audiokinetic

The Wwise Project Adventure

インタラクティブオーディオ制作のための ハンドブック

By Damian Kastbauer

2014.1.1

The Wwise Project Adventure

インタラクティブオーディオ制作のための ハンドブック

By Damian Kastbauer

Wwise 2014.1.1 Build 5179

目次

アドベンチャーの開始

インタラクティブオーディオが複雑に展開し続ける一方で、その実装やワークフローの基本は変わりません。Wwiseのサウンドエンジンとオーサリングアプリケーションは、現行世代から次世代までのオーディオインテグレーションを柔軟でダイナミックな制作パイプラインで支援する、基礎から構築されたソリューションです。

本書“Wwise Project Adventure”では“Wwise Fundamentals”で説明したコンセプトを発展させ、架空のゲームプロジェクトを使ってインタラクティブかつダイナミックなオーディオを実装する方法を紹介します。全てのプロセスを包括的に説明しているので、付属のWwiseプロジェクトと合わせて使えば、実際に作業しながらテクニックを学べます。このチュートリアルを読みながら実際に作業するか、付属のプロジェクトに目を通すだけにするかは、読者次第です。

本書の対象者は、ゲーム用オーディオの基礎を理解し、デジタルオーディオ制作の経験も多少ある方です。様々な実装分野をさらに詳しく説明した関連情報へのリンクも文中に掲載されています。なお、本書はユーザードキュメンテーションを置き換えるものではなく、Wwiseオーサリングアプリケーションの機能を具体的な例や図で分かりやすく説明することが目的です。

“Wwise Project Adventure”で作成するサンプルゲームは、いたる所に危険が潜む深い森の中で繰り広げられる、ファンタジー系アドベンチャーゲームです。ヒーローがダンジョンに入り、暗い隅に潜むスコージをパージするチャンスをつかもうとします。強力な剣と揺るぎない勇気で武装したヒーローは、すぐ先に待つ宝に向かい前進していきます。

“Wwise Project Adventure”へようこそ！

第1章 アンビエントの設定

概要

ゲームに最初の物体が配置される前から、ゲームの舞台設定は始まります。作業の目的はシンプルで、これから繰り広げられる何時間にもおよぶ深い森の中の旅を実現するために、繰り返しのない豊かなアンビエントバックグラウンドを作り上げることです。

本章では、以下の操作を説明します。

- オーディオファイルのインポート
- バックグラウンドでループするアンビエントオーディオのソース設定
- バックグラウンドのアンビエントエレメントのランダム再生
- エレメントの3D配置のランダム化
- クロスフェード機能を使ったブレンドコンテナに対して、RTPC（リアルタイムパラメータコントロール）の適用
- アンビエントデザインにおけるサウンドエミッタの使用
- 減衰シェアセットの作成
- ゲームオブジェクトの減衰の設定
- SoundSeed Air - Windの使用
- マスターミキサー階層の構成

基盤の設定

オーディオファイルのインポート

最初にアンビエントバックグラウンドとしてループさせる1つのオーディオファイルを、Wwiseの新規プロジェクトにインポートします。プロジェクトにコンテンツを取り込む時に、気をつける点がいくつかあります。

Wwiseでは、以下の2種類のアセットを、最も基本的なレベルで区別しています。

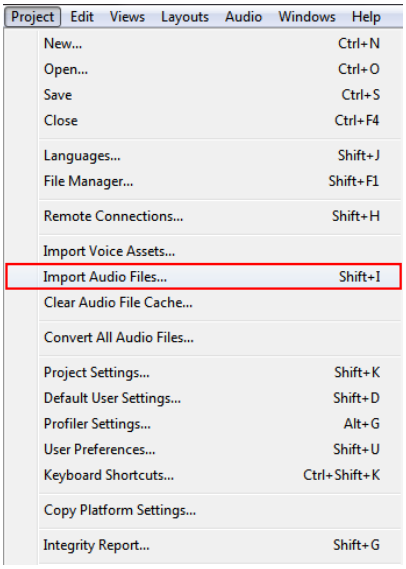
- Wwiseにインポートしたオリジナルのアセット
- 多様なプラットフォーム向けに作成したこれらのアセットの各種バージョン

Wwiseでは、この2種類のアセットをプロジェクトフォルダの別の場所に保存するので、別々に管理できます。まずインポートしたアセットは全て、デフォルトでプロジェクトの[Originals]フォルダに保存されます。通常、インポートされたアセットはチーム内の数人がシェアするので、この[Originals]フォルダはネットワーク上のどこに置くこともでき、既に使用中のソースコントロールシステムで簡単に管理できます。

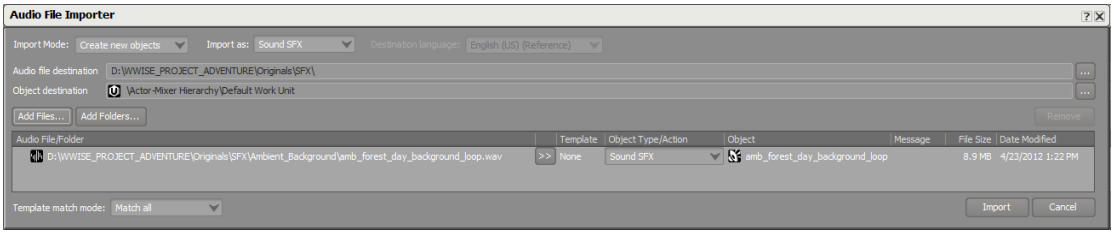
Wwiseのインポート操作によって、以下の処理が行われます。

- 元のメディアファイル（wavファイル）の検証と、プロジェクトの[Originals]フォルダへの保存
- メディアファイルからオーディオソースの作成
- オーディオソースを含むサウンドオブジェクトやミュージックオブジェクトの作成と、Project Explorer画面のAudioタブの該当する階層への表示

Audio File Importer機能を使い、個別ファイルや複数ののファイルが入ったフォルダをアクターミキサー階層にインポートできます。インポート機能は、Projectメニューから選択するか、オーディオファイルをインポートしたいワークユニットやコンテナを右クリックして選択します。



ProjectメニューからImport Audio Filesを選び、インポートを開始



Audio File Importer画面で、ファイル1つをSound SFXとしてインポート

またフォルダからWwiseの階層に、直接オーディオファイルをドラッグ&ドロップできます。

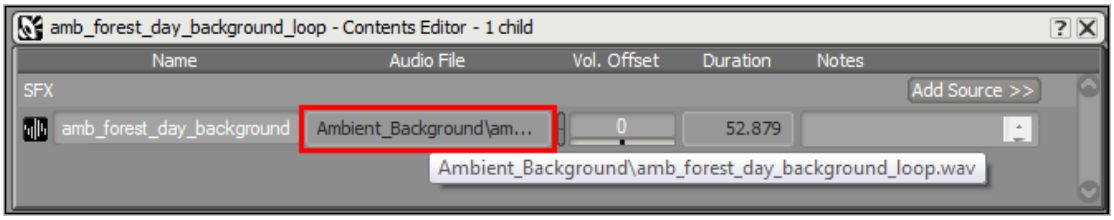
次の例では、アンビエントサウンドシステムの基盤として使うアンビエントサウンドファイルをインポートします。使用するファイルは、以下のプロジェクトフォルダにあります： /Originals/SFX/Ambient_Background/ambient_forest_day_background_loop.wav

インポートされたサウンドファイル

Designer Note

Project Explorer画面のAudioタブに表示されるサウンドオブジェクト、ミュージックオブジェクト、モーションエフェクトオブジェクトは、ソースが現在のプラットフォーム向けに変換されていない場合は青字で表示されます。サウンドソースを対象プラットフォームのフォーマットに変換すると、白字に変わります。また、有効なソースがない状態でサウンドオブジェクトが作成された場合は、赤字で表示されます。

インポートしたメディアファイル（単数または複数）への参照情報と共に、インポートしたファイルはサウンドオブジェクトに追加されます。サウンドオブジェクトをProperty Editorにロードすると、ソースがContents Editor画面に表示されます。

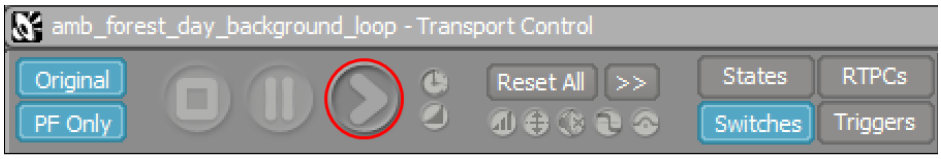


サウンドSFXオブジェクトのNameとソースのAudio Fileの場所

Designer Note

1つのサウンドオブジェクトに代替バージョン、サイレンス、プラグイン、ローカリゼーション用の他言語バージョンなど、複数のソースを入れることができます。

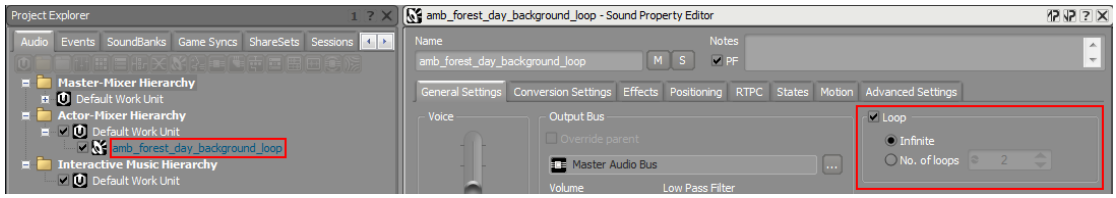
サウンドオブジェクトは、Transport Control機能で試聴できます。サウンドオブジェクト “ambient_forest_day_background_loop”をダブルクリックし、再生ボタンをクリックすると、Transport Control画面にロードされたオブジェクトを再生できます。スペースバーを押してサウンドオブジェクトを再生・停止することもできます。



Transport Control画面の再生ボタン

ループサウンドの設定

これで昼間の森のシーンで使う、アンビエントバックグラウンド用のオーディオファイルをプロジェクトに追加できたので、ループを設定して連続再生させます。サウンドオブジェクトやモーションエフェクトオブジェクトのループ再生は、無限ループかループ回数指定のどちらかを選択できます。



サウンドSFXのループ設定

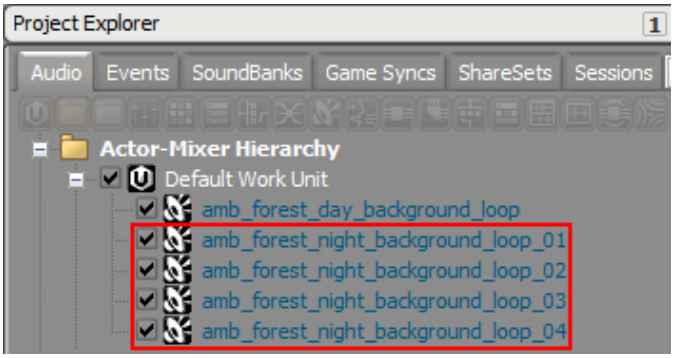
オブジェクトをダブルクリックすると、デフォルトでオブジェクトのプロパティが表示されます。この動作により、Sound Property Editorのフォーカスが、選択されたオブジェクトに移ります。

複数のサウンドファイルのループ設定

ループポイントが聞こえないような1つのサウンドファイルをループ再生させる方法は今日でも使われますが、最近は複数のサウンドファイルをランダムに組み合わせてループさせる方法も頻繁に採用されます。例えば、2分間のサウンドファイルを再生する代わりに、30秒のファイルを4つ、ランダムに再生します。

それでは、この手法で夜間の森のシーンで使うアンビエントバックグラウンドを設定します。まず、1つのアンビエントサウンドファイルを4つの短いファイルに編集し直し、これをインポートしてランダムコンテナに子ファイルとして入れます。この4つのファイルの再生順序をランダムにすることで、ループ再生の編成が毎回変わります。

最初に4つの個別ファイルをインポートしますが、前述のAudio File Importer機能を使うか、元のフォルダからDefault Work Unitにドラッグ&ドロップします。



インポート後の4つのサウンドファイル

ゲーム内の複雑なオーディオ要請に対応するために、異なるタイプのオブジェクトをWwiseのプロジェクト階層に取り込むことができます。アセットをまとめたりプロジェクト構成を作り上げる時に、以下のオブジェクトタイプを組み合わせることができます。

- サウンドオブジェクト
- モーションエフェクトオブジェクト
- アクターミキサー
- コンテナ

アクターミキサー階層には、以下の4種類のコンテナがあります。

- 

  - ランダムコンテナ：1つまたは複数のサウンドオブジェクト、モーションエフェクトオブジェクト、および/またはコンテナをグループ化したもので、これらはランダムな順番で再生する。

- 

  - シーケンスコンテナ：1つまたは複数のサウンドオブジェクト、モーションエフェクトオブジェクト、および/またはコンテナをグループ化したもので、特定のプレイリストに従い再生する。

- 

- スイッチコンテナ：1つまたは複数のサウンドオブジェクト、モーションエフェクトオブジェクト、および/またはコンテナをグループ化し、いくつかのスイッチやステートに割り当てたもので、スイッチやステートは、ゲーム中にある1つの要素の様々な選択肢に対応している。

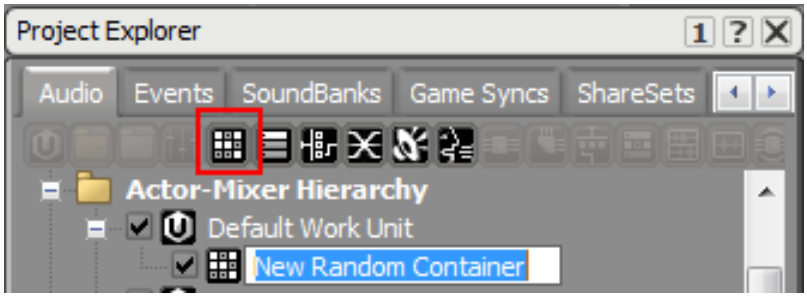
- 

- ブレンドコンテナ：同時に再生される、単数または複数のサウンド、モーションエフェクトオブジェクト、コンテナのグループ。ブレンドコンテナにあるサウンドやコンテナは、まとめて別のブレンドトラックに入れて、RTPCを使ってサウンドのプロパティをゲームパラメータ値にマッピングできる。また、ブレンドトラック内のサウンドに対し、ゲームパラメータ値に基づいたクロスフェードを適用することも可能。

次に、複数のオーディオファイルをランダムな順番で再生します。今回はランダムコンテナを1つ作成します。ランダムコンテナやその他のサウンドオブジェクトを作るには、以下の方法があります。

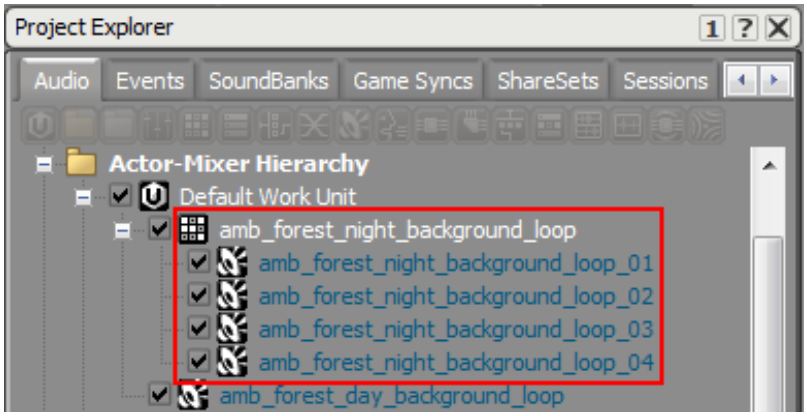
- Project Explorerツールバーから
- 右クリック（コンテキスト）メニューから
- ショートカットキーから（Helpメニューの Wwise Shortcuts Quick Reference Cardを参照）

Default Work Unitを選択してProject Explorerツールバーのランダムコンテナのアイコンをクリックし、選択したワークユニットまたはサウンドオブジェクトの中に、新規ランダムコンテナを追加します。



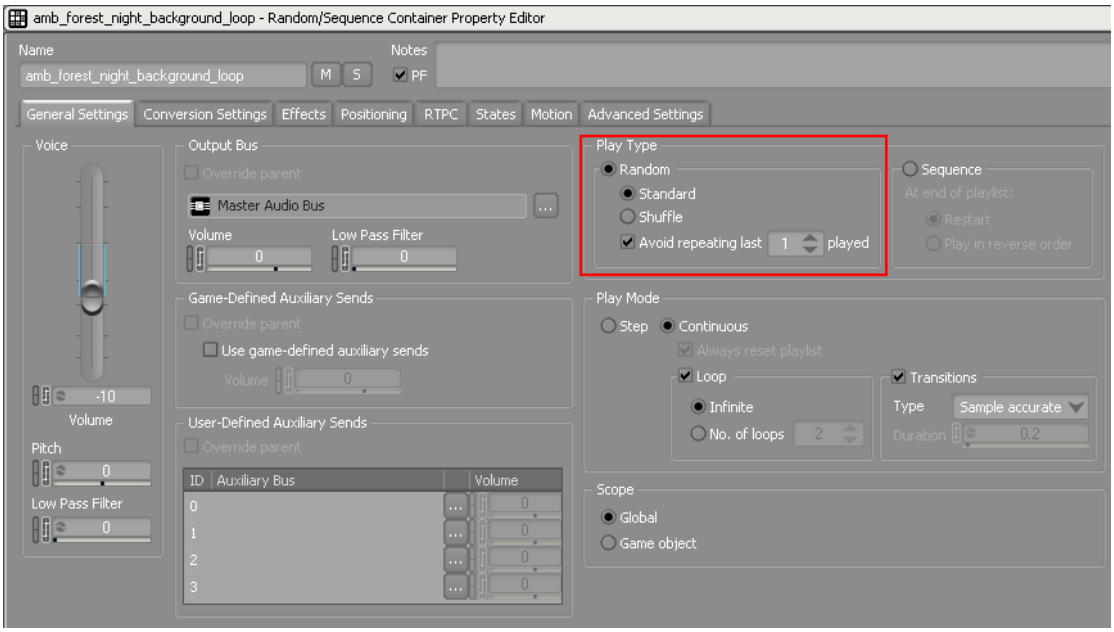
Project Explorerツールバーからランダムコンテナを作る

ランダムコンテナを作り名前をつければ、サウンドオブジェクトを直接ドラッグ&ドロップして入れられます。



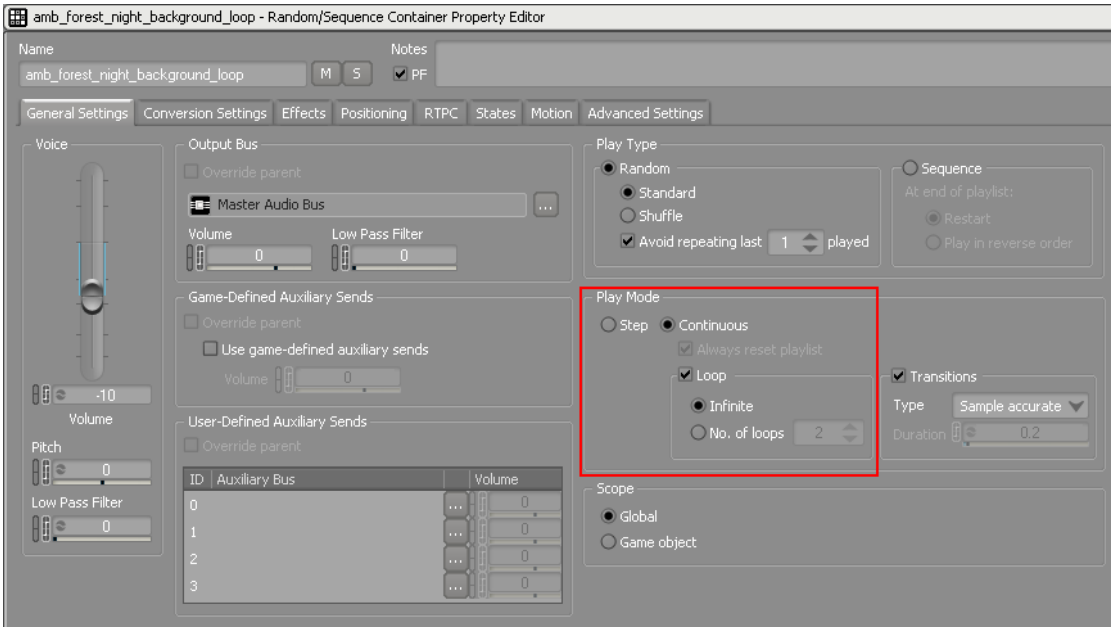
ランダムコンテナに入れた4 つのアンビエントサウンドファイル

これで、夜の森のアンビエントとしてループさせる4つの短いファイルがコンテナに追加され、次に、再生の順序をRandom Container Property Editor画面で設定します。Wwiseには2種類の「ランダム」があり、コンテナ内のオブジェクトを同じ確率で選択して再生する通常のランダム再生と、一度再生したオブジェクトを選択群から外すシャッフル選択のどちらかを選びます。ランダムコンテナをダブルクリックすると、ランダムプロパティが表示されます。



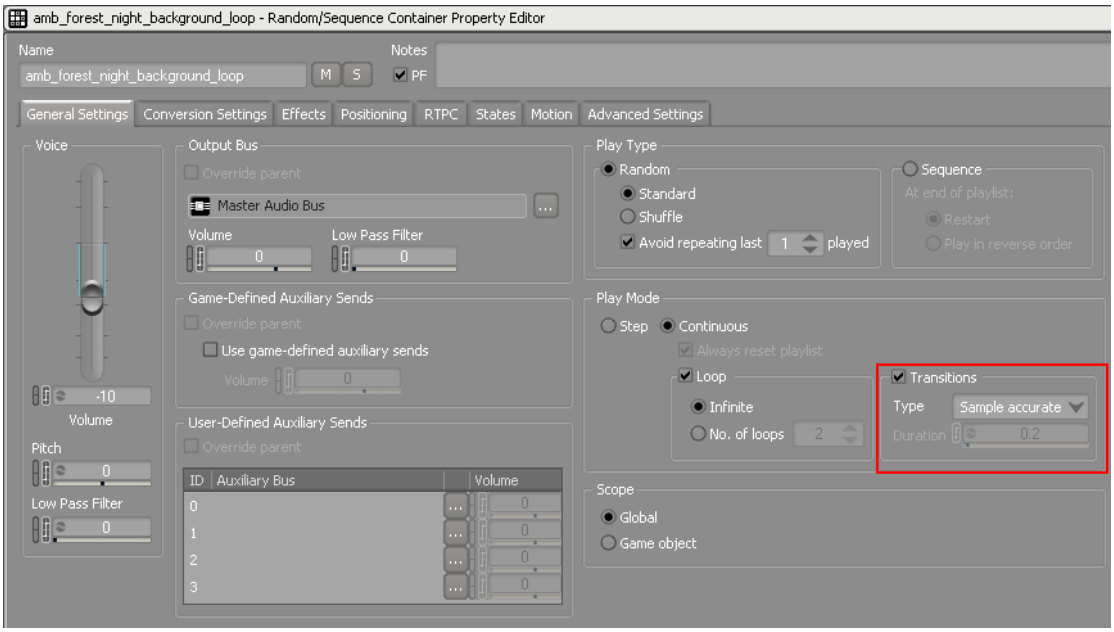
ランダムコンテナのPlay Typeの設定

このループ機能はランダムコンテナのプロパティでも設定できるので、コンテナ内でオーディオファイルのグループを同様にランダムにループさせることもできます。



**Play ModeをContinuous（継続）に設定し
てランダムコンテナのサウンドをループする**

これで複数ファイルをランダムにループさせ続ける設定が完了すると、次に、ループ再生中に1つのサウンドから別のサウンドにトランジション（移行）する時の詳細を設定します。ループのトランジション設定を有効にすれば、コントロールの範囲がさらに広がります。



ループするサウンドファイルの間のTransitionタイプを追加設定する

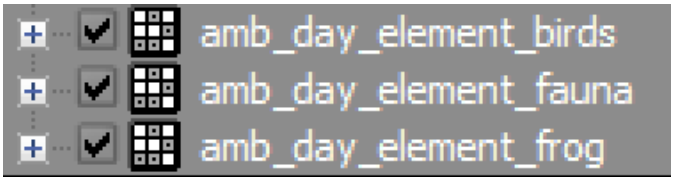
上記の例では、トランジションをSample accurate（サンプルアキュレート）に設定したことで、シームレスにループするアンビエントファイルが作られました。これ以外にも、トランジション設定の選択肢として、ディレイ（サイレンス）、一定パワーでのクロスフェード、一定振幅でのクロスフェード、トリガーレートがあります。

この手法の強みは、ランタイムで複数のサウンドファイルを組み直して実現するサウンドが、長さの決まった1つのファイルを使用した場合よりも、はるかに多様性に富んでいることです。さらに、サウンド間のトランジションを設定できるため、多様かつ繰り返しのないループで複雑に変化するサウンドが生まれます。これらのループ方式は、モノ、ステレオ、4チャンネル、サラウンドのいずれのアンビエントバックグラウンドにも対応しています。

以上でアンビエントバックグラウンドシステムの制作が開始されました。ループを自由にデザインしてクリエイティブに再生することにより、限られたリソースで豊かな環境を表現できます。

展開するサウンドスケープに詳細を追加

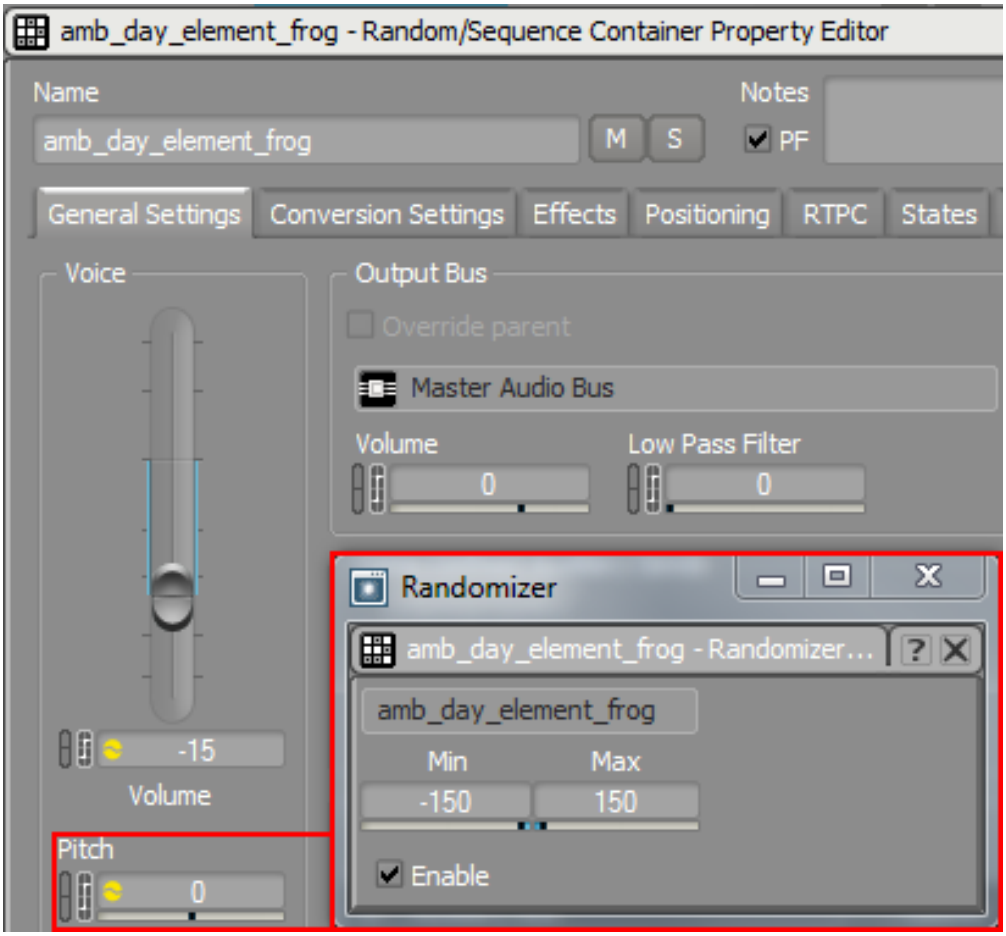
アンビエントバックグラウンドが完成した後は、1つのファイルや複数のファイルをループ再生するだけでは得られないダイナミック性と多様性を追加します。昼間の森の中では様々なサウンドが、一斉に聞こえてきます。そこで複数の鳥、コオロギ、フクロウ、オオカミの声が入った昼用アンビエントエレメントのコンテナをいくつか用意し、より複雑なアンビエントバックグラウンドを作り上げていきます。



昼間のアンビエントエレメントが入ったランダムコンテナ

プロパティ値のランダム化

Property Editor画面でボリューム、ピッチ、LPFなどを固定値で設定する以外に、ランダマイザー機能で値を変化させることができます。プロパティ値の範囲をランダマイザー機能で設定すれば、その範囲内でランダムに変化します。プロパティボックスのランダマイザーボタンをダブルクリックし、変化させる範囲の最低値と最高値を入力します。今回は、ピッチのランダマイザーを有効にし、ランダム範囲を-150から150セントまでとします。



ピッチのRandomizer（ランダマイザー）範囲を -150セントから +150セントまでに設定

ランダマイザーを有効化すると、アイコンがグレーから黄色に変わります。

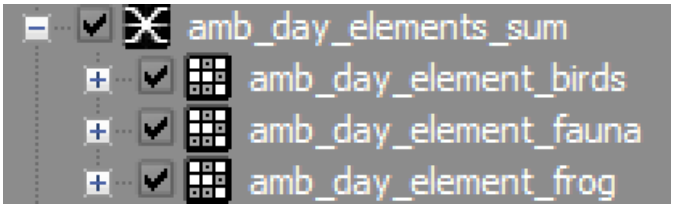
Designer Note

コンテンツのグループごとに、最適なランダム設定の範囲を研究することで、各グループのサウンドを最大限に活かすことができます。コオロギの場合は1つの鳴き声のピッチを大幅に変化させても不自然に聞こえないのに対して、動物の鳴き声を極端に変化させると、異常に聞こえてしまうかもしれません。

ブレンドコンテナを使ってサウンドを組み合わせる

個別のサウンドをランダムコンテナに追加し、ピッチ、ボリューム、LPFなどを変化させる作業が完了した後は、これらを組み合わせて動物のコーラスをつくり出します。それぞれにエレメントが入った複数のコンテナを再生する時、ブレンドコンテナを使うことができます。ブレンドコンテナにブレンドトラックがない場合、全ての子オブジェクトが同時に再生されます。この手法は、複数のサウンドをまとめて整理するのに便利です。

さらに各コンテナ内のコンテンツをループさせ、昼間のアンビエントバックグラウンドと共に、各エレメントのサウンドが時間をかけて再生されるようにします。

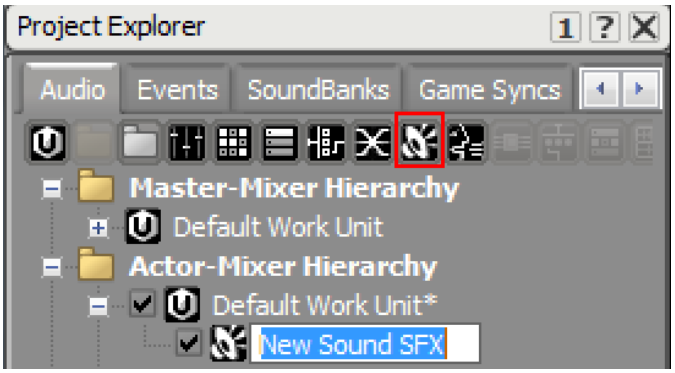


ブレンドコンテナにまとめられた、3つの“ambient daytime element”（昼間のアンビエントエレメント）

サイレンスのプラグインを使い、ランダム性にウェイト付けする

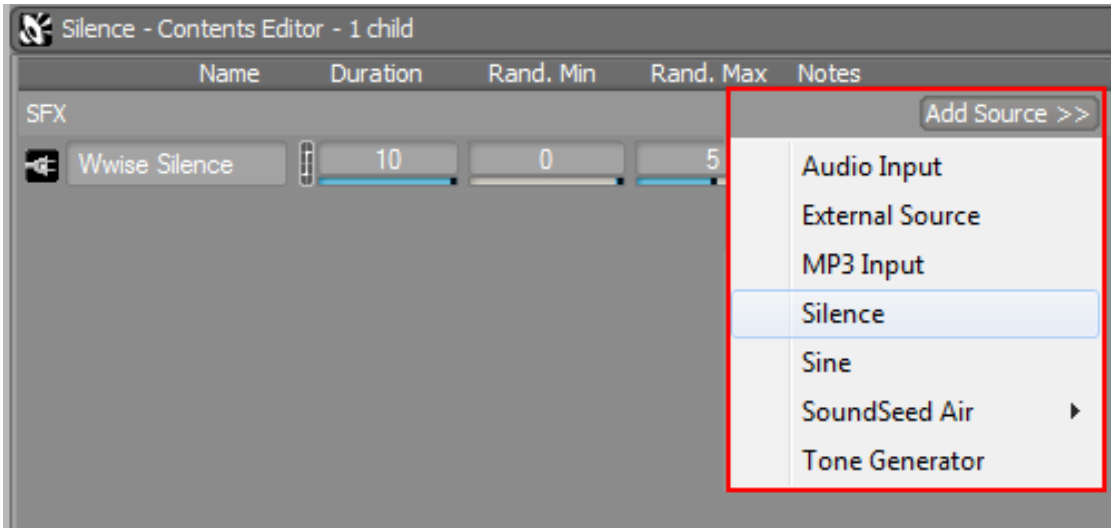
これらのサウンドが全て、絶え間なくループした状態では圧迫感があります。そこで、サウンドファイルの間に無音の時間を導入しサウンドの再生頻度を減らせば、昼間の森のサウンドスケープがバランス良く聞こえます。サウンドSFXオブジェクトにWwiseのサイレンスプラグインを追加することで、サウンドファイルのランダム性にサイレンスが挿入されます。

Project Explorerツールバーのサウンド SFX アイコンをクリックし、Default Work Unitに新規サウンドSFXオブジェクトを追加すると、新しいサウンドSFXが作られます。また、コンテキストメニューやショートカットキーからサウンドSFXを作ることもできます。



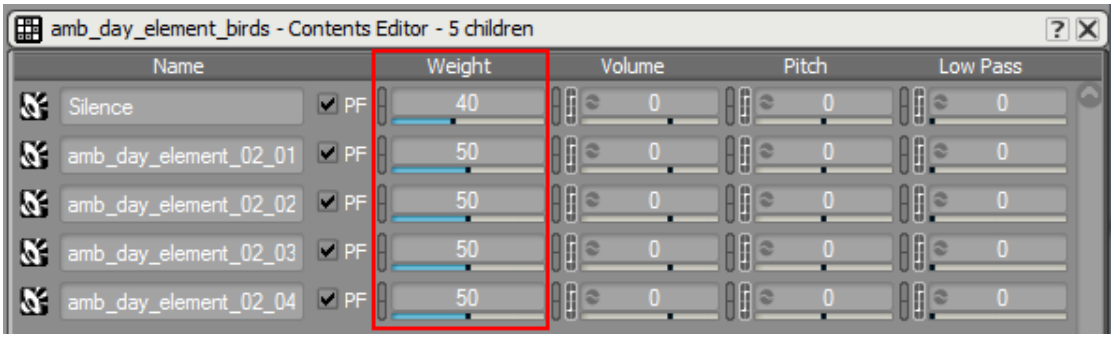
Project ExplorerツールバーからNew Sound SFX（新規サウンドSFX）を作成

新しいサウンドSFXをダブルクリックし、Contents Editor画面にあるAdd Sourceメニューから Silence（無音）を追加します。Wwise のサイレンスプラグインは、静的なDuration（長さ）と、長さをランダムに変えていくためのRand. Min（ランダム時の最短の長さ）とRand. Max（ランダム時の最長の長さ）を設定できます。Rand. Min 値をDuration値から引いた数値が、最短の長さとなります。



サウンドオブジェクトのSilenceのソースプラグインを作成

サウンドSFXのウェイトを設定すれば、ランダムコンテナ内で各オーディオファイルが再生される確率を調整できます。サイレンスオブジェクトのウェイト付けが適切であれば、各オーディオファイルを繰り返す頻度を、不自然のない程度に維持できます。例えば、鳥がひっきりなしにさえずる状況を避け、次の鳴き声が聞こえるまでの沈黙の時間をランダムに変化させます。



Weight（ウェイト）付けでサウンドの再生の確率を調整

Designer Note

ウェイト付けの合計を100%にする必要はなく、入力された値に従い、Wwiseが自動的に計算します。

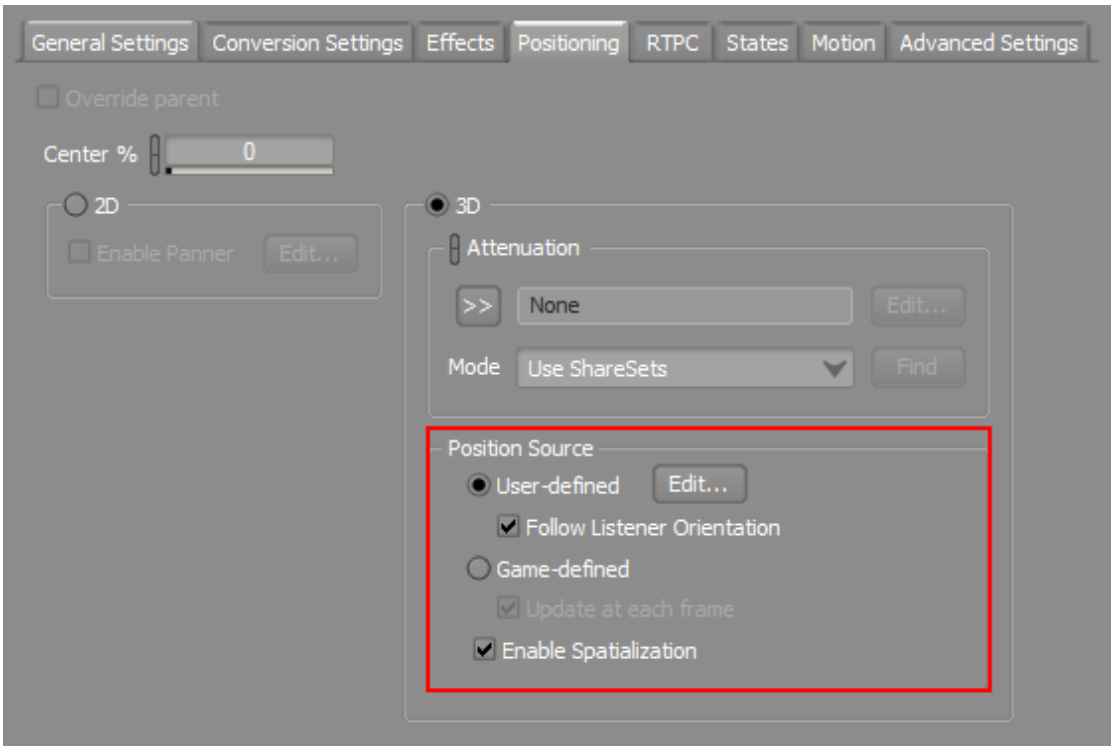
サウンドの位置のランダム化

ランダムに再生される数種類の昼の自然界のサウンドがまとめられたブレンドコンテナができました。次に、ユーザーが定義する3Dポジショニング機能を使い、生き生きとしたサウンドスケープを作り出します。

