、**DCP_the_core** SoundBankへドラッグします。

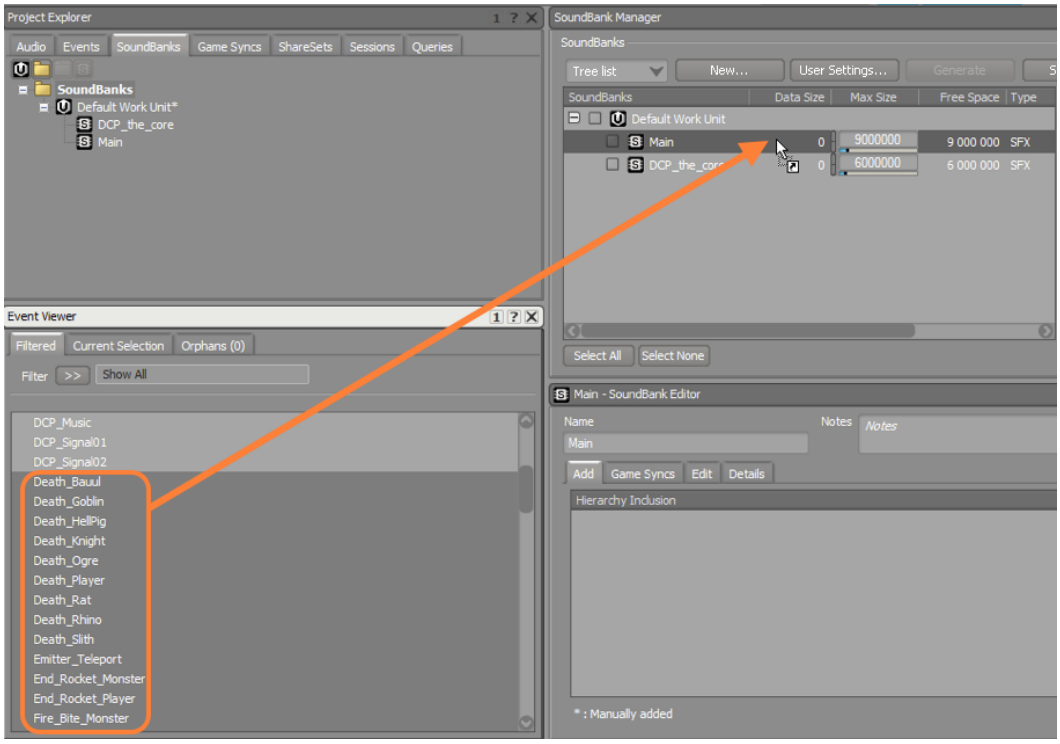


注記

これまでのレッスンで使用したプロジェクトでは、 DCP_Music イベントは、あなたが全てのゲームレベルのミュージックトラックを聞くことができるように、メインサウンドバンク内に格納されていました。しかしながら、本レッスンでは、DCP_Music イベントを 、このゲームの最後に使用される特定ゲームレベルがロードされた際に使用される、DCP_the_coreサウンドバンクに格納しています。 つまり、本レッスンの後半で使用するレベルと、異なるゲームレベルをプレイする限り、このミュージックを聴くことはありません。

DCP_the_core のデータサイズは増加しません。それは、まだあなたがサウンドバンクを未生成で、実際にはメインサウンドバンクにデータを追加してから行ないます。

10残りの全てのイベントを**Main** SoundBankへドラッグします。

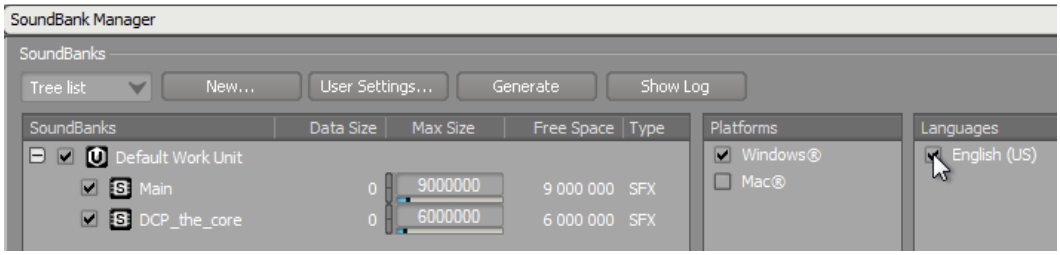


ティップ

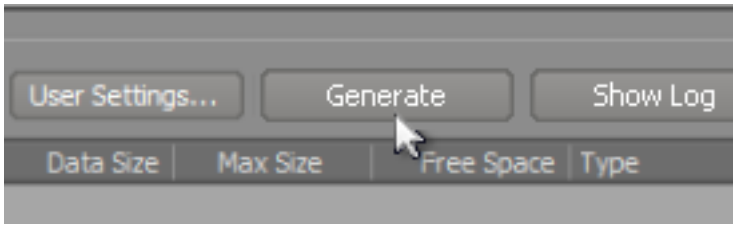
イベントビューアーで、Death_Baaulを選択し、リストの最後までスクロールし、Shiftを押しながら、一番最後のイベントであるWeapload をクリックすることで、素早くイベントの全てをワンステップで選択します。

どのサウンドバンクを生成するか指定しなければなりません。

11 **Main** と**DCP_the_core**サウンドバンクと、Windows（もしくはMac）プラットフォームと言語はEnglishのチェックボックスを選択します。

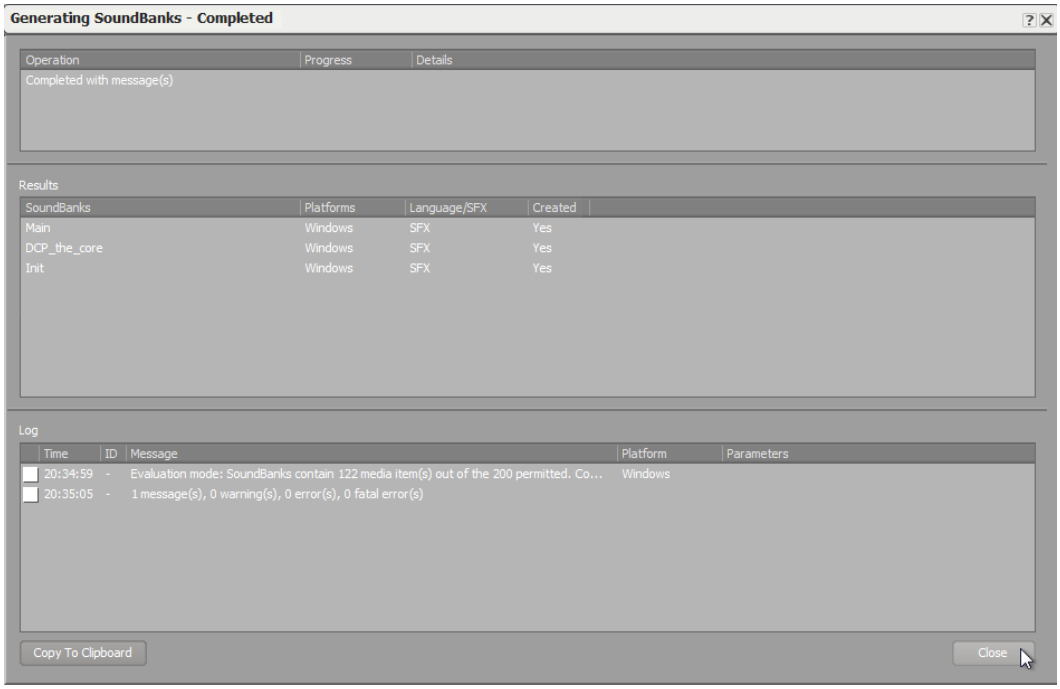


12 **Generate**をクリックします。

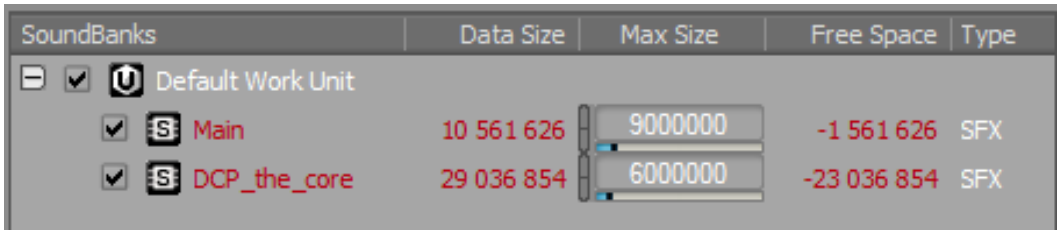


プログレスバーが表示され、システムが、サウンドバンクに追加されたイベントを再生するために必要なオーディオアセットを全て検出し、変換する進捗を表示します。

13Generating SoundBanksウインドウを閉じます。



両方のサウンドバンクのデータサイズが、予算設定を越えてしまったことが確認でき、空きスペースが赤い文字で負数が表示されています。これが予算範囲内にサウンドバンクを収めるために削減が必要なデータのbyte数になります。