Designer Note

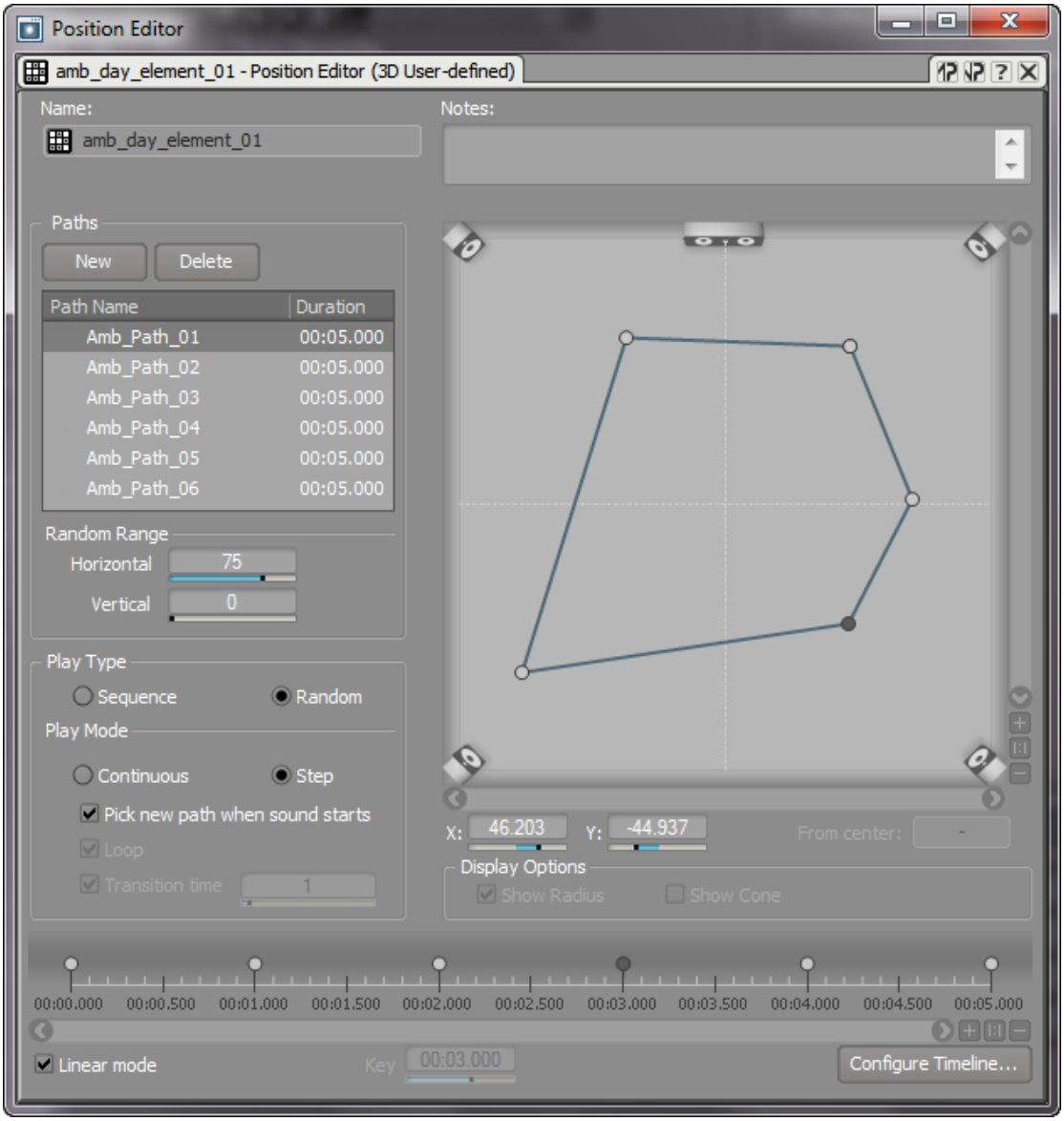
Position Editor (3D User-defined) 画面ではアニメーションパスを使って、サラウンド環境におけるオブジェクトの空間位置を定義します。1つのパスに対して複数のバージョンを作り、これらのパスに沿ってサウンドやミュージックをアニメーション化できる機能です。それぞれのパスの再生方法も選べます。ここで作成したパスは、ゲーム内のポジションやオリエンテーションよりも優先されます。

サウンドが入ったランダムコンテナを1つ選び、Property Editor画面のPositioningタブで設定を変更します。Position Source（ポジション源）の設定をデフォルトのGame-defined（ゲームが決定）からUser-defined（ユーザー決定）へ変えると、Position Editor機能が使用可能になり、サウンドの再生位置をユーザー定義で設定できます。この手法は、静的なサウンドの位置を動かしたり、準備した環境サウンドの位置をランダムに変えたりするために使います。



アンビエントエレメントのポジションをUser-defined（ユーザー定義）に設定

複数のパスを作り、それらのパスを順番に再生したり、ランダムに再生すれば、複数のサウンドがそれぞれ別方向から聞こえてくるエフェクトをもたらします。



Position Editor画面で、スピーカーの配置に従ってサウンドの再生位置を定義

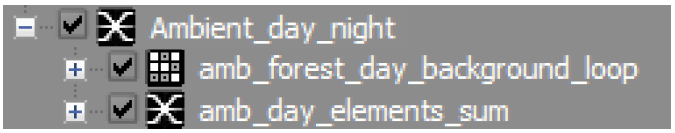
Designer Note

Position Editor 機能の詳細は、チュートリアルビデオ： Video Tutorial - Positioning and Distance Attenuation ドキュメンテーション： Wwise Help > Wwise Reference > Positioning > **Position Editor (3D User-defined)** を参照してください。

付属のプロジェクトに入っている昼間と夜間のアンビエントエレメントは、ここで説明したユーザー定義の3Dポジショニング機能を使って作られていますので、本機能をよりよく理解するためにご参照下さい。

昼間の森林アンビエントを組み合わせる

ループするアンビエントバックグラウンドが入ったランダムコンテナと、複数の昼間のエレメントが入ったブレンドコンテナができたら、両者を1つのブレンドコンテナにまとめます。



ブレンドコンテナにまとめた、アンビエントバックグラウンドのループと昼間のエレメント

これで、ランダム化された昼間のエレメントと、ループするアンビエントバックグラウンドが同時に再生され、毎回異なるサウンドスケープが実現します。もちろん、サウンドファイル自体のバリエーションを増やしたり、プロパティをランダム化する余地もあります。コンテンツと実装方法のバランスをとることが、リソース消費を最適化してプレイヤーの試聴負担を軽減することにつながります。2つの実装方法の性質を上手く組み合わせることで、ゲームの規模に合った繰り返しのない再生システムが確立されるでしょう。

本節のまとめ

本節で、以下のサウンドオブジェクトを作成しました。

- 昼間の森を表現する複数のアンビエントバックグラウンドファイルが、トランジションを経由してループするランダムコンテナ
- ウェイト付けされたサイレンスとランダムな3Dポジションを適用した昼間のアンビエントエレメントのブレンドコンテナ

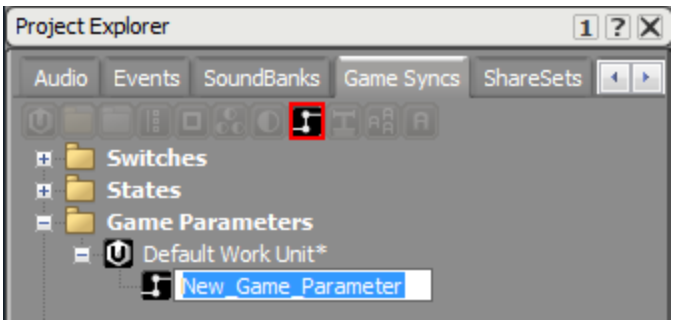
昼と夜のサイクルの始動

一日中、森の中を旅した後に日は沈み、不吉な雰囲気が漂い始めます。木々の影が伸びるに従い、のどかな野道も暗く恐ろしい細道に変わります。時間の経過を効果的に表現するためには、ビジュアル環境をサウンドに正確に反映し、同等の憂鬱感と不安をかもし出すべきです。

ゲームパラメータの設定

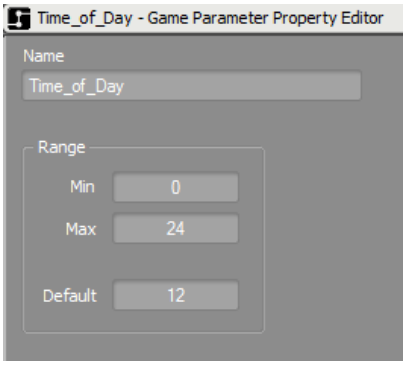
最初に、ゲーム側が時間の経過をビジュアルに表現するために使っている情報を、ゲームパラメータとしてWwiseにも送出できるように、プログラマーに依頼します。アンビエントバックグラウンドや個別のエレメントの入ったブレンドコンテナにこの情報を利用できれば、アンビエントサウンドスケープをゲームの他の動きに合わせることができます。

まず、“Time_of_Day”（時間）という新規ゲームパラメータを作成します。そして、Project Explorer画面のGame Syncsタブを開き、Default Work Unitを選択します。すると、Project Explorerツールバーのゲームパラメータアイコンをクリックして、新規ゲームパラメータを追加できます。また、コンテキストメニューにあるゲームパラメータのワークユニットを選択するかショートカットキーを使って、ゲームパラメータを作ることもできます。



Project Explorerツールバーからゲームパラメータを作成

ゲームパラメータの名前を入力してから、パラメータ範囲をProperty Editor画面で設定します。ここではゲームパラメータ名を“Time_of_Day”とし、0をAM12:00とする 0-24の値を使うことで、サウンドのパラメータ範囲を、0-24時に合わせます。

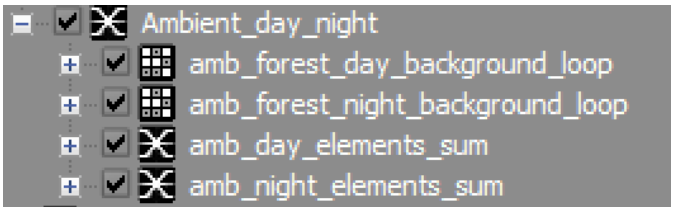


ゲームパラメータ値“Time_of_Day”の範囲設定

アンビエントシステムの作成

一日の時間の経過を表現するためのゲームパラメータを準備した後は、ループするバックグラウンドと、配置がランダム化されるアンビエントサウンドを同時に再生するブレンドコンテナを導入します。全ての設定が完了すると、ダイナミックに変化する昼と夜のサイクルが森のアンビエントとして成立し、この変化をオーサリングアプリケーション上でゲームパラメータを使ってコントロールしたり、試聴することができます。

前節で説明したテクニックを使い、昼間のサウンドに盛り込んだ機能を夜間のアンビエントにも取り込めます。今回は昼用と夜用のコンテナをそれぞれ別の親ブレンドコンテナに入れるのではなく、全てを“Ambient_day_night”という1つのブレンドコンテナにまとめて、ここで森のアンビエントシステム全体を管理することにします。

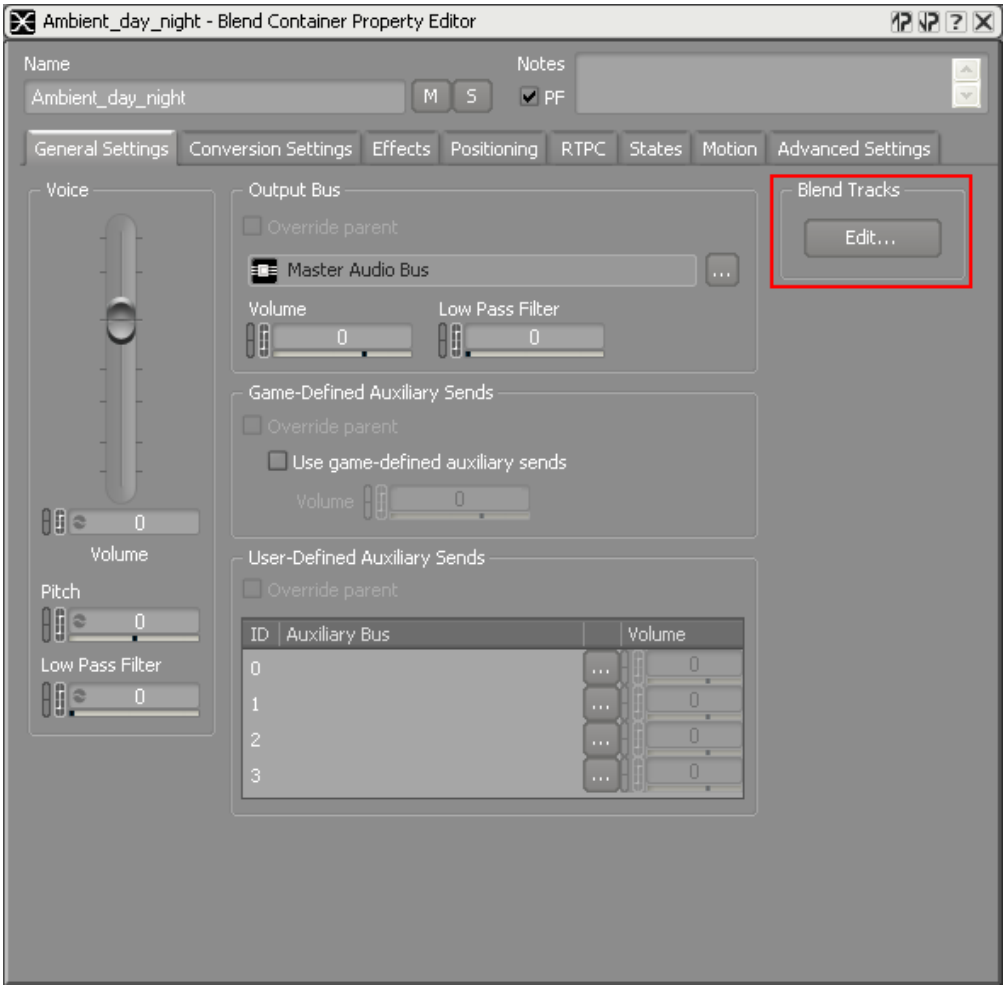


**昼夜の全てのコンテナを1つにまとめた、
ブレンドコンテナ“Ambient_day_night”**

ブレンドトラックエディタの使用

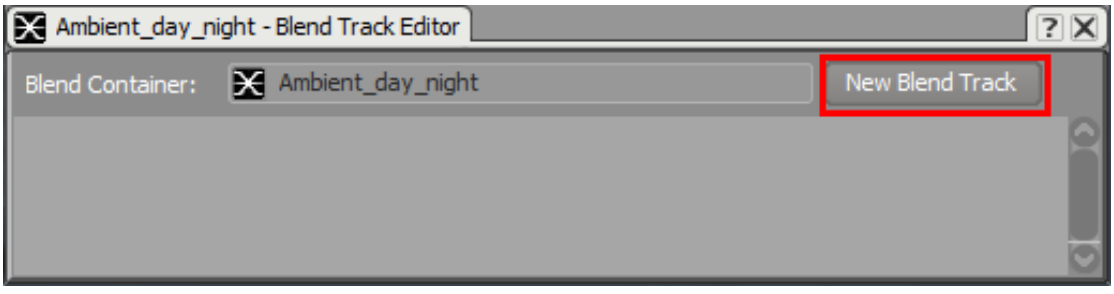
全てをブレンドコンテナに入れて、同時に再生される状態になったので、次に、昼間のエレメントと夜間のエレメントを分け、ゲームパラメータTime_of_Dayで統合する方法が必要です。ブレンドコンテナ独自の機能であるBlend Track Editorを使って、コンテナの中身を調整してプロパティをRTPCカーブやクロスフェード機能でコントロールします。

ブレンドトラック機能は、Blend Container Property Editor画面のGeneral Settingsタブに表示されます。



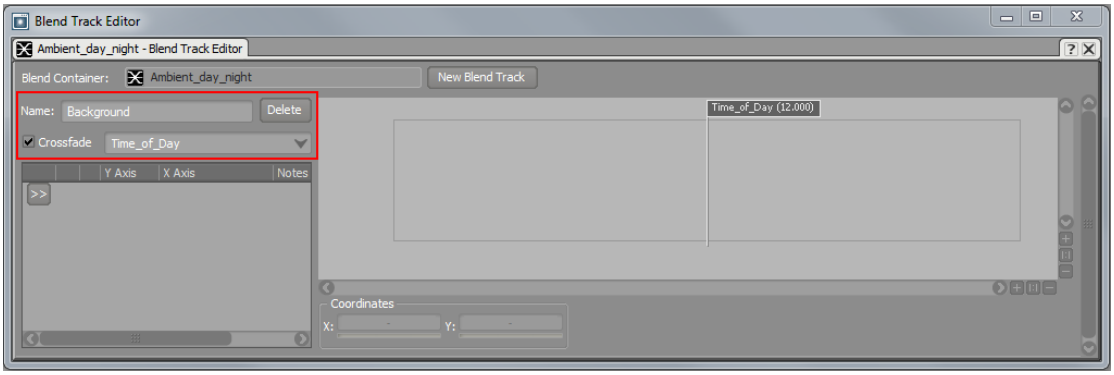
Property Editor画面からBlend Track機能を選択

Blend Track Editor画面は主に2つに分かれ、ブロックをクロスフェードさせるエリアと、グラフ上でRTPCカーブを追加したり変更するエリアがあります。これらのエリアで作業するには、Blend Track Editor画面で新規ブレンドトラックを追加します。



Blend Track Editor画面で、空のブレンドコンテナに New Blend Track（新規ブレンドトラック）を追加

追加した新規ブレンドトラックに、アンビエントバックグラウンドにちなんだ名前（ここではBackground）を付けます。オプションでクロスフェード機能を使えば、ブレンドトラックにあるコンテナ間や他のサウンドオブジェクト間を移行する時のゲームパラメータを選択できます。



新規ブレンドトラックのCrossfade（クロスフェード）機能を適用

ブレンドトラックBackgroundに対して、ゲームパラメータ“Time_of_Day”を使ったクロスフェードを適用すれば、ゲームパラメータの値に従ってコンテナを配置できます。ブレンドトラック上のコンテナの配置によって、オーバーラップやクロスフェードの範囲が決まります。

Designer Note

ゲームパラメータが変異の幅が異なる別のゲームパラメータに変わる時、ブレンドトラックのオーディオブロックのレンジに影響することがあります。そのような場合、オーディオブロックをStretch（ストレッチ）するのか、ゲームパラメータ変更の後もポジションをPreserve（維持）するのかを決めなければなりません。

1つのブレンドトラックが追加され、クロスフェード機能も設定できるようになったので、次にContents Editor画面の中でブレンドトラックに対しコンテナを割り当てます。

Contents Editor画面のBlend Tracksエリア

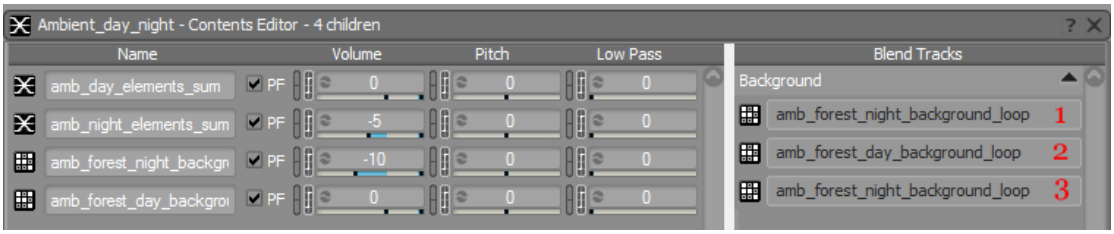
ブレンドトラック内でコンテナの順番を決める

これで、ブレンドコンテナにあるコンテナを、Blend Tracksエリアにドラッグ＆ドロップできます。Contents Editor画面でブレンドトラックを設定する時は、**表示させたい順番にコンテナを並べること**が重要で、Blend Track Editor画面では左から右に表示されます。

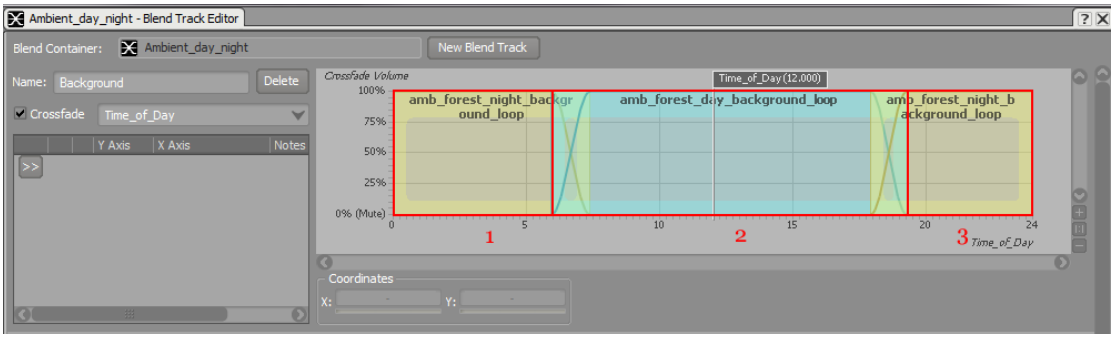
順番と並べ方の関係を下図に示します。

Blend Track: Background

1. **Random container:** amb_forest_night_background_loop
2. **Random container:** amb_forest_day_background_loop
3. **Random container:** amb_forest_night_background_loop



Contents Editor画面のBlend Trackエリアにある、ランダムコンテナの順番

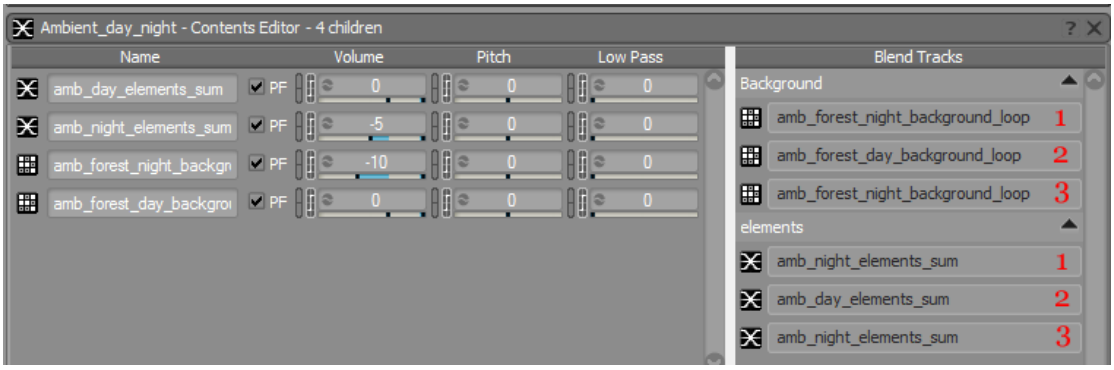


順番に並んだコンテナの入ったブレンドトラックの追加

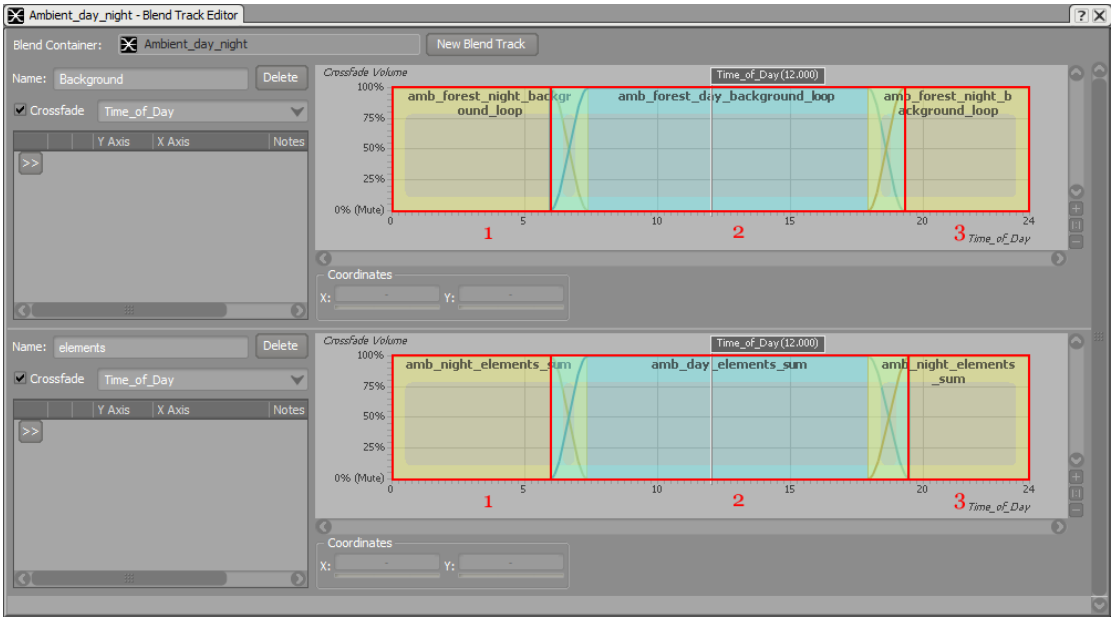
アンビエントシステムを完成させるには、昼間と夜間のアンビエントエレメントも専用のブレンドトラックに追加します。

Blend Track: Elements

1. **Random container:** amb_night_elements_sum
2. **Random container:** amb_day_elements_sum
3. **Random container:** amb_night_elements_sum

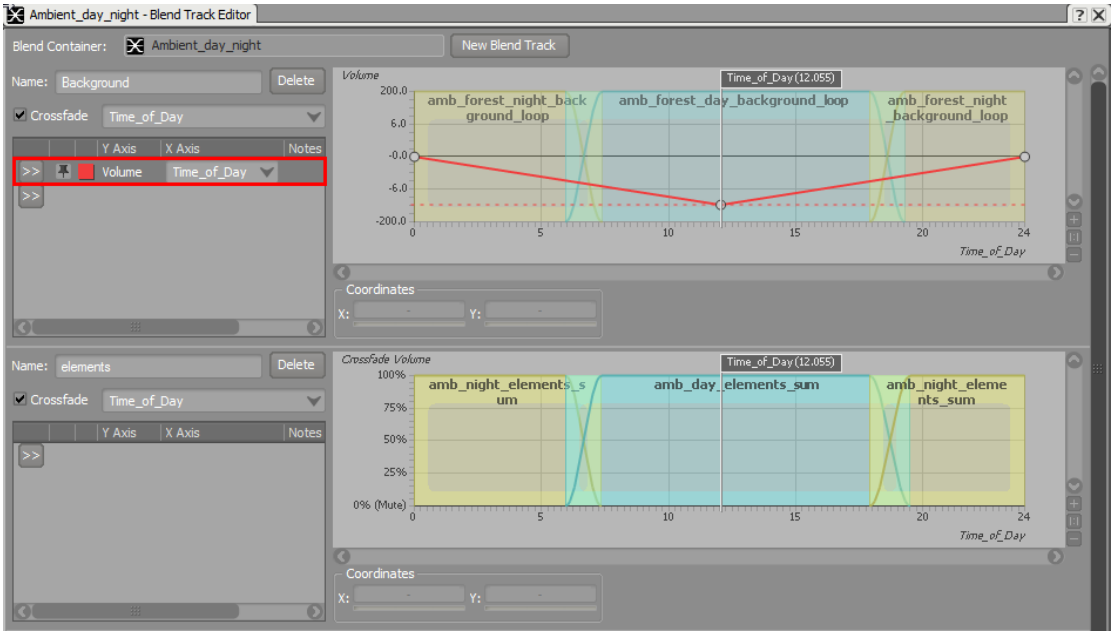


Contents Editor画面のBlend Trackエリアにある、コンテナの順番



順番に並んだコンテナの入ったブレンドトラックの追加

ゲームパラメータ“Time_of_Day”のカーソルを使い、バックグラウンドと環境エレメントの両方に関して、コンテナ間に設定したクロスフェードを試聴できるようになりました。グラフのカーブのポイントやクロスフェードの種類は、再生中にリアルタイムで編集できます。さらに、ゲームパラメータとブレンドトラックの結び付きでボリュームやピッチ、ローパスフィルタを変化させることもできます。



ボリュームとゲームパラメータを結び付け、ブレンドトラックを変化させる例

イベントの準備

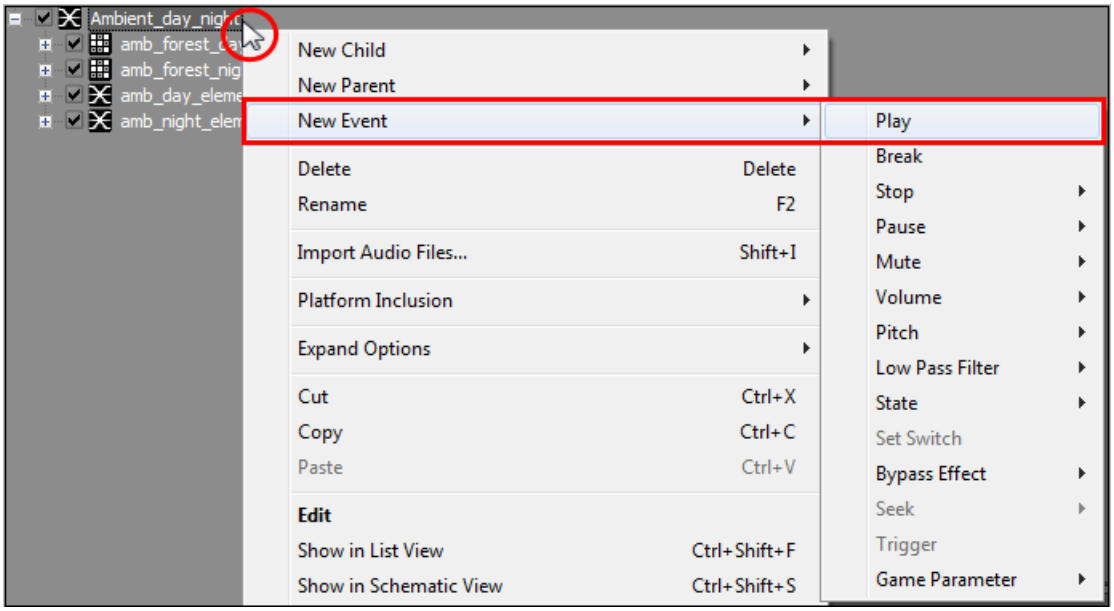
ダイナミックなアンビエントシステムができたので、次にゲームエンジンによってループの再生や停止ができるようにします。Wwiseはイベントを使ってサウンドやミュージック、ダイアログの指示をゲームに出します。ゲームのある時点で再生するサウンドやミュージック、モーションやダイアログなどはイベントによって決まり、様々なアクションを通して他の面をコントロールすることもできます（再生、一時停止、停止などや、ボリューム、ピッチ、ローパスフィルタの変更など）。

イベントを利用して、プロジェクト階層にある様々な構成に対してアクションを適用することもできます。イベントにはアクションが1つ、または一連のアクションが複数、入っています。指定したアクションによって、Wwiseオブジェクトの再生、一時停止、停止などが決まります。

それでは、今回のゲームのヒーローが森に入り、そして森を出る時に、ゲームエンジンがアンビエントシステムを再生し、そして停止するように、以下の2つのイベントを作ります。

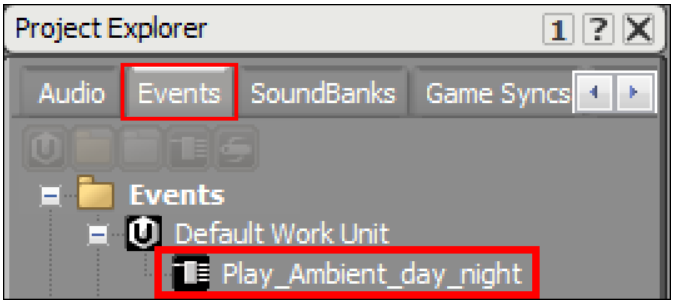
- Play_Ambient_Forest
- Stop_Ambient_Forest

新規の再生イベントを作るには、ブレンドコンテナ“Ambient_day_night”のコンテキストメニューから New Event > Playを選択します。



コンテキストメニューからNew Eventを作成

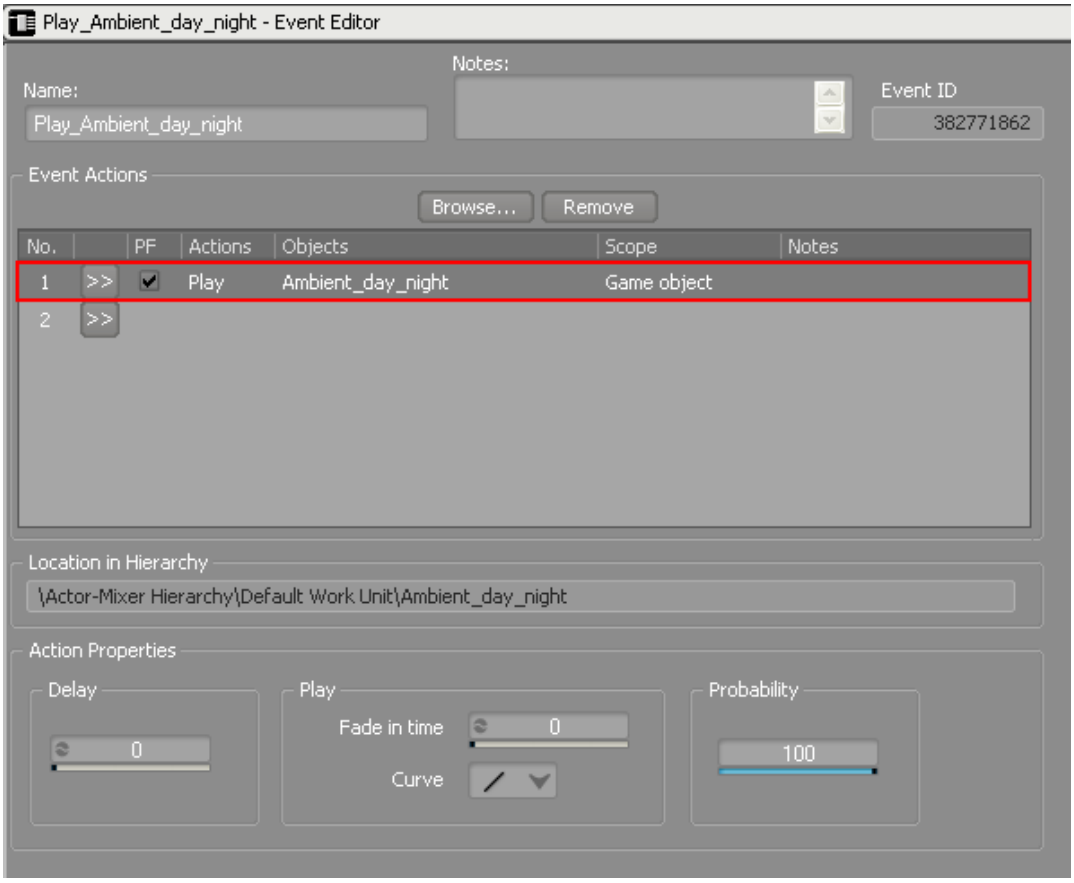
これにより、コンテナにちなんだ名前の新規のPlayイベントが、Project Explorer画面のEventsの下に、Default Work Unitの一部として作られます。



Project Explorer画面のEventsタブに表示された新規イベント

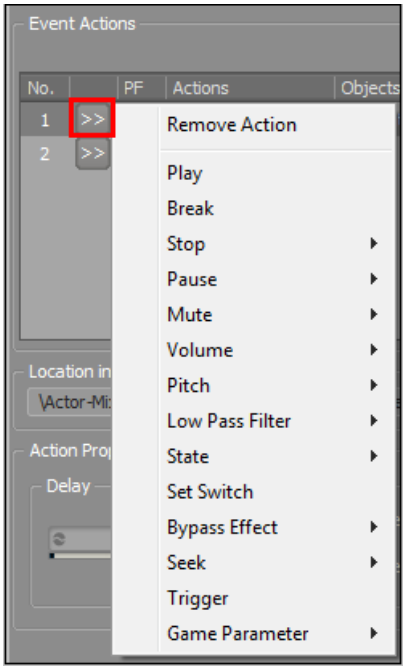
イベントアクションを完成させる

イベントが作成されると、Event Editor画面が表示され、新規イベントが反映されます。新規イベントにはブレンドコンテナ“Ambient_day_night”が追加され、アクションとしてPlayが設定されています。



アクションPlayをブレンドコンテナ“Ambient_day_night”と結び付けた新規イベント

セレクタを使って、イベントアクションのRemove（削除）や追加も可能です。



セレクタでイベントアクションの他のオプションを選択

Event Editor画面には、各イベントアクションのオブジェクトのLocation（場所）や、そのイベントアクションのAction Propertiesも表示されます。

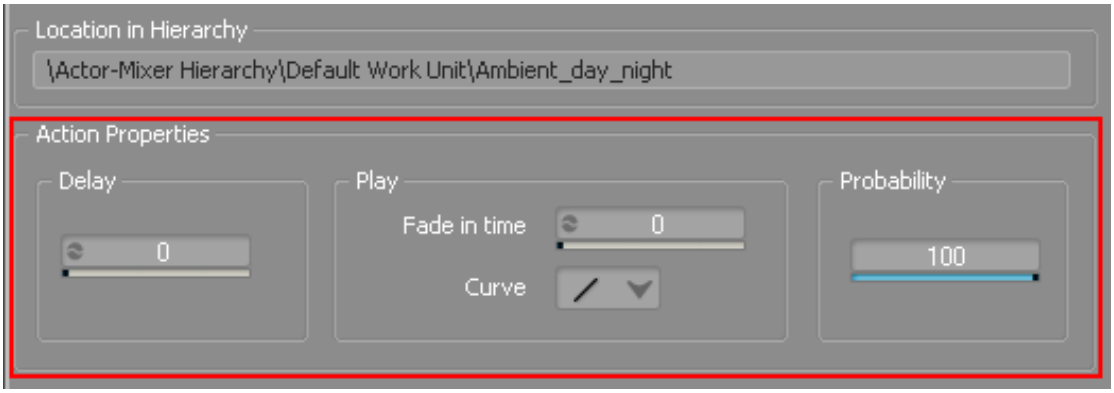
Designer Note

イベントアクションによって設定できるプロパティが異なり、これらのプロパティを調整して希望するエフェクトを作ります。全てのイベントアクションの種類やプロパティを確認するには、以下のHelpドキュメンテーションをご参照下さい。

Wwise Help > Wwise Reference > Events > **Event Editor**

Wwise Help > Where to Begin? > Wwise Fundamentals > Understanding Events > **Action Events**

Wwise Help > Interacting with the Game > Managing Events > Overview > **Types of Event Actions**



イベントアクションPlayで設定できるプロパティ項目

イベントアクションPlayは、以下のプロパティを変更できます。

- **Delay**（ディレイ）- アクションが実行されるまでに経過する時間。
  - デフォルト値「0」、デフォルトのスライダ範囲「0 - 10」、入力範囲「0 - 600」、単位「秒」
- **Fade in time** （フェードイン時間）- オブジェクトのフェードインにかかる時間。
  - デフォルト値「0」、デフォルトのスライダ範囲「0 - 10」、入力範囲「0 - 60」、単位「秒」
  - Delayが設定してある場合は、それが終了してからフェードインを開始する。
- **Curve** （カーブ）- オブジェクトのフェードインの様子をカーブの形状で定義したもの。
- **Probability**（確立）- イベントアクションが再生される確率。
  - デフォルト値「100」、デフォルトのスライダ範囲「0 - 100」、入力範囲「0 - 100」、単位「%」