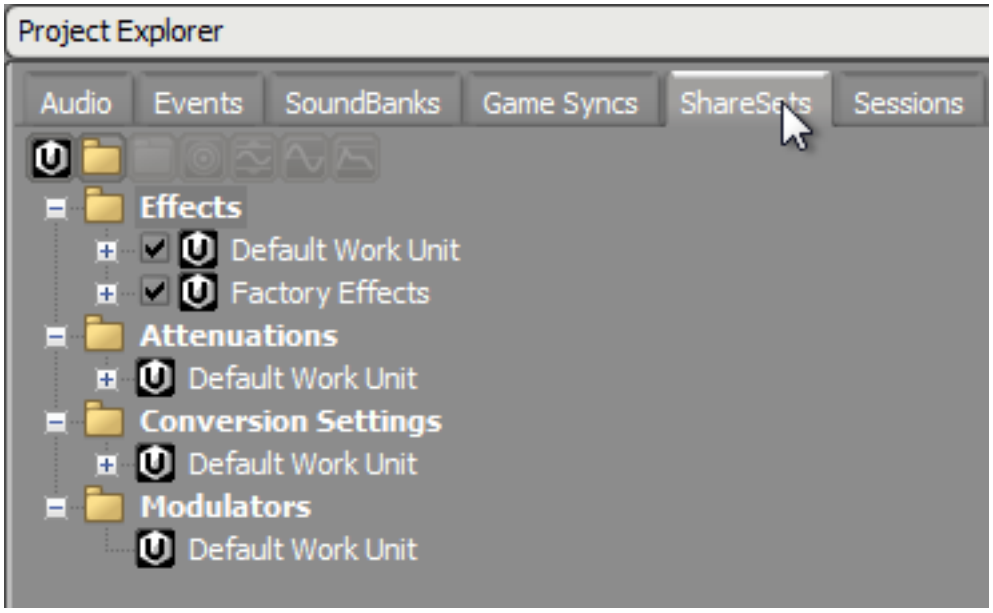
変換設定(Conversion Settings)の作成

次にサウンドバンクのサイズをどのように削減するかを理解する演習を行ないます。これには様々な手法があり、その一つはプロジェクトにインポートされ

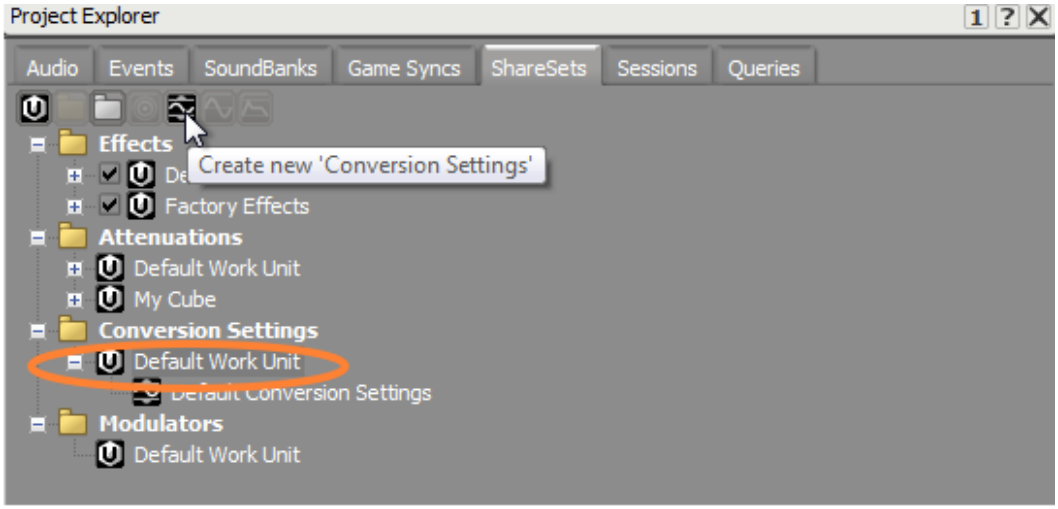
た.wav オーディオファイルを、もっとコンパクトな形式に変換することです。サンプルレートを下げたり、モノへのダウンミックスなどチャンネル数を減らしたり、また別のファイル形式に変換することが考えられます。これは通常、サウンドクオリティの低下につながることにはなりますが、時には必要な妥協判断になります。Wwiseの強みとして、インポート時にこれらの変換設定について気にすること無く、あなたが状況を十分に把握した上で、必要に応じて変換を行なうことが出来ます。

ファイルを1つづつ変換することも出来ますが、通常サウンドのグループに対して適用する変換設定を決めてから、それらをまとめて適用するのが一般的です。具体的には、オーディオオブジェクトに適用するプリセットとして変換設定シェアセット(Conversion Setting ShareSet)を作成します。

1. プロジェクトエクスプローラーで、ShareSetsタブをクリックします。

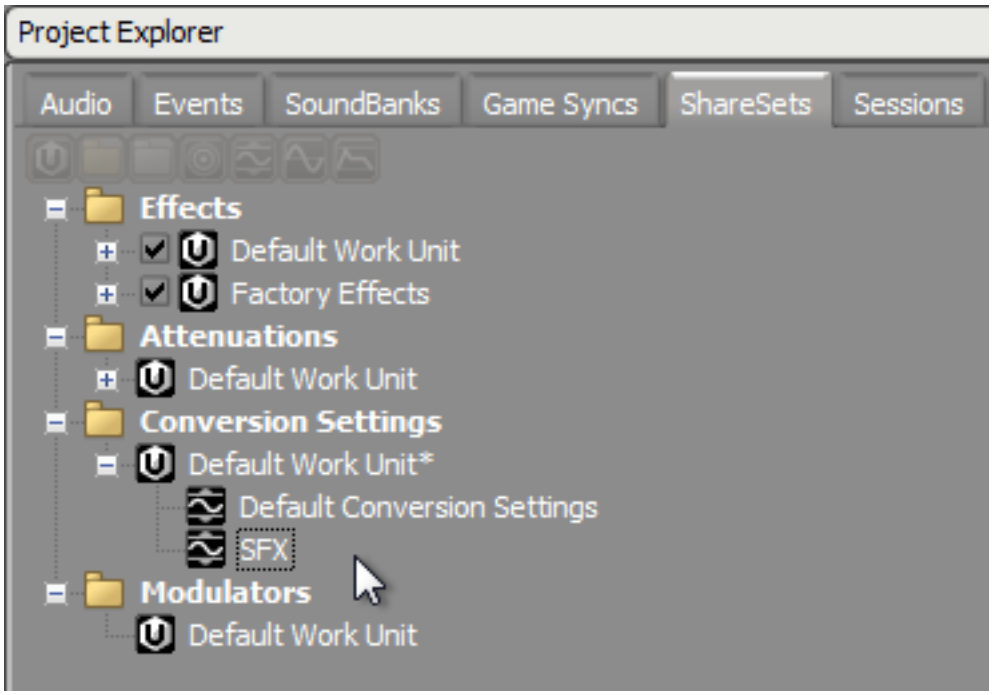


2. **Conversion Settings Default Work Unit** を選択し、**Create new 'Conversion Settings**をクリックします。



あなたのサウンドの多くは一般のサウンドエフェクトから構成されますので、これらの種類のサウンドに一般的に適用する変換設定を作成します。

3. このConversion Settings ShareSetを **SFX**と命名します。



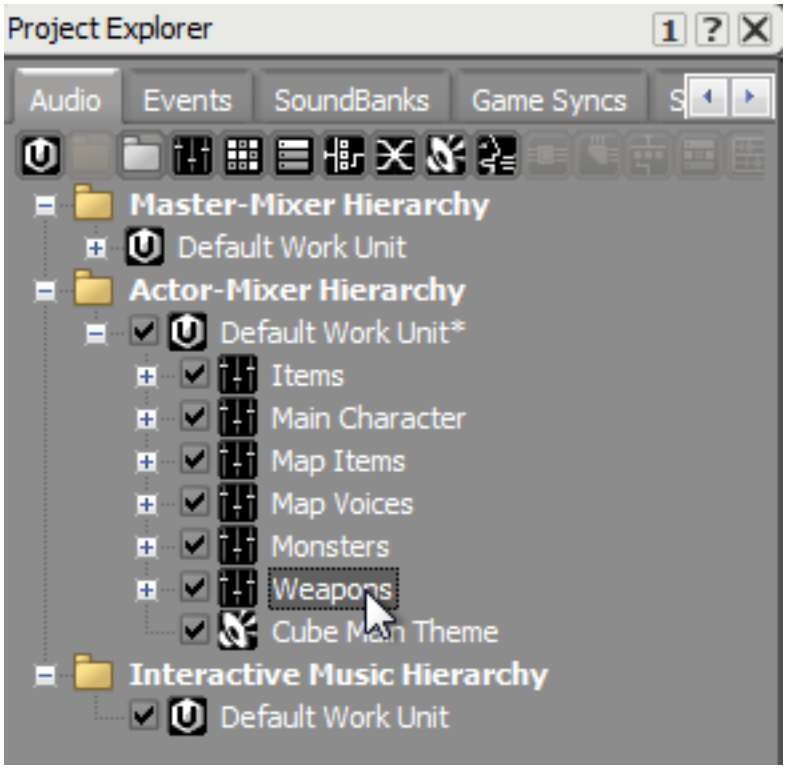
ティップ

追加のConversion Settings ShareSets(変換設定シェアセット)も別の種類のサウンド用に作成できます。例えば、ダイアログは周波数帯域が限られているので、影響が気にならない範囲で、サンプルレートを下げることが出来ます。また、ミュージック要素については、低いオーディオクオリティについては敏感になりがちなので、フル周波数として、別扱いしたいと思うかもしれません。

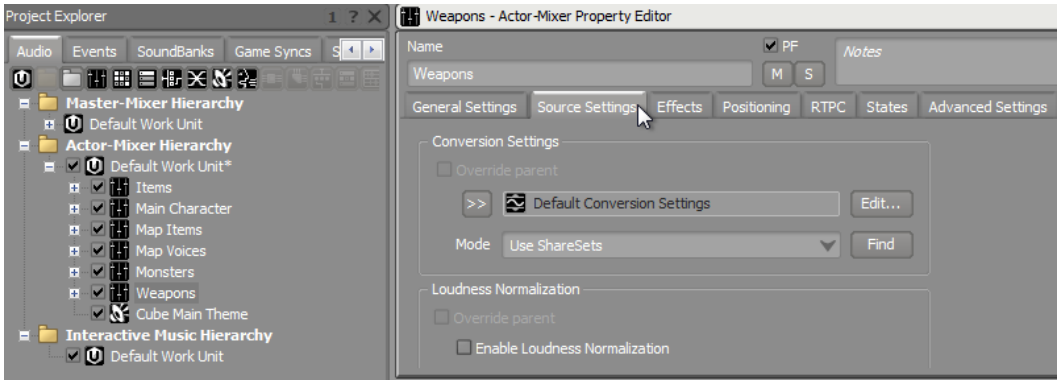
変換設定をアサインする

あなたのプロジェクト用のConversion Settings ShareSet(変換設定シェアセット)が用意できたので、それをプロジェクト階層内のオブジェクトにアサインしていきます。変換設定ShareSetのアサインは、他のオブジェクトプロパティのように、親から子へ継承されます。つまり、変換設定ShareSetをアクターミキサーにアサインすると、それ以下の全てのコンテナやオブジェクトが自動的に同一のShareSetを使用するようになります。

1. **Designer** レイアウトに切り替え、プロジェクトエクスプローラー内で、Audioタブをクリックし、**Weapons** Actor-Mixerを選択します。



2. プロパティエディタ(Property Editor)でソース設定(Source Settings)タブを選択します。

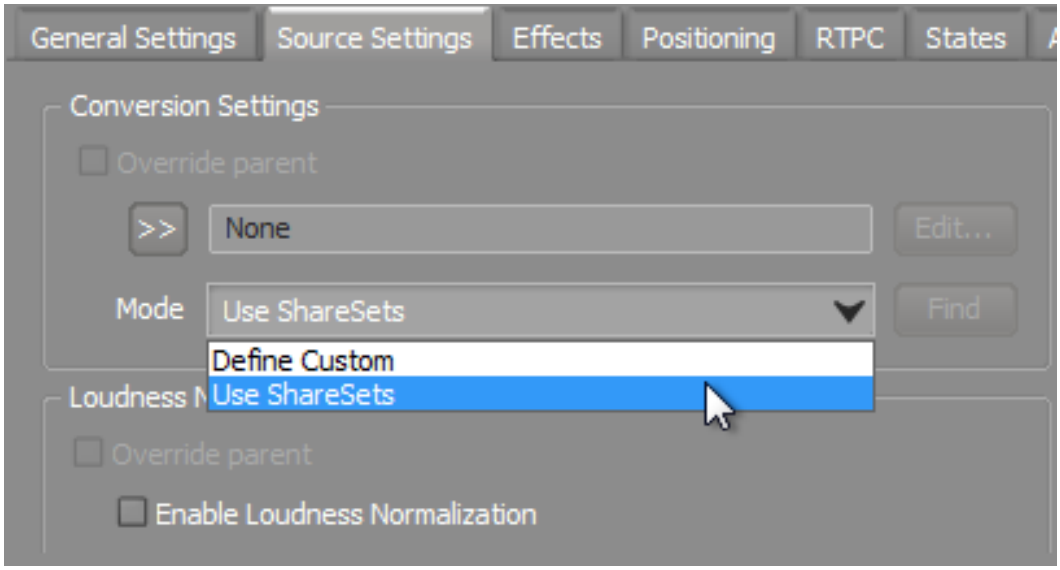


変換設定(Conversion Settings)グループボックスで、**Define Custom**と設定されたモードがあります。ShareSetを持たない、特別な変換設定を適用したい場合にはカスタム定義(Define Custom)を使用します。**ShareSet**を生成し、**Mode**パラメータを**Use ShareSets**に変更しているので

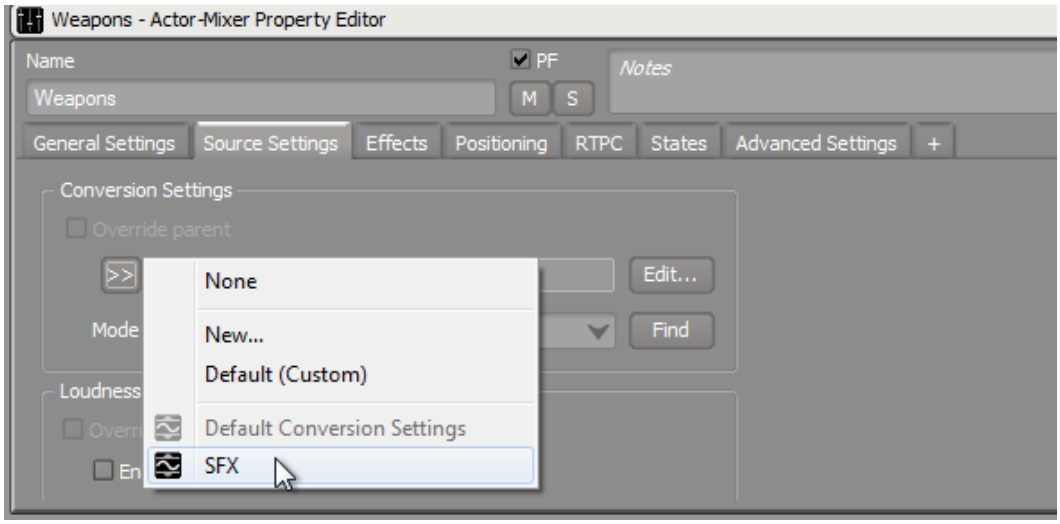
注記

新規オブジェクトはデフォルトでShareSetを使用(Use ShareSets)の設定になっています。

3. **Mode**パラメータを選択し、**Use ShareSets**を選択します。

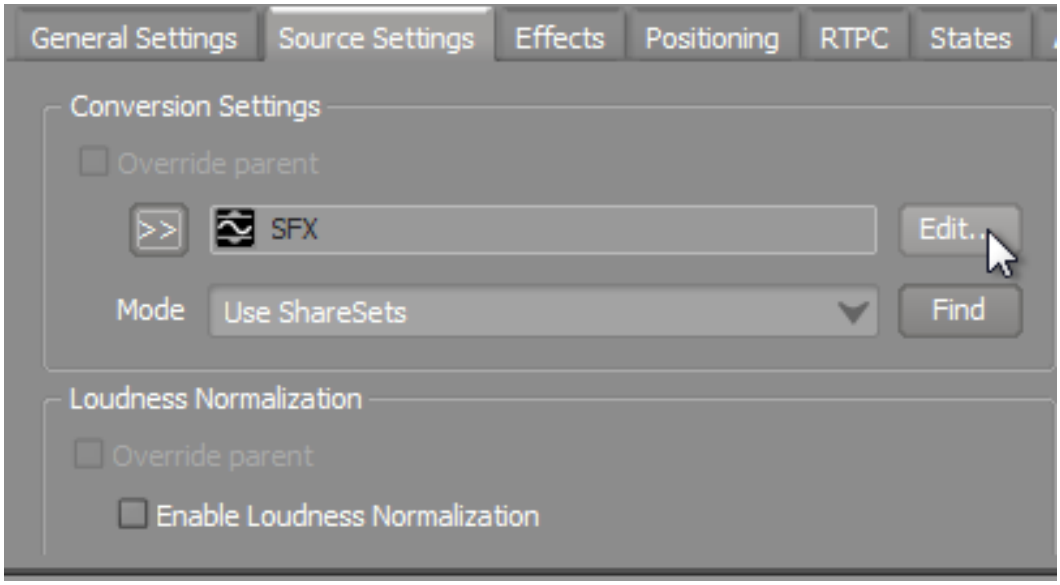


4. セレクターをクリックし、作成したShareSetのリストを表示し、**SFX**を選択します。



次にSFX変換ShareSetの変換パラメータを調整する必要があります。

5. SFX ShareSetの右の**Edit**をクリックします。

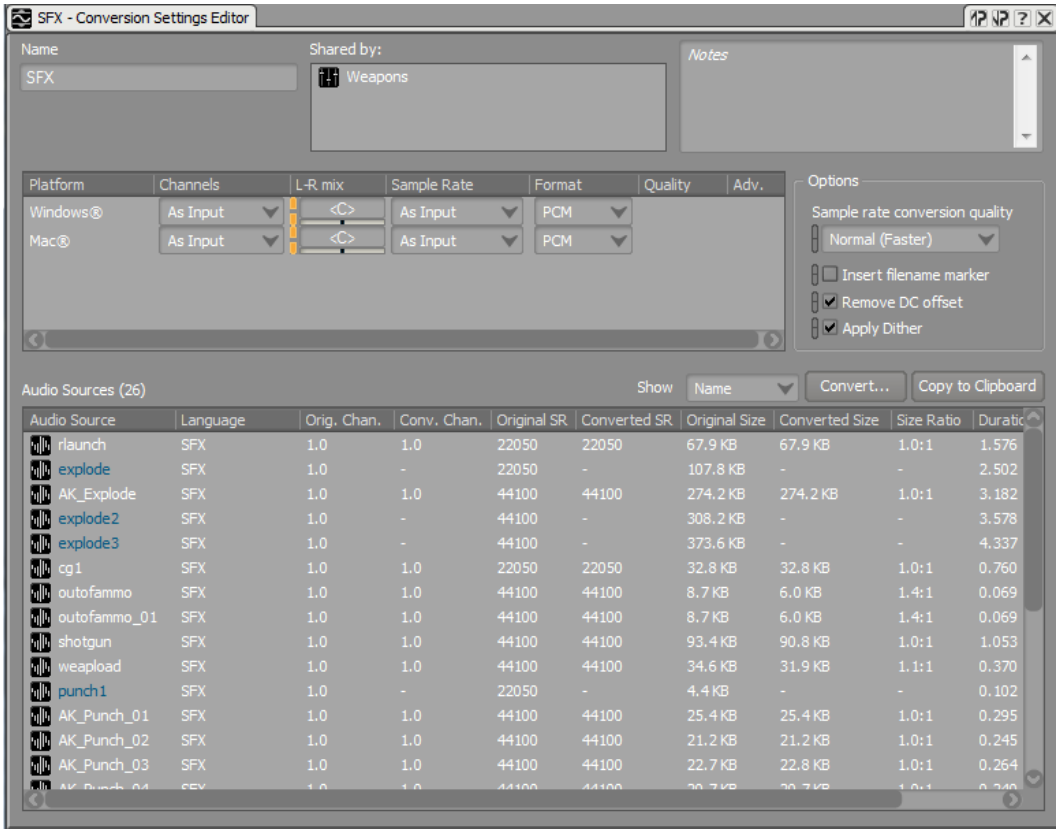


変換設定エディターが開きます。ウインドウ上部にはこのShareSetにアサインされた任意のオブジェクトに適用する変換パラメータが表示されています。ウインドウ下部には現在ShareSetにアサインされたオブジェクトのリストが、チャンネル数、サンプルレート、最も重要なオリジナルのファイルサイズなどの詳細が表示されています。

いくつかのオーディオソース名称はブルーで表示され、変換済みサイズ行が空白であることを確認して下さい。これはこれらのファイルに対して変換処理がまだ行なわれていないからです。ホワイトで表示されるオーディオソースは、本レッスンでサウンドバンクを生成した際に、既に変換済みです。

注記

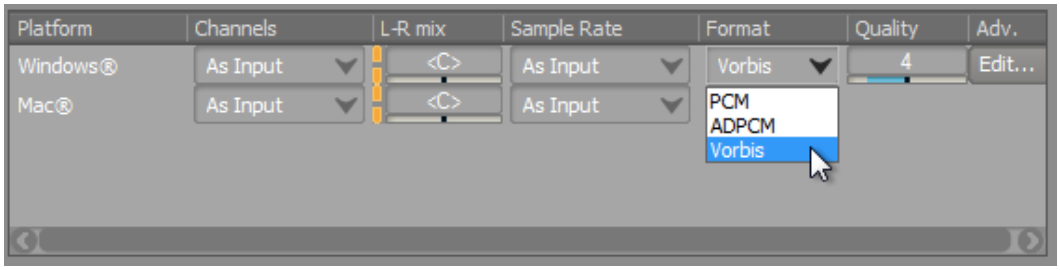
一番左端にAudio Source名が見えなければ、行ヘッダの右端をドラッグすることで行の幅を広げることが出来ます。