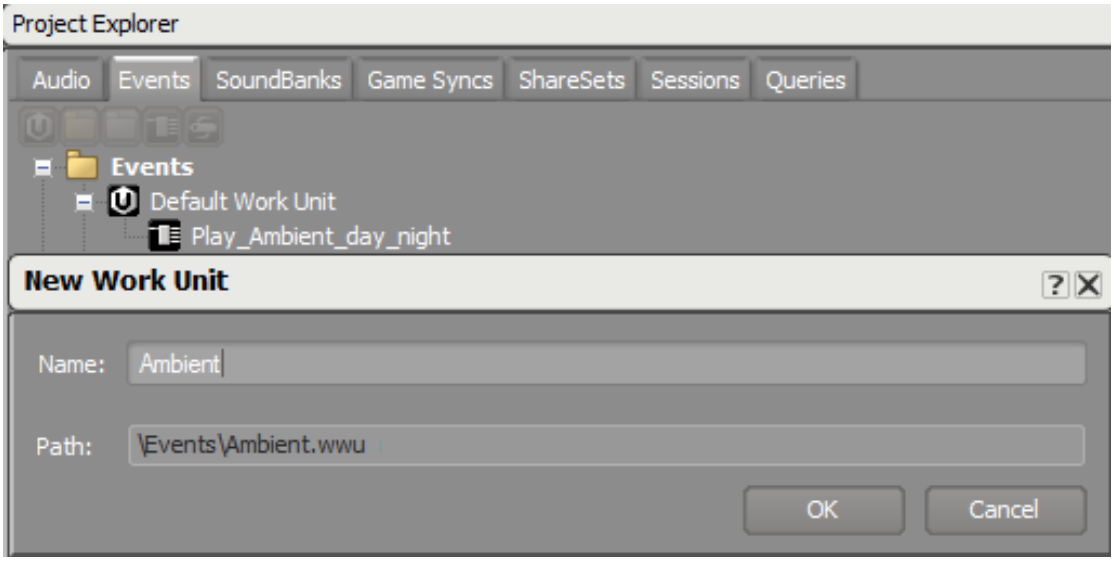
ワークユニットの作成

次に、ブレンドコンテナ“Ambient_day_night” のStopイベントを、別のアプローチで作成します。

最初に、Project Explorer画面 のEventsタブで“Ambient”という新しいワークユニットを作り、プロジェクトの様々な構成要素を抜き出します。

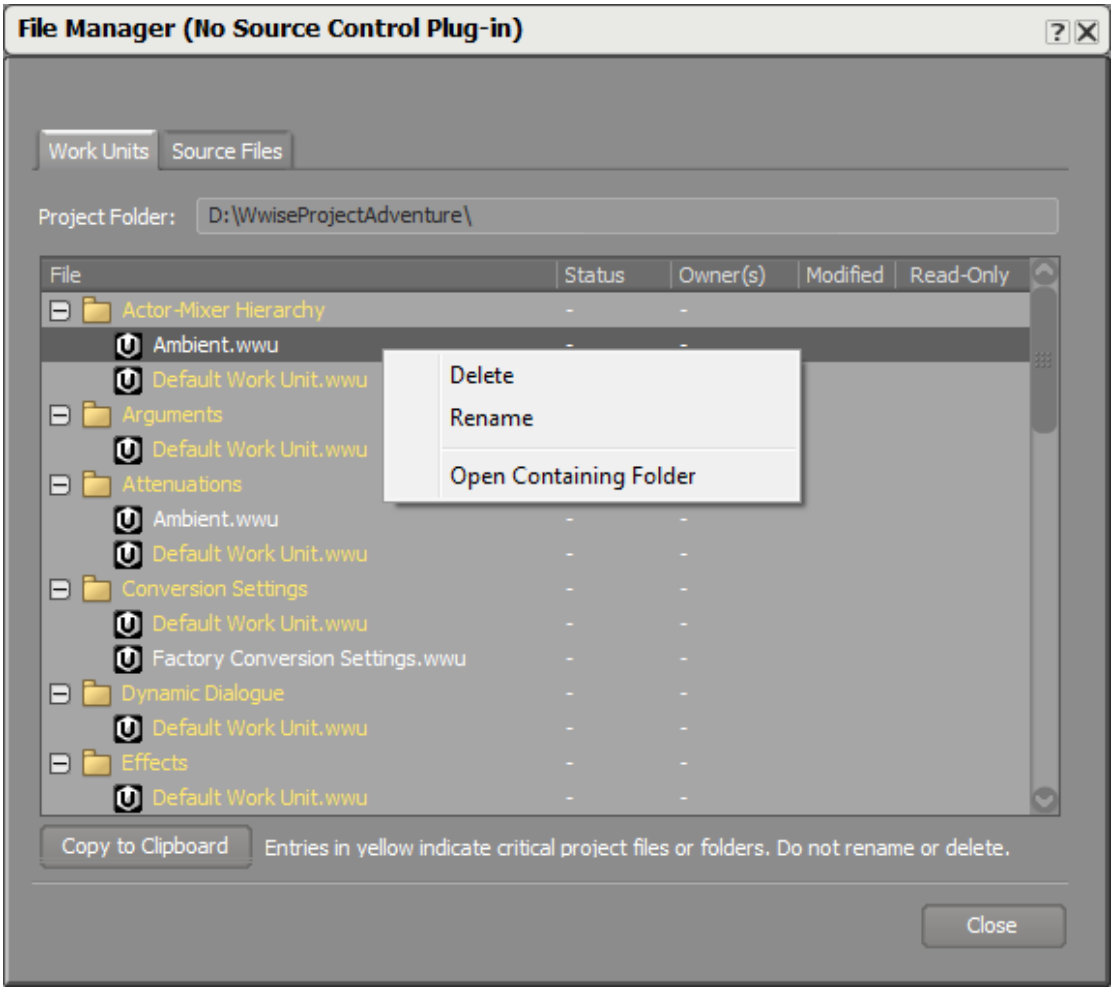
Wwiseのワークグループの基盤となっているのが、ワークユニットです。ワークユニットは個別のXMLファイルで、プロジェクト内の特定の部分やエレメントに関する情報が入っています。プロジェクトに存在する様々な要素を整理し管理するために使います。ワークユニットをネスト化させた階層を作り、物理的なフォルダやサブフォルダに入れて整理します。

Project Explorerツールバーのワークユニットアイコンをクリックし、New Work Unitを作成します。また、イベントのヘッダータイトルのコンテキストメニューから、ワークユニットを作ることもできます。



Work Unitの作成とName入力

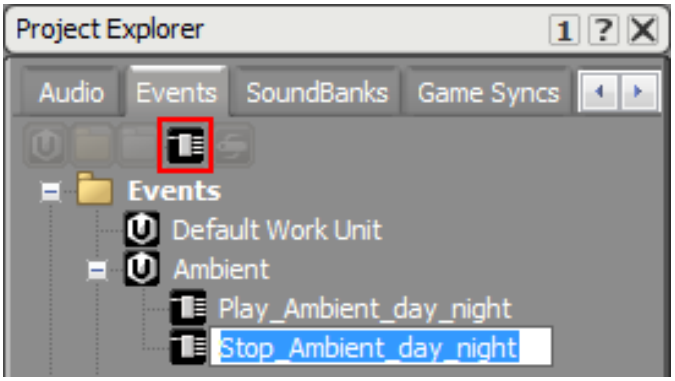
作成したワークユニットは、Project Explorer画面で削除や移動、名前変更ができます。また、File Manager画面からワークユニットやソースファイルを変更することもできます。File Manager画面はProjectメニューから選択できます。



File Manager画面で、ワークユニットやソースファイルのDelete、Rename（名前変更）、移動などが可能

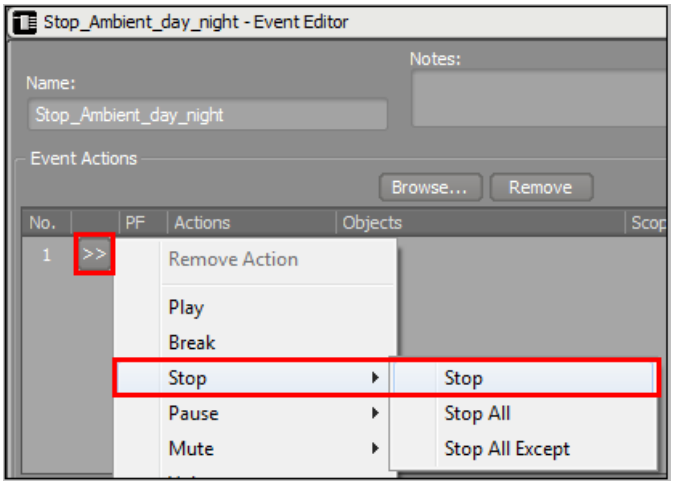
作成したイベントワークユニット“Ambient”に、既に作成してあるイベント“Play_Ambient_day_night” をDefault Work Unitからドラッグ＆ドロップで入れます。

次に、Project Explorerツールバーのイベントアイコンをクリックし、新規イベント“Stop_Ambient_day_night”を追加します。イベントは、コンテキストメニューやショートカットキーからも作成できます。



ワークユニット“Ambient”の中に、新規イベントを作成

新しいイベント“Stop_Ambient_day_night”をダブルクリックすると、Event Editor画面が開きます。次に、イベントアクション用セレクタでイベントアクションの一覧を開き、Stop>Stopを選択してアクションStopを追加します。

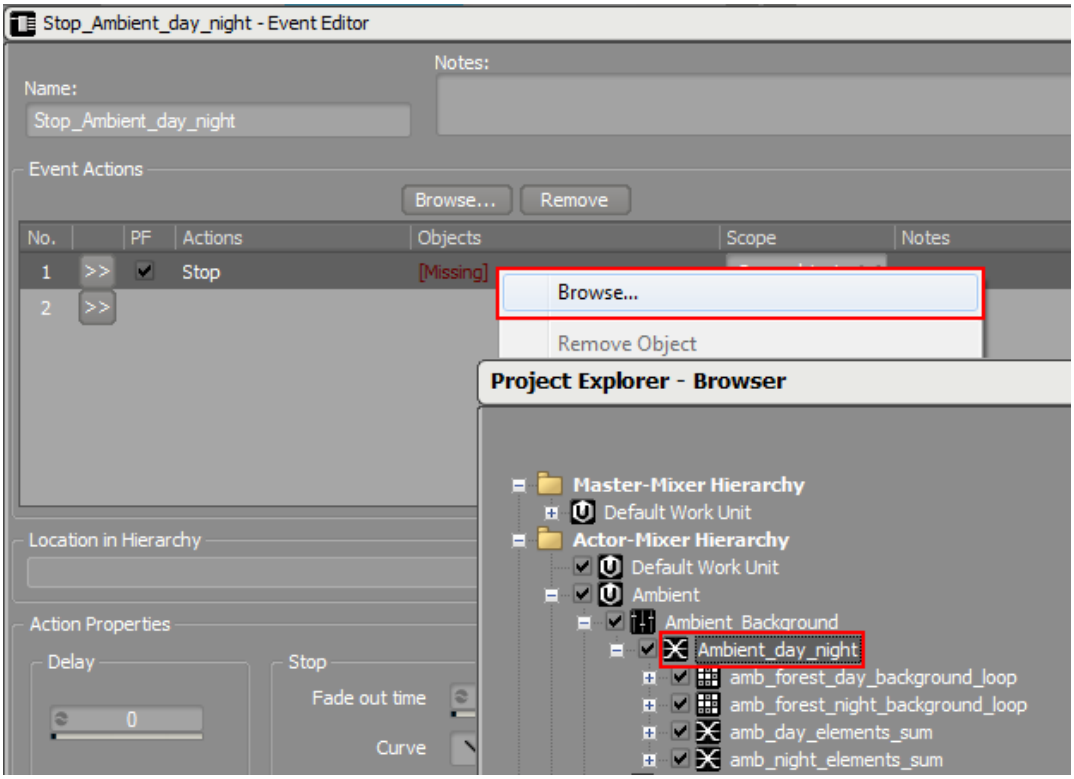


新規イベントにイベントアクションStopを追加

イベントにイベントアクションを追加した後、このアクションの対象となるオブジェクトを指定しますが、対象オブジェクトがない場合、Objects欄にはMissing（欠如）と表示されます。ゲームがイベントを要求した時点で、指定されたアクションを以下の4種類のオブジェクトのいずれかに適用させる必要があります。

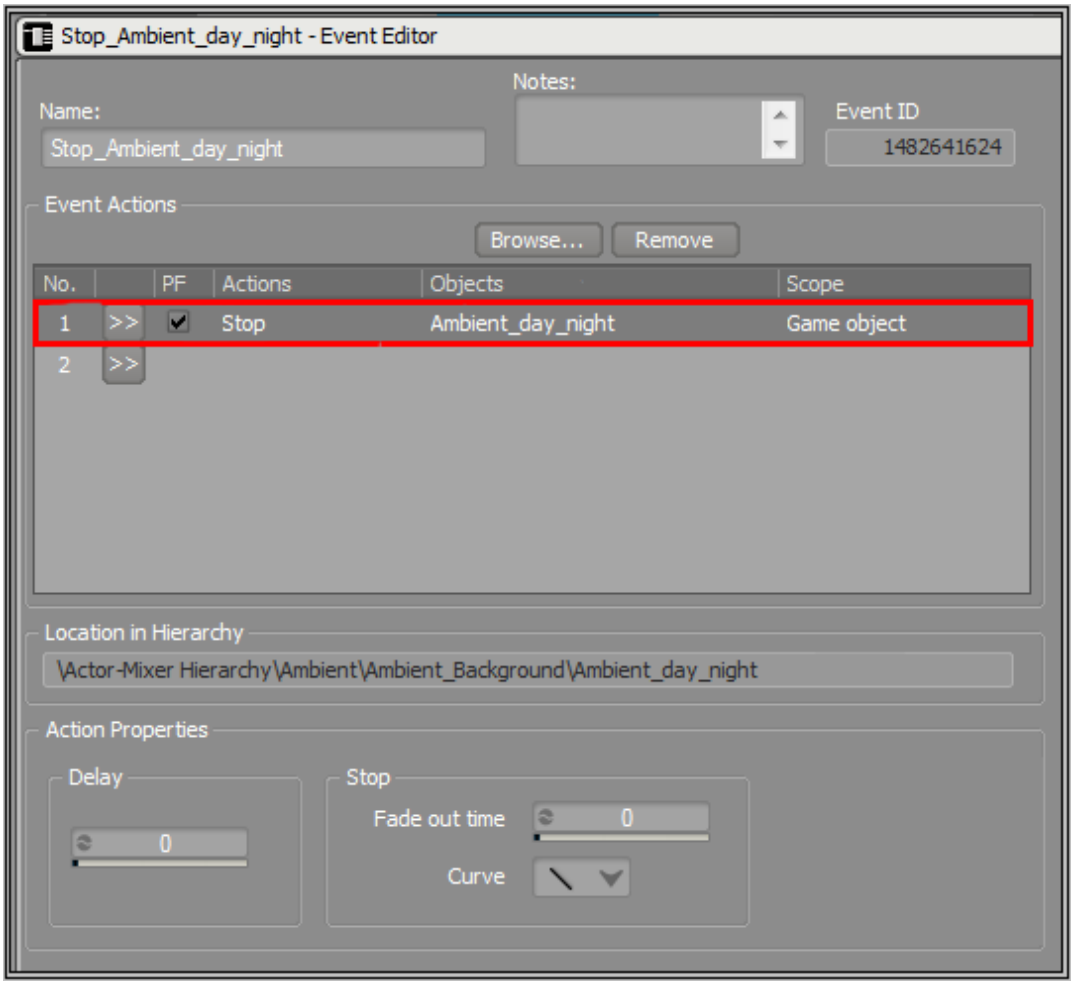
- サウンドオブジェクト
- モーションFXオブジェクト
- アクターミキサー
- コンテナ

これらのオブジェクトは、Project Explorer - Browser画面で選択するか、Project Explorer画面からEvent Editor画面のイベントアクションのObjects欄までドラッグ＆ドロップすることもできます。



Project Explorer - Browser画面でブレンドコンテナ“Ambient_day_night”を選択

オブジェクトを追加すると、階層内のオブジェクトの位置が更新され、アクションのプロパティが編集できるようになります。



アクションStopとブレンドコンテナ“Ambient_day_night”を結び付けて作成したイベント

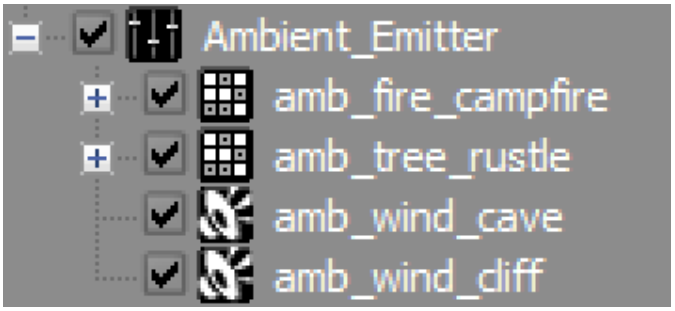
作成したイベントをゲームエンジンに実装すれば、ゲーム中に適宜、呼び出すことができます。イベントは抽象的なので、ゲームエンジンに実装した後も、イベントの構成要素をオーサリングアプリケーション上で引き続き調整することができ、ゲームエンジンに実装し直す必要はありません。

本節のまとめ

ブレンドトラックを導入すれば、ブレンドコンテナ内でオブジェクトの順番を決めクロスフェードさせることができ、さらにゲームパラメータを使ってダイナミックに変えることも可能です。またイベントやイベントアクションを使い、ゲーム中の特定のタイミングでどのサウンドやミュージック、モーションやダイアログを、どのように再生するのかを決められます。一日の長旅が終わり日が沈む頃、小鳥のさえずりがコオロギの鳴き声やオオカミの遠吠えに変わる中、ダイナミックに変化するアンビエンスの効果が実感できるでしょう。

ゲームワールドの中にあるサウンドエミッタ

バックグラウンドに、一日を通して変化するループ再生を設定できたところで、例えば川、滝、たき火、枝にとまった小鳥など、森の中の特定の場所に配置されたサウンドを追加して環境を埋めていきます。最初に、ランダムコンテナをいくつか設定して、ワールドに特徴を与えていきます。



3D Emitterとして使うサウンド

減衰シェアセットの作成

ここまで実施してきたアンビエントバックグラウンドの設定と比較して、あえて3D空間に配置したサウンドの最大の違いは、リスナーの位置によってサウンドが変化することです。リスナーとはゲーム内にある仮想マイクのことで、スピーカーにサウンドを割り当てるために使い、これによって3D環境をシミュレーションします。Attenuation（減衰）のシェアセットをオーサリングすれば、リスナーとの位置関係を使って、距離に基づいたサウンド再生コントロールが可能です。

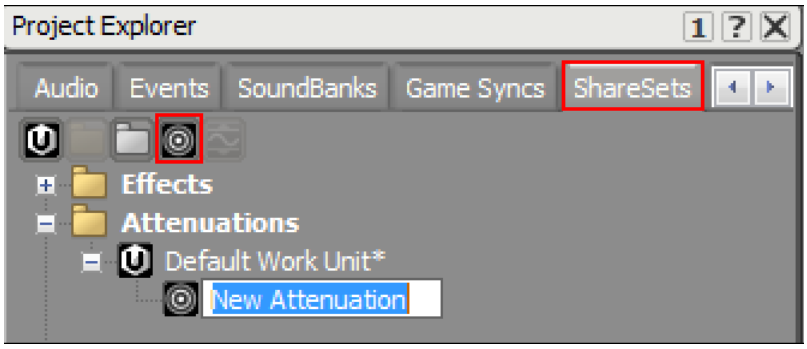
Designer Note

Attenuation Editor 画面で、距離に基づいたオブジェクトの減衰プロパティを定義できます。ボリュームやローパスフィルタなどのWwiseプロパティに対して、音を発信しているエミッタソースからリスナーまでの距離がどう影響するのかを、グラフ上のカーブを使って定義すれば、ゲーム内のサウンドやミュージック、モーションが距離によって減衰する状況を精巧にシミュレーションできます。リスナーに対するゲームオブジェクトの向きに基づいて減衰をシミュレーションするためのサウンドコーンを使えば、減衰の質がさらに向上します。

サウンドエンジンでトリガーするイベントは必ずゲームオブジェクトと結び付いているため、ゲームオブジェクトはWwiseの中心的な概念といえます。一般的にゲームオブジェクトは、サウンドを発信するゲーム内のオブジェクトやエレメントを表し、キャラクターやウェポン、たいまつなどのアンビエントオブジェクトなどがその例です。

一般的な減衰設定の作成と確立

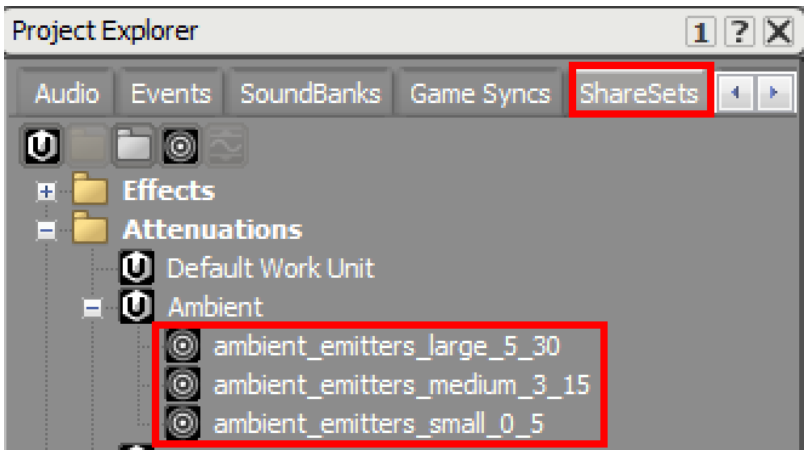
Project Explorer画面のShareSetsタブでDefault Work Unitを選択して、Project Explorerツールバーのゲームパラメータアイコンをクリックし、新規Attenuationを追加します。減衰は、減衰ワークユニットのコンテキストメニューやショートカットキーからも作成できます。



New AttenuationのShareSetを追加

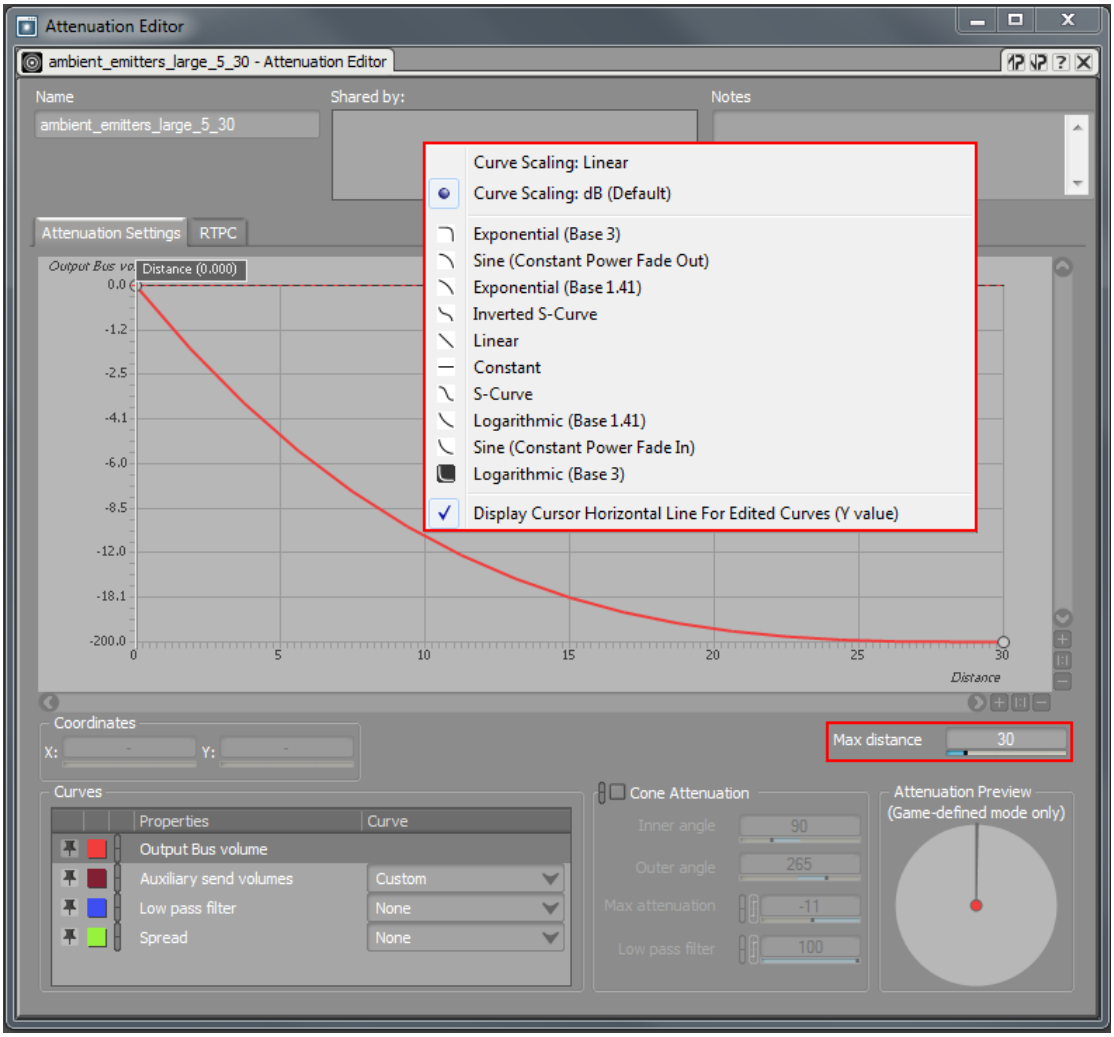
最初に、多くのシナリオで利用できそうな一般的な減衰設定を作成すると良いでしょう。プロジェクトを進めて行く中で、距離関連の具体的な問題やスペシャルエフェクトに対応する減衰を必要に応じて追加します。

まず、Ambient Emitter用の一般的な減衰として、Large（大）、Medium（中）、Small（小）の減衰設定を用意します。この時、減衰設定の名前の一部に距離に関する情報を挿入して使用目的が分かるようにします。プロジェクトの一貫したネーミングルールを決める時に、作成したサウンドオブジェクト、イベント、ゲームシンク、シェアセットなどを、使う人が簡単に理解できるように心がけることが大事です。



一般的な減衰が表示されたProject Explorer画面のShareSetsタブ

作成した減衰カーブのプロパティを調整して、リバーブ、LPF、スプレッドの程度をコントロールします。減衰設定で決めたMax distance（最大距離）に基づき、それぞれのカーブで減衰のプロパティを定義します。最大距離とは、サウンドが最低レベルとる時の、エミッタソースからの距離です。最大距離を超えると、そのオブジェクトの減衰は一定になります。

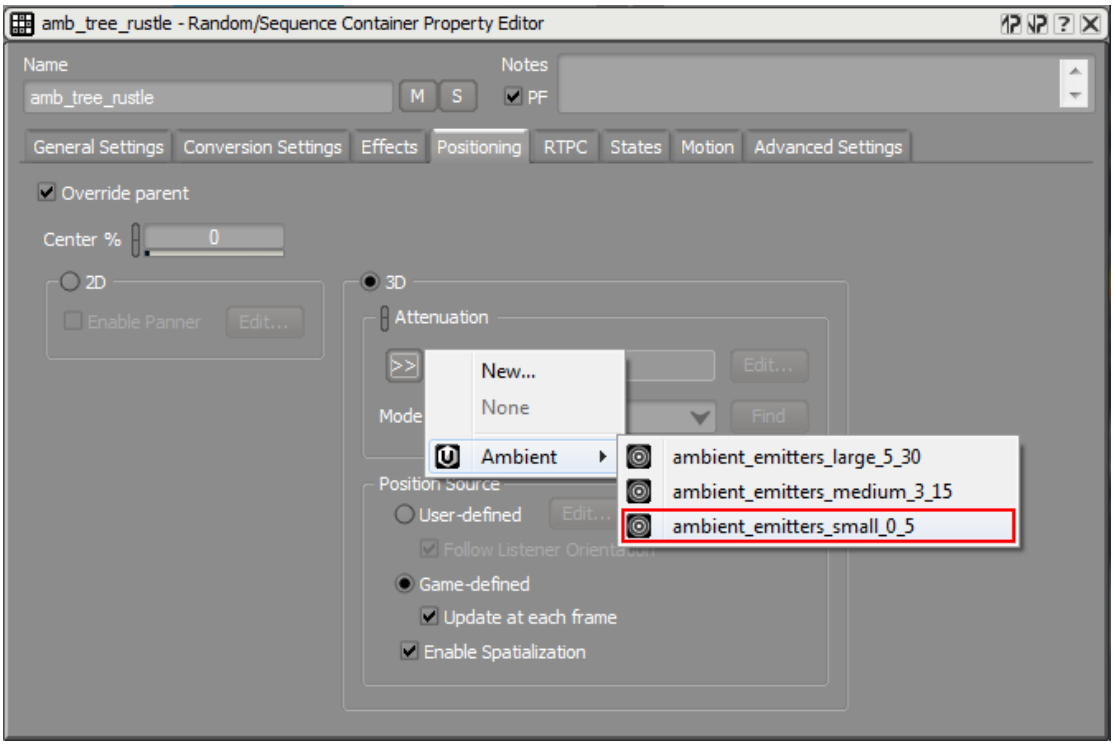


カーブのProperties、Max distance、減衰機能の一覧

減衰にサウンドオブジェクトを登録する

アンビエントエミッタ用の一般的な減衰をいくつか設定したので、アクターミキサー階層のどのレベルに対しても適用することができます。なお、親サウンドオブジェクトが登録した減衰設定は、オーバーライド（無効化）しない限り、子サウンドオブジェクトにも継承されます。次の例では、サウンドオブジェクト“ambient_tree_rustle”を減衰“ambient_emitters_small_0_5”に登録し、エミッタからの距離が5ゲームユニット以内の時だけ、木々の音が聞こえるようにします。

Positioningタブを開いて、もし親オブジェクトの下にある場合はそれをオーバーライドし、サウンドを3Dに変えます。セレクタボタン（>>）をクリックすると、減衰シェアセットの階層が表示されるので、登録先とする減衰を選択します。

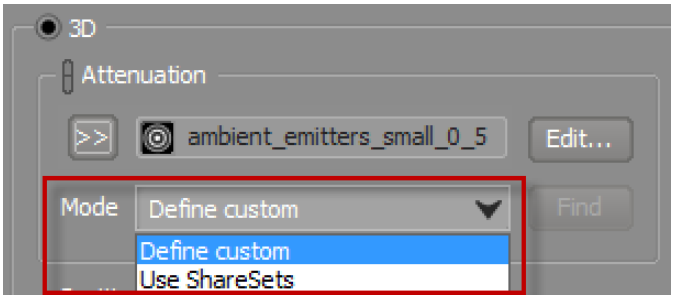


サウンドオブジェクトをAttenuation設定に登録

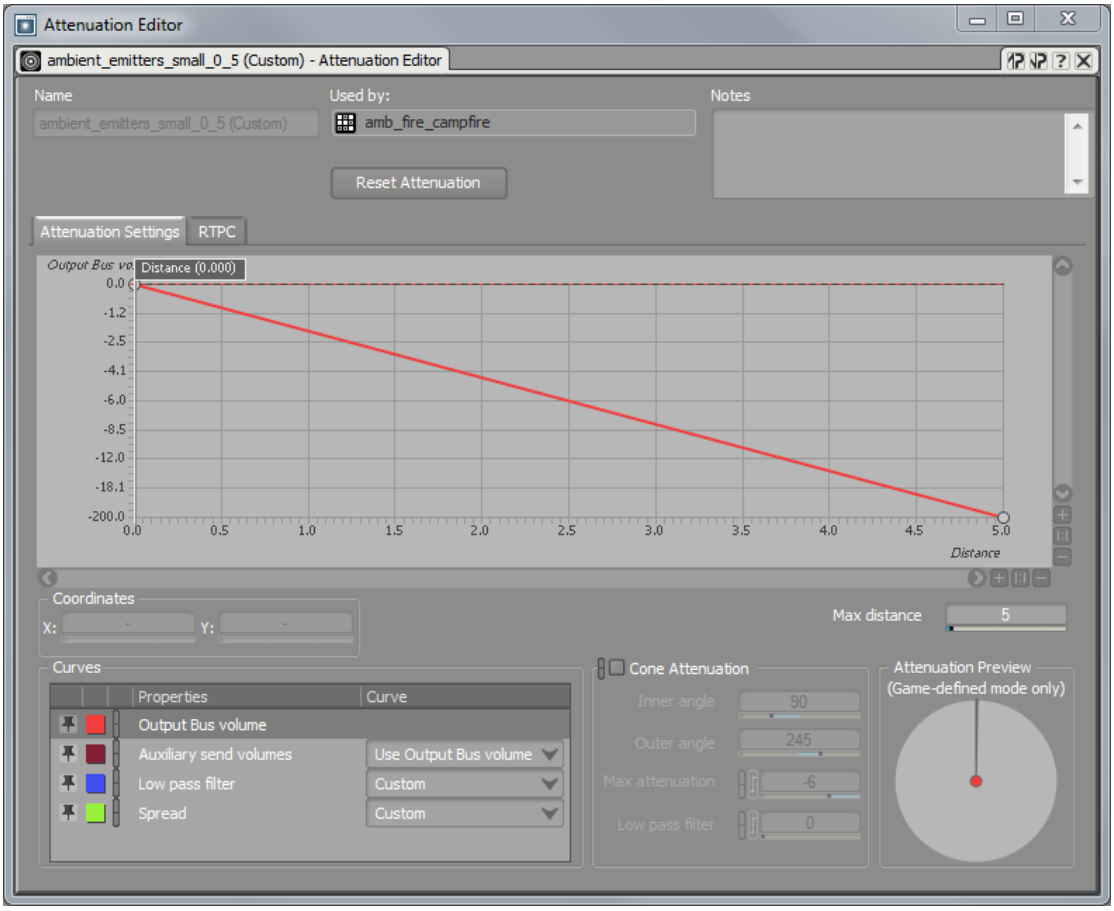
これで、サウンドオブジェクト“ambient_tree_rustle”を再生すると、減衰“ambient_emitters_small_0_5”の定義に従って、リスナーがエミッタの近くに来た時に木の葉の揺れる音が聞こえます。減衰のプロパティが変更されると、この減衰に登録している全てのサウンドオブジェクトを変更されます。

減衰にスプレッドを設定する

次の例でも、同じ減衰のシェアセット“ambient_emitters_small_0_5”に、サウンドオブジェクト“ambient_fire_campfire”を登録します。今回のモードはDefine custom（カスタム設定）として、同じ減衰ambient_emitters_small_0_5を利用しながらも、このオブジェクトの近くではサウンドをSpread（スプレッド）することにします。



減衰のモードをDefine custom（カスタム設定）とする



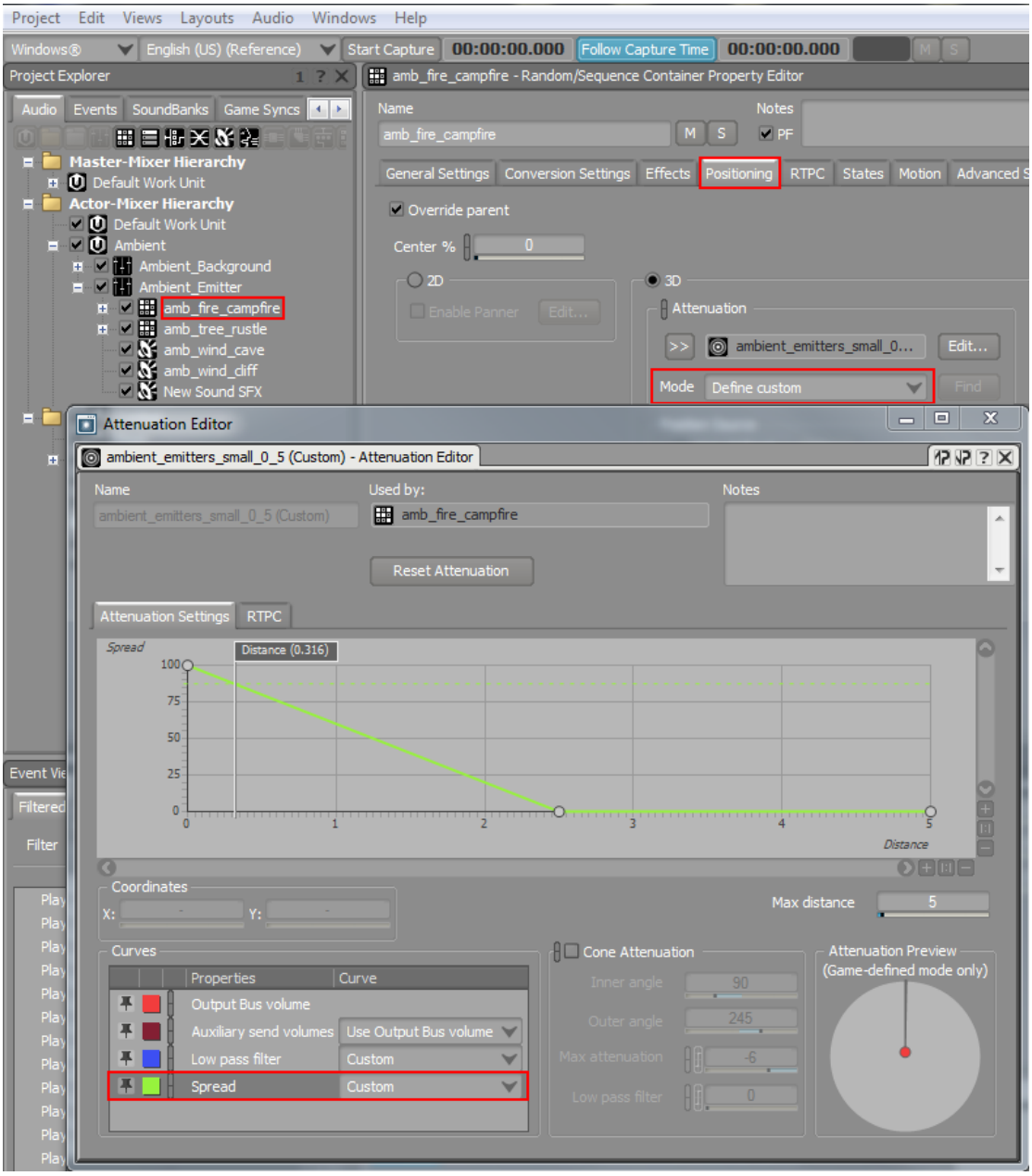
Attenuation Editor画面でカスタム設定できるプロパティ

サウンドオブジェクト“ambient_fire_campfire”の減衰モードをDefine Custom（カスタム設定）に変更したので、他のサウンドオブジェクトとシェアしない（シェアできない）独自の減衰インスタンスが使われます。

Designer Note

Spreadは音が隣のスピーカーに広がる量または割合を指定する機能で、距離値が低い時は点音源、高い値では完全に拡散した伝搬に変わります。マルチチャネルのサウンドでは、各チャンネルで別々にスプレッドします。

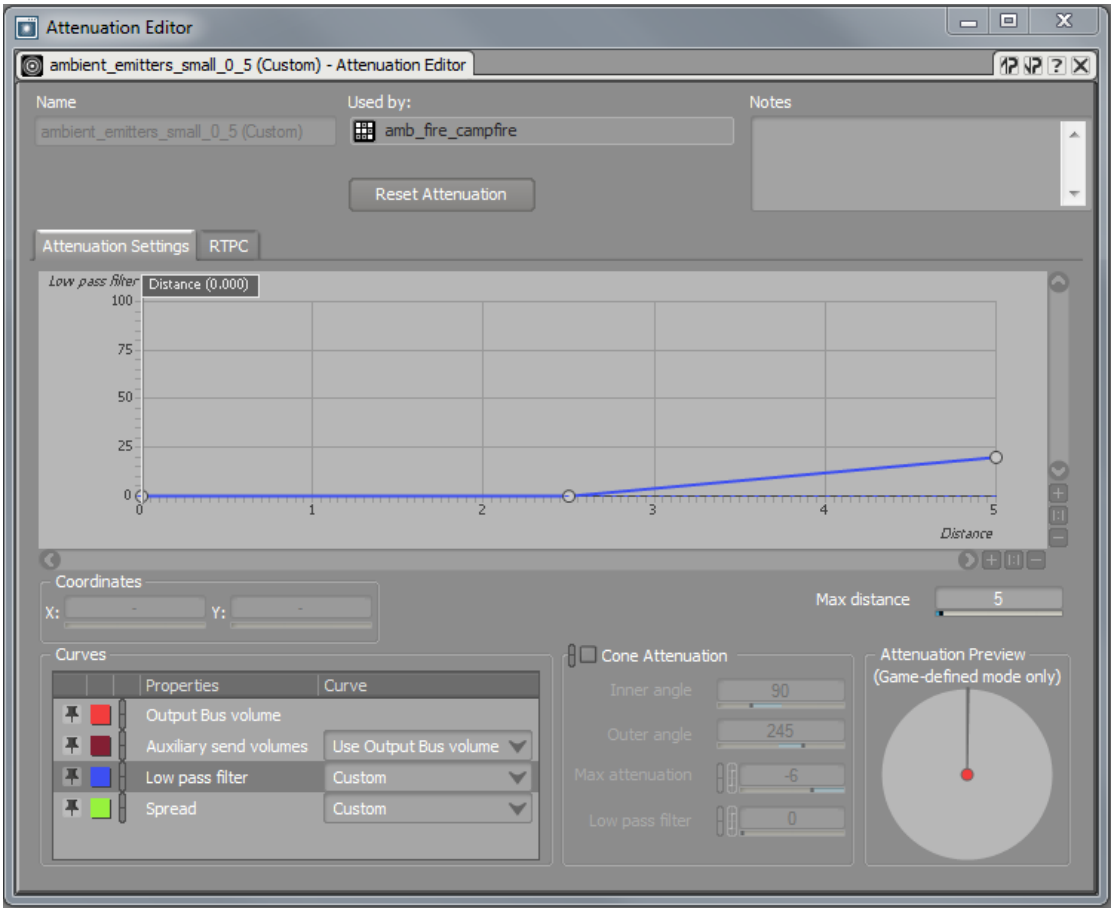
下記の例では、キャンプファイヤーのサウンドが、利用可能なスピーカーの間を最大距離の半分（2.5）でスプレッドし始め、最小距離（0）で100% まで広がります。