サイズ削減に役立つ、オーディオファイルに適用可能な変換プロセスには様々な種類があります。ステレオチャンネル音源をモノ音源にしたり、サンプルレートを下げることはファイルサイズの削減には役立ちますが、オーディオの聞こえ方にはっきり分かる変化をもたらす可能性もあります。

もう一つのオプションはフォーマットの変更です。デフォルトで、Wwiseにインポートされるオーディオは、一般にPCMファイルで、ファイル自体に対してデータ圧縮が適用されていません。MP3ファイル等に使用されるデータ圧縮は、ファイルサイズの削減に大きく役立ちますが、その適用の仕方次第では、あまり変化が見られないことがあります。WwiseではMP3圧縮のオプションは提供していませんが、同様な圧縮フォーマットとして、クオリティについては様々な意見があるようですが、ゲームオーディオのインテグレーションにおいて標準となっているVorbisを提供しています。一つ考慮すべきことは、ファイルが圧縮された際には、再生時にはゲームシステム側でそれを伸張する必要があり、システムプロセッサへの負荷が加わることです。こういうわけで、ファイルサイズとプロセッサ負荷のバランスを考慮しなければなりません。この演習では、Vorbis圧縮を使用することにします。

6. WindowsとMacフォーマット用の**Format** プルダウンメニューをクリックし、**Vorbis**を選択します。

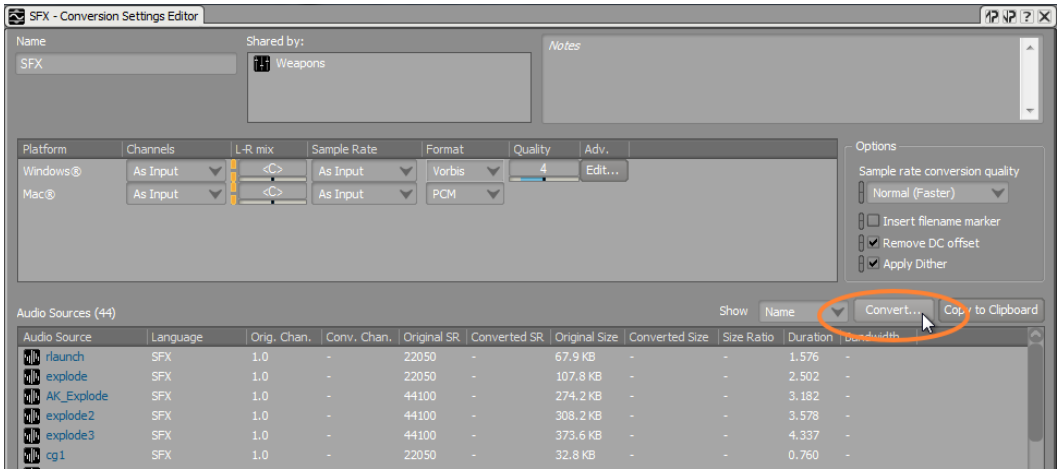


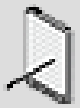
オーディオソース表示がブルーになり、新規に選択された変換設定ではまだ変換されていないことを示しています。

Vorbisオプションに特有のクオリティプロパティ値が表示されています。本項目がVorbis変換後のサウンドのクオリティを表し、最低レベルの-2から10までの範囲になります。より高い値はより優れたサウンドクオリティに対応しますが、データ削減の面では効果が少ないことを意味します。4という値が出発点としては有効で、ファイルサイズを劇的に削減できるのと同時に、大抵の場合で納得のいくサウンドを得得られます。

ここで設定を適用します。

7. **Convert**をクリックし、変換設定を適用します。

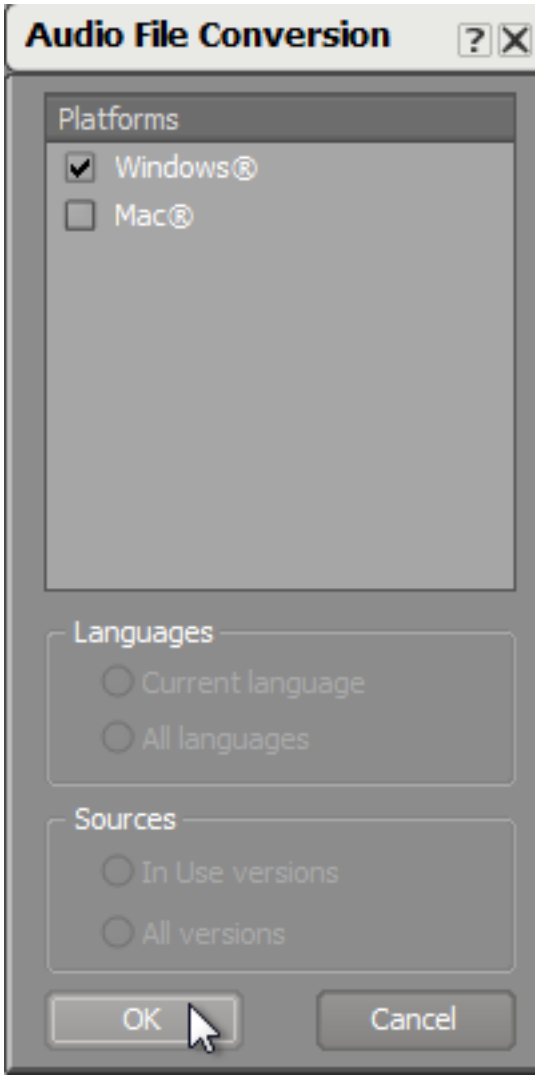




注記

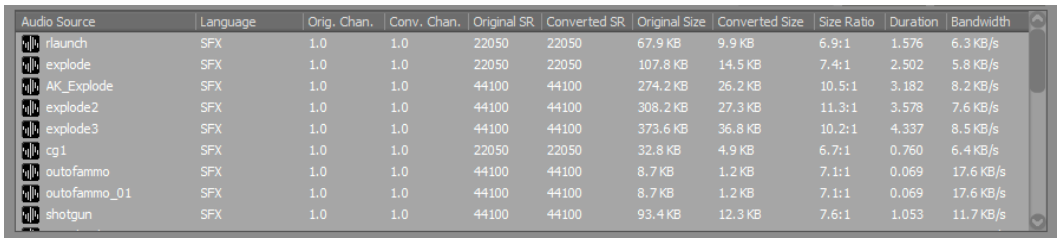
サウンドバンク生成時にも、変換設定が適用されます。

オーディオファイル変換ダイアログボックス(Audio File Conversion dialog box)が開き、どのゲームシステムに適用するかを確認します。



8. あなたのシステムに適切なオプションを選択し、**OK**をクリックします。

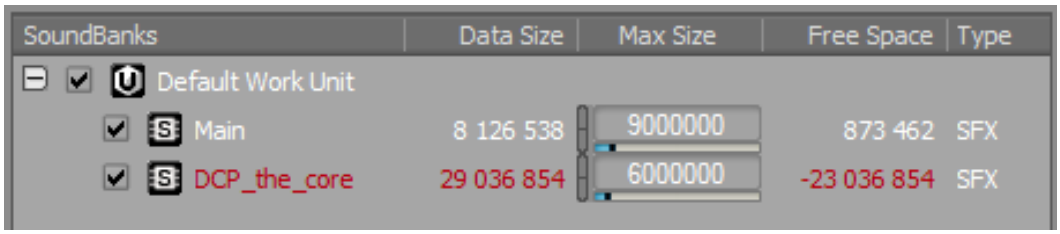
処理がすぐに終わると、リスト中のオーディオファイル名がホワイトに変わり、全てが変換されたことを示します。変換後のファイルサイズを確認します。Vorbis変換により、多くのファイルのサイズが70-80%程度削減できました。これは大きなスペース削減です。



Audio Source	Language	Orig. Chan.	Conv. Chan.	Original SR	Converted SR	Original Size	Converted Size	Size Ratio	Duration	Bandwidth
rlaunch	SFX	1.0	1.0	22050	22050	67.9 KB	9.9 KB	6.9:1	1.576	6.3 KB/s
explode	SFX	1.0	1.0	22050	22050	107.8 KB	14.5 KB	7.4:1	2.502	5.8 KB/s
AK_Explode	SFX	1.0	1.0	44100	44100	274.2 KB	26.2 KB	10.5:1	3.182	8.2 KB/s
explode2	SFX	1.0	1.0	44100	44100	308.2 KB	27.3 KB	11.3:1	3.578	7.6 KB/s
explode3	SFX	1.0	1.0	44100	44100	373.6 KB	36.8 KB	10.2:1	4.337	8.5 KB/s
cg1	SFX	1.0	1.0	22050	22050	32.8 KB	4.9 KB	6.7:1	0.760	6.4 KB/s
outofammo	SFX	1.0	1.0	44100	44100	8.7 KB	1.2 KB	7.1:1	0.069	17.6 KB/s
outofammo_01	SFX	1.0	1.0	44100	44100	8.7 KB	1.2 KB	7.1:1	0.069	17.6 KB/s
shotgun	SFX	1.0	1.0	44100	44100	93.4 KB	12.3 KB	7.6:1	1.053	11.7 KB/s

これがメインサウンドバンクのメモリ予算にどのように影響を与えるかを確認するには、サウンドバンクを生成する必要があります。

9. **変換設定(Conversion Settings)**ウインドウを閉じ、サウンドバンクレイアウトに戻り、**Generate**をクリックします。



メインサウンドバンクはメモリ予算内に収まっていることが確認できますが、DCP_the_core サウンドバンクはまだ予算を超えている状態です。これについては後で対応します。

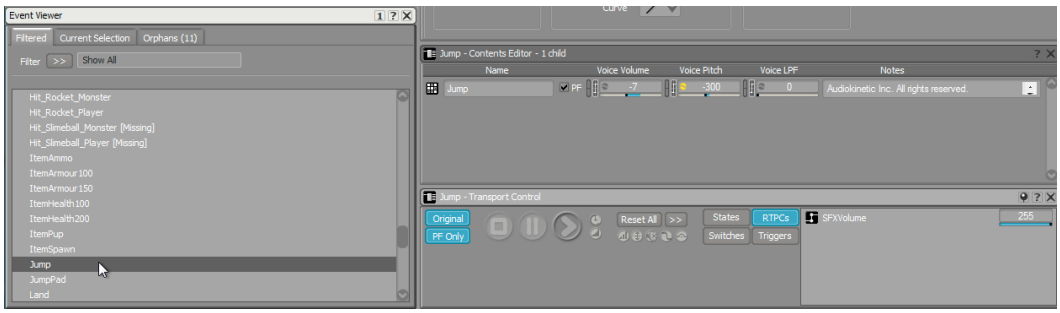
注記

あなたのプロジェクト用に変換設定ShareSetsを作成した後に、そのうちどれをデフォルトとして使用するかを指定することが出来ます。デフォルト変換ShareSetは新規オブジェクトが生成された際に使用されますが、その新規オブジェクトがトップレベルの親オブジェクトである場合に使用されます。オブジェクトに親があれば、親の変換設定が継承されます。オブジェクトに変換設定ShareSetがアサインされていなければ、デフォルトShareSetが、サウンドバンク生成時にオブジェクトが変換される際に使用されます。デフォルトShareSetは、プロジェクト設定(Project Settings)のソース設定(Source Settings)タブでアサインできます。

変換されたオーディオを比較する

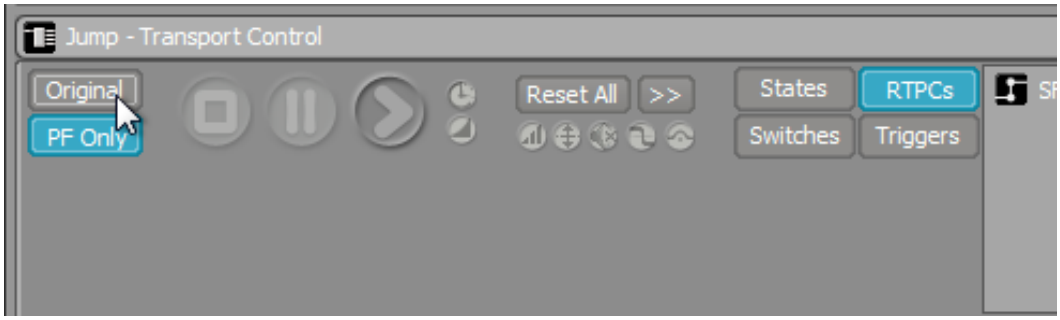
一般にメモリ削減は良いことですが、オーディオクオリティに影響が大きく、目立つようであれば問題があります。Wwiseでは、オリジナルのオーディオと、変換後のオーディオを容易に比較できる方法を提供しています。

1. Designer Layoutに戻り、Event Viewerで、**Jump** イベントを選び、数回再生してみます。



Jumpイベントを再生する際に、オリジナルの未変換のうめき声を聞くことが出来ます。これはトランスポートビューの左上のコーナーにあるオリジナルボタンが選択されているからです。

2. **Transport** の**Original** ボタンをOFFにして、同じ**Jump** イベントを再生してみます。



今回は、変換の結果を聞いています。オリジナル版と変換版の比較が、オリジナルボタンをクリックするだけで行えます。

注記

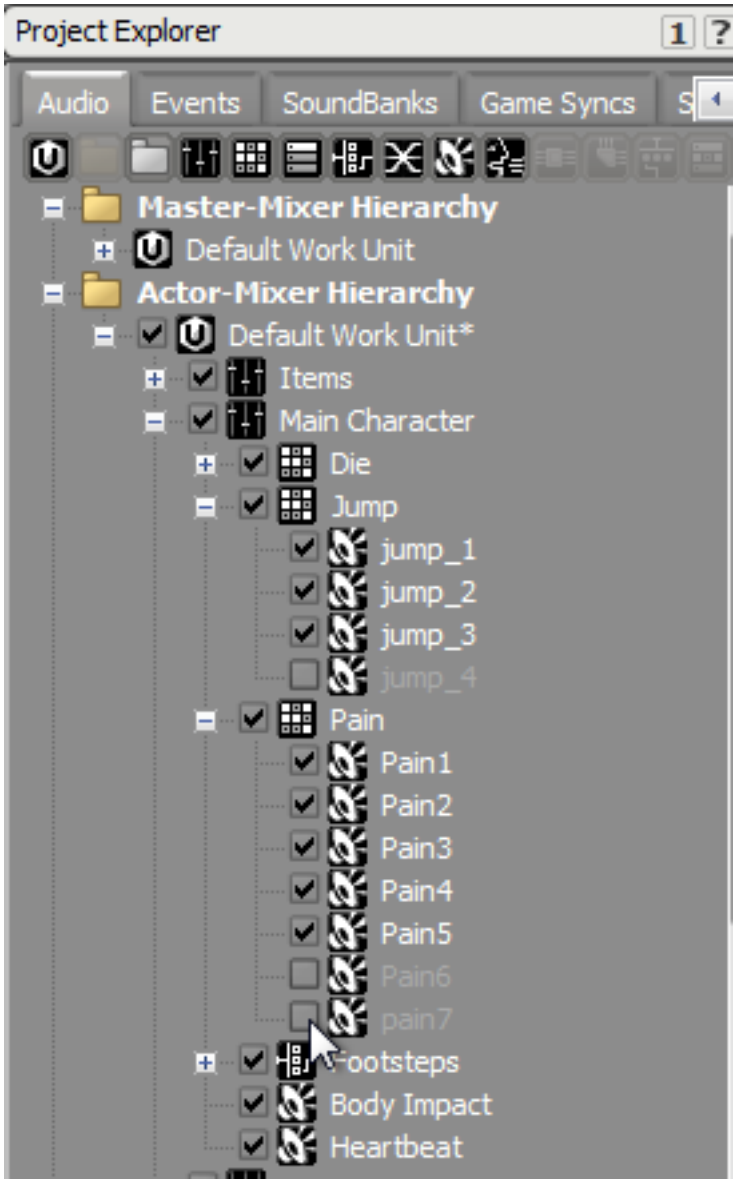
ブルーで表示されるオブジェクトは、関連するオーディオファイルが変換されていないことを示します。本イベントでは、オリジナルボタンが選択されていなければ再生することもできません。

オーディオのインクルード、エクスクルード

場合によっては、ファイルの圧縮のみではメモリフットプリントを低減するのに不十分ではなく、ゲームからオーディオ要素の一部を削除する必要があります。銃声のように、任意のアイテムで用意された唯一のサウンドの元となるオーディオファイルを削除するのは適切ではありません。しかしながら、複数のオーディオファイルが、ランダムコンテナの一部として使用されている場合、、サウンドのリザーブから幾つか削除することは聞く側にとって気がつかない程度の変更であり、予算内に収めるのに役に立ちます。

オブジェクトを削除するのでは無く、プロジェクトエクスプローラーの各オブジェクトの隣にあるチェックボックスをクリアすることで、サウンドバンクのビルド時に除外することができます。

1. ランダムコンテナ**Jump** と**Pain** を探して、それらに含まれるSFXオブジェクトのチェックボックスをクリアします。



オブジェクトが非選択になった際、そのオブジェクトは元々そこに存在していなかったように扱われます。

2. **Jump**と**Pain**のランダムコンテナを再生します。

   これらオブジェクトが再生された際にバラエティは限られるかもしれませんが、十分なバラエティがあり、コンテナではより少ないメモリを使用します。

ストリーミング

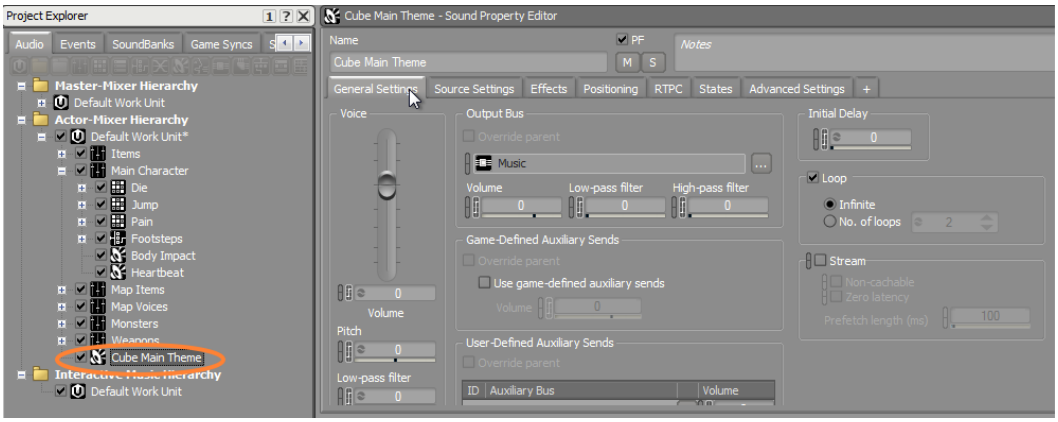
メモリを削減するもう一つの方法は、ゲームシステムのRAMに全て一度にロードしないことです。そうではなく、ストリーミングでは、光学ディスクやハードドライブのようなメモリストレージから直接オーディオを再生する方法を提供します。このオプションは、ゲームデベロッパーにとって、どのようにメモリを管理するか別の方法を提供します。ストリーミングの使用は、各ゲームの技術的な要求に依存

しますが、ストリーミングはミュージックのように数分にわたるような大きなオーディオファイルで特に有効です

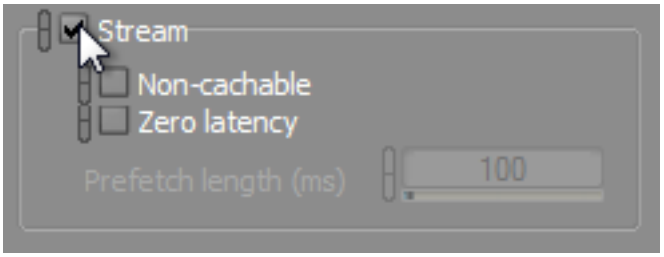
ストリーミングのデメリットは、ゲームエンジンがサウンドをコールした際に、サウンドの再生が始まるまでに多少の時間がかかる可能性があります。それは、ファイルを特定し、ストリーム処理が開始されるまでに時間がかかるからです。これは銃声のような音には不向きですが、環境音や、ミュージックが連続再生するようなものについては有効です。ストリーミングのもう一つの制限は一度にストリームから読むことの出来るデータ量に技術的な制限があり、同じストリームでビデオのようなゲームの他の処理と帯域幅を奪い合うことになる可能性があります。