Attenuation Editor画面でオーサリングしたSpreadをDistanceパラメータで調整

減衰にローパスフィルターを設定する

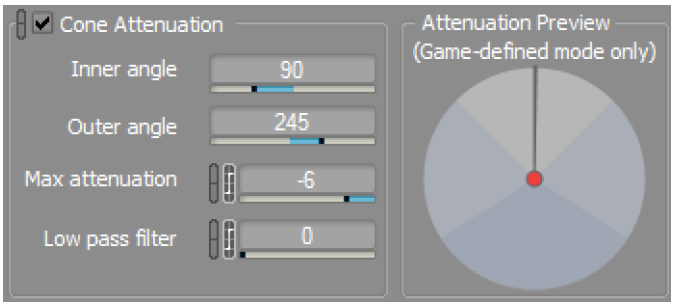
炎の遠のくサウンドをよりリアルにするために、リスナーが遠ざかると周波数の高い音が徐々にカットされるような、ローパスフィルタをカスタム設定できます。



Attenuation Editor画面でオーサリングした
Low pass filter をDistanceパラメータで調整

減衰コーンのプロパティの調整

Cone Attenuation（減衰コーン）プロパティを調整して、1つの減衰シェアセットに登録されたサウンドの方向性を左右させることもできます。ゲームオブジェクトのオリエンテーション（方位）に従ってサウンドの伝搬の角度を制限すれば、ゲームワールド内のサウンドに向きを持たせることができます。例えばビークルや弾丸など、フォーカスを絞った位置関係が要求される物体に特に便利です。



Cone Attenuation（減衰コーン）のプロパティ

本節のまとめ

減衰とは、ゲーム内にあるオブジェクトのサウンドが、距離と共にどう変化するかを決定するルールのセットです。細かなカスタム設定が可能で、ゲームのビジョンに応じてクリエイティブに、またはリアルに使うことができます。高度なリアル感を求めるのか、クリエイティブで印象に残る美しさを求めるのか、デザイナーのニーズに合わせて、ソニックプロパゲーション（音の伝搬）の幅を調整できます。

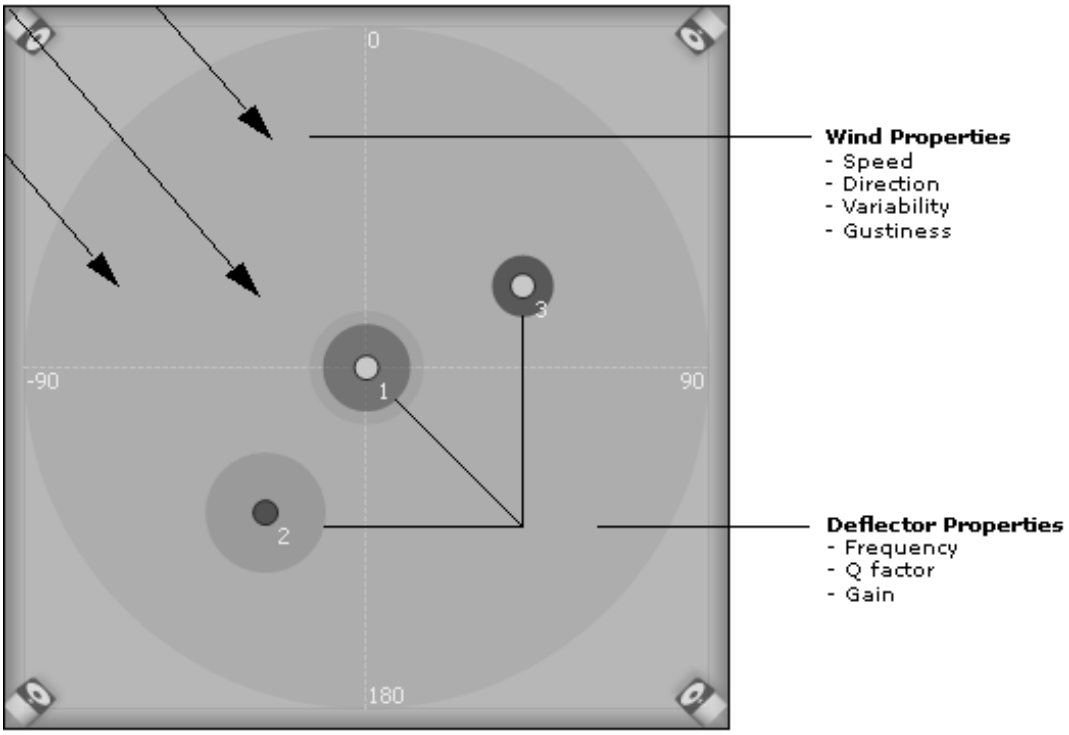
SoundSeed Air -Wind

ゲームでこれから進む道を崖の縁から見下ろしている時も、ダンジョンの入り口に立って影に潜む相手を想像している時も、そのシーンに壮大な雰囲気をもたらすのに役立つのが風の音です。プラグインのSoundSeed Windは、ダイナミックな「風」のエレメントをオーサリングする難しさを解消する手だてとして開発された、ランタイム効率の優れたソリューションです。

SoundSeed Windは風が物体の中や周りを通り抜ける時の音を生成するWwise用のソースプラグインです。合成アルゴリズムを時変パラメータを使って動かすことで音が生成されます。風の音は完全な合成音なので、ソースのオーディオファイルは不要です。長いwavファイルを風のアンビエンス用に準備してループさせる必要がないため、ゲーム中のメモリを節約できます。

SoundSeed Wind は、シーンの中を通り抜ける風の流れを模倣し、方向設定によって、そのシーンに吹く風のエントリーポイントが指定されます。つまり圧力波は、エントリーポイントに最も近いディフレクタ（偏向板）に最初に当たることになります。その後、流れは新しく入ってくる風に押されて、シーン中に伝搬します。伝搬するにつれ、エントリーポイントから離れて配置されたディフレクタに当たる圧力波が聞こえてきます。なお、エントリーポイントで早い風速で入ってくると、遅い風速よりも早い速度で流れを押すことになります。

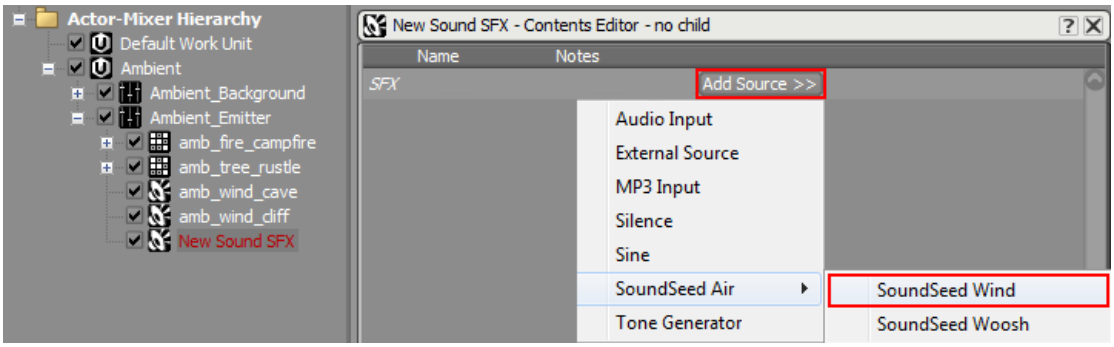
風とディフレクタをプロパティ設定で定義した典型的なシーンを下図に示します。図中のディフレクタは、割り当てられた周波数やゲインのプロパティ値によって大きさや色の濃淡が変わります。



Deflector（偏向板）とWind（風）のプロパティ

合成の風を利用する最大のメリットは、リアルタイムでダイナミックに風モデルを変更できることと、毎回異なるランダムなサウンドを生成できること、そして操作からオーディオコンテンツを除外することでメモリを節約できることです。従来は数秒の長さにおよぶオーディオファイルを要するところが、Wwise内でオーサリングした現実味のあるモデルで置き換えられます。オーサリングされたサウンドはランタイムでゲームパラメータを使ってさらに調整できるほか、ゲームに接続しながらオーサリングアプリケーションで修正できます。

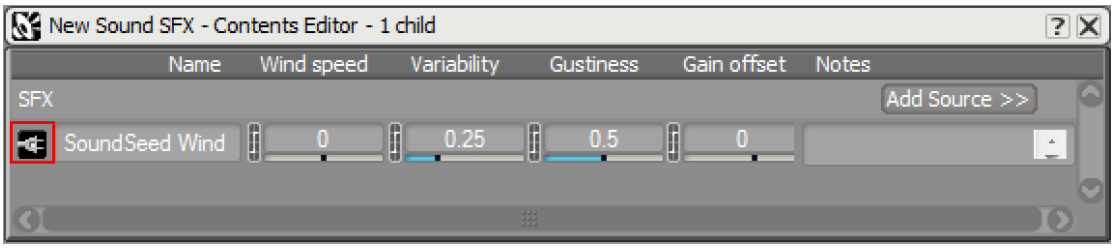
プラグインSoundSeed Windを新規サウンドSFXに追加するには、本章で説明したサイレンスのプラグインの追加と同様に行います。



空のSound SFXにSoundSeed Air Windを追加

SoundSeed Wind - Deflectors

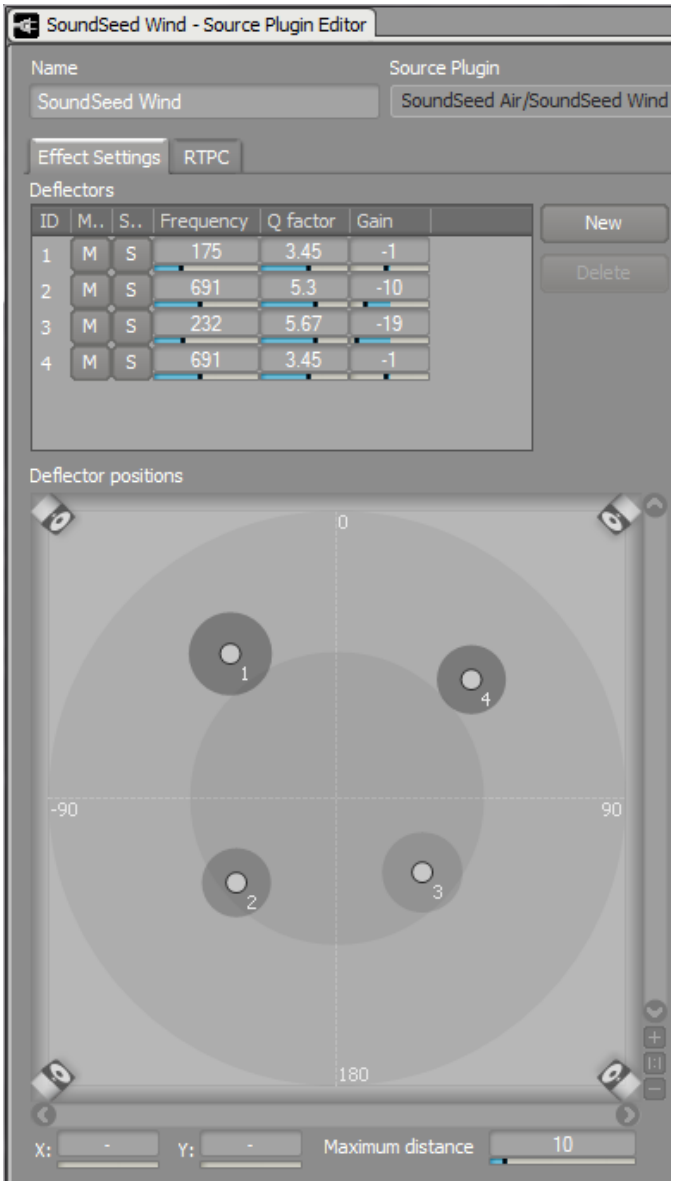
風のモデルを作成するには、まずディフレクタ（単数または複数）を配置して、物体の中や周りを通り抜ける時の風を発生させます。Contents Editor画面のプラグインアイコンをダブルクリックすると、Source Plug-in Editor画面が表示されます。



プラグインWind Sourceのアイコン

以下の属性を設定できます。

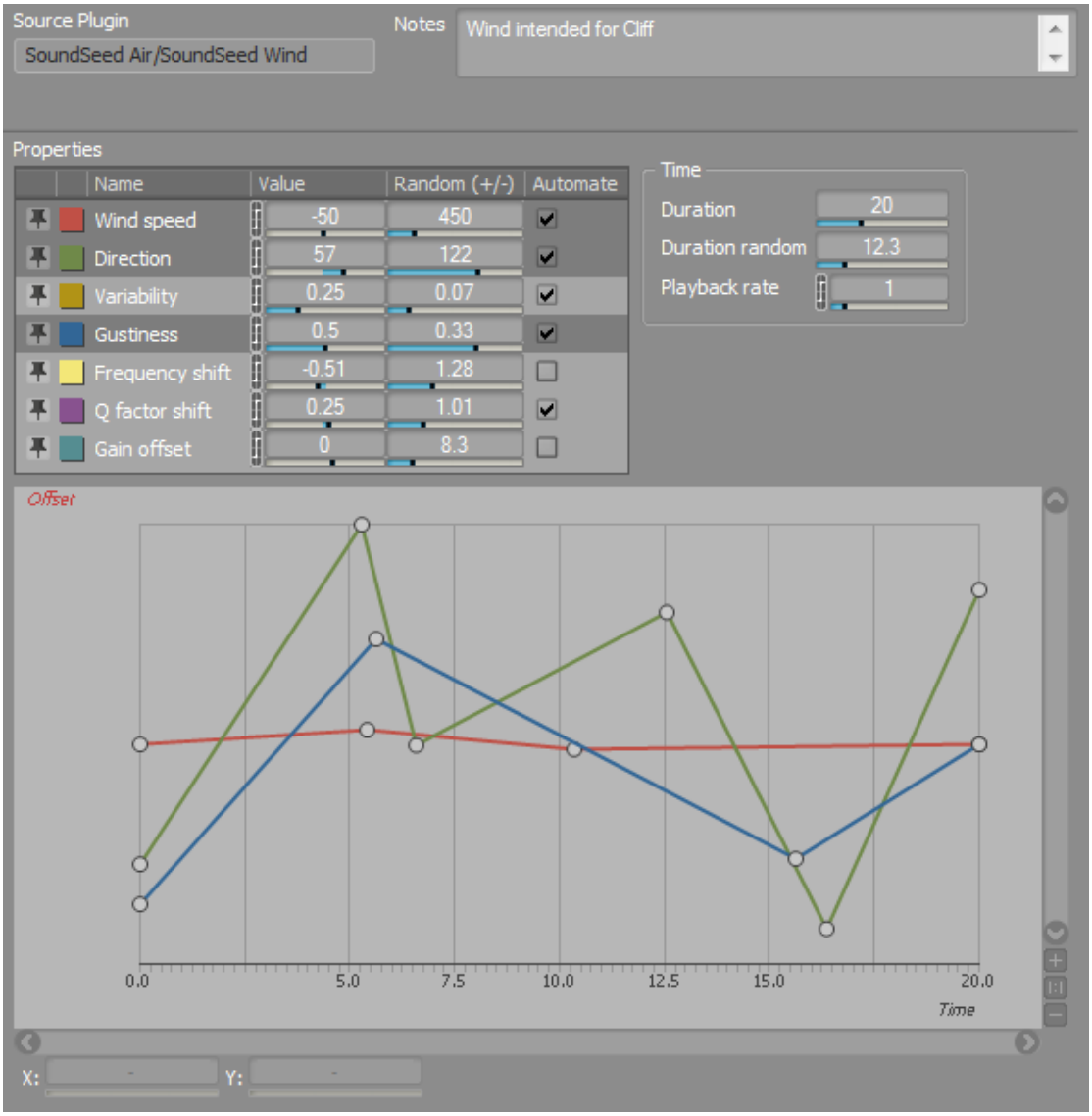
- **Frequency**（周波数）：ディフレクタの大きさを表す。
- **Q factor**（Q係数）：ディフレクタ表面の種類をシミュレーションするために使用。
  - 高いQ値は均一な表面、低いQ値は丸い表面や不均一な表面。
- **Gain**（ゲイン）：ディフレクタの振幅をコントロール。



ディフレクタの配置

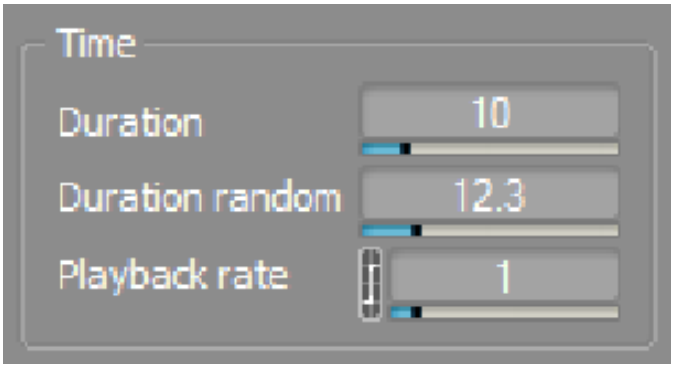
SoundSeed Wind - Properties

他のツールと同様に、実験しているうちに希望するサウンドを発見できるでしょう。風とディフレクタの様々なプロパティ設定は、まず固定値を決めてから、それに対してプラスまたはマイナスの変化を決めます。また、どのプロパティもカーブに自由にポイントを加えて時間と共に起きる変化を自動化できます。



風のプロパティと、カーブの編集機能を使ったプロパティの自動化

RTPCインジケータのある他のプロパティ同様に、Playback rate（再生レート）は、Properties領域にあるTime設定でRTPCを使って変更できます。ループのDuration（固定の長さ）やDuration random（長さのランダム性）は、Time設定で指定して、ループは、他のサウンドオブジェクトと同じくGeneral Settingsタブで有効にします。



時間に関連したプロパティ

SoundSeed Wind - RTPC

サウンドのプロパティがゲームパラメータにマッピングされているため、ゲームの情報を利用して、風の音の様々な特性をコントロールできます。ゲームでプレイヤーがジャンプした後に急落する時の風の速さをプレイヤーの落下速度に合わせてコントロールしたり、山を登るプレイヤーの登山高度のゲームパラメータに突風のサウンドをマッピングできます。風のモデルにパラメータコントロールを適用することで、ゲームプレイやシナリオの意図をダイナミックに表現して忠実に反映できます。

アンビエントのまとめ

本章では、変化に富んだダイナミックなサウンドスケープを形成するための、いくつかの基本的なプロセスを確立させました。本章で説明したテクニックの多くはこのドキュメントを通してこれ以降も登場し、インタラクティブオーディオの基礎となります。

本章では、以下を説明しました。

- 様々な種類のサウンドオブジェクトの作成
- サウンドオブジェクトを使用したアンビエントサウンドスケープシステムの基礎の確立
- ループするバックグラウンドの確立
- 個別のアンビエントエレメントをランダム再生するシステムの作成

プロセスの説明

- コンテナに単純なサウンドをインポート
- ブレンドコンテナに複数のサウンドオブジェクトをまとめ、同時に再生
- プロパティを変化させるゲームパラメータの作成
- コンテナ間のクロスフェード にパラメータ“Time_of_Day”を使用

詳細設定の説明

- 減衰シェアセットを使った距離に基づくフォールオフの管理
- SoundSeed Air - Windを使ったリアルでダイナミックな風のオーサリング
- Position Editor機能のパスを活用した、3D空間における個別アンビエントエレメントのランダム化された配置

オブジェクトの作成方法

- ループするバックグラウンドのランダムコンテナ
- コンテンツ間のクロスフェードをゲームパラメータ“Time_of_Day”に基づくブレンドトラックを使って実行する、ブレンドコンテナ“Ambient_day_night”
- 位置を有するサウンドエミッタとして使えるSoundSeed Windモデル
- アンビエントエミッタ用の一般的な減衰シェアセット

参考ドキュメントとチュートリアル

Wwise Knowledge Base - How does Wwise handle multichannel sources with 3D positioning?

Wwise Knowledge Base - How do I simulate a sound that is not a point sound source?

Video Tutorial - Using the interface

Video Tutorial - Importing Audio Files

Video Tutorial - Building Sound Hierarchies

Video Tutorial - More Learning Resources

Video Tutorial - SoundSeed Air Wind Overview

Video Tutorial - Using the Blend Container

Video Tutorial - Relation between Sound- Source and Audio File

第2章 キャラクターの構築

概要

ヒーローが森に入った瞬間の足音や、ダンジョンを通り抜けた後にアップグレードされたアーマーなど、動作を表すサウンドはプレイヤーをゲームワールドに引き込む役割を担います。ゲームの種類によってキャラクターサウンドに対する要求も大きく変わりますが、どのジャンルにも共通する基本があります。

本章では、以下の操作を説明します。

- プレイヤーとノンプレイヤーキャラクター（NPC）の動作設定
- スイッチグループやスイッチの目的と使い方
- 以下の情報を管理するマルチレベルのスイッチの設定
  - 地面の種類
  - プレイヤーの種類
  - ステップ（足どり）の種類

キャラクターのサウンドは、ゲームワールドにある全てのエリアやシナリオに常に存在しています。ゲームプレイ中に頻繁に聞こえてくるサウンドのデザインや実装を計画する際は、特に注意してください。Wwiseでは、あらゆる状況の動作サウンドに対応するための方法がいくつか用意しています。

足音と動きのニーズを探る

オーディオコンテンツのニーズを明確にする前に、ゲームプレイをサポートする動作サウンドについて理解すべき点がいくつかあります。単純な足音システムを作成するのは簡単ですが、多様性のあるシステムの方が優れたプレイ体験を提供します。様々な足音サウンドをゲームに入れる際は、使用可能なメモリを使い切ることなく、適切な場所にリソースを費やし、バランスの良いシステムを設計することが大事です。

これから作成する足音システムの適切な規模を決定するために、以下の点を検討して下さい。

- キャラクターの足音は何種類か。（歩く、走る、ひきずる、曲がるなど）
- サウンドで表現しなければならない地面は何種類あるか。（土、岩、砂など）
- 足音サウンドを何でトリガーするのか。（アニメーション、プログラムなど）
- 足音が続けて聞こえてくる頻度はどれくらいか。（2 ～ 5分、10 ～ 20分など）

シンプルな足音

単純な足音システムでは、キャラクターの歩みがアニメーションの一定フレームに到達する度に1つの足音のオーディオファイルを再生するようなランダムコンテナが考えられます。アニメーションに依存した足音の場合、足音サウンドを再生すべきアニメーションの各フレームをWwiseのイベントでマークする（タグを付ける）方式で行うのが一般的です。そのアニメーションのフレームに対し、歩く、走る、曲がる、引きずる、跳ねる、着地するなどの適切な足音タイプを直接タグ付けして、正しい足音サウンドの入ったイベントに対応します。

単純な足音システム

このテクニックは足音の少ないゲームでは使えますが、このような単純なシステムでは同じ音の繰り返しが気になり、サウンドデザイナーもオーサリングアプリケーション上で足音の各種バージョンを管理できないので、限界があります。

スイッチシステムの導入

スイッチは、1つのゲームオブジェクトに存在する、複数の選択肢を表します。サウンド、ミュージック、モーションの各オブジェクトを整理してスイッチに割り当てて、ゲームで現在の選択肢から他に変わった時に、該当するサウンドオブジェクトやモーションオブジェクトを再生します。1つのスイッチにアサインされた複数のWwiseオブジェクトは、スイッチコンテナにまとめられます。変更を指示するイベントがあると、スイッチコンテナがそのスイッチを検証し、正しいサウンド、ミュージック、またはモーションのオブジェクトを再生します。

それでは、いくつかのスイッチグループをカスケード式に並べ、それらが一体となり以下の条件で選ばれる正しい足音を再生するようなシステムを作成します。

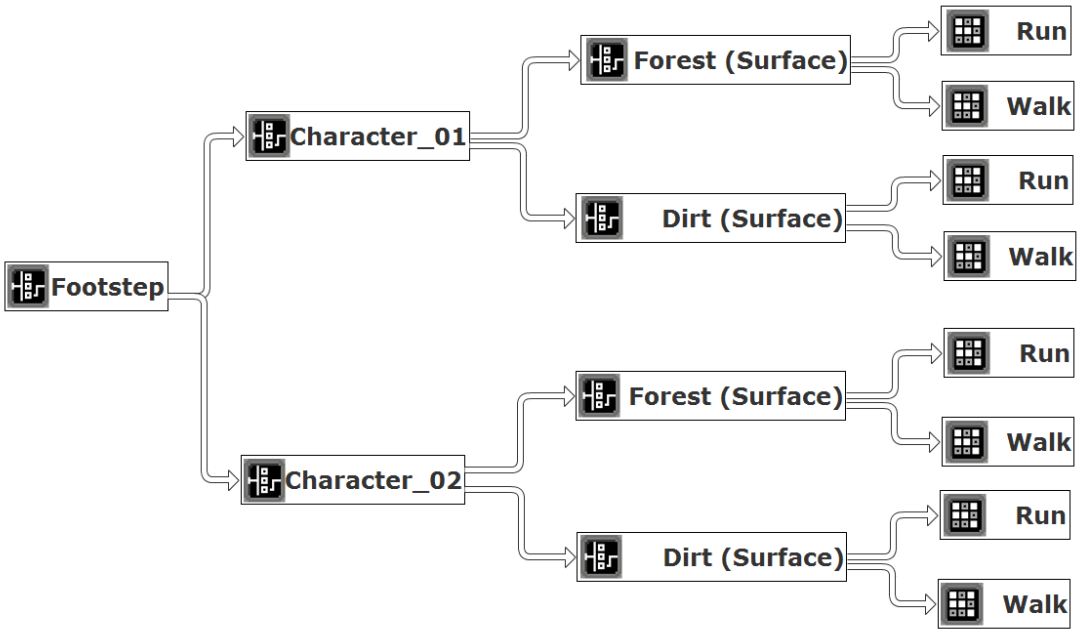
- ステップ（足どり）の種類
- 地面の種類
- キャラクターの種類

スイッチの威力を最大限に活かすためには、プログラマーがこれらをゲームエンジン側で定義してシステムを動かす必要があります。

ゲームに出てくるアセットは、岩のテクスチャーからキャラクターモデルに至るまで、各アセットの内容とプロパティがそのアセットのメタデータとして含まれることがよくあります。 この既存の定義情報を使ったり、この情報を作成してスイッチシステムをコントロールすれば、データ主導型のパイプラインに近づき、開発中のアセット管理や拡張がしやすくなります。

Designer Note

ゲームが送出する情報を使ってWwiseのスイッチシステムを動かすことについては、開発サイクルのできるだけ早い段階でオーディオプログラマーと話し合ってください。実装に関連して起きる課題の対処を自動化できる可能性が多々あるからです。

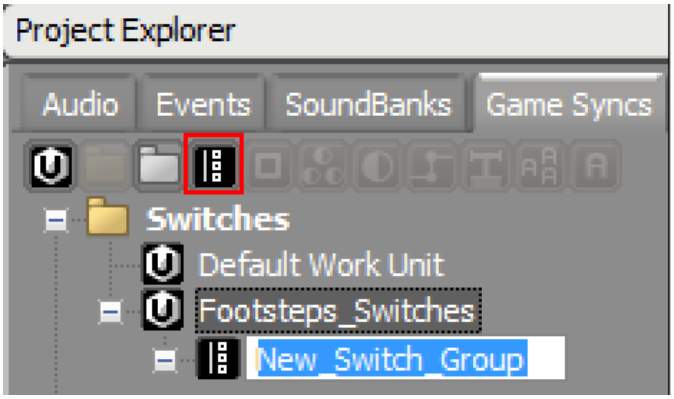


マルチレイヤーのスイッチシステムで、キャラクター、地面、ステップの種類に応じて、スイッチ（切り替え）を実行

これは単純な足音システムと比較して一見複雑そうに見えますが、ゲームが送出する情報をサウンドに利用できる柔軟性があるため、サウンドに対するコントロールの幅が広がり、サウンドデザイナーがよりクリエイティブに表現することが可能になります。

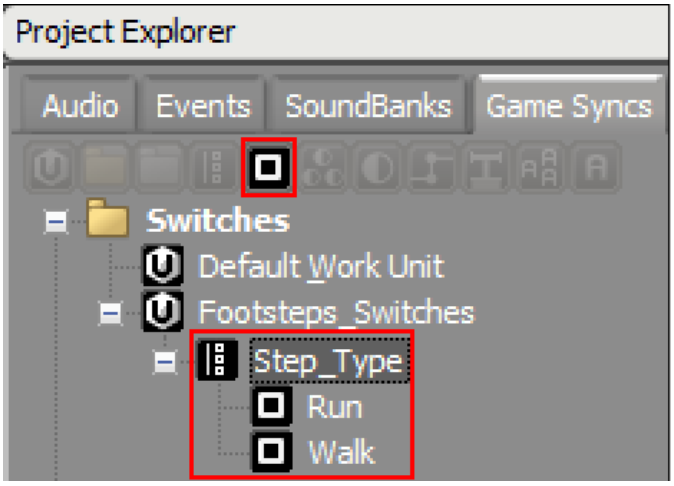
足音の種類の定義

まず、Footstep（足音）スイッチ用の新規ワークユニットを作成して、歩きスイッチと走りスイッチ間で切り替えるスイッチグループ“Step_Type”を追加します。新規ワークユニット“Footstep_Switches”を選択して、Project Explorerツールバーのスイッチグループのアイコンをクリックして、新規スイッチグループを作成します。



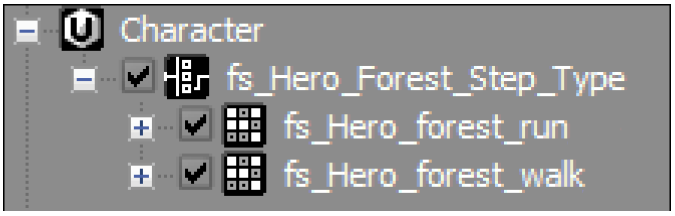
ワークユニット“Footstep_Switches”に新規スイッチグループを作成

新規スイッチグループ“Step_Type”を作成した後、新規スイッチグループを選択してProject Explorerツールバーのスイッチアイコンをクリックして、スイッチが追加します。今回はWalk（歩く）とRun（走る）の2種類のステップに対応する2 つの新規スイッチを作成します。また、コンテキストメニューやショートカットキーからスイッチやスイッチグループを作ることもできます。



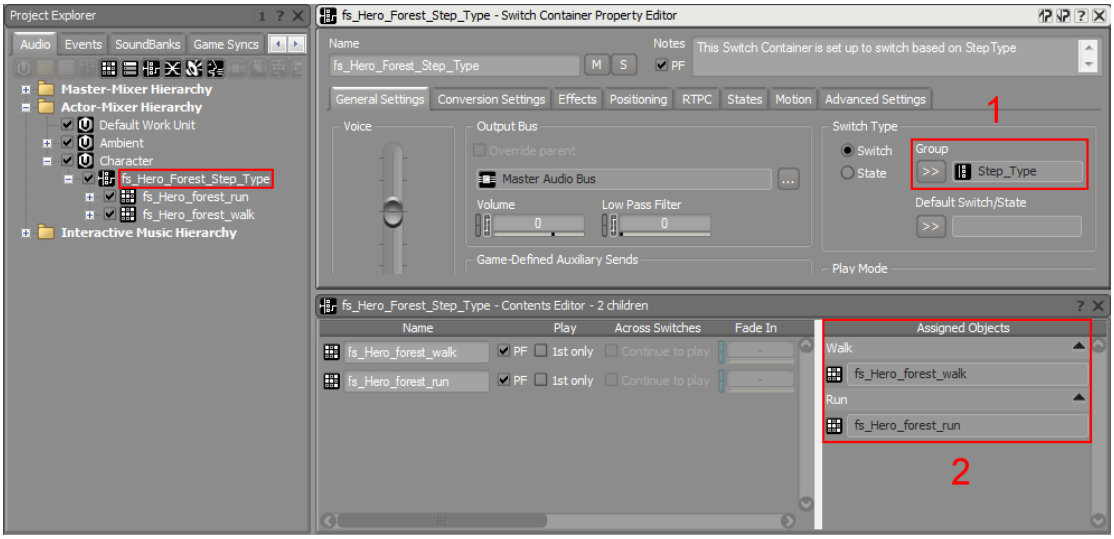
スイッチグループ“Step_Type”と、その下のスイッチ“Run”と“Walk”を作る

次に、“run”と“walk”の2 つのランダムコンテナが入った新規スイッチコンテナを作成します。



2 つのランダムコンテナ“run”と“walk”が入ったスイッチコンテナ

そして、Switch Type（スイッチの種類）のセレクタボタン（>>）から新しいスイッチグループ“Step_Type”を選び（1）、スイッチの種類を指定します。次に、ランダムコンテナのwalkとrunを割り当てるために、これらをContents Editor画面のAssigned Objects（アサインされたオブジェクト）エリアまでドラッグ＆ドロップします（2）。



スイッチグループStep_Type を追加し、オブジェクトをアサイン

各種ステップ（足音）は、Transport Control画面で試聴できます。スイッチコンテナ“fs_Hero_Forest_Step_Type”をダブルクリックで選択し、再生ボタンをクリックするかスペースバーを押すと、オブジェクトが再生されます。スイッチを選ぶには、Transport Control画面のゲームシンクエリアでSwitchesを押します。オブジェクトが利用できる関連ゲームシンクが表示されます。

Transport Controlでゲームシンクスイッチを選び、選択中のゲームオブジェクトに利用できるスイッチを表示

地面の種類の定義

多様性を増やすのに最も有効な方法の1つであり、プレイヤーに場所の雰囲気をすぐに伝えることができるのが、足音用の地面の種類です。地面の様子はゲーム中にビジュアルとして表現されますが、キャラクターが横切るその地面をスイッチで取り扱うには、情報伝達に多少の工夫が必要です。理想的には、移動中の地面の種類をゲームエンジンが特定し、その情報をスイッチの引き金としてWwiseに送出します。

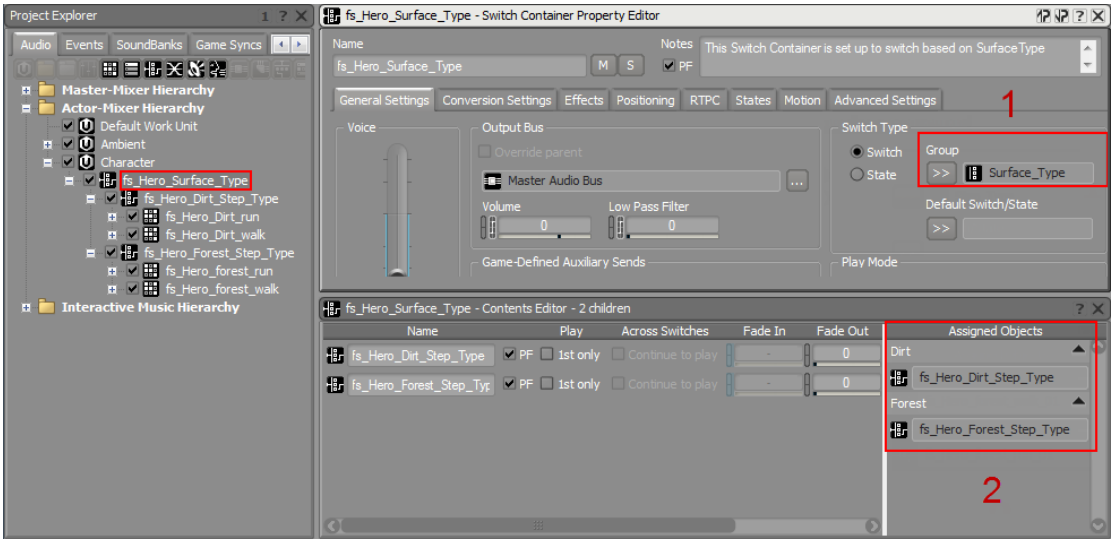
Programmer Note

Wwise Programmer SDKに、分かりやすい足音インテグレーションの例があります。

Designer Note

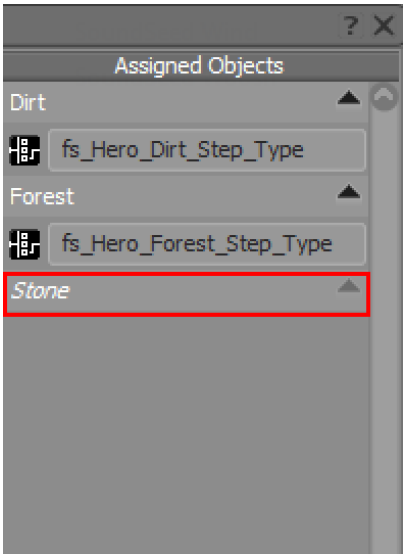
Wwiseに伝える地面情報の使い方や活用方法は、足音に限られません。例えば、ダイナミックな雨音システムを構築する場合、ゲームワールドの様々な地面に雨があたるインパクトを場所によって変更できます。また、物理的インパクトのサウンドの場合も、インパクト（衝撃）を受ける面の材質が分かれば効果的になることがよくあります。例えば、土の地面に倒れる木材のサウンドは、コンクリート面と比べて鈍い音になります。

前節と同じように、新規スイッチコンテナを作成し、新規スイッチグループ“Surface_Type”に登録します（1）。次に、Contents Editor画面のAssigned Objectsエリアで、“Dirt”（土） と“Forest”（森）のスイッチに適切なランダムコンテナを追加します（2）。



スイッチグループ“Surface_Type”を追加し、オブジェクトをアサイン

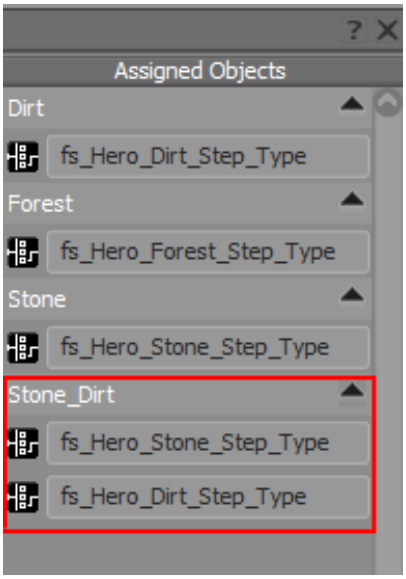
複数のレベルで成立するスイッチシステムを使うことで、Wwiseの階層構造がサポートする整理されまとまった作業方式が成立します。また、実装作業を分かりやすく簡潔に保つのにも役立ちます。さらに、ゲームに新種の材質などが導入されたときは、スイッチをスイッチグループに追加して拡張できるので、分かりやすく単純です。例えば“Stone”（岩）という地面のスイッチを追加したい場合、それをスイッチグループに追加することで、“Surface_Type”グループに登録された全てのサウンドオブジェクトのContents Editor画面で、Assigned Objectsエリアに表示されます。



既存のスイッチグループにスイッチを追加

また、“Stone”の足音サウンドのコンテンツがまだ完成していない場合は、とりあえず手元にあるオーディオコンテンツを使い、準備ができたら置き換えます。一時的にコンテンツをアサインすることで、すぐにフィードバックを確認し、システム稼働の検証ができます。さらに確認するには、ゲームが返すキャプチャーログをWwise Profiler画面で見ることもできます。(第10章の「10.4 Wwiseの各種プロファイリング機能について」を参照。)

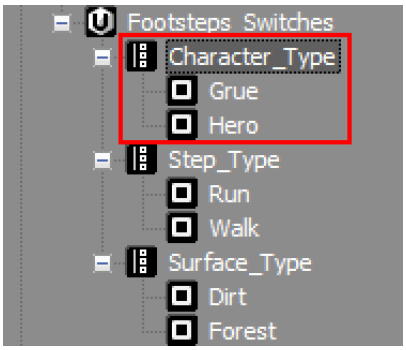
このように抽象化した方式を採用しているので、スイッチ間でコンテンツをシェアすることもでき、基本的な要素同士を組み合わせれば、コンテンツを追加することなくバリエーションを増やせます。例えば、“Dirt”（土）“Stone”（岩）のサウンドを組み合われば、既にあるコンテンツを使って、ほこりっぽい岩の地面が新たにできるかもしれません。



1つのスイッチに複数のサウンドオブジェクトを設定

キャラクターの種類の定義

この手法を発展させて、“Character_Type”のスイッチグループも設定できます。



スイッチグループ“Character_Type”と登録されたスイッチ

合わせて再生する

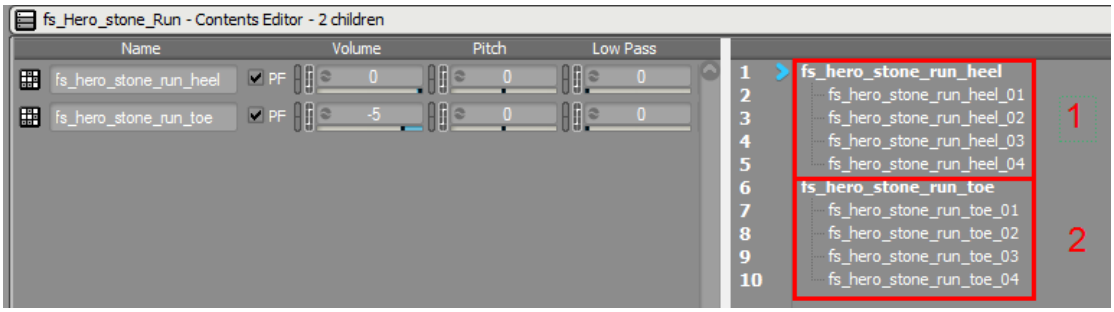
これで、キャラクターサウンドの以下3つの性質を示すスイッチ情報が、ゲームからWwiseに伝達されます。

- ステップ（足どり）の種類
- 地面の種類
- キャラクターの種類

これでイベント“Footstep”がゲームによってトリガーされると、上記のスイッチが組み合せられ、正しいキャラクター、ステップ、そして地面の各サウンドオブジェクトに対応するサウンドが、再生されます。このため、複数のイベントを把握しておく必要はなく、“Footstep”という1つのイベントをゲームから送ることで、ゲームが定義するサウンドをスイッチが自動的に、正しく再生します。

Designer Note

コンテンツのレベルで時々使われるテクニックの1つに、つま先と踵を別々に設定する方法があります。ランダムに組み合わせるとさらに多様なサウンドが生まれ、適切にデザインされたコンテンツを使えば、バリエーションが飛躍的に増え、繰り返しの可能性を減らせます。新しいシーケンスコンテナを作成して、ランダム再生のコンテナ“Heel”（踵）（1）と、それを追うランダム再生のコンテナ“Toe”（つま先）（2）を設定すれば、足音が要求された時に一歩一歩をオンザフライで作れます。また、踵やつま先の要素を他の地面でも利用できるのであれば、さらにメモリを節約できます。このようにWwiseオーサリングアプリケーションのテクニックを基本要素に適用すれば、バリエーションを増やすと同時にメモリフットプリントを減らせるので、どのような開発プロジェクトにとっても有利す。



一歩の足音を作るために、踵とつま先を続けて再生

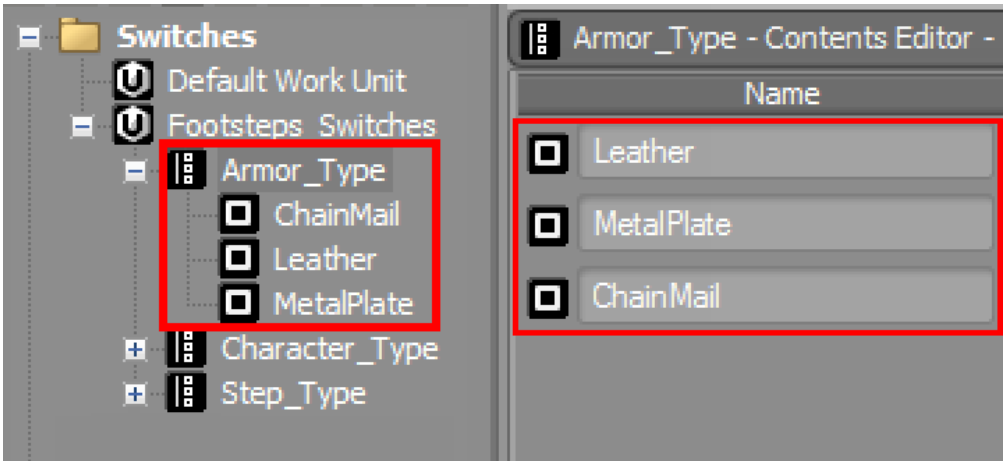
ゲームや実装方式によって求められる細かさは異なりますが、ゲームエンジンからの情報に依存して再生されるコンテンツのグループがある限り、スイッチは効率的な処理方法を提供します。

動作

動作に伴うサウンドのもう1つの重要な要素が、ワールドを旅するキャラクターが着る服の素材の音です。着古した“Leather”（レザー）や“Chain Mail”（鎖かたびら）、“Metal Plate”（鎧）など、特徴ある質感が足音サウンドと共に信憑性を一段と高めます。また、ゲームプレイ中に別の種類の服装や装備に変更したり追加できるのであれば、動作のサウンドが特に重要です。可能なときに必ず音を切り替えることで、プレイヤーが服装などが変わったことを意識し、ゲームの体験がよりリアルに感じられます。

アーマータイプの定義

足音に使ったスイッチシステムと似た方法で、スイッチ“Character_Type”と“Step_Type”を使い、新しく“Armor_Type”というスイッチを追加します。



スイッチグループ“Armor_Type”とそのスイッチ

各スイッチがスイッチシステムを動かすために使われるので、適切なサウンドコンテンツをスイッチで選べるように階層を整理します。

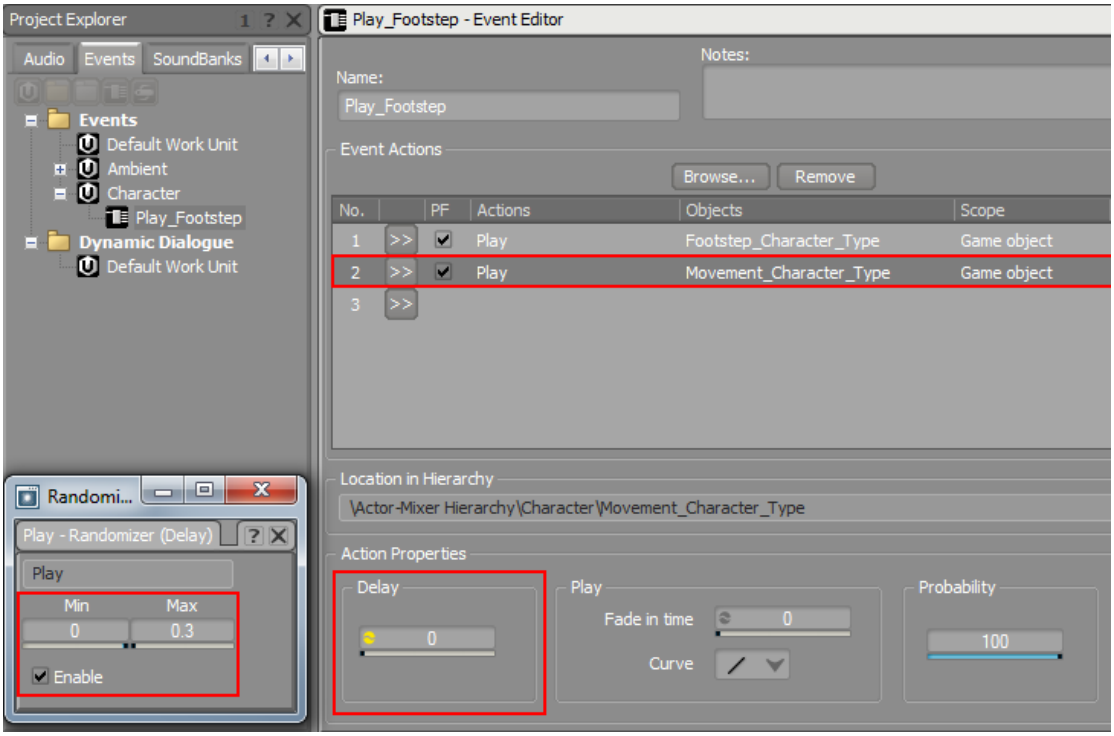
Movement（動作）のスイッチと、アサインされた“Step Type”

動作イベントの作成

適切な選択を行うスイッチとコンテンツが揃ったところで、アクションPlayを使って動作用の親スイッチコンテナを足音イベントに追加します。

Design Note

動作のサウンドコンテンツによっては、足音サウンドと同時にアーマーの素材サウンドを再生するのが好ましくないこともあります。足音サウンドがはっきりと聞こえるように間を置いた方が良ければ、動作の再生にディレイのプロパティを設定できます。さらに、動作サウンドに一定の長さのディレイのオフセットを追加してディレイのランダム設定と合わせれば、ランダム性が増してよりリアルな表現となります。



スイッチMovementのDelayオフセットを設定

このテクニックを使うと、足音と足音の間の動作サウンドがより現実的に表現でき、毎回の足音イベントにバリエーションと独自性が増えた感覚が得られます。

これ以外のスイッチ機能として、ゲームパラメータに基づいてスイッチを変化させる機能や、スイッチの継続モードやフェード設定など、実装をよりクリエイティブにするテクニックがあります。またスイッチは使いやすい整理ツールにもなり、ゲーム情報を活用してワークフローを効率化するためにも便利です。

キャラクターのまとめ

本章では以下の通り、スイッチグループやスイッチを分かりやすく説明し、ゲーム情報を利用して動くシステムの一部としてスイッチをどう活用できるのかを見てきました。ゲームから受け取る情報を使ったシステムを構築した場合の強みは、コードを変更せずにツール内でコンテンツを変更したりコントロールして、調整できることです。ゲームとは独立して動作のコントロールをサウンドデザイナーが担えるようになります。

本章では、以下を説明しました。

- 単純な足音システムの設定
- スイッチグループやスイッチの目的と使い方
- サウンドオブジェクトのプロパティのランダム化
- Contents Editor画面のAssigned Objects領域内へのコンテナの設定

プロセスの説明

- 以下の情報を利用した、プレイヤー用とノンプレイヤーキャラクター（NPC）用の足音のランダムコンテナの設定
  - キャラクターの種類
  - 地面の種類
  - 足音の種類
- 以下の情報を利用、プレイヤー用とノンプレイヤーキャラクター（NPC）用の動作のランダムコンテナの設定
  - キャラクターの種類
  - 足音の種類
  - アーマーの種類
- 以下の情報を管理するスイッチの設定
  - 地面の種類
  - プレイヤーの種類
  - ステップ（足どり）の種類
- 足音と動作のためのマルチレベルのスイッチシステムの作成

詳細設定の説明

- イベントの中でディレイオフセットを使い、自然なランダム性を動作サウンドに適用

オブジェクトの作成方法

- 足音イベントを再生する時に、キャラクターや地面、ステップの種類を判断する、マルチレベルのスイッチシステム
- 足音イベントにランダムなディレイオフセットを設定した、動作サウンド用のマルチレベルのスイッチシステム
- 足音と動作の両方のスイッチシステムの再生アクションで構成された、足音イベント

参考ドキュメントとチュートリアル

Wwise Help > Where to Begin? > Wwise Fundamentals > What are Game Syncs? > **Understanding Switches**

Wwise Help > Interacting with the Game > **Working with Switches**

Wwise SDK - Windows » Sound Engine Integration Walkthrough » Integrate Wwise Elements into Your Game » **Integrating Switches**

Video Tutorial - Creating Footsteps using Random and Switch

第3章 戦闘の準備

概要

これまでに作成したアンビエントサウンドスケープの中をヒーローが戦い進んで行く時に、シーンを盛り上げてくれるのが戦闘サウンドです。ゲーム中の新しい出会い、敵キャラクターの別バージョンの出現、ウェポンのアップグレードなど、サウンドはスクリーンで展開するドラマに見合ったダイナミックオーディオでプレイヤーを興奮させます。

本章では、以下の操作を説明します。

- 様々なウェポンのサウンドセットの定義
- SoundSeed Whooshを利用したウェポンスイングの生成
- インパクトサウンドの様々な実装テクニック
- プレイヤーやNPCの減衰設定

戦闘サウンドの作成という、時にして大がかりな作業に取りかかる1つの方法として、まずゲームで入手できるウェポンの種類を確認します。ウェポンの種類を一覧にしてから、それぞれのウェポンをトリガーする方法や、プロジェクト内で整理する方法を理解しておくことが重要です。

戦闘シーンについて、以下の点を検討します:

- ゲームエンジンは、戦闘サウンドをどのようにトリガーするのか
- 戦闘アクションの大部分は、アニメーションシステムに依存するのか
- インパクト用のヒット（命中）検知は、アニメーションとは別に扱うのか

ゲームで悪名高い敵と死闘を繰り広げる時に、攻撃が成功したかどうかをプレイヤーに伝えるのが、サウンドの担う役割の1つです。戦闘サウンドの再生方法を決めてから、各アクションを適切なサウンドで表現できるように設定します。

各種ウェポンのサウンドセットの定義

基本的なレベルでは、様々なウェポンのスイングやインパクトの再生が、同じ実装方法で行われることが一般的です。各ウェポンにはそれぞれ特有の技や動きがありますが、一般的には複数のウェポンで共通する攻撃方法と、特別な扱いが必要なものとを区別する一定のロジックがあります。1つのアクションに対して1つのサウンドをデザインする方式を採用するのか、基本的なサウンドをいくつか準備して部品として組み立てる方式を採用するのか、サウンドがゲームに実装される方法を知ることで、サウンドデザインのアプローチも決まります。

今回は、プラグインSoundSeed Air - Whooshを使って作成した個別のウェポンスイング音を使います。プラグインWhooshは、シミュレーションされた物体が空気を切る音を生成するものです。接近戦でウェポンを振る音や弾丸が通り過ぎる音など、モーションサウンドの生成に最適です。

次に別のアプローチで、ウェポンのインパクト（衝撃）サウンドを、インパクトを受ける素材面の種類（単数または複数）に合わせて、様々な素材サウンドを組み合わ作成します。今回は足音用に作った素材面の種類に応じたスイッチを活用するので、主要なオーディオファイル一式から多様性に富んだサウンドを引き出せます。

SoundSeed Air - Whoosh

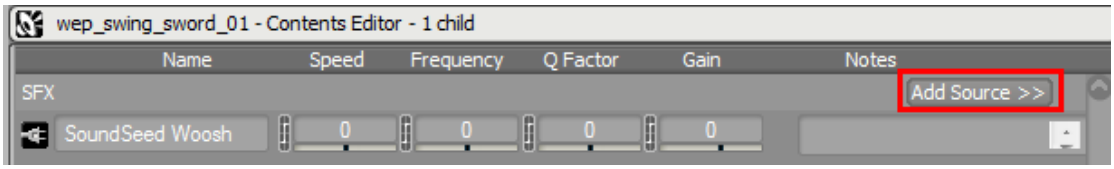
プラグインのSoundSeedシリーズは、Wwiseオーサリングアプリケーションの機能をさらに拡充させ、よくある問題に対してユニークなソリューションを提供するために開発されました。SoundSeedは、リソースに考慮しつつ、ゲーム中に繰り返し使用されるサウンドを新鮮かつダイナミックで多様性に富んだ印象に保ちながら、サウンドデザインをゲームの趣向に近づけます。

プラグインSoundSeed Whooshは、物体が空を切る時のサウンドを生成するソースプラグインです。サウンドを作り出すには、ディフレクタ（偏向）物体の特徴や、動く時の軌跡とスピードを定義します。Whoosh（「ヒュー」）音は完全な合成音なので、元となるオーディオファイルは一切ありません。つまり複数のwavファイルを保存する必要がないため、ゲーム中の使用メモリが節約できます。またランダムの機能も多数あり、Whooshのソースから別のバリエーションを作成することもできます。

それでは、多数あるヒーロー用の“Sword”（剣）のサウンドの1つを作成してみます。まず、アクター・ミキサー“Swing”の中にワークユニット“Combat”を設定し、その下に空のSound SFXを準備します。

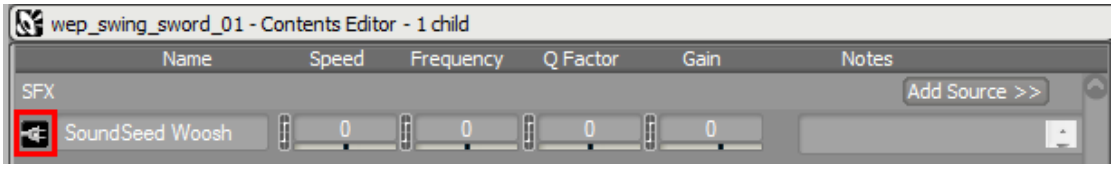
Whooshのソースを追加するために準備した、空のSound SFX

次に、Contents Editor画面の右上にあるAdd Source（>>）をクリックして、SoundSeed Airシリーズのプラグイン SoundSeed Whooshをソースとして追加します。

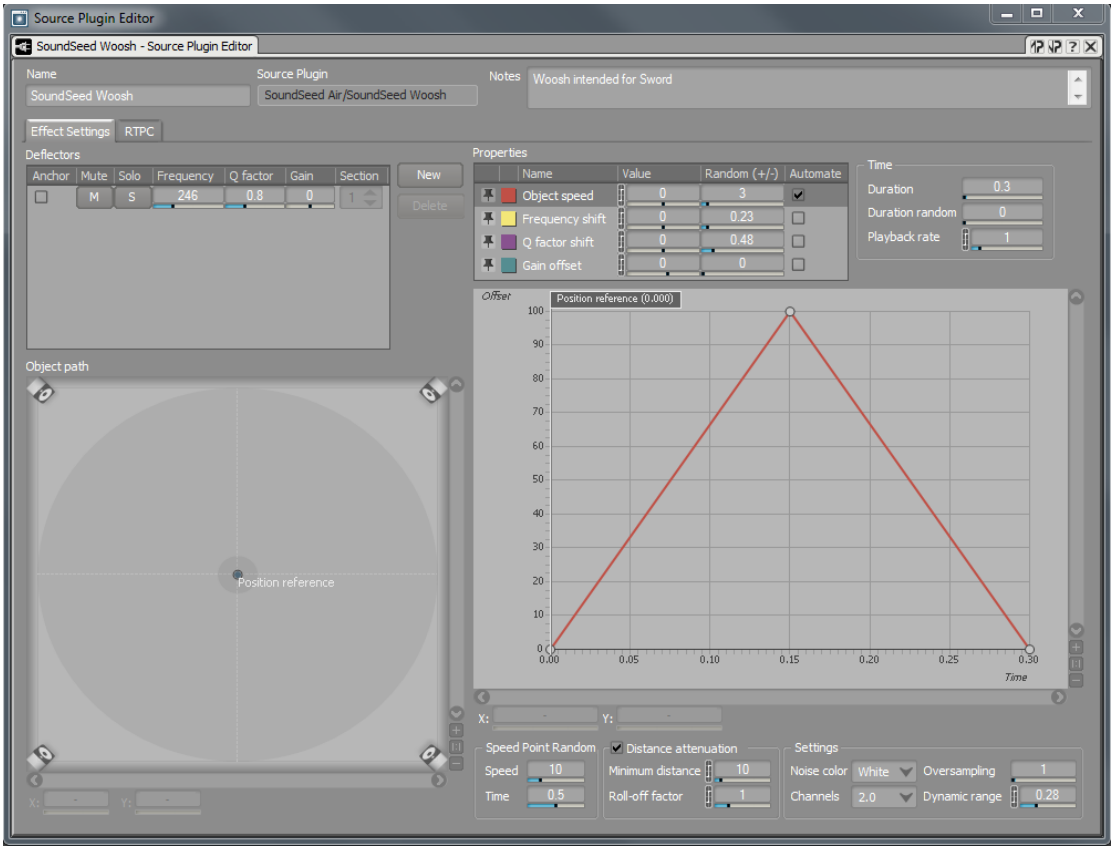


ソースプラグインWhooshを追加するためのAdd Source（>>）ボタン

Source Plugin Editor画面は、Contents Editor画面のプラグイン アイコンをダブルクリックすると開きます。



ソースWhooshのプラグインアイコン

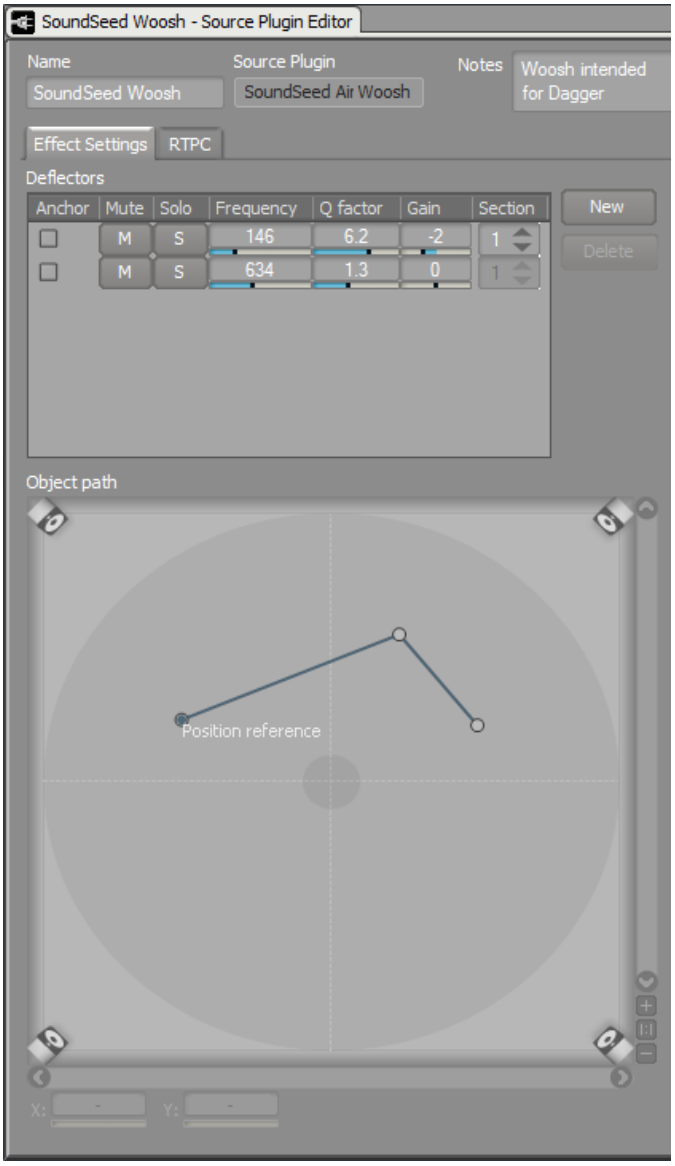


Source Plugin Editor画面でWhooshサウンドをオーサリング

Whooshのサウンドは、ある物体を表すディフレクタ（単数または複数）とその軌跡の組み合わせで決まります。ディフレクタ設定で物体の形状を決め、Anchor（錨）の設定で物体が回転するかどうかを決めます。

- Frequency（周波数）：物体の大きさを表す。
- Q factor（Q係数）：ディフレクタの表面素材をシミュレーションする。（高いQ値は均一な表面/低いQ値は丸い表面や不均一な表面）
- Gain（ゲイン）：振幅幅をコントロールする。

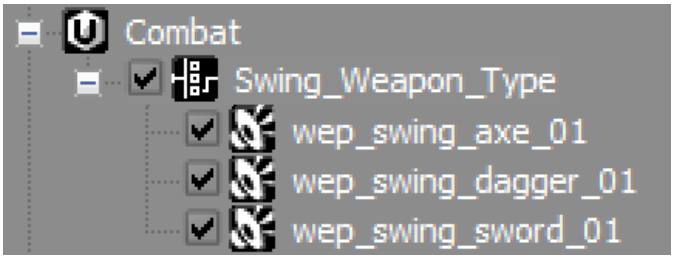
Object path（物体の軌跡）で軌跡を示すポイントを設定し、空間を移動する方向を決めます。



ディフレクタのプロパティとObject path（軌跡）

物体が移動する軌跡を描くためのポイントは無制限に設定でき、また軌跡は、Settings画面で設定したスピーカーの配置やチャンネル数に合わせて適切にパンします。

今回のウェポンのSwing（スイング）には、2チャンネルの配置と、プレイヤーの前で左から右に円弧を描く軌跡を組み合わせて使います。Wooshの例を分かりやすくするために、“Sword”（剣）、“Axe”（斧）、“Dagger”（短剣）の3種類のウェポンを設定します。

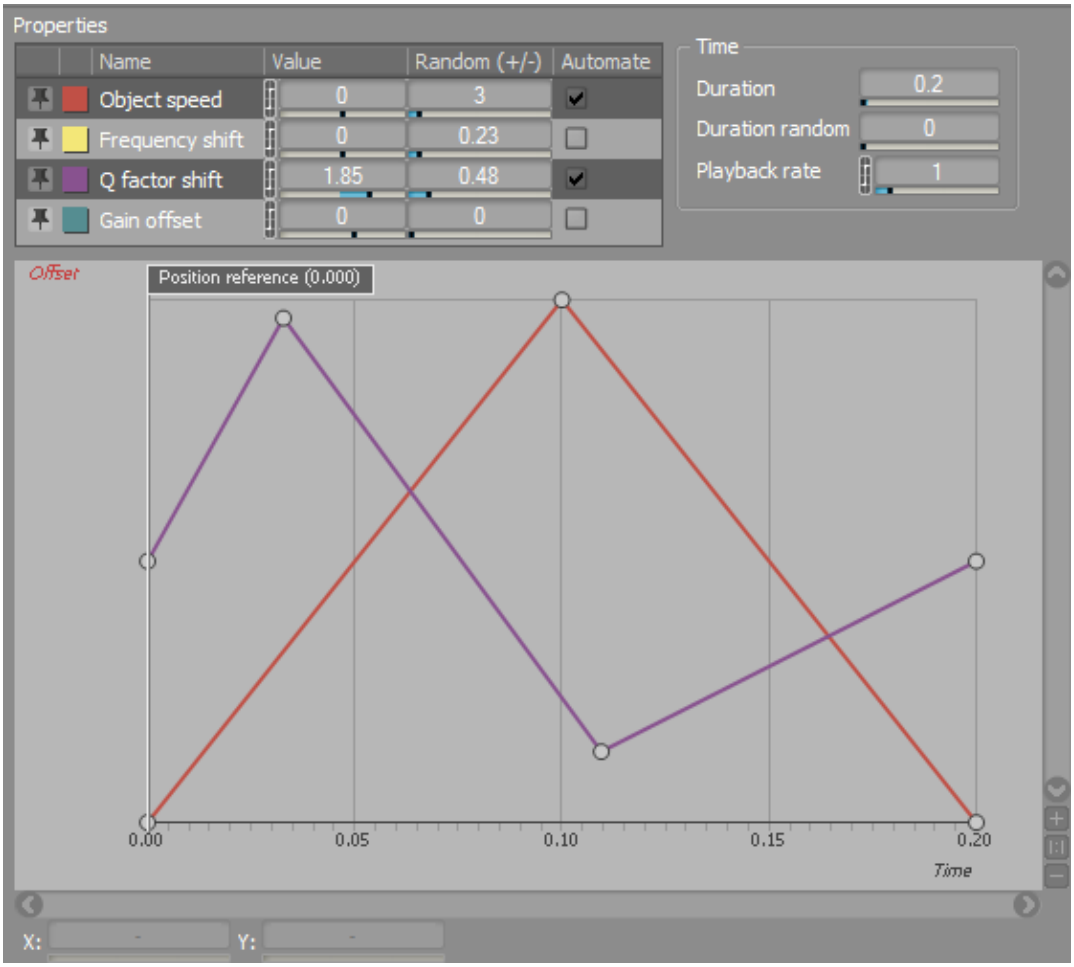


ウェポンの“Swing”音として用意した、3つのWhooshサウンド

Designer Note

ウェポンが空を切る短い音以外にも、軌跡を円形にして時間設定を長くすれば、スペシャルエフェクトを表現する動作サウンドとして利用できます。生成されるサウンドなので、毎回異なるシミュレーションやサウンドとなります。例えば、魔法をかけるエフェクトの時にランダムなWhooshエレメントをマルチチャンネルで再生すれば、プレイヤーの周りを渦巻くように音が降る効果が、メモリに全く負担をかけずに実現します。

SoundSeed Windの設定と同様に、オブジェクトの動きをコントロールするプロパティの設定は、固定値、ランダム化、またはRTPCに結びついたパラメータから選べます。自動的に作られるカーブをプロパティ設定に利用すれば、「ヒュー」というWhoosh音に動きを追加したり、設定した範囲内でのランダム性が可能になり、サウンドを立体化するのに役立ちます。



WhooshのProperties（プロパティ）、Automate（自動化）されたカーブ、Time（時間）の設定

サウンド生成の機能を利用することでメモリを節約できるのは当然のメリットとして、プラグインのSoundSeedシリーズは驚くほどクリエイティブになれる可能性を秘めています。標準的なサウンドデザインのツールをゲームオーディオのオーサリングアプリケーションに組み込むことで、ランタイムに合成音を生成してデザインと実際の実装との間のギャップを埋めることができます。さらにゲームパラメータを使えばプロパティをリアルタイムで変化できるため、ダイナミックな柔軟性と効率的なメモリー使用という2つのメリットが同時に得られます。

複数のウェポンスイング音を1つのスイングサウンドSFXの中で納めた結果、Whooshの機能を利用すれば、大量のコンテンツのバリエーションを準備する必要がなくなります。

インパクトについて

ウェポン作りに取りかかったところで、インパクトサウンドと悪者の頭を殴るアクションなどの詳細を見ていきます。物体同士の衝撃は、複合サウンド（インパクトがもたらす心理的な意図を伝達する手段）を採用するか、マルチレイヤーのシステム（リアルなサウンドを再現する手段）を作成することが一般的です。方向性を決める時は、ゲームの種類とプレイヤーのスタイルや期待をサポートする最良の方法がどれかを検討します。

最小限のバリエーションがあれば充分なシステムでは、ランダム再生されるいくつかのオーディオファイルをインパクト用に用意するだけで、全てのウェポンと全ての表面素材の表現に対応できるかもしれません。少し改良して、表面素材に関係なくウェポンの種類によってインパクトサウンドを変えることもできます。さらに改良して、ウェポンの種類によって、そしてゲームに存在する各種の表面素材に合わせて、異なるインパクトサウンドを使う、つまり、剣のインパクトが岩に当たる時と土に当たる時で異なり、また斧が木に当たる時と異なるように設定します。

物体同士の接触を抽象的に表現する場合でも、よりリアルに感じるモデルを取り入れる場合でも、Wwiseオーサリングアプリケーションはニーズに応じて対応します。

ウェポンタイプの定義

ランダムでバリエーション豊かな効果を求め続けるのであれば、前例の足音や動作と同じようにスイッチを使いこなします。今回はキャラクターやステップの種類ではなく、ウェポンタイプに着目し、これまで以上に多くの表面素材と組み合わせてさらに細部にこだわります。

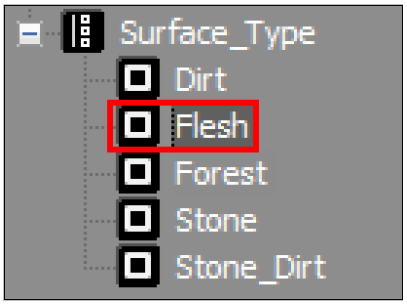
スイッチグループ“Weapon_Type”と、その下のスイッチ

スイッチグループ“Weapon_Type”は、プレイヤーまたはNPCが振り回すウェポンの種類で代わり、各タイプに異なるサウンドオブジェクトをアサインします。ウェポンタイプ用のスイッチグループが出来たところで、まず各種スイングを親となるスイッチコンテナに入れ、スイッチグループを設定、オブジェクトをアサインして、前節で作成したスイングを実装します。

スイングサウンドがアサインされたスイッチグループ“Weapon Type”

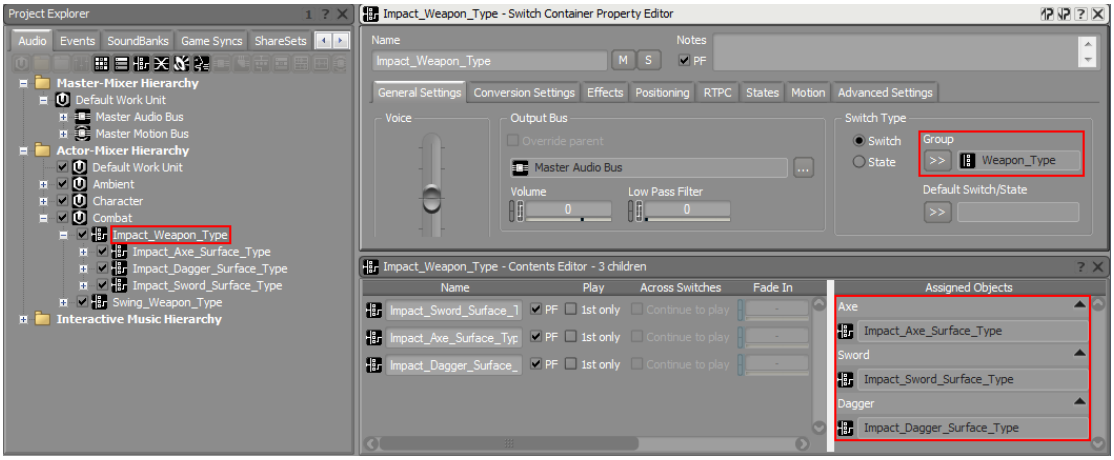
ウェポンインパクトのシステム

ヒーローが無数のバトルで戦う過程で、相手の刃に当たってしまうこともあるでしょう。インパクトシステムの細部にこだわるために、“Surface”（表面）素材のスイッチグループに“Flesh”（皮膚）を追加します。

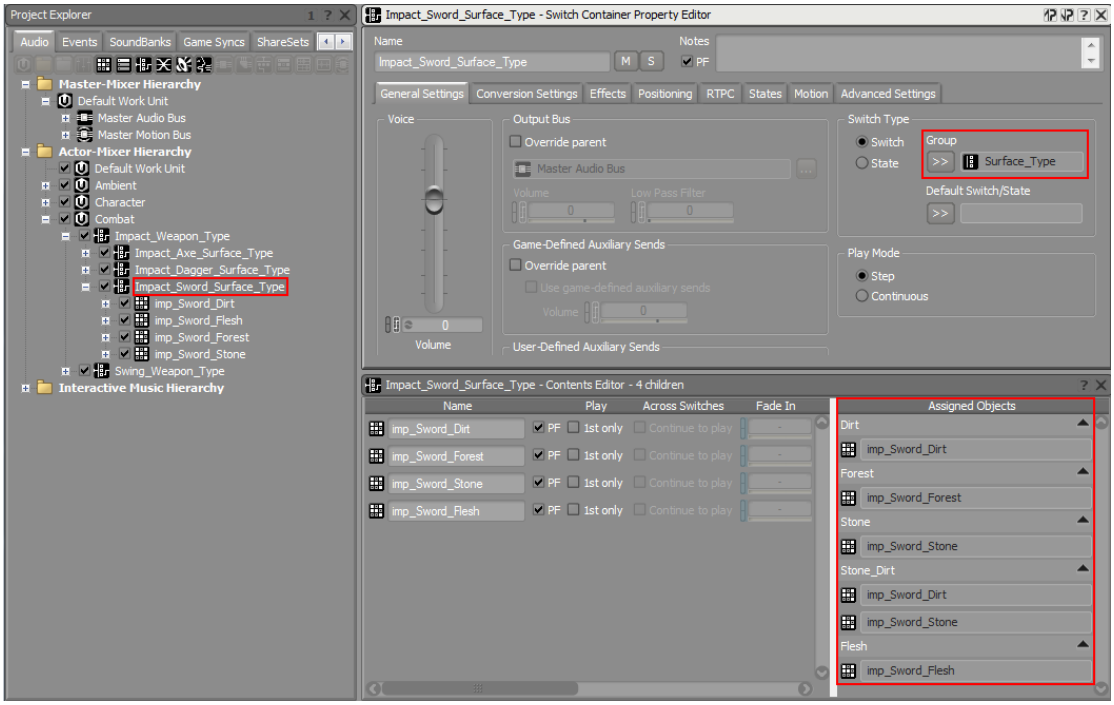


新しい“Surface Type”（表面素材）のスイッチ

ゲーム中の各種スイッチを定義し素材の設定が完了すると、要求されたゲームオブジェクトのプロパティに従いスイッチが作動し、ウェポンと表面素材の組み合わせに該当する正しいサウンドを再生します。



アサインされた“Weapon Type”のオブジェクト



アサインされた“Surface Type”のオブジェクト

今日のゲームで表面素材は増え続ける一方なので、メモリの許す限りサウンドの細部にこだわることができます。前述したテクニックを利用してスイッチ間で複数の表面素材をシェアすれば（第2.2章 キャラクター）、コンテンツを拡張すると同時にディテールを増やすことができます。よくできたサウンドデザインと効率的なシステムがあれば、実装方法を工夫して様々な制約を乗り越えることが可能です。

プレイヤーとNPCの減衰設定

これでゲームに出てくる全てのウェポンに対応するインパクトサウンドのスイッチシステムが完成したので、次に、距離が遠くなるにつれてサウンドが聞こえなくなる減衰の設定と登録が重要な作業となります。アンビエント音の説明で単純な減衰設定について触れましたが、戦闘サウンドでも減衰が大いに役立ちます。

厳戒態勢では

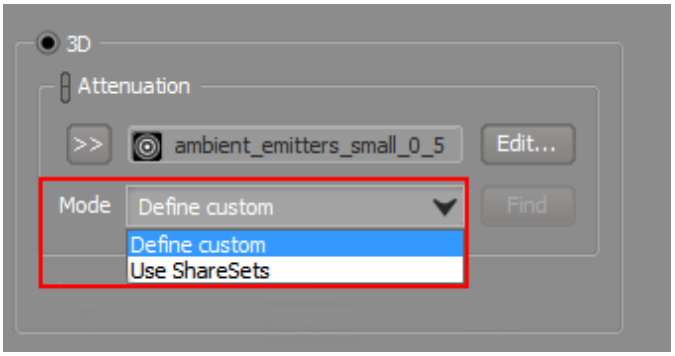
ゲームでバトルが始まる前から、敵のアイドル音や足音を聞かせることで攻撃の範囲外にいる時から、近くにいる敵の存在をプレイヤーに警告することができます。サウンドを利用して危険を知らせることで、プレイヤーは警戒心を高めると同時に、差し迫る争いに向けた準備ができます。プレイヤーは敵がアタックするまでの距離を知り、減衰による音の変化で闘争が始まる前から相手の動きを聞き取ることができます。例えば、距離15ユニット以内にいるプレイヤーに挑んでくる設定であるとしましょう。敵の動作サウンドの最大減衰の距離を20ユニットまで延ばし、近づくバトルに対してプレイヤーに充分な装備時間を与えます。