Cubeデモでは、 DCP_the_core サウンドバンクがメモリ予算を超えてしまう問題を解消する手段になります。 Cubeデモのメインテーマミュージックは、前のレッスンでDCP_the_coreサウンドバンクに追加されたDCP_Music イベントでトリガされ、それが現在メモリ予算をオーバーしています。ミュージックのストリーミングはメモリ使用量を減らす便利な方法で、サウンドのクオリティに影響を与えません。

1. **Designer**レイアウトで**Cube Main Theme** オブジェクトを選択し、プロパティエディタのGeneral Settingsタブが選択されていることを確認して下さい。



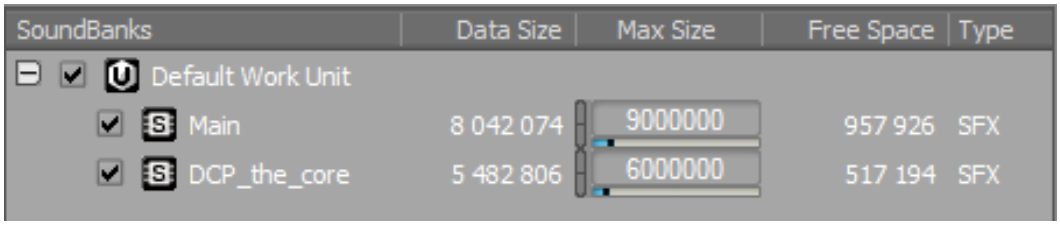
2. プロパティエディタで**Stream** チェックボックスを選択します。



その他のストリーミングプロパティが表示されますが、ここでストリームを開始するまでのレイテンシーを最小化するのでなければ、これらパラメータは変更する必要はありません。

オブジェクトの一部を含まないようにしたことと、ストリーミング処理によるメモリ削減の効果を確認するには、サウンドバンクレイアウトで、サウンドバンクを生成する必要があります。

3. **SoundBank**レイアウトを選択して、**Generate**をクリックします。



DCP_the_core サウンドバンクにおいて、ストリーミング処理に切り替えたことで、大きな影響があったことが分かります。具体的には29,000,000 bytesで、元々の予算よりも6,000,000 bytes少ない値です。

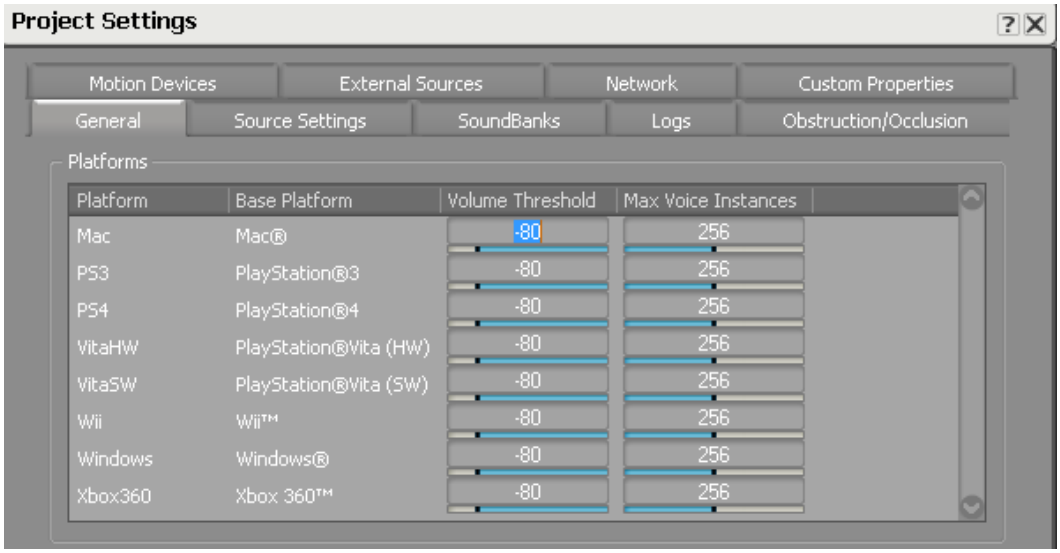
プロセッサーにおける最適化

ゲームプレイ中、サウンドが再生される度、ゲームシステムのCPUでは多少の処理能力が必要になります。オーディオのプロセッサにかける負荷は比較的小さいですが、システムに多大な負荷を与える3Dグラフィクス描画など、ゲームの他の処理とプロセッサーを共有しなければならないことを考慮する必要があります。こうした理由から、オーディオエンジンがプロセッサーを最大限に生かし、処理サイクルを無駄にしないようにする方法を学ぶ必要があります。

ゲーム内で再生されるサウンドそれぞれは、それが再生されるにあたりボイスと呼ばれるものが必要です。本レッスンの中で以前ダイアログに関連して使用されたボイスオブジェクトと混同しないようにしてください。この場合は、ゲーム内のサウンドを生成する論理メカニズムとしてボイスを理解してください。全てのアクティブなボイスはプロセッサーの処理サイクルを消費します。

ゲームによっては、特にCubeデモのような3Dワールドのゲームでは、ボイス数が急激に増加することがあります。あなたのキャラクターが30体の悪役に対して武器を発砲する一方で、これら悪役も同時に打ち返してくる場合、これらのサウンド、及び全てのキャラクタの足音と、その他全てのバックグラウンド環境音は、すぐにボイス数を増大させてしまいます。実際には、これらのサウンドの多くは聞こえることが重要でありません。100m先を歩いている悪役の足音が遠くで静かに再生されているかもしれませんが、これらの音は、同時に10体の他の悪役が10mの距離で武器を発砲している場合には、聞こえそうにありません。遠くの足音の処理には、2mの距離で発砲される銃声の処理と同じくらいの時処理負荷がかかりますので、聞こえそうも無いサウンドをレンダリングすることは処理の無駄になります。この無駄を最小限に留めるために、あなたのプロジェクトでは、ボイスを介して実際にレンダリングする必要があるサウンドの最小ボリュームを指定する設定があります。

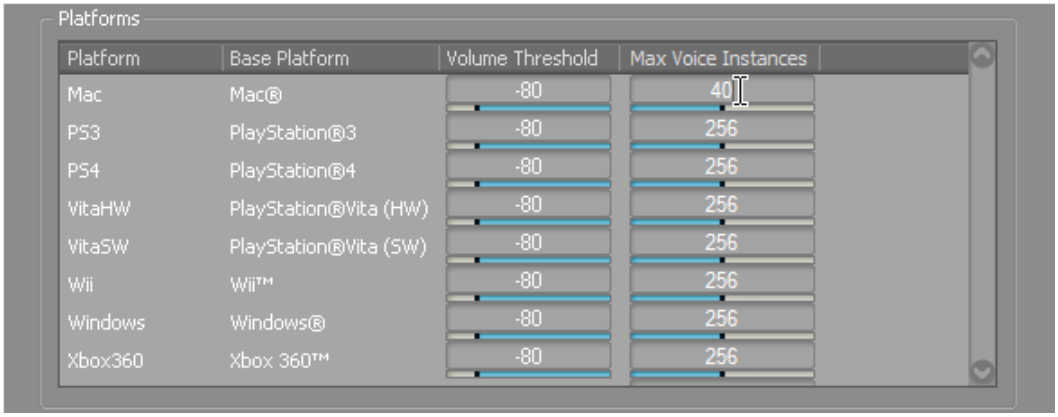
1. メインメニューで、**Project Settings**を選択し、Generalタブを選びます。



開発対象のゲームシステム毎に異なる構成を設定をすることができます。例えば、Wii UはPlayStation4と比べ処理能力が低いため、これらの値をゲームシステム毎に個別に調整することが出来ます。音量しきい値は、デフォルト設定で -80で、単位はデシベルになります。つまり、このゲームでは80dBのダイナミックレンジがあるということです。ゲームシステムへの負荷を軽減したい場合には、音量しきい値をより高く設定することでダイナミックレンジを押さえることで可能ですが、その場合、微妙な環境音のように、あなたが聞かせたい音もカットオフされてしまう危険性があります。

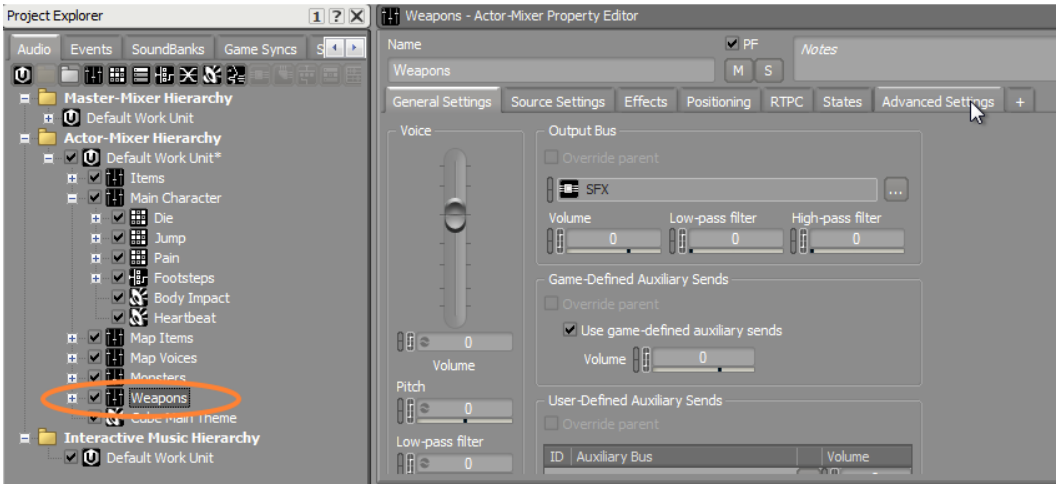
また、同時にシステムを介して再生できる最大ボイス数を定義する別のパラメータもあります。こちらも併せて、システムへの過負荷を回避することが出来ます。このパラメータを低く設定することで、オーディオエンジンはCPUへの負荷を押さえることが出来ますが、特に多くのアクションが行なわれている戦闘中などで、あなたが聞かせたいサウンドがカットオフされてしまう可能性があります。ゲームが最も緊迫した状況において、あなたのメインの武器のサウンドを失うことは、ゲーム体験から離れてしまうこととなり、最悪の状態となります。でもご心配は無用です、この後の演習でプライオリティ付けの方法を学びます。

2. あなたのシステムの**Volume Threshold** を-50に変更し、**Max Voice Instances**を40にセットし,**OK**をクリックします。