リスナーに関する配慮

多くの場合、リスナーの場所はプレイヤーの位置、またはカメラの位置、または両者間の（パラメータなどで）調整された位置にあります。リスナーがNPCに接近するとサウンドが変化し、減衰設定でサウンドのフォールオフが決まります。このシナリオは3Dゲームで一般的に使われているもので、ゲームオーディオでは広く受け入れられています。しかしプレイヤーのサウンドの場合、ゲーム内のリスナーの位置に特別な配慮が必要となることがあります。

状況によっては、プレイヤーの上で2D再生するのが有利なこともありますが（サラウンドの魔法エフェクトなど）、具体的に、ゲーム環境の中の別の場所で発生中のアクションを見せるために、カメラがワールドの中を移動していくようなごく一般的なシナリオの場合、好ましくない結果に陥ることもありです。この時プレイヤーの位置で再生されるサウンドが2Dに設定されていると、画面にプレイヤーがいなくなってもサウンドだけが聞こえてしまいます。これを避けるための対策として、プレイヤーの減衰設定も前例のNPCの減衰と同じように3Dサウンドとするのが安全です。

最初はいくつかの共通する減衰設定を使い回したとしても、後から階層の好きなレベルに独自の減衰シェアセットを設定することができます。また、1つのオブジェクトだけに適用されるカスタム設定の減衰も作成できます。サウンドオブジェクトの減衰設定のモードをUse ShareSets（シェアセット使用）からDefine Custom（カスタム設定）に変えれば、他のサウンドオブジェクトとシェアしない（できない）減衰設定が適用されます。



減衰のシェアセットから特定サウンドオブジェクト用のカスタムプロパティを作成

減衰のカスタム設定を例外として活用すれば良いソリューションとなりますが、複数のシェアセットを作成して一カ所で管理する方が、コントロール性も透明性も優れています。

ゲーム内でサウンドがどのように伝搬するかを感覚的に覚えると、新しいサウンドを作った時に直感的に減衰設定のシェアセットが使えるようになります。上図に定義された柔軟な減衰設定が準備できれば、ゲームに適した距離減衰モデルの作成が可能です。

戦闘のまとめ

本章では、スイングやインパクトをウェポンごとに設定する方法を説明しました。その過程で、SoundSeed Whooshの機能を使い、オーディオファイルのバリエーションを用意する従来のモデルを、強力な合成ツールセットで置き換える方法を確認しました。さらに、減衰設定やリスナーの配置を工夫して両者を組み合わせることで、ゲームプレイを向上させ伝搬モデルを構築しました。

本章では、以下を説明しました。

- 様々なウェポンタイプのサウンドをデザインする時にクリエイティブ面で必要な決定事項と、その実施の方法
- サウンドコンテンツのバリエーションに依存するのではなく、SoundSeed Whooshを利用してウェポンのスイング音を生成する方法
- インパクトサウンドの様々な実装テクニック

プロセスの説明

- SoundSeed Whooshを使用して、様々なウェポンタイプ用のスイング音の作成
- ウェポンタイプに応じてウェポンのスイング音を選ぶスイッチシステムの作成
- ウェポンタイプと表面素材に応じて、ウェポンのインパクト音を選ぶスイッチシステムの作成

詳細設定の説明

- 素材で変わるインパクト音の複雑な仕組みへの対応
- プレイヤーとNPCの減衰設定
- リスナー位置を選ぶ時の注意事項

オブジェクトの作成方法

- SoundSeed Whooshでオーサリングした剣、斧、短剣のスイングサウンドSFXオブジェクト
- ウェポンタイプに対応したスイングのスイッチシステム
- ウェポンタイプと表面素材に応じてインパクトを判断するウェポンインパクトのスイッチシステム

参考ドキュメントとチュートリアル

Video Tutorial - SoundSeed Introduction

Video Tutorial - SoundSeed Air Whoosh Overview

第4章 マジックの表現

概要

怪物の攻撃を魔力で防ぐのも、暗黒のウィザードに襲撃されるのも、サキュバスを暗い王国から誘い出すのも、はるか昔の時代より、マジックのルールは魔術師から魔術師へと世代を超えて伝えられてきました。音の魔術を展開して、強力な破壊力を秘めたサウンドを解き放つため、Wwise のオーサリングアプリケーションを使ってサウンドで様々な試みを重ねてみましょう。

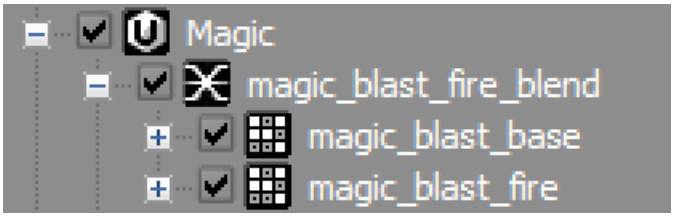
本章では、以下の操作を説明します。

- ブレンドコンテナを使ったサミングとレイヤリング
- エフェクトの使用
- リアルタイムパラメータコントロール（Real-time Parameter Control: RTPC）の使用
- 距離に基づいたサウンドの遠近システムへのブレンドトラックの活用
- モジュレーターエンベロープとLFOを利用して、サウンドのプロパティを多様化させる
- Wwise Synth Oneを使って合成的要素を作成する

他にない魔法のような効果へのアプローチにはまずインスピレーションが必要です。そこから、サウンドファイルと合成が、結果的な効果のサウンドパレットを定義することになります。個々のサウンド要素の多様性もしくはサウンドのレイヤーを利用、またはそれらのブレンドコンテナで再び組み合わせることで、あなたの作り出すエフェクトにおける創造性の度合いを高めることができます。ゲームからのパラメーターを利用して、ダイナミックな変化を加えましょう。

ブレンドコンテナを使ったサミングとレイヤリング

アンビエントの章で説明した通り、ブレンドコンテナは中身を全て同時に再生します。サウンドエレメントの個々のバリエーションやサウンドデザインのレイヤーを個別に扱い、ブレンドコンテナの中で組み合わせを変えることで、驚くような新エフェクトや混合音が生まれ、素材に広がりをもたらします。特にランダム化されたプロパティやアクターミキサー階層どのレベルでも利用可能な追加エフェクトと一緒に使うとさらに効果的です。



マジックブレンドブレンドコンテナは、複数のランダムコンテナで構成されています

距離に基づくブレンドトラックの作成

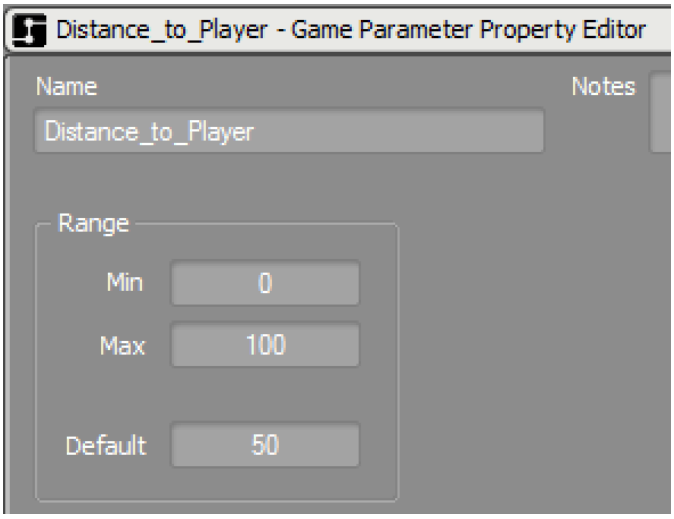
ブレンドトラックのもう1つの使い方として、距離によって変わる複雑なブレンドを作成して聞こえ方を変化させる方法があり、簡単に言えば、距離に応じてサウンドコンテンツを変えるのです。このテクニックは、距離に幅があるウェポンや、動いている物体、そしてマジックのエフェクトを表現する際に非常に効果的です。

ブレンドコンテナを使い、マジックインパクトのブラスト（爆発）がプレイヤーの近くにある時と、遠くの視点から聞こえる時とでサウンドを変えます。マジックブラストのイベントをトリガーすると、プレイヤーからブラストしたゲームオブジェクトまでの距離に基づいて、1つの視点（または複数の視点の組み合わせ）のサウンドが再生されます。近い距離では高周波のディテールを増やして危機感を表現し、遠くではややローパスフィルターをかけたリバーブのある音とします。

マジックエフェクトは多くの場合、ゲーム環境において粒子エフェクトやビジュアルエフェクトが関係します。粒子エフェクトは様々なテクニックから成り立ち、最終的なビジュアルエフェクトとして成立しますが、通常は原点があり、これがエフェクトの中心点となって、ゲームオブジェクトとして3D空間を移動するので、サウンドを添付することが可能です。サウンドからプレイヤーまでの距離情報をエフェクトの一部として利用することで、サウンドが移動する感覚を生み出し、位置に関する編集要素を付加してダイナミックなエフェクトをさらにドラマティックにします。

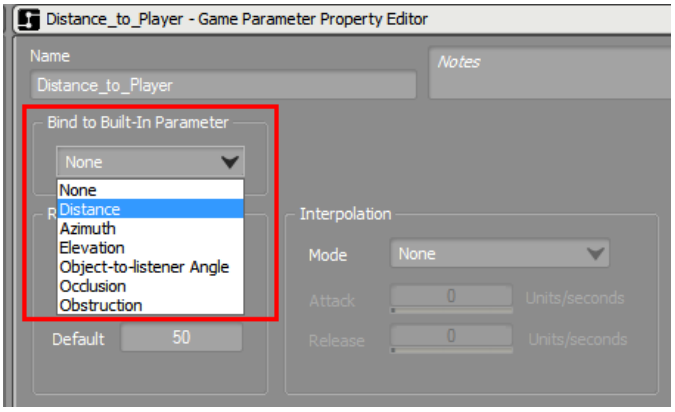
ゲームパラメータの設定

ゲームオブジェクトとプレイヤーの間の距離情報をゲームエンジンから受け取り、プロジェクト全体を通して“Distance_to_Player”（プレイヤーまでの距離）という新規ゲームパラメータとして利用できます。



ゲームパラメータ“Distance_to_Player”の設定

Wwiseオーディオエンジンから直接距離の情報やその他の値を直接得ることは簡単です。"Bind to Built-In-Parameter" ドロップダウンメニューを使って、オーディオエンジン内で計算されたビルトイン値のリストからリスナーとゲームオブジェクトの間の距離を選んでビルトイン値として登録できます。



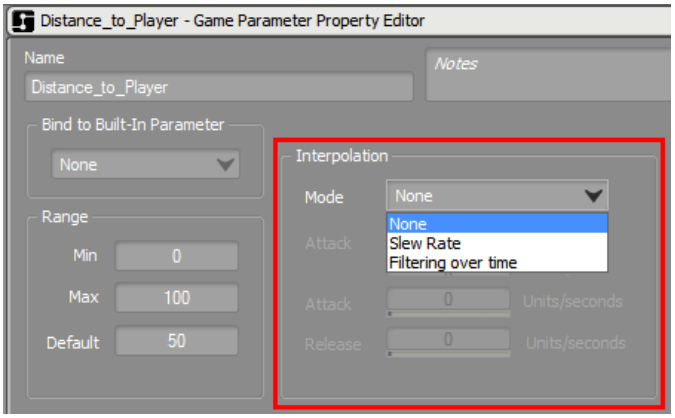
ビルトイン距離値を"Distance_to_Player"ゲームパラメータにバインドする

オーディオエンジンが次のビルトインパラメータを計算し、ゲームパラメータにバインドすることができます。

- Distance - 距離。ゲームオブジェクトまでの距離。
- Azimuth - 方位。水平の角度。
- Elevation - 高度。垂直の角度。
- Object-to-listener Angle - オブジェクトとリスナーとの角度。オブジェクトの向きとリスナーの間の角度。
- Occlusion - オクルージョン。そのゲームで、ゲームオブジェクトに設定された値。
- Obstruction - オブストラクション。そのゲームで、ゲームオブジェクトに設定された値。

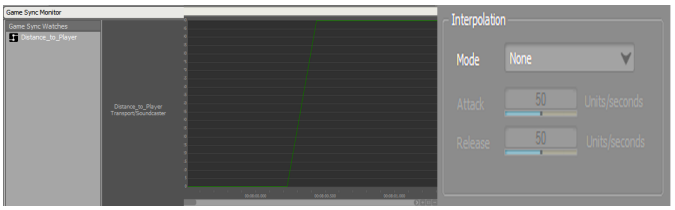
補間を利用して、オーディオエンジンから送られる値に対して、所謂「スムーズにする」効果を適用することもできます。ターゲット値が上昇する（Attack）または下降（Release）する際、モード（None（なし）以外）を利用して、レート（Units（単位）による）または時間（秒）を修正することができます。Interpolation（補間）モードは、ゲームエンジンからの値が「ジャンプ」したり、「ステッピング」が早くなり、たとえばボイスの音量が急に大きくなるなどの現象を防ぎます。補間プロパティを使うと、ゲームもしくは Wwise オーディオエンジンから送られるゲームパラメータ値の間の移行を制御することができます。

この例では、"Distance_to_Player" ゲームパラメータの動作が、Game Parameter Property Editorで利用できる補間プロパティを使って修正されています。



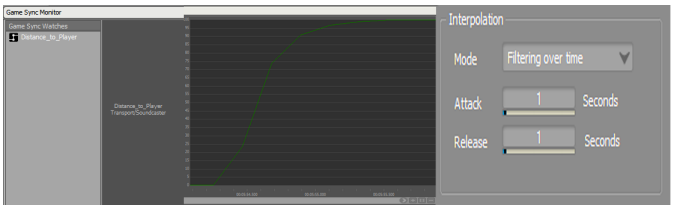
ゲームパラメータ補間モードの設定

Interpolation（補間）モードは、Mode ドロップダウンリストを使って設定することができ、これには次のモードが含まれています。



- **Interpolation Mode - None:**なし。ターゲット値に即移行する。

- **Interpolation Mode - Slew Rate:**スルーレート。ゲームパラメータを特定のAttackおよびReleaseレートの変位率に限定する。 （Unitsごと）



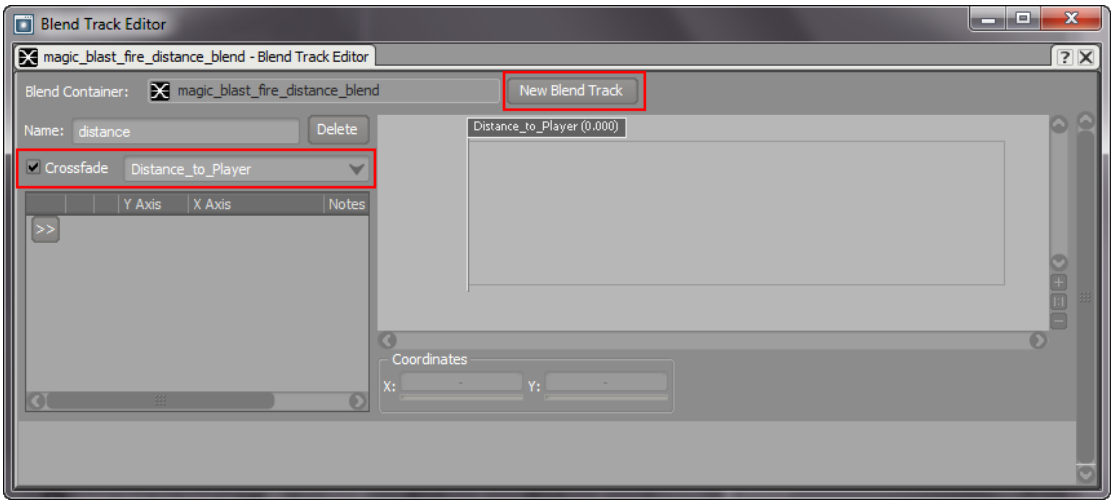
- **Interpolation Mode - Filtering Over time:** 時間によるフィルタリング。ゲームパラメータを特定のAttack/Release時間内のターゲット値99.5%以内を目標にフィルターする。（秒）

"Distance_to_Player" ゲームパラメータのInterpolation モードをAttack 50 Unit、Release 50 Unitの「スルーレート」で設定することで、望ましい効果を作り出します。

ブレンドトラックのコンテナ間に Crossfade（クロスフェード）機能を適用する

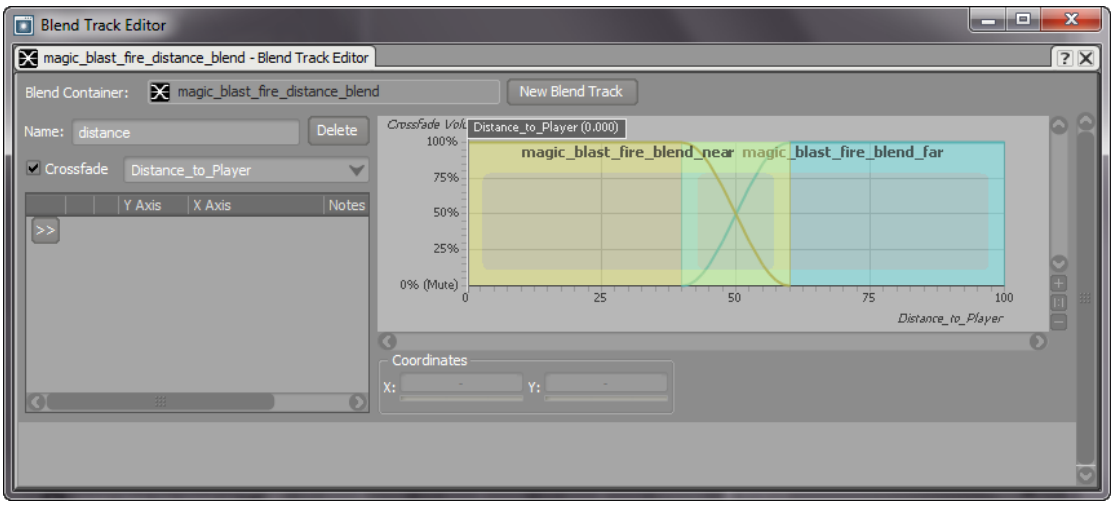
これでオブジェクトとプレイヤーの間の距離をトラッキングできますので、この情報を、ブレンドコンテナ“magic_blast_fire_distance_blend”の中にあるブレンドトラックの様々なコンテナに対して、コンテナ間のクロスフェードを作動させるために使います。

アンビエントシステムのために作成したブレンドコンテナと同様に、新規ゲームパラメータ“Distance_to_Player”に基づいたクロスフェード機能を利用した新規ブレンドトラックを作成します。



ゲームパラメータ“Distance_to_Player”に基づいたクロスフェードを採用した、新規ブレンドトラックの作成

次に距離のブレンドトラックにあるAssigned Objectsエリアで、ランダムコンテナを追加します。前述の通り、Assigned Objectsエリアでコンテナを並べた順番が、ブレンドトラック上の順番として反映されます。



正しい順番でブレンドトラックに並べられたコンテナ

ゲームから情報が正しく伝達された後、ゲームパラメータ“Distance_to_Player”でコンテナ間のクロスフェード機能をコントロールし、ゲームパラメータのカーソルの下にくるコンテナ（複数可）だけが再生されます。

- ゲームオブジェクトによって再生されるイベント"magic blast"が、“Distance_to_Player”（プレイヤーからの距離）にして「0〜25」離れている時は、“magic_blast_fire_blend_near”が単独で聞こえる。
- ゲームオブジェクトによって再生されるイベント“magic blast”が、“Distance_to_Player”（プレイヤーからの距離）にして「25〜75」離れている時は、両方の視点のブレンドが聞こえる。
- ゲームオブジェクトによって再生されるイベント“magic blast”が、“Distance_to_Player”（プレイヤーからの距離）にして「75〜100」離れている時は、“magic_blast_fire_blend_far”が単独で聞こえる。

Designer Note

距離情報を使ったブレンド操作に複数の減衰設定を使うことでも可能ですが、ブレンドトラック内で管理するとワークフロー上のメリットがあります。

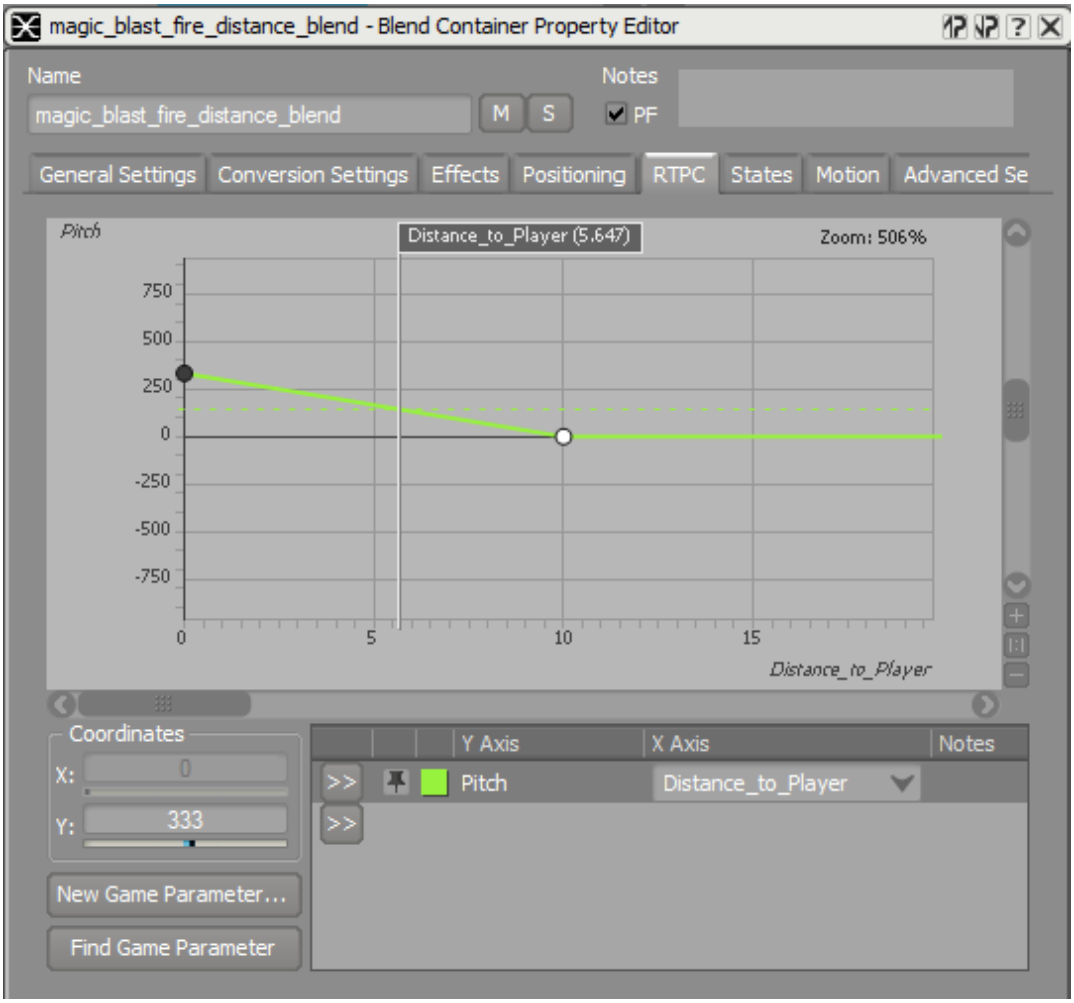
本節のまとめ

これで、場所とゲームオブジェクトからプレイヤーまでの距離で変わるオーディオコンテンツを使ったマジックブラストを実装できました。このテクニックは、コンテンツのレベルにおいて、視点を変えたり精密に表現する必要がある場合に威力を発揮します。出てくるサウンドのトーンを変更するために単純なフィルターをかけることもできますが、コンテンツ自体を適切にデザインすることでより細かいレベルまでこだわりをみせることができます。

リアルタイムパラメータコントロール（RTPC）

マジックエフェクトをさらにダイナミックにするには、アクターミキサー階層のサウンドオブジェクトやミキサーマスター階層のオーディオバスに対して、RTPC（Real Time Paramenter Control / リアルタイム パラメータコントロール）を適用します。RTPCタブはサウンドオブジェクトのProperty Editor画面から表示し、またセレクターボタンを使って新しい項目を追加できます。

ブレンドコンテナ“magic_blast_fire_distance_blend”でPitch（ピッチ）の増減を設定するために、オブジェクトのRTPCとしてゲームパラメータ“Distance_to_Player”（プレイヤーまでの距離）を使います。距離の変化を示すカーブを使ってマジックブラストのサウンドをダイナミックに変えるために、マジックブラストからプレイヤーまでの距離によって、ピッチを変化させます。これに似た設定は、ブレンドトラック内のRTPCを使用することでも可能です。



Pitchの量をコントロールするために、パラメータ“Distance_to_Player”を使ったRTPC

ゲームからの情報に基づいてサウンドの変化させるダイナミックな可能性を開くことで、RTPCはさらにサウンドオブジェクトのプロパティを変化させます。この例では、マジックブラストのサウンドをコントロールしてプレイヤーに近づくとピッチが高まるようにしています。その結果、マジックブラストのサウンドが接近する様子がピッチの変化で表現されます。

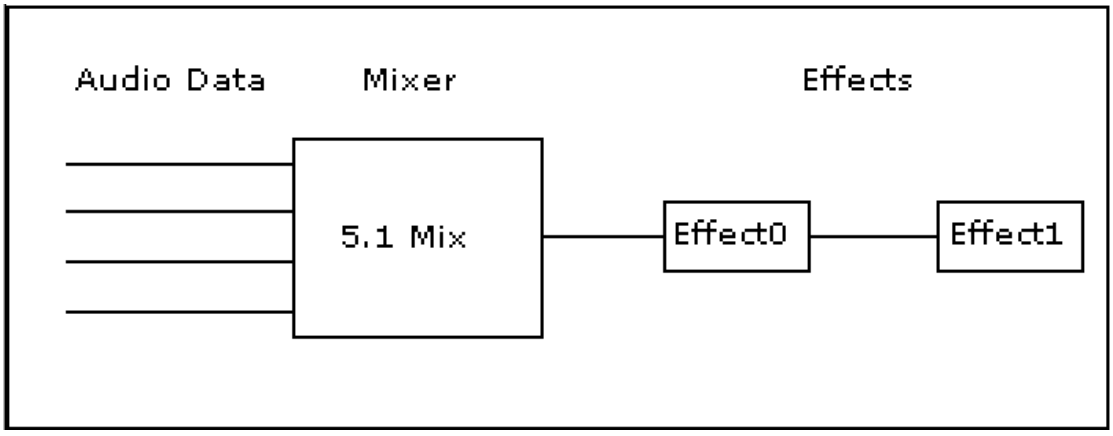
リアルタイムエフェクトの活用

ブレンドコンテナに対して使えるユニークな機能のほか、コンテンツをインポートした後の編集にWwiseに同梱されたDSPエフェクトのスイートを使えるので、オーサリングアプリケーションで行うサウンドデザインの幅がさらに広がります。

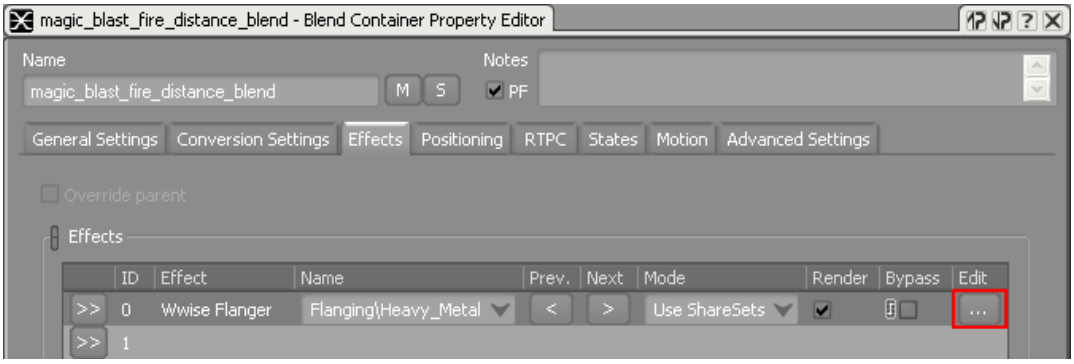
これらのエフェクトは、全てのオーディオオブジェクトのProperty Editor画面や、マスターミキサー階層の中で設定でき、サウンドを事前設定に基づいて変化させたり、ゲームパラメータを使ってランタイムにダイナミックに変化させることができます。

Designer Note

バスに対してエフェクトを適用すると、入力されるオーディオデータをサブミックスしてからエフェクトを適用します。複数エフェクトを適用する場合は、リストに表示された順序で適用します。

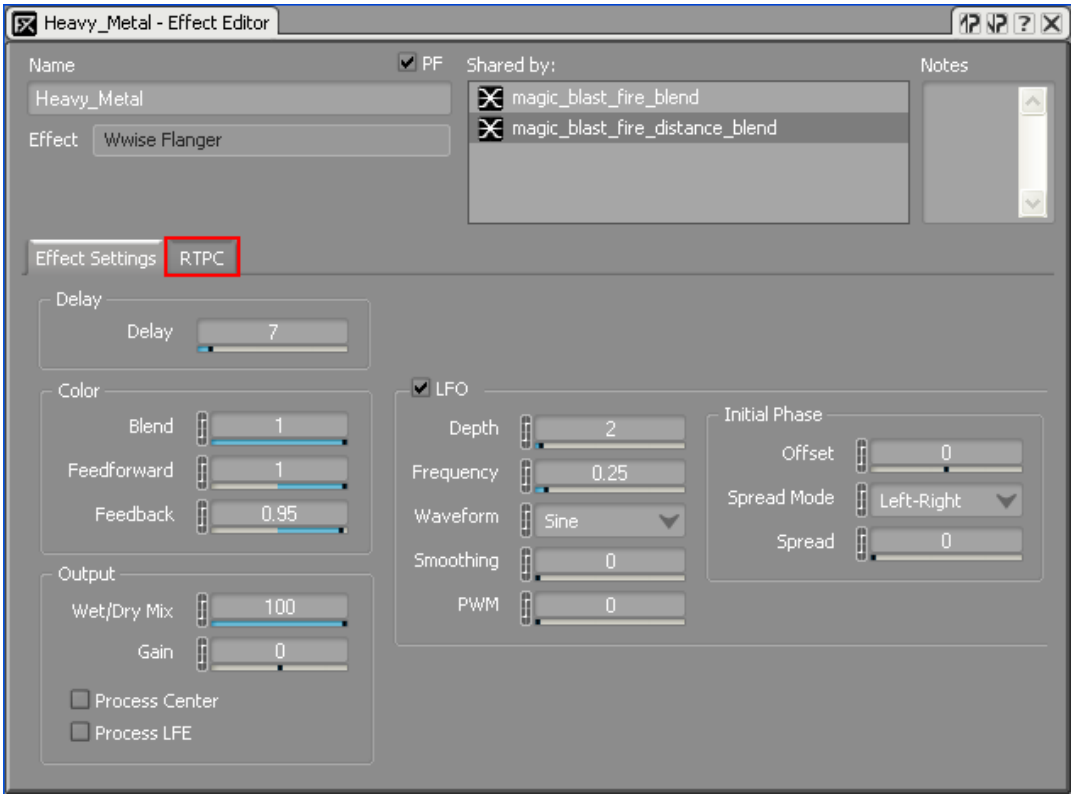


Effect Editor画面で有効になっているエフェクトのEditボタンをクリックすると、エフェクトのRTPCを表示。



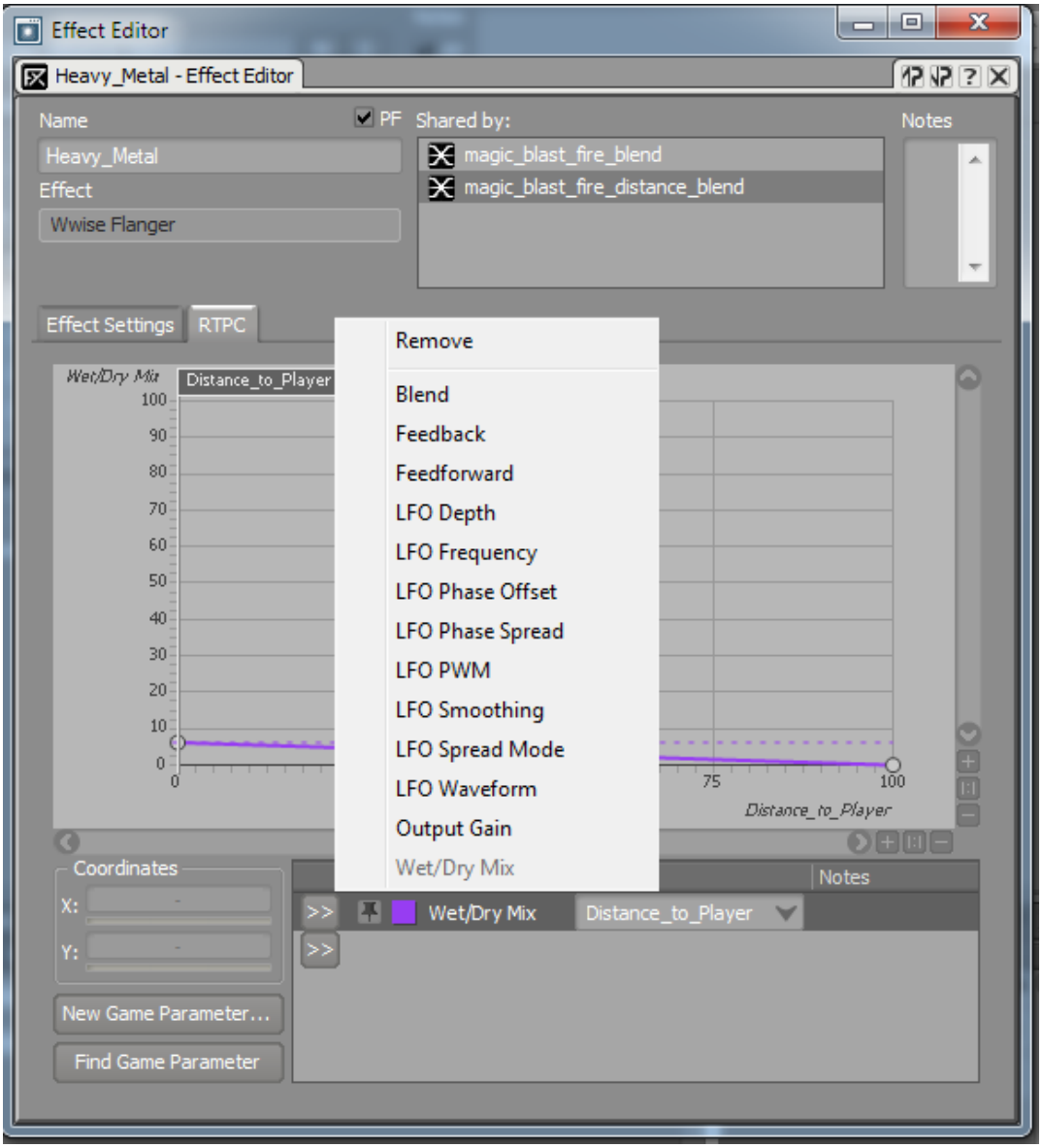
Effect Editor画面を開くためのEditボタン

Effect Editor画面でエフェクトの設定を全て確認でき、またRTPC機能を利用できるプロパティもここで編集できます。



Effects SettingsパネルとRTPC タブへのアクセス

RTPCタブでエフェクト設定とゲームパラメータを関連付けると、ゲームからの情報に基づくユニークなエフェクトを作り出すことができます



"Distance_to_Player"ゲームパラメータに追加されたRTPC設定

ゲームパラメータをDSPエフェクトの設定と組み合わせることで、ダイナミックに変化するコンテンツの全く新しい世界が開けます。ゲームパラメータとエフェクトを組み合わせると、以下のような使い方ができます。

- 川の水流サウンドに対して、遠くにいる時に限り、軽いフランジャーエフェクトを使い不気味に聞こえるようにする。
- プレイヤーが戦闘中に身体的な理由で死ぬ時、全ボーカルサウンドにステレオディレイをかける。
- エアコンの送風機のループに対して、ファンの回転数に応じて変化するトレモロを加える。

誰もがすぐに分るような、魔法をかける時の象徴的なサウンドをデザインする時も、自由な組み合わせでバリエーションを増やそうとする時も、RTPCやダイナミックDSPエフェクトの威力を最大限に活用すれば、インタラクティブゲームの世界でしか実現できないようなスペシャルエフェクトを作り出すことが可能です。

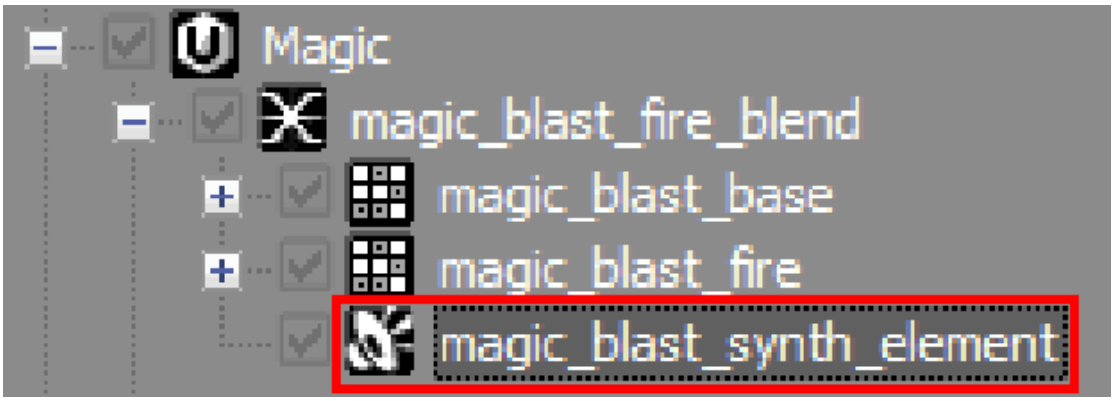
ダイナミックな合成を解き放つ

オーディオファイルを置き替える、効果を増大させる、または補足する他の手段として、 Wwise Synth One プラグインの利用があります。合成は、システムのメモリ（RAM）に頼らない柔軟性のあるダイナミックなオーディオプラットフォームを提供、つまりランタイムでCPUを使ってサウンドを生成します。おそらく、DAWのサウンドデザインの一部として合成を既に使用しているのではないでしょうか。Wwise Synth Oneの機能を使えば、合成がWwiseオーサリング環境での作業の一部になります。

ゲームパラメータ、MIDIメッセージ、モジュレーターLFO、モジュレーターエンベロープを、どのようなサウンドオブジェクト、オーディオバス、AuxバスRTPCの一部として追加することが可能であり、その後Wwise Synth One、SoundSeed Whoosh、SoundSeed Air、またはいずれのDSPプラグインのプロパティを修正するのに利用できます。ランタイムで合成を使ってサウンドを編集できるため、ゲームもしくはWwiseオーサリング環境内からのパラメータを利用してプロパティを変更するより大きな柔軟性が生まれます。

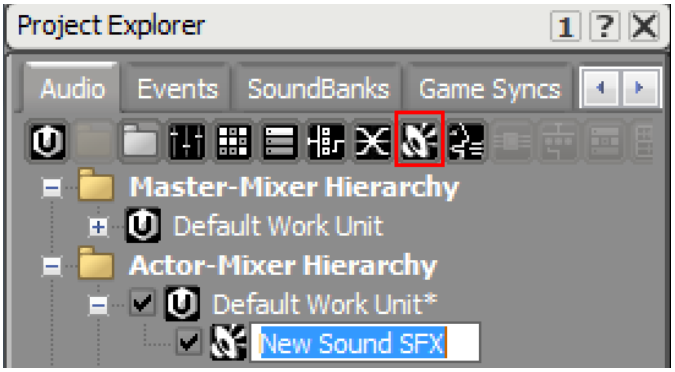
Wwise Synth One

まずは、第3節で作った"magic_blast_fire_blend"の一部として、Wwise Synth Oneプラグインを含む"magic_blast_synth_element"を作るところから始めます。