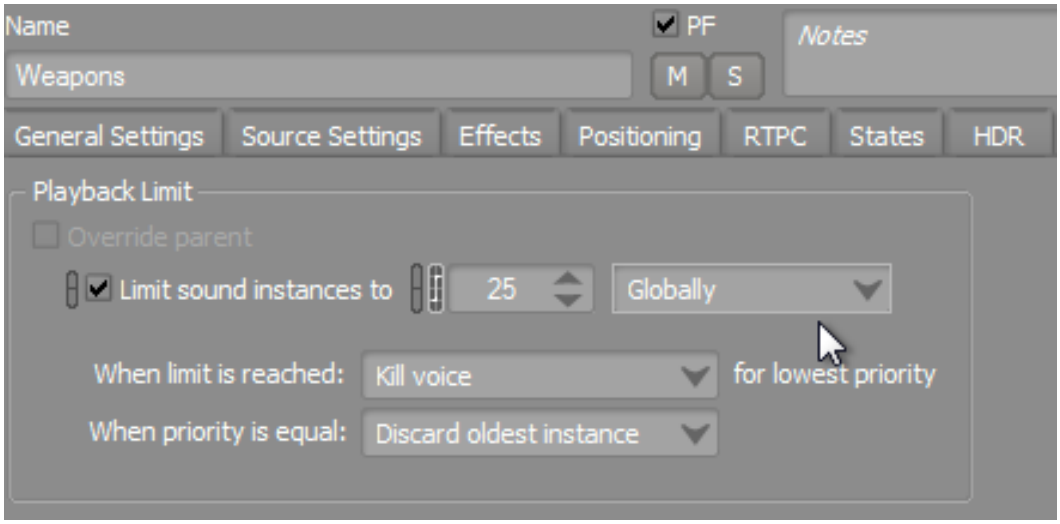
特定のサウンドは、そのものがボイス負荷の高いものがあります。Cubeデモでは、チェインガンのような武器は、連射される性質上、発射音が重なり、多くのボイスを急激に生成する可能性があります。このような状況に対応するために、任意の単一オブジェクトが、ゲーム中において同時に扱うことが出来るボイス数を管理することが出来ます。

3. **Designer**レイアウトに戻り、Project Explorerで、**Weapons**アクターミキサーを選択し、Property Editor内で、Advanced Settingsタブをクリックします。



Playback Limitエリアでは使用可能な最大ボイス数を設定することができます。これを設定することによって、各ゲームオブジェクトにこの制限値が適用されます。つまり、各キャラクタはそれぞれ特有の武器用に50種類のボイスを使用することが出来る、もしくはこれをグローバル対象に設定することで、同時に聞こえるのが1種類あたり50個の武器ボイスに制限されます。後者は特に有用で、それはプレイヤーが同時に再生される同一タイプのサウンドを、数種類以上判別することはなかなか出来ないからです。

4. **Limit sound instances to**チェックボックスを選択し、インスタンス値を25に設定し、プルダウンメニューから**Globally** を選びます。



サウンドによっては、ゲーム中何が起ころうとも、Killされることが無いようにしたいものもあります。このような場合に対応するために、Wwiseにはボイスのプライオリティシステムがあり、ゲーム中で聞かせる重要性に従った、サウンドの種別をランク設定することができます。このように、ゲームシステムがボイスをKillする際に、本当に聞かせる必要があるサウンドは無音にならないようにすることができます。

意図せずオフにしたくないサウンドの一例としては、プレイヤーのダイアログや、ミュージックなどがあります。この場合、ミュージックオブジェクトに対して高いボイスプライオリティを設定することでプロテクトすることができます。

5. **Cube Main Theme**オブジェクトを選択し、**Priority**値をvalue to 80に変更します。

これで、ゲームプレイ中にミュージックが不注意からオフになることがほとんど無くなります。

ティップ

Offset priority チェックボックスと、設定値により、減衰エディターで指定された最大距離値に達した際に、オブジェクトのプライオリティがオフセットされます。これは足音など、リスナーの近くにある場合にプライオリティが高くなり、遠くにある場合に

はプライオリティが低くなるオブジェクトのプライオリティ設定に有用です。

ティップ

同じくAdvance Settingsに、Virtual Voiceエリアがあります。バーチャルボイスは、通常は再生されるのに、そのプラットフォームで設定された音量しきい値よりも音量が低いため、バーチャルボイスリストに追加されたサウンドを表しています。バーチャルボイスリストは、サウンドの特定のパラメータがサウンドエンジンによりモニタされる仮想環境になります。サウンドは、ボイスレベルに従い、可聴フィジカルボイスと無音バーチャルボイス間を移行します。ボリュームが音量しきい値を超えたり、再生サウンド数が同時再生サウンドの制限を下回っている場合には、オブジェクトは自動的にフィジカルボイスに戻ります。

プロファイラーを使用したリアルタイムモニタリング

ここまでで、メモリとプロセッサーに関する考慮をした調整を行なってきましたが、想定したとおりに動作しているか検証します。あなたは、これまでのレッスンでプロファイラーを使用したように、ライブのゲームプレイをプロファイラーを使用してモニタし、確認することが出来ます。

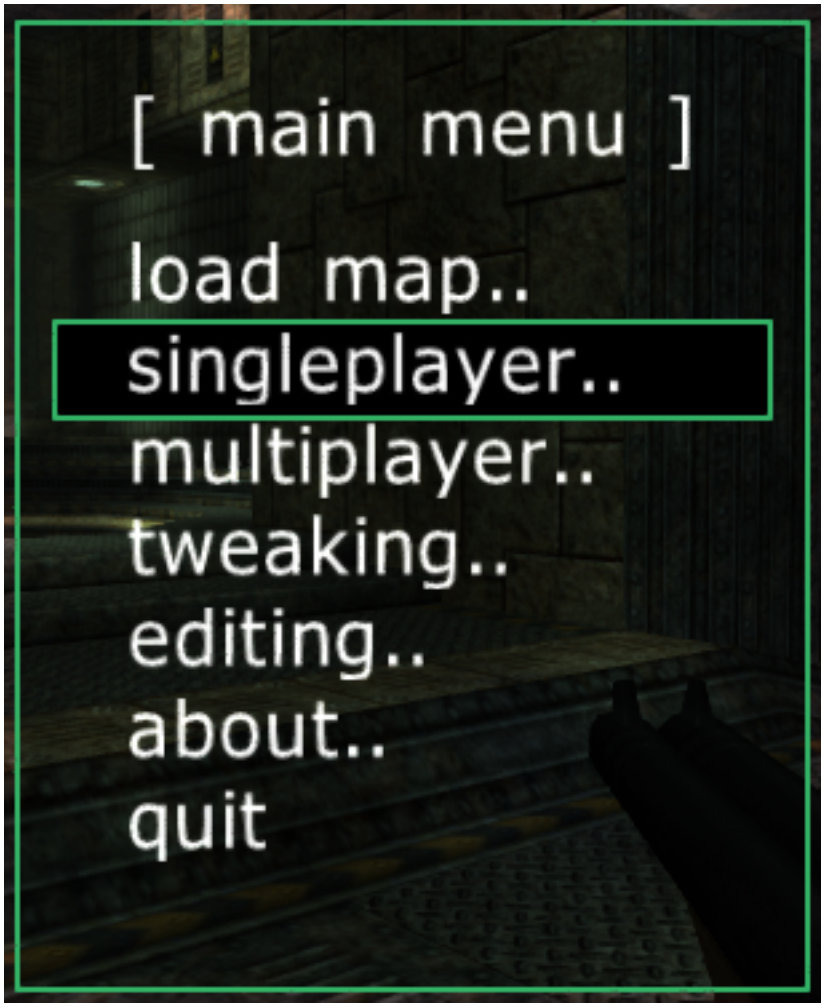
いかなる変更も有効にするには、サウンドバンクを生成する必要があります。

1. **SoundBanks**レイアウトで、**Generate**をクリックしてあなたが行なった変更を含むサウンドバンクを生成します。
2. **Profiler**ビルドのCubeデモを起動します。
3. Wwiseであなたのコンピュータで実行されているCubeデモに接続し、キャプチャを開始します。

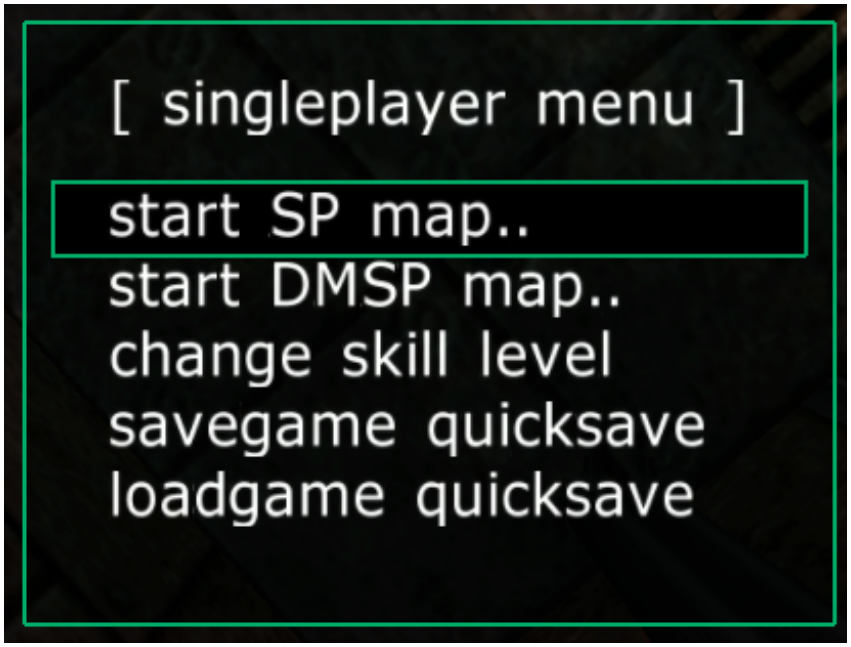
   レベル専用のミュージック等のサウンドを含む DCP_the_core レベルの用のサウンドバンクを生成し、このレベルをプレイしテストします。同時にあなたが生成したDCP_the_core サウンドバンクが正しく動作しているかを検証することが出来ます。
4. Cubeデモで、**escape**キーを押します。