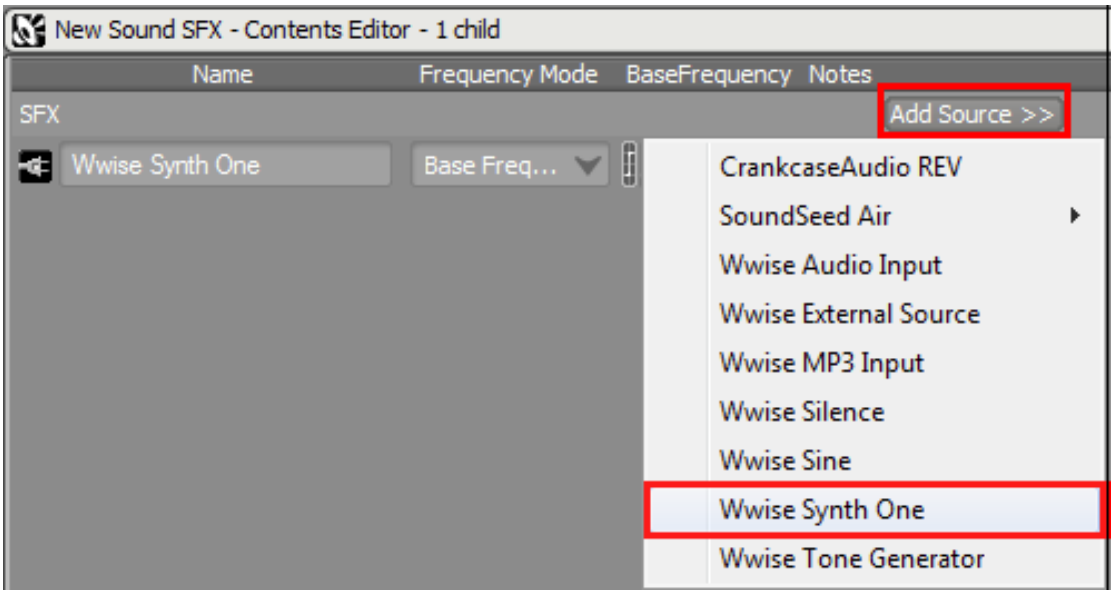
"magic_blast_fire_blend"にWwise Synth Oneプラグインを含むサウンドSFX

第1章のSilenceプラグインの追加と同様に、Wwise Synth Oneプラグインは、サウンド SFX（またはSound Voice: ボイス用サウンド）への入力ソースとして追加できます。最初に、Project Explorerツールバーのサウンドアイコンをクリックし、デフォルト ワークユニットにサウンドSFXオブジェクトを追加します。これで新しいサウンドSFXが作成されました。 また、コンテキストメニューやショートカットキーからサウンドSFXを作ることもできます。



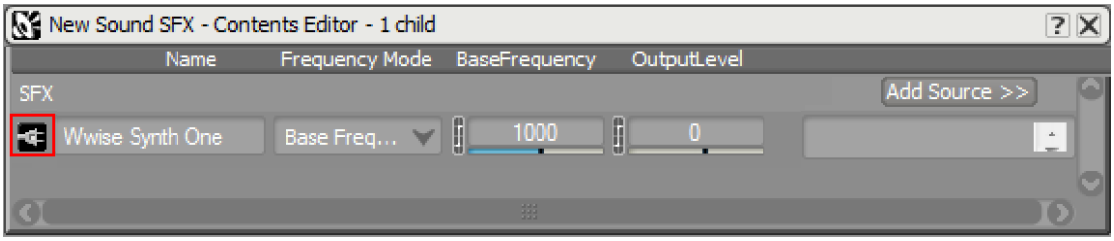
Project ExplorerツールバーからNew Sound SFX（新規サウンドSFX）を作成

ダブルクリックで新しいサウンドを選び、Contents Editor画面にあるAdd Sourceメニューから Wwise Synth Oneプラグインを追加します。

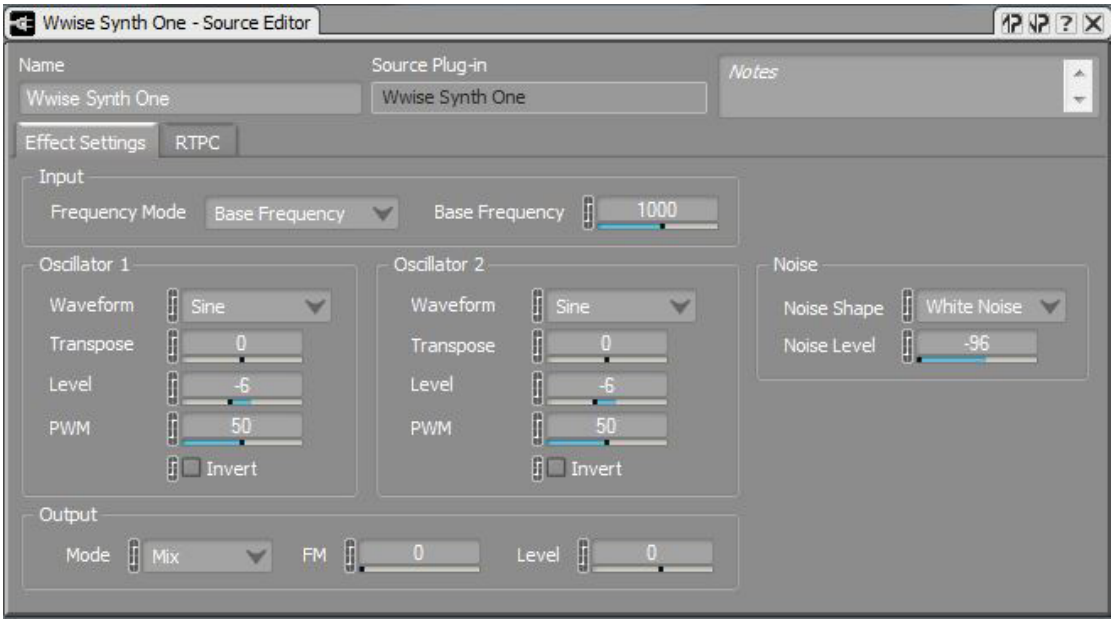


サウンドオブジェクトのWwise Synth Oneのソースプラグインを作成

Contents Editor画面にあるプラグインアイコンをダブルクリックしてSource Editor画面を表示します。



Wwise Synth One プラグインアイコン



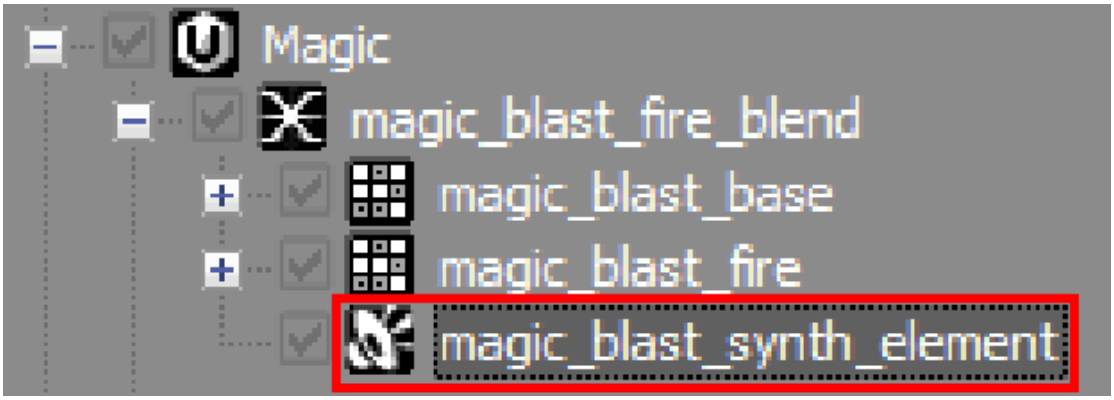
Wwise Synth One - Source Editor画面

Designer Note

Wwise Synth Oneプラグインの機能については、第8章のMIDIと合成を参照してください。

本章の前の部分で既に作成したオーディオファイルに追加して再生され"magic_blast_fire_blend"ブレンドコンテナに追加の合成的要素を作成します。この要素は、サウンドに合成的特徴を加え、Wwise Synth OneプラグインのOutput Level（出力レベル）プロパティにあるモジュレーターエンベロープを使ってコントロールします。 これに加え、モジュレーターエンベロープと同時にモジュレーターLFOと"Distance_to_Player" ゲームパラメータを使って、Wwise Synth Oneプラグインの様々なプロパティをコントロールします。

さあ、第3節で作成した"magic_blast_fire_blend" にWwise Synth Oneプラグインを含む"magic_blast_synth_element"を加えましょう。

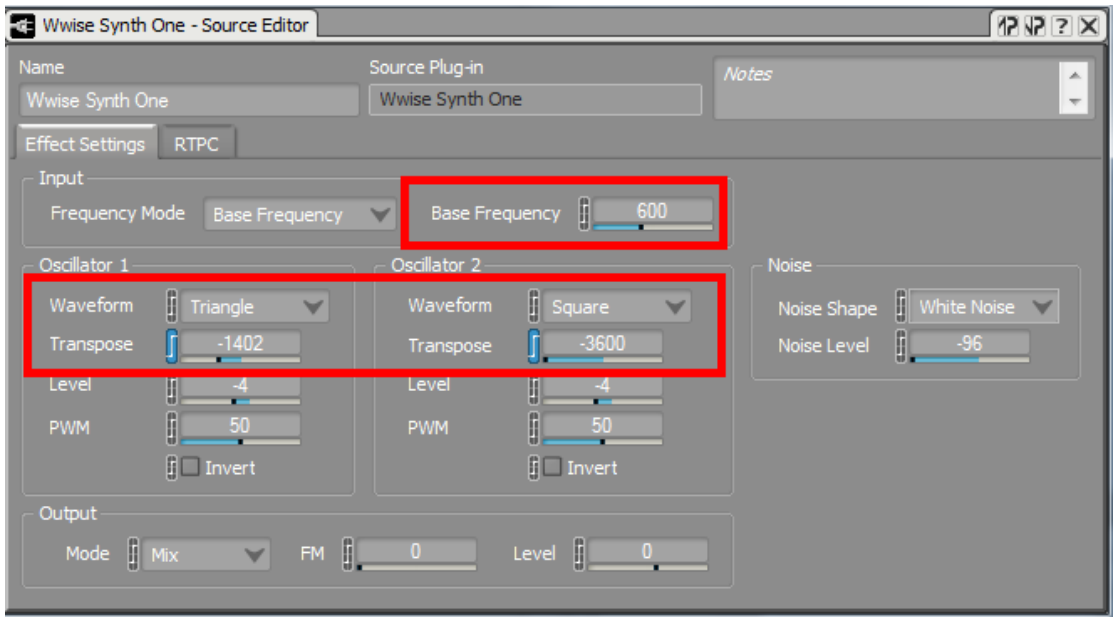


"magic_blast_fire_blend"にWwise Synth Oneプラグインを含むサウンドSFX

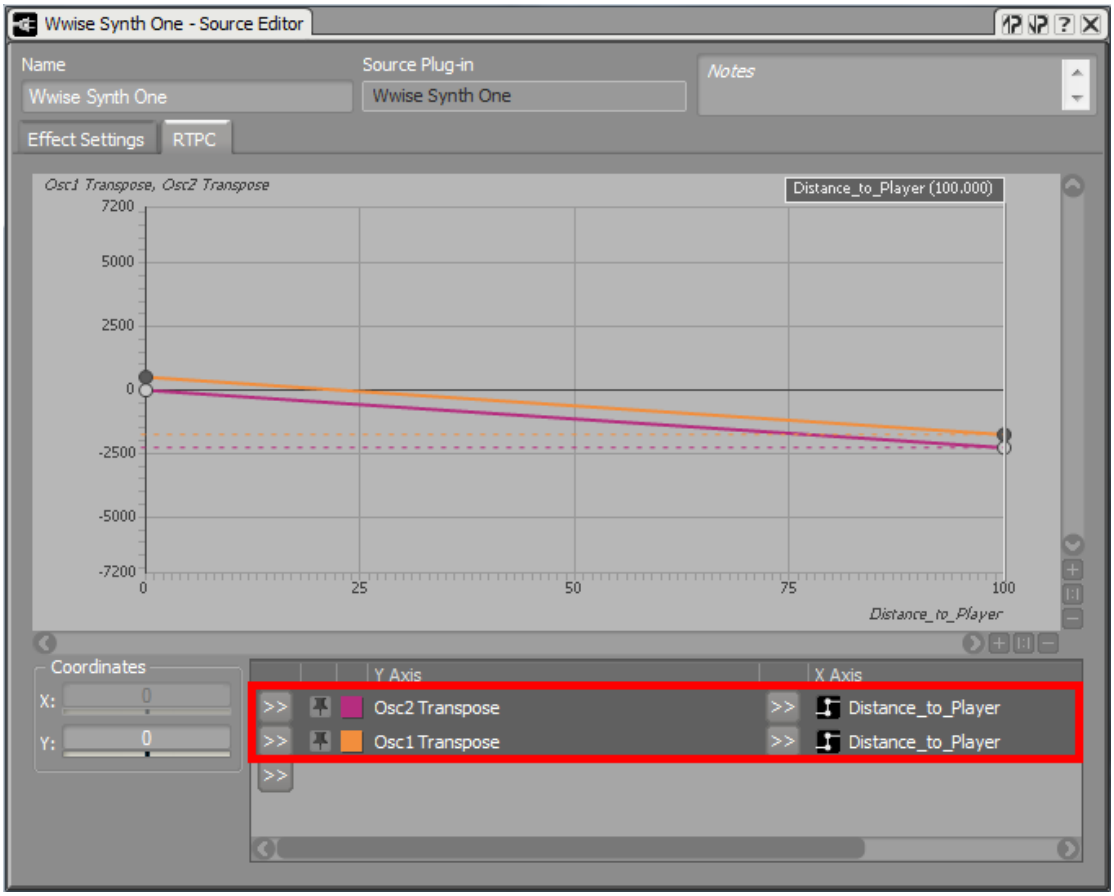
"magic_blast_synth_element" は低い基底周波数と三角形ならびに矩形の波形で始まります。"Distance_to_Player" ゲームパラメータを使って、各Oscillator（オシレーター）はさらにトランスポーズ、変化し、マジックブラストがプレイヤーに近づくにつれ、トランスポジションが上昇します。

Designer Note

ゲームパラメータに関する詳細は第1章のアンビエントの設定 - 昼と夜のサイクルの始動、ゲームパラメーターの設定を参照してください。



Wwise Synth One Base Frequency（基底周波数）およびWaveform （波形）プロパティ



Wwise Synth One Oscillator 1 & Oscillator 2 Transpose プロパティ RTPC

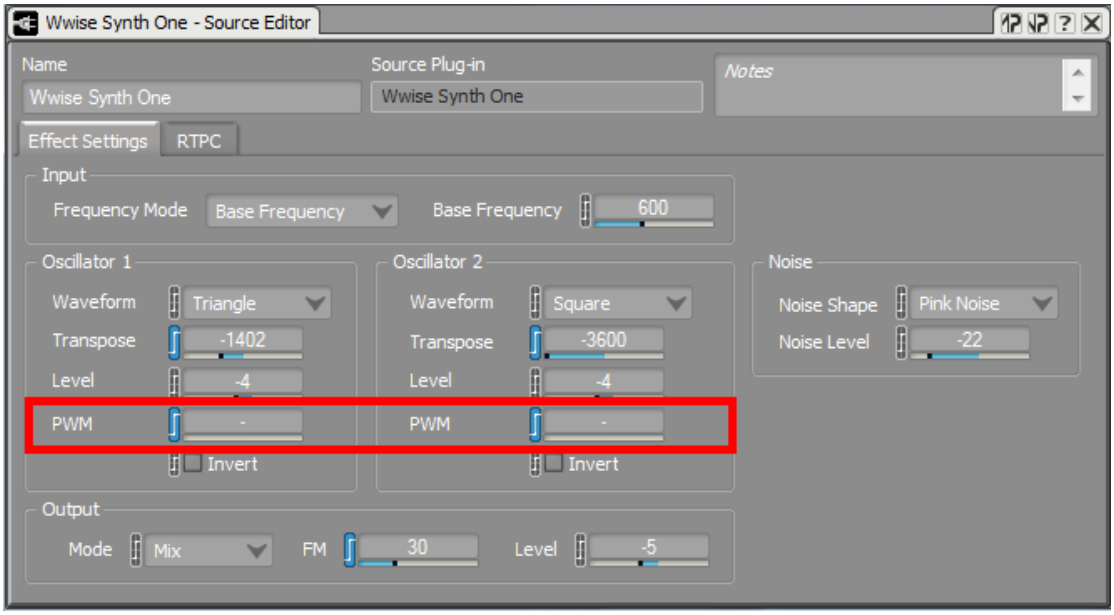
モジュレーター

Wwiseプロジェクト全体のプロパティをパラメーター化することは、ダイナミックな創作にとって強力なツールとなります。"amb_tree_rustle" アンビエントエミッターループのピッチを、低周波オシレーター（Low Frequency Oscillator: LFO）のプロパティでランダム化してコントロールする、または声の音量にランダム化した減衰時間を適用することでオーディオファイルの長さを変化させることを想像してください。ゲームパラメータに加えて、オブジェクトのプロパティを修正するのに使用できる一連のモジュレーターがあります。これらには、LFO、エンベロープ、そしてMIDIメッセージがあります。

モジュレーターLFO

モジュレーターLFO（低周波オシレーター）を使って、徐々にプロパティの値のモジュレーションを作成できます。RTPCに追加すると、モジュレーターLFOは値の範囲の間でプロパティ値を変化させます。 加えて、LFOプロパティの値はさらにパラメータ化もしくはランダム化でき、高レベルの変異性を達成することが可能です。

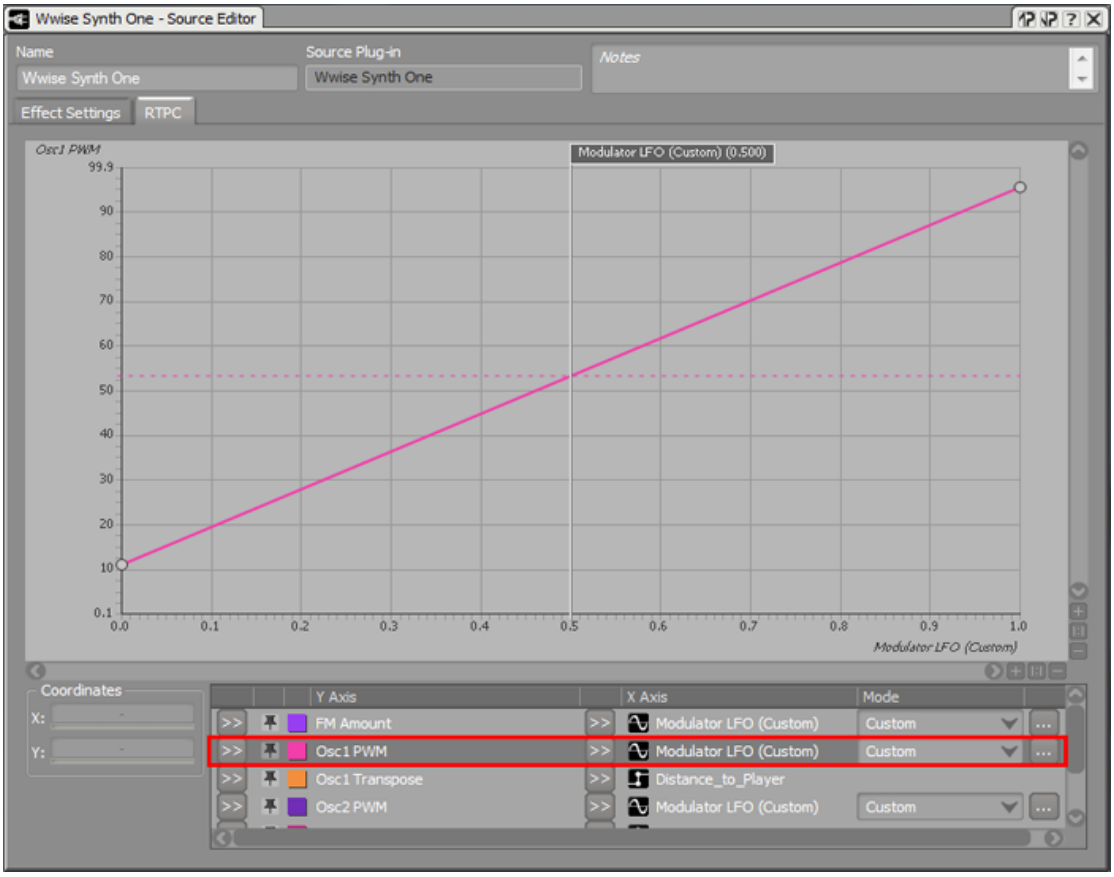
"magic_blast_fire_blend" 要素のために合成を多様化するため、各Oscillators PWM（Pulse Wave Modulation: パルス幅変調）プロパティにRTPCとしてモジュレーターLFOを加えることにより、サウンドにさらなる「動き」と振動を与えます。



LFOにアタッチされたWwise Synth One PWM（パルス幅変調）プロパティ（設定はRTPCタブで編集できます）

Designer Note

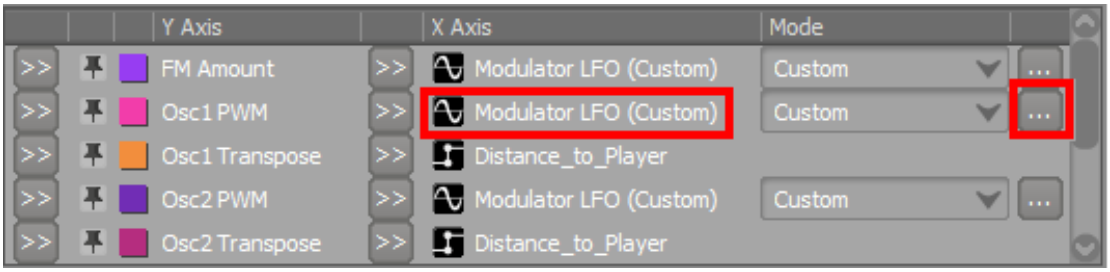
Wwiseには、追加式のプロパティ（Voice Volume、Voice Pitchその他）と、排他的なプロパティがあります。追加式のプロパティにLFOを追加すると、LFOモジュレーションはそのプロパティの現在の値に追加されます。排他的プロパティにLFOを追加すると、そのLFOモジュレーションがそのプロパティの現在の値と置き換わります。LFOで修正されたプロパティは、Propertyフィールドに破線（"-"）が示されます。



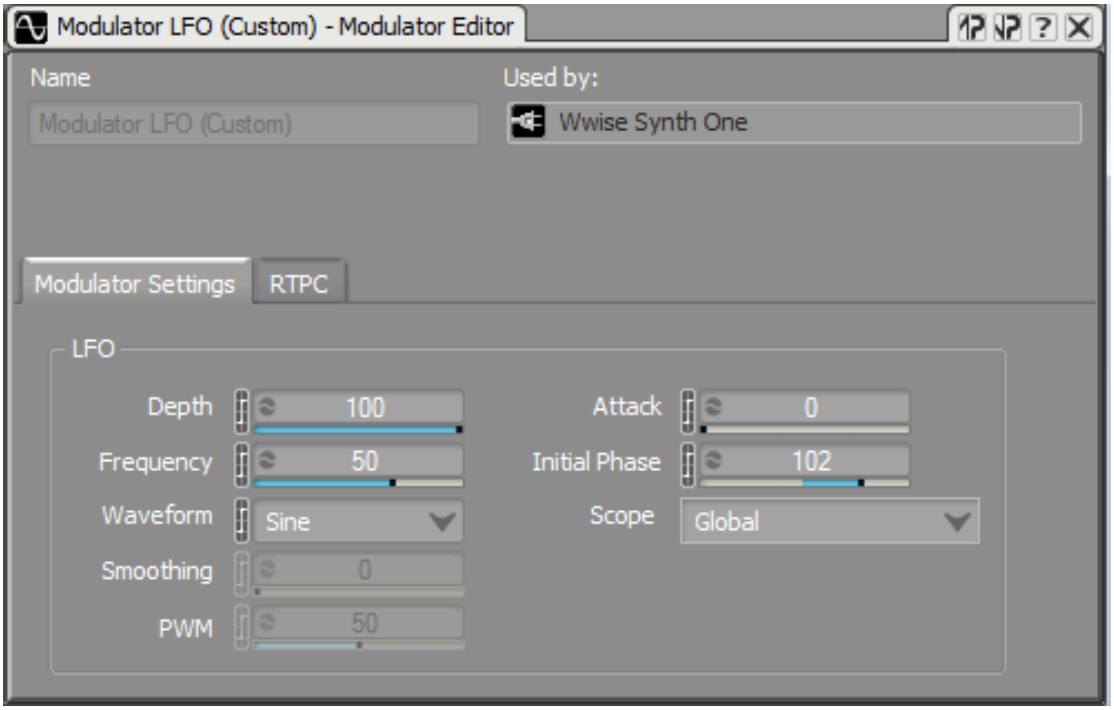
Wwise Synth One Oscillator 1 に示されるLFOによって変調されたPWMプロパティの効果的な範囲。

LFO（Low Frequency Oscillator）は、徐々にプロパティ値のモジュレーションを作成するのに使用します。Sound Property Editor 画面にあるLFOエンベロープの最高ならびに最低のRTPC値は、LFOの基底とトップ（ゼロとピーク値）に影響します。最低と最高値を入れ替えると、効果的にLFOを位相反転します。

RTPCダイアログのモジュレーターLFOをダブルクリックするか、[...] ボタンをクリックしてModulator Editor画面を開きます。



RTPCダイアログのモジュレーターLFOをダブルクリックするか、[...] ボタンをクリックしてModulator Editor画面を開きます。



Modulator LFO Modulator Editor画面

モジュレーターLFOのプロパティは次の通りです。

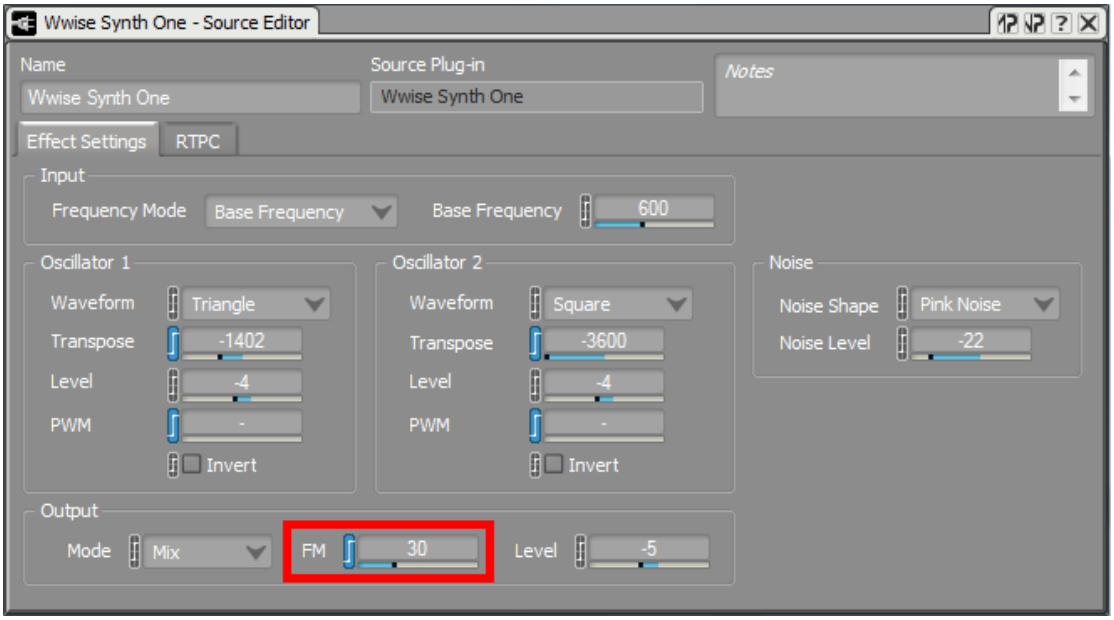
- **Depth**
  - 深さ。オシレーターの振幅推移（パーセンテージ）。最高振幅は 1.0。
    - デフォルト値: 100
- **Frequency**
  - 周波数。毎秒のサイクル数（Hz）。
    - デフォルト値: 1
- **Waveform**
  - 波形。モジュレーターの形（サイン形、三角形、矩形、のこぎり形）。
    - デフォルト値: サイン
- **Smoothing**
  - スムージング。波形に対するローパスフィルターで鋭いエッジをスムーズにする（パーセンテージ）。
    - デフォルト値: 0
- **PWM (Pulse Width Modulation)**
  - パルス幅変調。パルス波の幅であり、矩形の波形のみに適用される（パーセンテージ）。
    - デフォルト値: 50
- **Attack**
  - アタック。オシレーターが最大振幅に達するまでに要した時間（秒）。

    - デフォルト値: 0
- **Initial Phase**
  - 初期位相。オシレーター波形の初期（度）。
    - デフォルト値: 0
- **Scope**
  - スコープ。LFOの作成法をコントロールします。
    - **Voice:** ボイス。MIDIのコンテキストで使用する場合、LFOのインスタンスはシンセの各ボイスに作成される。
    - **Note/ Event:** ノート／イベント。MIDIのコンテキストで使用する場合、LFOのインスタンスはシンセの各再生インスタンスに作成さる。
    - **Game Object:** ゲームオブジェクト。ゲームエンジンに関連した各ゲームオブジェクトにLFOのインスタンスが作成される。
    - **Global:** グローバル。プロジェクト全体のために、すべてのインスタンスでグローバルに使用される単一のLFOが作成される。
  - デフォルト値: Voice

Designer Note

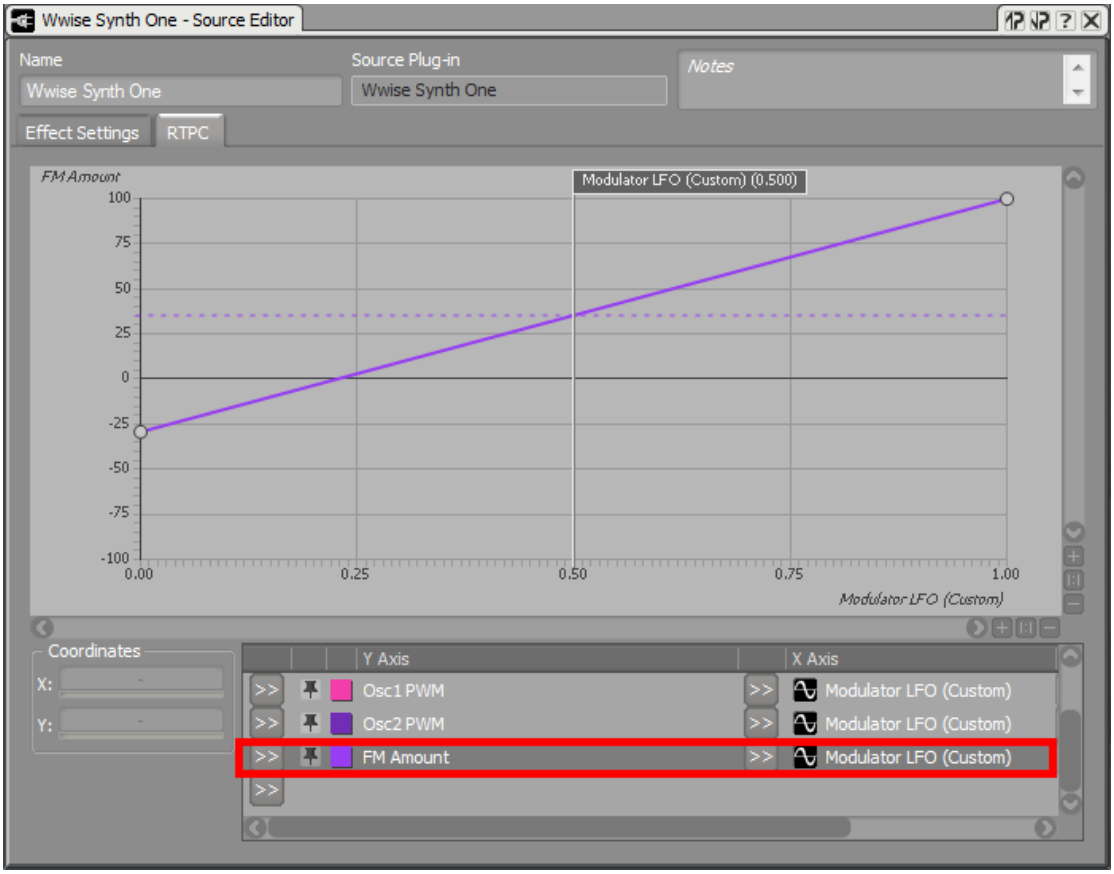
LFOでの作業に必要な追加情報はWwise Help文書を参照してください: Wwise Help > Interacting with the Game > Working with RTPCs > Working with LFOs

これに加えて、２つのOscillatorsの間でFrequency Modulation（周波数変調: FM）を使用してサウンドを「柔らかく」し、 サンプルをベースとしたmagic_blast_fire_blendへのブレンドを助けます。



Wwise Synth One FM（周波数変調: Frequency Modulation）プロパティ

この「FM量」のプロパティはLFOモジュレーターでさらに変化させることができます。



Wwise Synth One FM Amount プロパティ RTPC

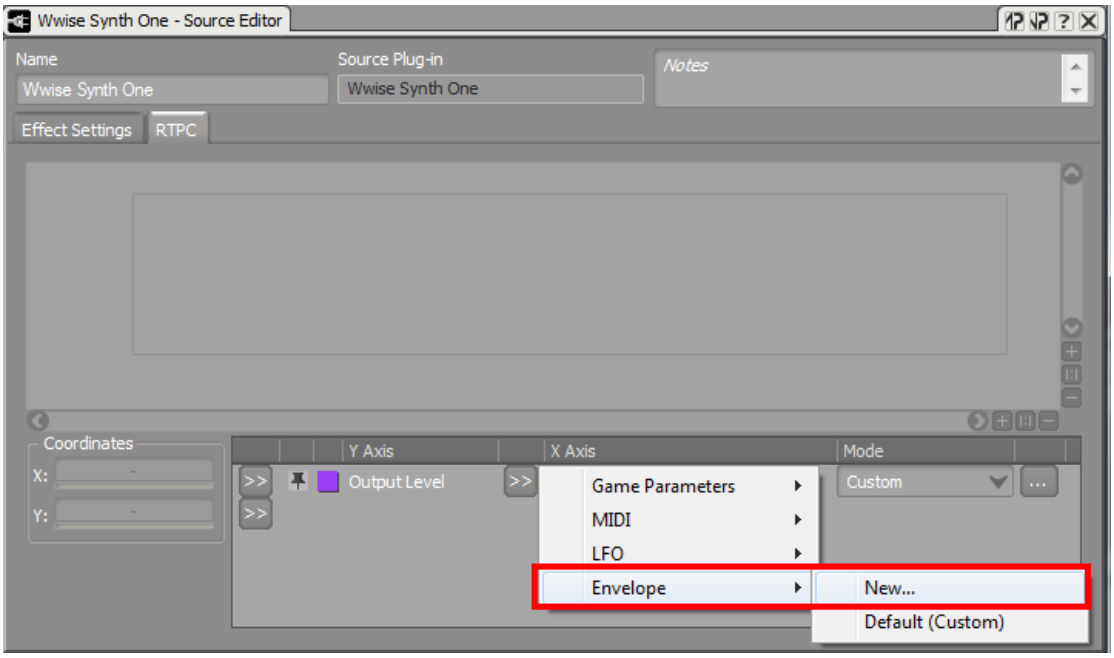
Designer Note

LFO オブジェクトは、Custom（カスタム）もしくはShareSet（シェアセット）として作成できます。カスタムオブジェクトは、それを持つオブジェクトに中に直接、適切に保存されます。シェアセットは別のワークユニットに保存され、複数のオブジェクト間で再利用できます。

Modulation Envelope（モジュレーションエンベロープ）

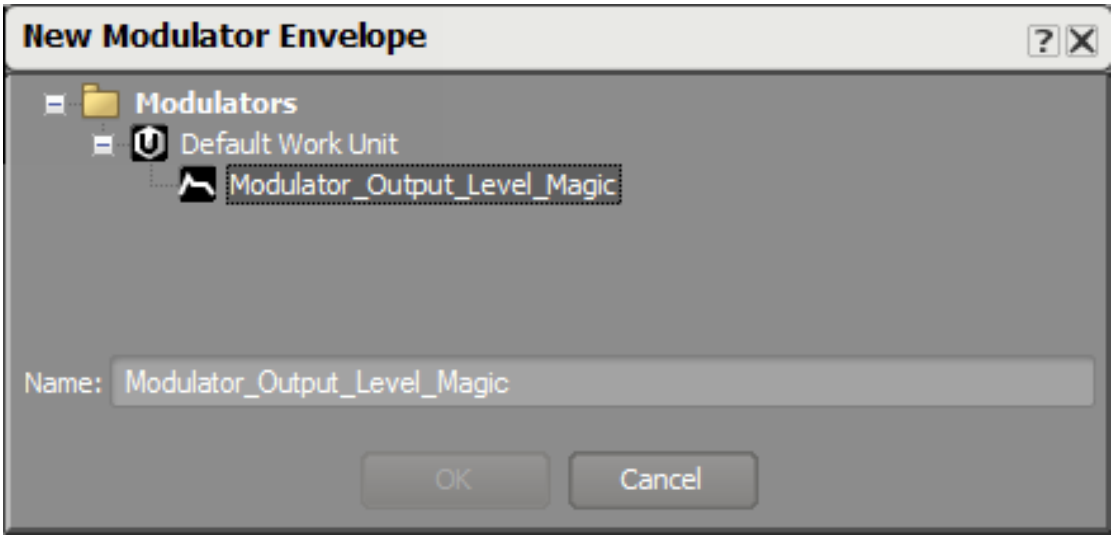
モジュレーターエンベロープで変更したOutput Level RTPCを加えて、Wwise Synth Oneの振幅エンベロープをコントロールできます。このモジュレーターエンベロープは、Attack Curve（アタック曲線）、Sustain Time（サステイン時間）に加えて、アタック、減衰、サステイン、リリース（Attack、Decay、Sustain、Release: ADSR）を提供します。モジュレーターエンベロープはまた、エンベロープ付きの再生を止めることにも利用できます。モジュレーターの多くのプロパティは、RTPCを使ってランダム化ならびに変化させることができます。

"magic_blast_synth_element"にはRTPCタブから、Selectorを使って新規のモジュレーターエンベロープシェアセットで変化させるOutput Level（出力レベル）パラメーターを加えます。