Cubeメニューが表示されます。

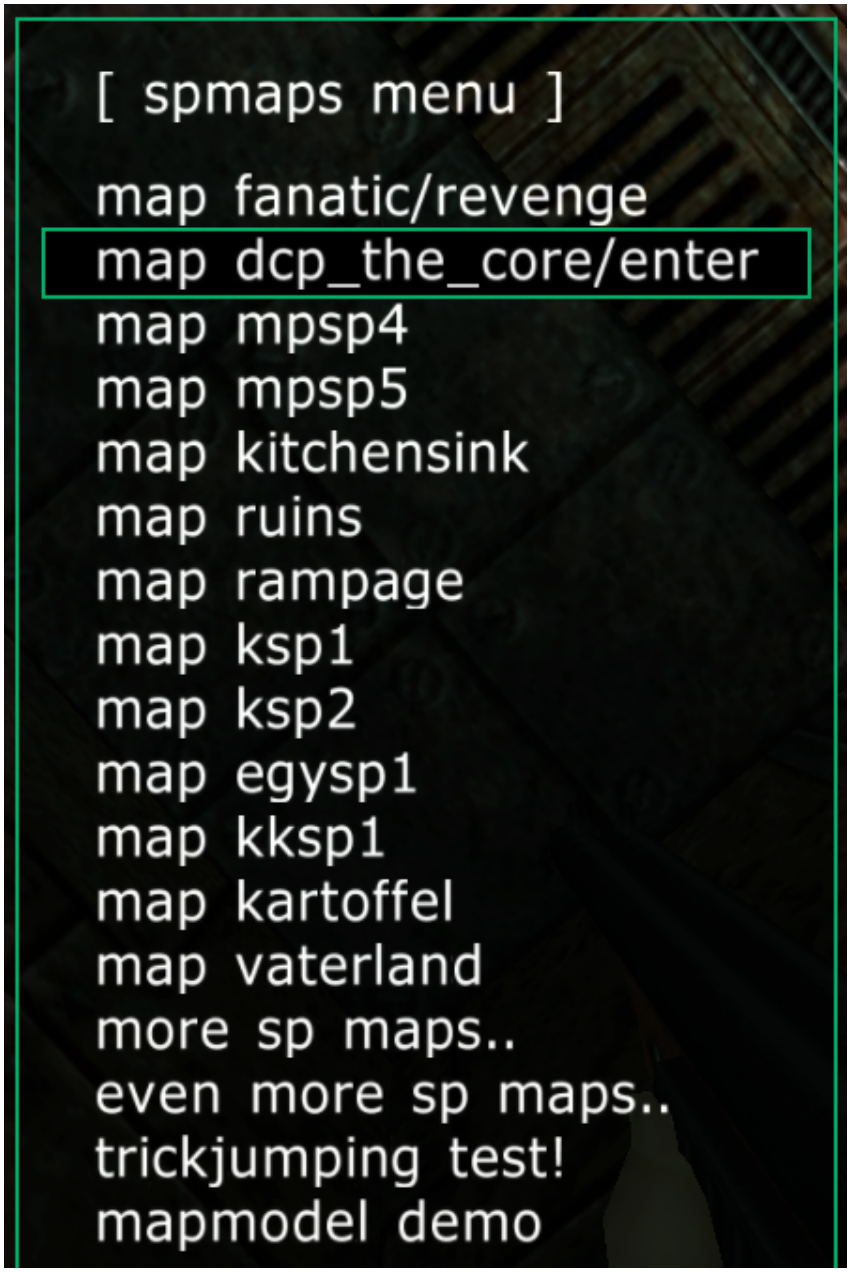
5. **singleplayer**を選択します。



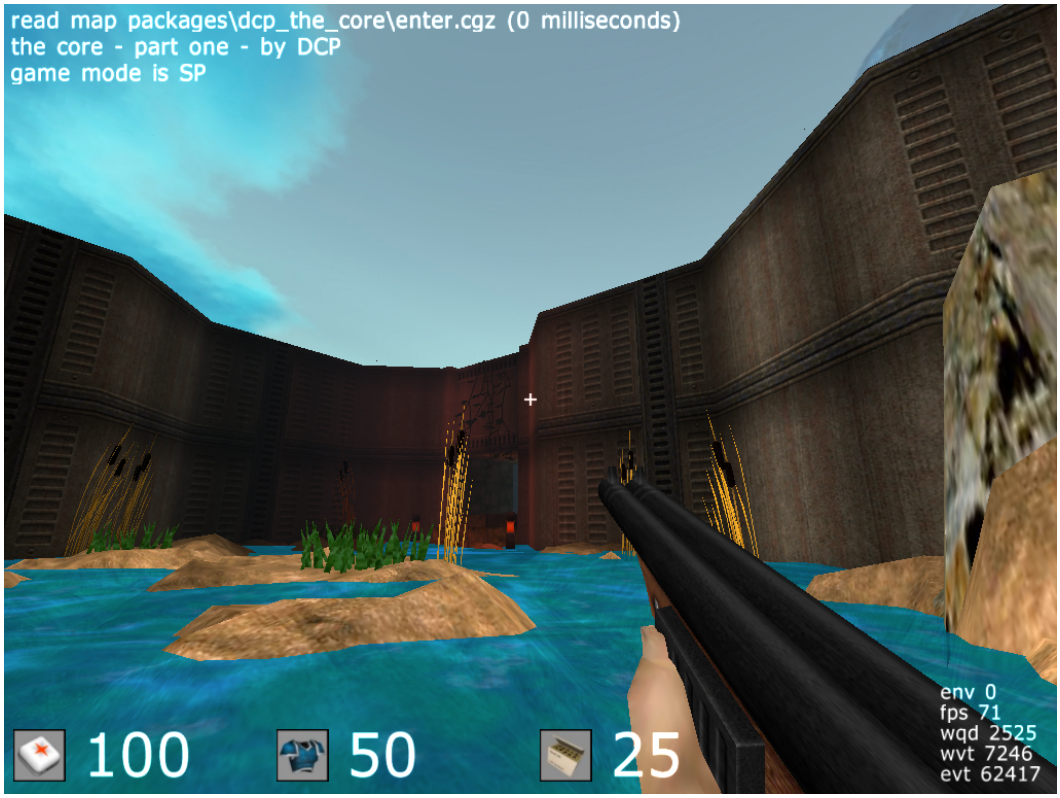
6. **start SP map..** を選択します。



7. **map dcp_the_core/enter**を選択します。



ゲームレベルがスタートします。



ゲーム開始時すぐに、ミステリアスなバックグランド環境音が聞こえます。これは実際にメインテーマオブジェクトが再生されており、あなたが作成したDCP_the_coreサウンドバンクが正しくロードされていることが分かります。

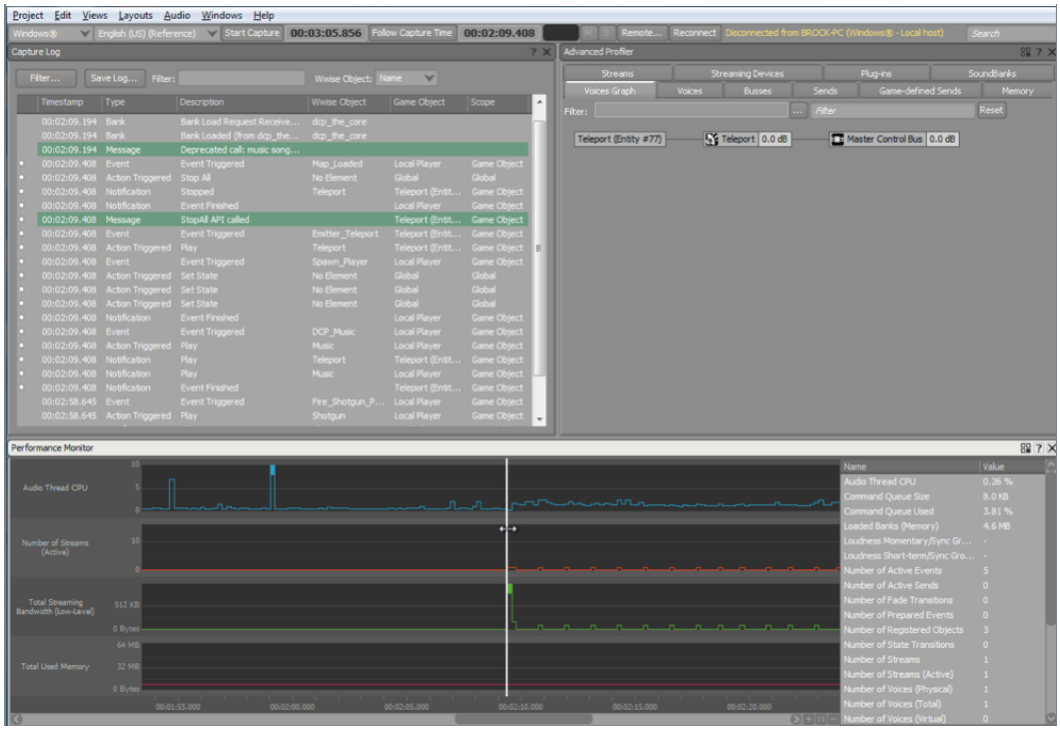
8. Wwiseをインストールするで行なったように、Wwise プロファイラーレイアウトでツールバーの**Remote** ボタンを使用して、ゲームに接続します。

一旦、接続が確立されると、画面下部に複数のグラフ表示が確認できます。一番上のグラフはCPU使用状況を示しています。 オーディオ処理が激しくなると、グラフ表示に動きが確認できます。その下には、ストリームの総数や、ファイルからストリームデータを読み出す際に使用している帯域幅などがグラフ表示されています。これがあなたの再生しているミュージックトラック(Music Track)を表しています。

ティップ

Performance Monitor のタイムカーソルを前後に移動させて、ゲームプレイ中のサウンドエンジンのパフォーマンス履歴を確認することが出来ます。DCP_the_coreレベルがスタートし、ミュージックストリームの再生が開始した際の状況を確認することが出来ます。

一番下のグラフでは、メモリー使用量が表示されており、非常に値が低いことが分かります。また、あなたが使用するサウンドの大半がゲームのスタート時にメモリへロードされているのでおおかた均一です。



常にゲームプレイをキャプチャし、これらのグラフを確認することが有用です。ゲームプレイに関連しない急激な変化に注意して下さい。場合によっては、プロファイラービュー無しには発見が難しかったバックグラウンドで起きている事象があるかもしれません。

ティップ

パフォーマンスモニターの表示はカスタマイズ可能で、確認したいパラメータを素早く表示させることが出来ます。表示領域で右クリックし、パフォーマンスモニター設定を選択し、 合計ボイス数やTotal Plug-in CPU使用状況などを表示に加えることが出来ます。