Output Level RTPCにモジュレーターエンベロープを加える

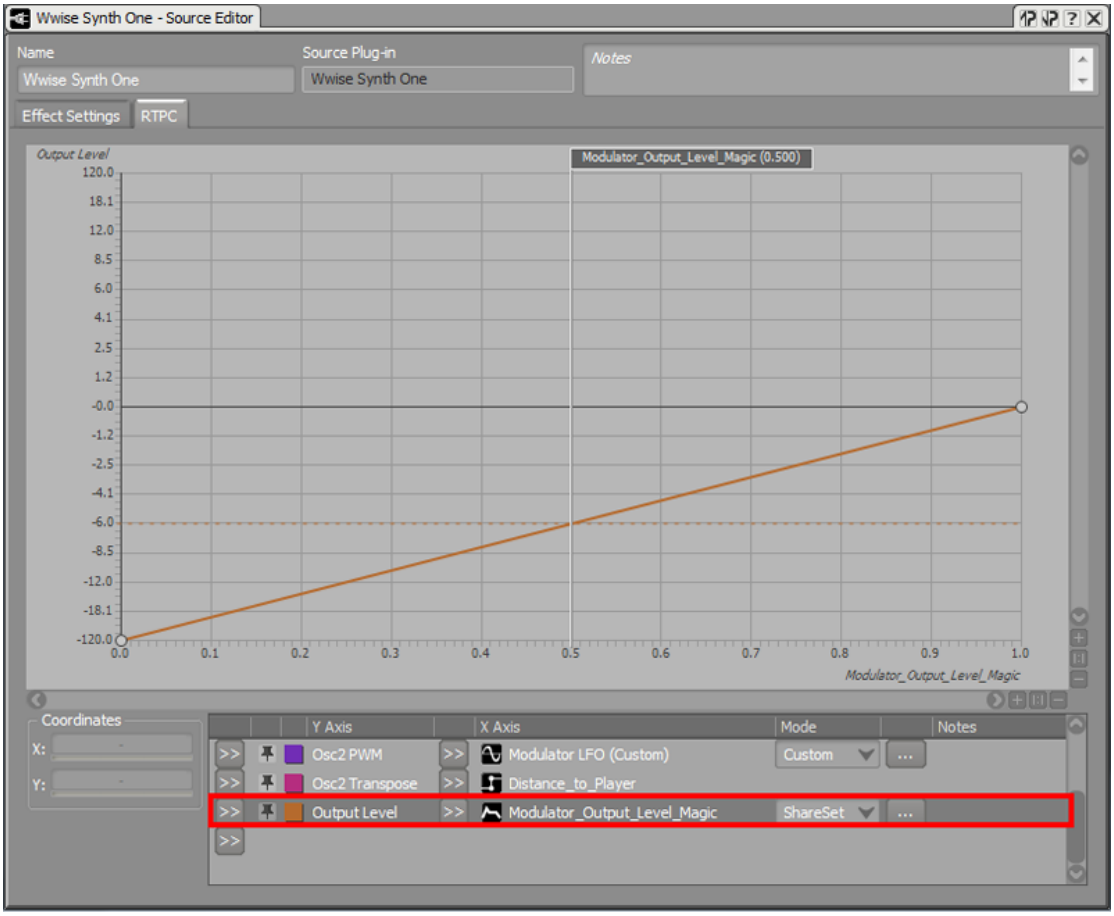
デフォルトでは、新規に制作されたモジュレーターエンベロープのモードはShareSet（シェアセット）に設定されています。



新規モジュレーターエンベロープシェアセットを作成して"Modulator_Output_Level_Magic"と命名

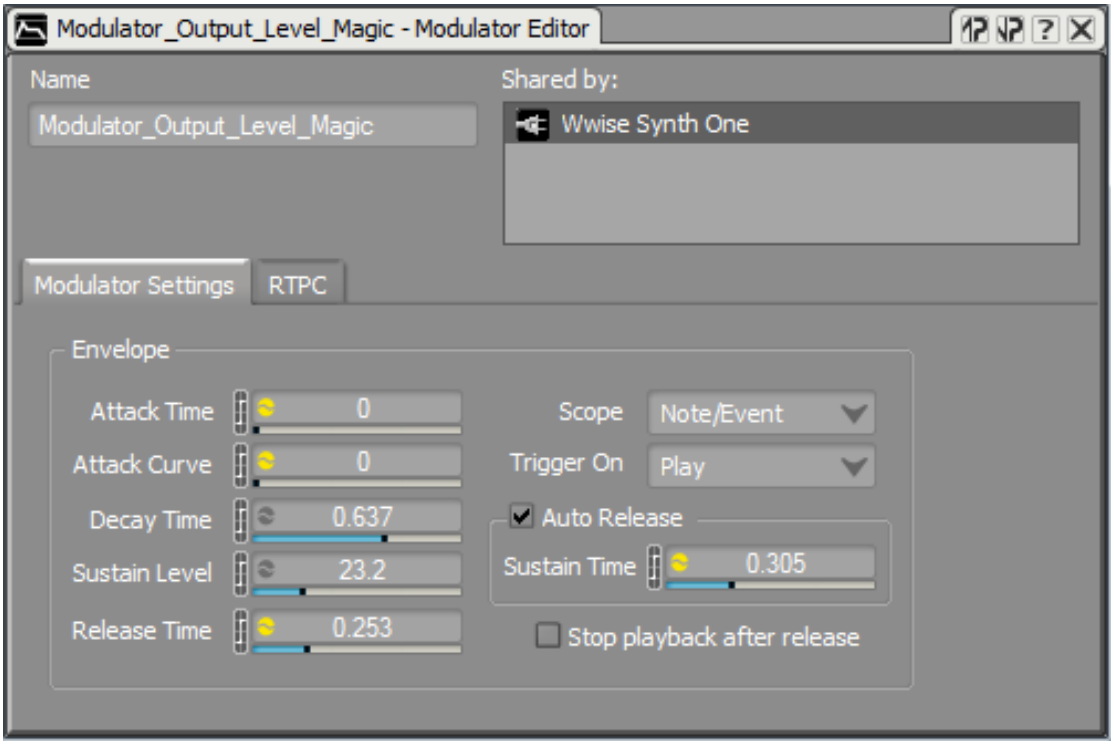
Designer Note

エンベロープオブジェクトはCustom（カスタム）としても ShareSet（シェアセット）としても作成できます。カスタムオブジェクトは、それを持つオブジェクトに中に直接、適切に保存されます。シェアセットは別のワークユニットに保存され、複数のオブジェクト間で再利用できます。



モジュレーターエンベロープをシェアセットに設定

Sound Property Editor 画面にあるモジュレーターエンベロープの最高ならびに最低のRTPC値は、ASDRの基底とトップ（ゼロとピーク値）に影響します。最低と最高値を入れ替えると、ADSRを位相反転します。



Envelope Modulatorの編集

Modulator Envelopeで利用できるプロパティは次のとおりです。

- **Attack Time**
  - アタックタイム。キーが最初に押されてから、ゼロからピークに達する初期の急上昇にかかる時間を定義します（秒）。
    - デフォルト値: 0.2
- **Attack Curve**
  - アタック曲線。Attack Curveを直線のデフォルトスロープ（50%）から次のいずれかに調整します。
    - 変化率が最初はゆっくりで後に早くなる指数スタイルのエンベロープ（0%）
    - 変化率が最初は早く徐々に遅くなる対数エンベロープ（100%）
      - デフォルト値: 50
- **Decay Time**
  - 減衰時間。アタックレベルのピークに続いて減衰し、指定されたサステインレベルになるまでにかかる時間を定義します（秒）。
    - デフォルト値: 0.2
- **Sustain Level**
  - サステインレベル。キーをリリースするまでの、サウンド持続時間の主となるシーケンスの間のレベルを定義します（範囲のパーセンテージ）。
    - デフォルト値: 100

- **Release Time（リリースタイム）**
  - リリースタイム。キーをリリースしてから、減衰がサステインレベルからゼロになるまでにかかる時間を定義します（秒）。
    - デフォルト値: 0.5
- **Scope**
  - スコープ。エンベロープの作成法をコントロールします。
    - **Voice:** ボイス。MIDIのコンテキストで使用する場合、エンベロープのインスタンスはシンセの各ボイスに作成される。
    - **Note/ Event:** ノート／イベント。MIDIのコンテキストで使用する場合、エンベロープのインスタンスはシンセの各再生インスタンスに作成さる。
  - デフォルト値: Note/Event
- **Trigger On**
  - トリガーオン。エンベロープをトリガーするアクション/MIDIイベント（つまりアタックフェーズに入る）:
    - **Play:** 再生。再生アクションまたはMIDIノートイベントのいずれか
    - **Note-Off:** ノートオフ。MIDIノートオフイベントのみ
  - デフォルト値: Play
- **Auto Release**
  - 自動リリース。エンベロープが、サステインフェーズを出て、リリースフェーズに入るためにアクション/MIDIエベントを必要とするかを定義。設定されると、エンベロープはSustain Time後、サステインフェーズを出ます。設定しない場合、エンベロープある条件に従ってサステインフェーズをでます。つまり、
    - **Release Envelope**イベントを経由して、エンベロープがゲームからリリースされます。
    - エンベロープがMIDIノートオフイベントによってトリガーされている場合、エンベロープはMIDIノートオフエベントを経由してリリースフェーズに入ることもあります。
    - デフォルト値: False
- **Sustain Time**
  - サステイン時間。Releaseが適用される前に、エンベロープがサステインに留まる時間を定義します（秒）。
    - デフォルト値: 0
- **Stop Playback After Release**
  - リリース後に再生を停止。設定すると、リリースフェーズが完了した後、関連するサウンドの再生を停止します。
    - デフォルト値: False

Modulator Editorを使って、モジュレーター エンベロープ プロパティにアクセス

Designer Note

エンベロープでの作業に必要な追加情報はWwise Help文書を参照してください: Wwise Help > Interacting with the Game > Working with RTPCs > Working with Envelopes

ゲームプレイのダイナミックスと組み合わせれば、合成、モジュレーション、そしてパラメーター化を組み合わせたパワーを最高に発揮することができます。プロジェクト内でサウンドのプロパティを変化させ、これに影響を与える大変多くのオプションを使うには、これらがプラットフォームのメモリやCPU性能にどれだけ影響を与えるかを理解することは大事なことです。

モジュレーターの処理時間はRTPCの使用に依存します。ほとんどのプロパティでは、モジュレーターはサウンドサンプル毎に評価されます。しかしVoice Volumeプロパティでは、関連するモジュレーターはフレーム毎に評価されます。モジュレーターはプラットフォームのメモリとCPUを大量に使用するので、モジュレーターは慎重に選択して使用します。常にターゲットプラットフォームの制限内で作業するようにします。

マジックのまとめ

魔法やおまじないをかける時のエフェクトの創造的なサウンドデザインは、サウンドデザイナーの想像力から生まれます。空想がサウンドとなり、マイクで録音され、最終的にはサウンドライブラリーに登録されてデジタルオーディオワークステーションを使って組み合わせ、幻想的なエフェクトを作成するベースとなります。次の段階では、最終的な実装に向けてコンテンツをどう準備するかを考えます。オーサリングアプリケーション内でゲームパラメータを活用しながらサウンドを組み合わせ、組み直し、ダイナミックに編集する作業手法をとれば、ゲーム本来のインタラクティブ性を活かすことができます。合成、LFO、そしてエンベロープの効果を組み合わせた、ゲームプレイとゲームのダイナミックな相互作用の可能性が、サウンドデザイナーの夢を実現します。

本章では、以下を説明しました。

- ブレンドコンテナを使ったサミングとレイヤリング
- RTPCを利用し、距離を元にサウンドの遠近感を変更
- エフェクトの適用や修正についての検討
- Wwise Synth One要素の作成

プロセスの説明

- ブレンドコンテナを使ったサウンドを再び組み合わせる
- エフェクトの使用
- リアルタイムパラメーターコントロール（Real Time Parameter Control: RTPC）の使用
- 距離に基づいたサウンドの遠近システムへのブレンドトラックの活用
- Modulator Envelope ShareSet（モジュレーターエンベロープシェアセット）を作成

詳細設定の説明

- エフェクトの順序付け
- エフェクトのオーディオバスのサブミキシング
- モジュレーター LFO

オブジェクトの作成方法

- マジックブラストの様々な遠近サウンド間をクロスフェードするために、距離に基づくゲームパラメータを使うブレンドコンテナ“magic_blast_fire_distance_blend”
- "magic_blast_fire_blend"のオーディオコンテンツを補足するWwise Synth One要素

参考ドキュメントとチュートリアル

Wwise Help > Using Sounds and Motion to Enhance Gameplay > Defining Object Playback Behaviors > **Defining the Contents and Behavior of the Blend Container**

Wwise Help > Wwise Reference > Actor-Mixer Objects > **Blend Containers**

Video Tutorial - Creating Dynamic Sounds Using RTPCs

Wwise Help > Interacting with the Game > Working with RTPCs > **Working with LFOs**

Wwise Help > Interacting with the Game > Working with RTPCs > **Working with Envelopes**

Wwise Help > Wwise Source/Effect Plug-ins > Wwise Synth One > **Wwise Synth One Plug-in**

第5章 ダイアログの設定と多言語対応

概要

恐怖におののく悲鳴が森中に響き渡ると、木々の枯れ葉が振り落とされて鳥が飛び立ちます。その後、人間のものとは思えないうめき声やうなり声に続き、助けを求める叫び声が聞こえてきます。

単純な会話のやり取りから、長編の冒険を多言語で物語る手段に至るまで、ダイアログの制作に高度なアセットマネジメントとシステムインテグレーションの技術が求められるようになりました。最近のゲーム機向けタイトルは台詞の量が増え続けていますが、ローカリゼーション計画が煩雑であれば、吹き替え作業も難題へと発展してしまいます。

本章では、以下の操作を説明します。

- Voiceへの入門
- 多言語サポートの準備
- ダイナミックダイアログ
- ポジショニングとパンの設定

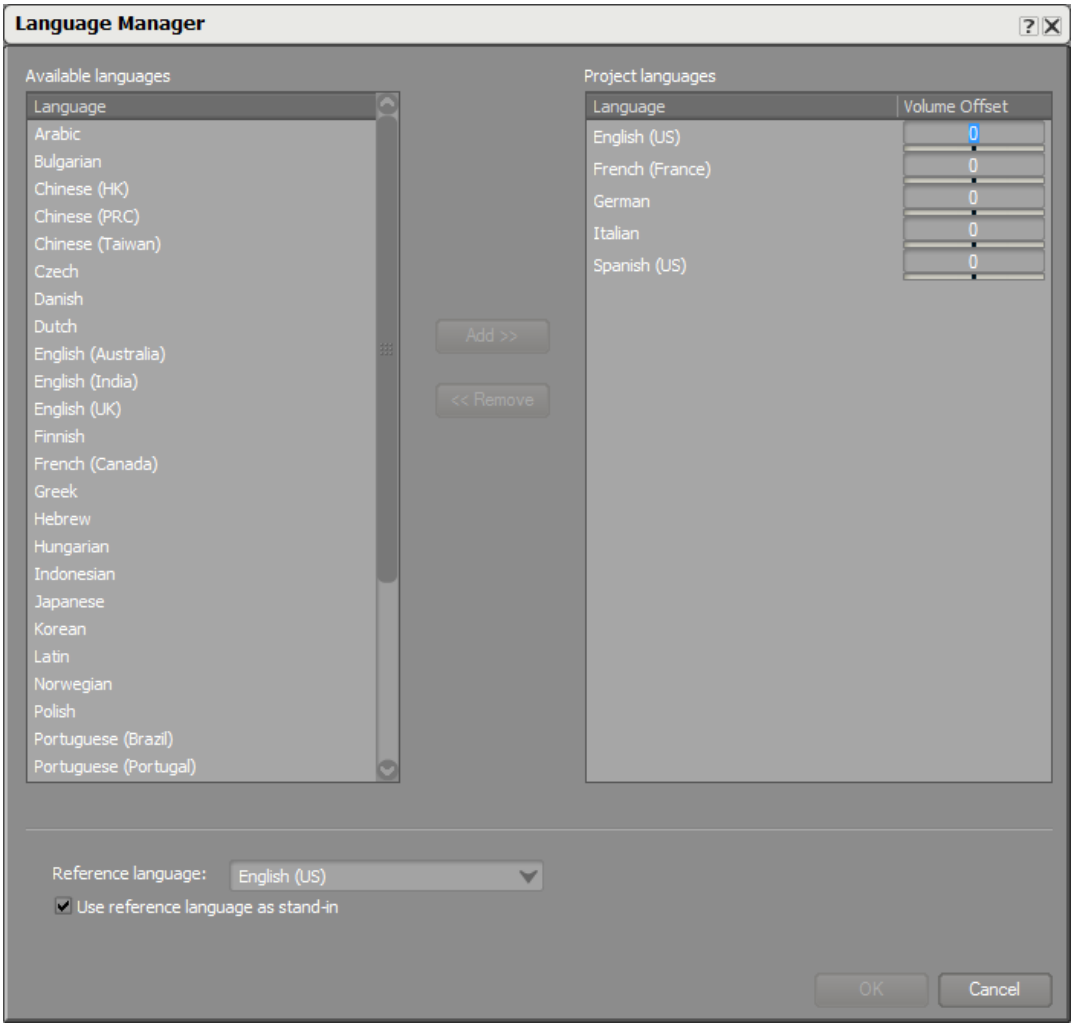
ダイアログと言葉のない音声の設定

ダイアログの作業では、言葉を使うダイアログと使わないダイアログを区別する必要があります。言葉を使うダイアログは各言語へのローカリゼーションが必要ですが、言葉を使わないダイアログは通常は翻訳しない感情表現の音声です。ローカリゼーション対象でないオーディオファイルの場合、他のサウンドエフェクトと同様に各種のバリエーションをサウンドオブジェクトに直接追加し、アクターミキサー階層内で整理して、イベントに追加できます。このようなダイアログの特徴に対応するために、ボイスアセットのインポートはミュージックやサウンドエフェクトのインポートと異なります。基本的な違いは、ローカリゼーションに向けてプロジェクトで使う言語を決める作業があることです。

メニューからWwise>Project>Languagesを選択して、Language Manager機能でゲームプロジェクトで使う言語を設定します。また、プロジェクト中に使うレファレンス（基準）言語をここで1つ選択します。

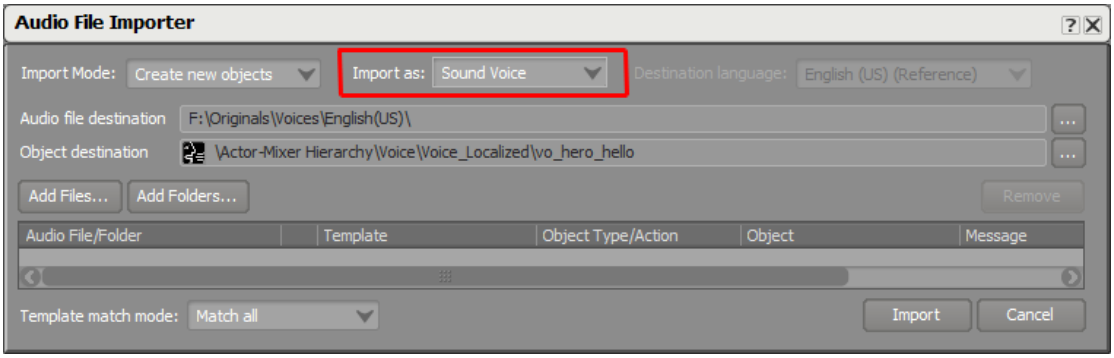
設定したレファレンス言語は、プロジェクト中に以下のような場面で使用します。

- **ランゲージファイルのインポート** - オーディオファイルをインポートする時、レファレンスファイルのインポートのコンバージョン設定を使用する
- **ランゲージファイルのコンバージョン** - 言語ファイルのコンバージョンには、レファレンス言語のコンバージョン設定が選択できる
- **ランゲージファイルが未完成** - あるランゲージ用のソースが準備中の時に、一時的にレファレンスランゲージで代用できる



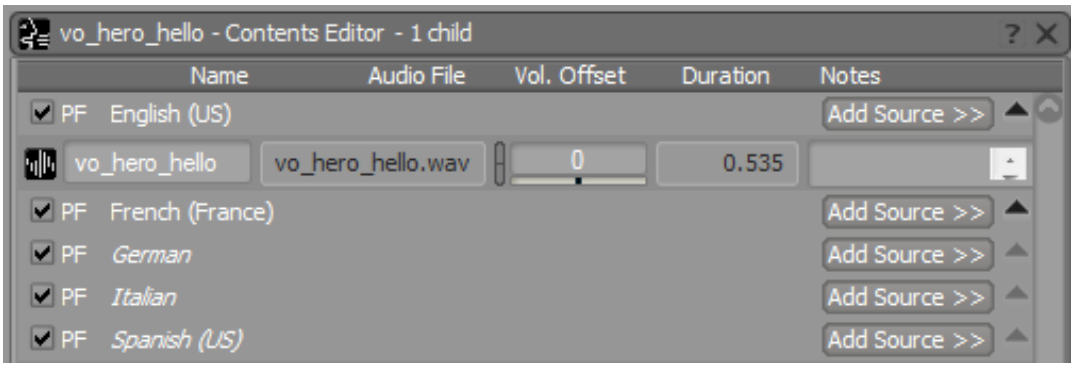
Language Manager画面でプロジェクト用の言語を選択

プロジェクト用の言語が決まれば、ローカリゼーション済みのボイスファイルをProjectメニューでインポートするか、Projectウィンドウにファイルをドラッグ&ドロップして入れます。Import as Sound Voice（ボイス用サウンドとしてインポート）を選択すると、Language Manager画面で設定したプロジェクト用ランゲージを元に、オブジェクトのローカリゼーション準備がされます。



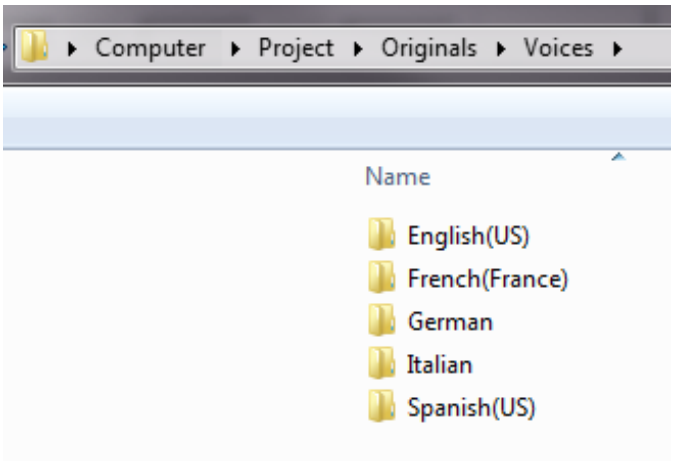
Audio File Importer画面でダイアログをSound Voiceとしてインポート

レファレンス言語でSound Voiceをインポートした後は、ローカリゼーションされたファイルを追加できます。



英語のオーディオソースの追加

次に、ソースであるオーディオファイルのコピーが、言語で分別されプロジェクトに追加されます。デフォルトで、プロジェクトのダイレクトリ内のOriginals/Voiceエリアにこれらのファイルのコピーが保存されます。

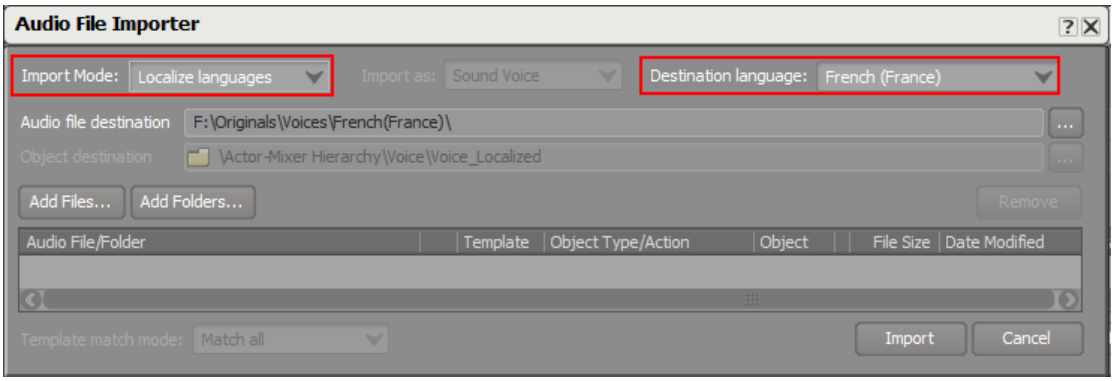


オリジナルのコピーが保存されたプロジェクトのVoicesフォルダ

他の言語の追加

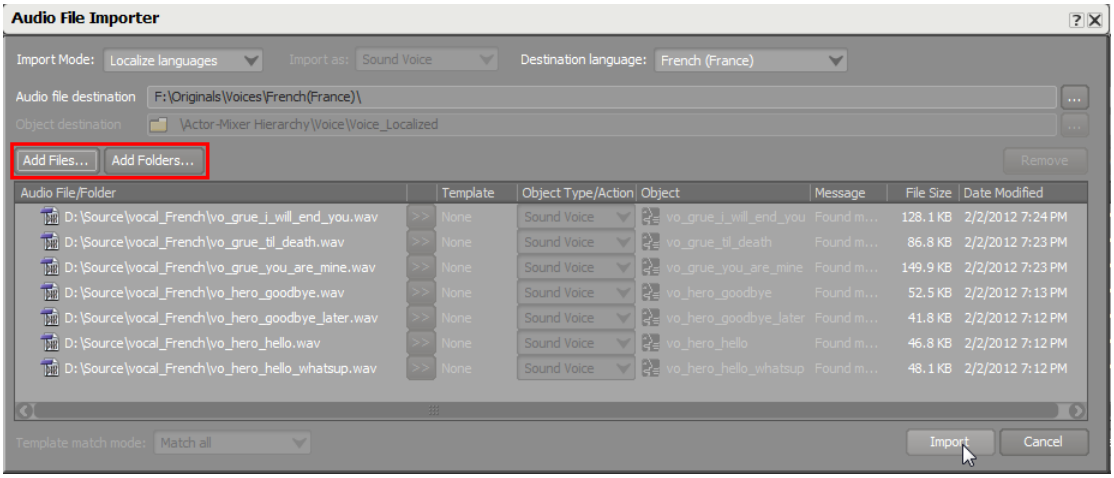
ローカリゼーションした言語ファイルが準備できれば、Audio File Importer機能を使ってプロジェクトに直接インポートできます。

まず始めに、インポートのモードをLocalize languages（言語のローカリゼーション）に変えて、ローカリゼーション用のDestination（行き先）言語を設定します。



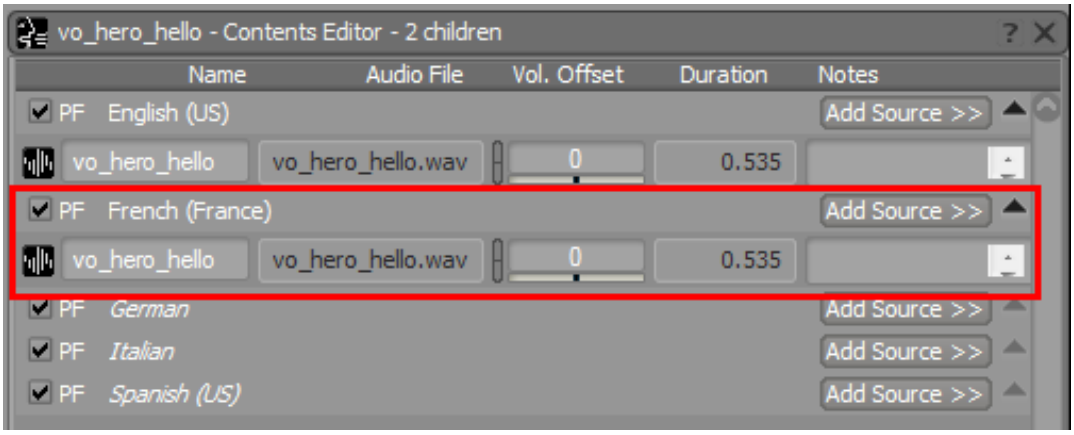
Audio File Importer画面でローカリゼーション済みのボイスファイルをインポート

次に、Destinationの言語用にインポートするファイルやフォルダを、Add FilesまたはAdd Foldersボタンをクリックして選択します。



既存のボイスオブジェクトにローカリゼーション済みファイルを追加

インポートが済むと、ローカリゼーション済みのファイルは関連するSound Voiceオブジェクトの Contents Editor画面に表示されます。



既存Sound Voiceにローカリゼーション済みダイアログを追加

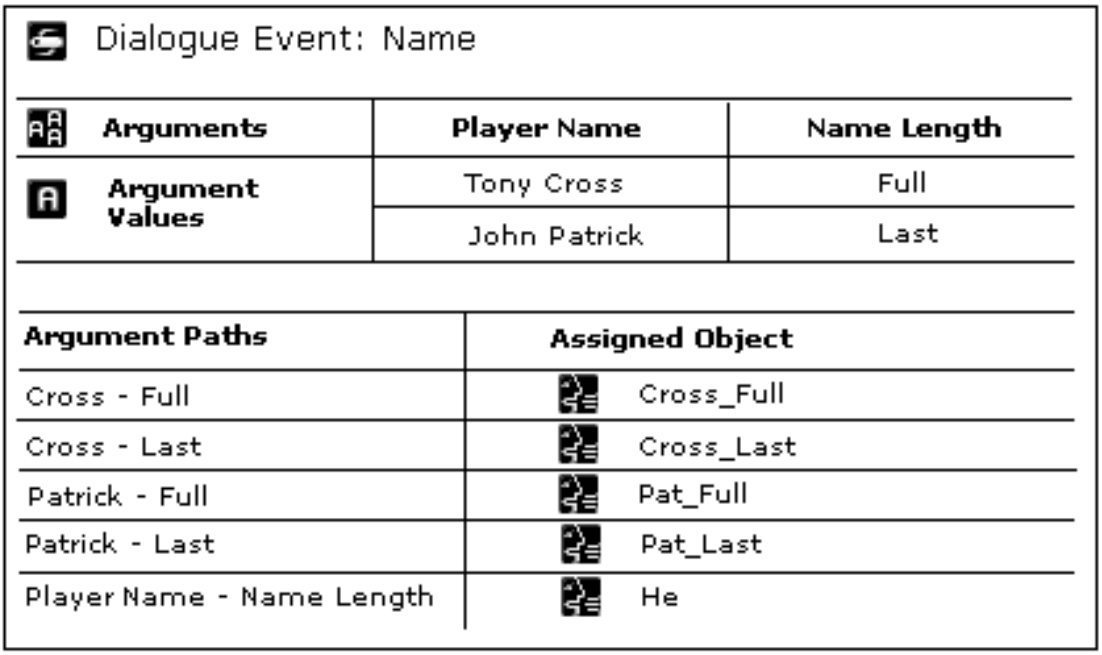
ダイナミックダイアログ

ゲームでは、冒険を続けるうちに道が分かれ、どちらかを選ばなければならない時が必ずやってきます。枝分かれするダイアログ計画の魅力は、プレイヤーが与えられた選択肢の中から自分で選び、会話を先に進めるところにあります。選択肢にある結果以外にも、それまでの会話で他のキャラクターに対して返答した内容を持ち越すこともできるでしょう。

Wwiseのダイナミックダイアログというシステムの中心にあるのがダイアログイベントで、どのダイアログを再生するかを決める際のルールや条件のセットを提供します。ダイアログイベントによって、ゲーム内で存在する様々なシナリオ、条件、結果などを再現できます。全ての状況に対応できるように、Wwiseでデフォルト条件や予備条件を作成することも可能です。

これらの条件は全て、引数や引数値を使って定義されます。さらに、この引数や引数値を組み合わせて作った引数パスが、ゲームで起きる特定の条件や結果を決めます。次にこのパスを1つずつ、Wwiseの対応するサウンドオブジェクトに結びつけます。ゲーム中にダイアログイベントが呼び出されると、ゲーム側で現在の条件をダイアログイベントの中にある複数のパスと比較します。ゲームにおける現在の条件に適合する、複数の引数パス、それぞれのパスのモード、確率、そしてウェイト付けによって、どのダイアログを再生するのか、または全く再生しないのかを決定します。

例えば、下記のダイアログイベントにはスポーツゲームの選手の名前に関連した引数が入っています。各引数の値を組み合わせて、可能性のある複数のパスや条件が作られます。この例では、解説者が選手のLast name（姓）またはFull name（姓名）のどちらかを使うという設定です。



万が一、ゲームの状況に合う引数値がないような場合のために、デフォルトまたは予備の引数値が入ったパスを作ることもできます。予備的なパスには通常、引数値の代わりに1つ以上の引数を入れ、一般的なサウンドオブジェクトに結びつけます。前例の場合、予備的引数パスは、プレイヤーの名前の代わりに「He（彼）」というサウンドオブジェクトに結びついています。

ダイアログイベントに対して起こりうる全ての条件を設定した後、ゲームエンジンに実装します。ゲームがダイアログイベントを呼び出すと、サウンドエンジン側で、ダイアログイベントに合う引数パスに対応するオーディオオブジェクトを返してダイアログイベントを完結させます。その後、サウンドエンジンはそのオーディオオブジェクトをダイナミックシーケンスにインサートして再生するかを判断します。オーディオオブジェクトを返す行為とオーディオオブジェクトをダイナミックシーケンスにインサートする行為の関係は、1:1とは限りません。完結したダイアログイベントで返したオーディオオブジェクトは、必要に応じて何回でもダイナミックシーケンスに入れることができます。

ゲームエンジンはダイアログイベントのイベント名を使うので、イベントを作成してゲームへのインテグレーションが済めば、イベントの中に入っているコンテンツの構築や微調整を行っても、再度インテグレーションをする必要はありません。この方法を取ることで柔軟性が大幅に広がり、引数値を自由に追加や削除し、色々なサウンドを試したとしても、追加のプログラミング作業は一切発生しません。

ダイナミックダイアログの詳細について、以下もご参照ください。

Wwise Help > Interacting with the Game > Working with Arguments > **Overview**

Wwise Help > Interacting with the Game > Managing Dynamic Dialogue > **Understanding the Dynamic Dialogue System**

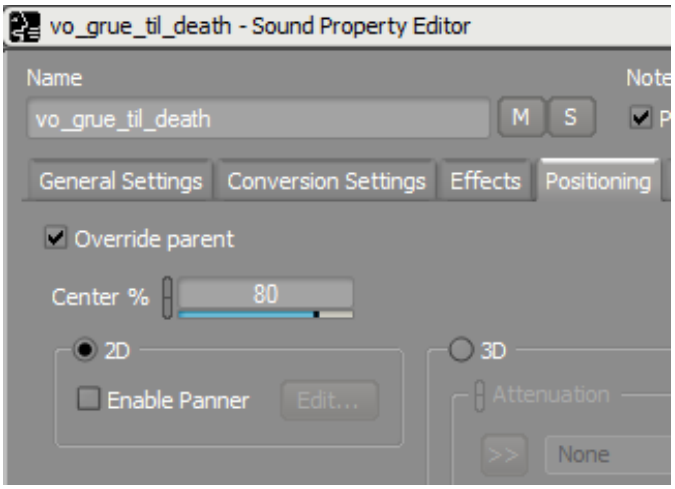
Wwise Help > Interacting with the Game > Managing Dynamic Dialogue > **Dialogue Events Tips and Best Practices**

Wwise Help > Interacting with the Game > Managing Dynamic Dialogue > **Creating Dialogue Events**

映画的ダイアログの配置

ゲームでのダイアログのポジションについて理解するために、ここでいくつかの例を検討します。映画やテレビなどストーリーがリニアのメディアは、ゲームの画面上でダイアログを提供する時の参考になりますが、ダイナミックに変化するやり取りを実行するには、検討すべき範囲がさらに広まります。

ゲーム中に、プレイヤーがコントロールできない場面で発生する映画的な会話は、一般的にフロントスピーカーに固定されているので、このような場面のダイアログポジションは2Dに設定することにより、その配置を適切に管理できます。伝搬設定がないサウンドオブジェクトの2Dオブジェクトは、デフォルトで左右のフロントスピーカーだけで再生されます。センターチャンネルを何パーセントか追加すると、コントロール範囲が広がります。



映画的なダイアログを2Dとして、Centerチャネルのボリュームを80%に設定

ゲームプレイのダイアログが再生されている最中にプレイヤーが存在し、ワールドの中を自由に動けるような状況では、ダイアログの再生ポジションを3Dとする必要があるかもしれません。台詞を言うキャラクターの位置にあるゲームオブジェクトのSound Voiceに減衰設定を登録するだけで、事前に定義された減衰設定に基きボリュームが既定通りに減少します。これは、ゲームのワールドに配置されたキャラクターが発言している最中に、プレイヤーが自由に動けるような場面で使える方法です。

ゲームの中でポジションが決まっているダイアログをリアルにする方法は、ボリューム減衰に限定されません。アンビエント音の説明でキャンプファイヤーの例をみましたが、一般的にボイスが接近しているとき、隣接するスピーカー間でスプレッドを追加するとリアル感が増します。さらにこだわって減衰コーンも設定すれば、ゲームオブジェクトの向きによってサウンドに方向性をもたせることができ、例えば、話しているキャラクターの口の向きによって聞こえ方を変えられます。減衰コーンを使えば、方向性をもつゲームオブジェクトの外や後ろにリスナーが移動するにつれ、サウンドが減衰しフィルターが適用されます。つまり、キャラクターの台詞がキャラクターの頭によって妨害される様子をシミュレーションできます。

Programmer Note

OrientationFront（前方向）ベクターで定義するのが、リスナーの顔の向きです。これは、リスナーの頭の傾斜を定義するOrientationTop（上部方向）ベクターに対して直角にあるべきです。リスナーが人間であれば、OrientationFrontベクターがリスナーの鼻の向き（顔から外に向かっている）となり、OrientationTopベクターはそれに対して直角関係にあり、リスナーの目の間を通って顎とは逆方向を向きます。

よりリアルなダイアログのシミュレーションを提供する上級テクニックを導入することも検証の価値はありますが、再生するダイアログがゲームにどれだけ重要かで、こだわる度合いを簡単に判断できます。例えば、ドラゴンを従わせる秘密の技をプレイヤーが聞きとるのが重要であれば、ダイアログをリアルにしようとフィルターやボリューム減衰を使って聞こえづらくするのは良くありません。ゲーム中の会話は全て、リアル感の追及とストーリーの伝達とのバランスが大切であり、ゲームプレイと結びついた時にプレイヤーが会話によって没入感を味わうことが理想的です。

ボイスのまとめ

ゲームでストーリーを表現する機会が拡大するにつれ、感情を表現してキャラクターを形成していく上で人間の声より効率的な手段は、あまりありません。言葉を使わないコミュニケーションから、会話を成立させる複雑なツリーの構築に至るまで、ゲームプレイの流れの中にダイアログを組み込むことを明確な目的としたツールセットをWwiseが提供します。

本章では、以下を説明しました。

- Voiceへの入門
- 言語的音声と非言語的音声の区別
- ローカリゼーションダイアログの作成手順の確立
- シナリオ別のダイアログポジショニングの役割

プロセスの説明

- 英語ダイアログファイル用のサウンドボイスオブジェクトのインポート
- Audio File Importer機能でローカリゼーションされたダイアログファイルのインポート
- 2Dポジションのサウンドのセンターチャンネルの割合の調整

詳細設定の説明

- 引数を使った枝分かれするダイアログの作成

オブジェクトの作成方法

- 会話用のダイアログSound Voiceの英語版と、フランス語ローカリゼーション版

参考ドキュメントとチュートリアル

Wwise Help > Interacting with the Game > **Managing Dynamic Dialogue**

Wwise Knowledge Base - Does Wwise provide support for integrating a lip-sync solution?

Wwise Knowledge Base - Importing a large number of files in Wwise

Video Tutorial - Managing Voices and Language Localizations

第6章 ユーザーインターフェースの開放

概要

ゲームでは一般的にユーザーインターフェースが軸となり、そのまわりにゲームのワールドが形成されます。確実なサウンド環境を提供するには、ユーザーインターフェースの流れをよく理解し、没入感を失うことなく、ゲーム内のワールドをサウンドで強化することが基本です。

本章では、以下の操作を説明します。

- 簡単なメニューイベントの作成
- 2Dポジションとパンの設定
- 複雑なイベントの作成

正確なサウンドを忠実に再生するというシンプルな機能から、一時停止ボタンを押した時に発生する複雑なやり取りに至るまで、ユーザーインターフェースはオーサリングアプリケーションであるWwiseのソリューションやテクニックのケーススタディーとも言えます。

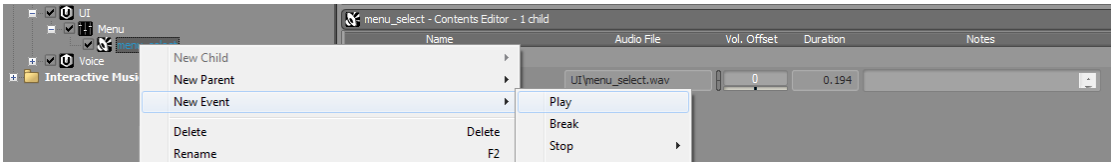
単純なメニューセレクトのサウンドを作成

それでは、ゲームのメニュー画面で項目を選択した時に再生されるサウンド“menu_select”のSound SFXを作成します。まずファイル“menu_select”をインポートしますが、オーディオファイルインポーターを使うか、ソースフォルダからデフォルトのワークユニットにドラッグ＆ドロップします。

サウンドオブジェクト“menu_select”

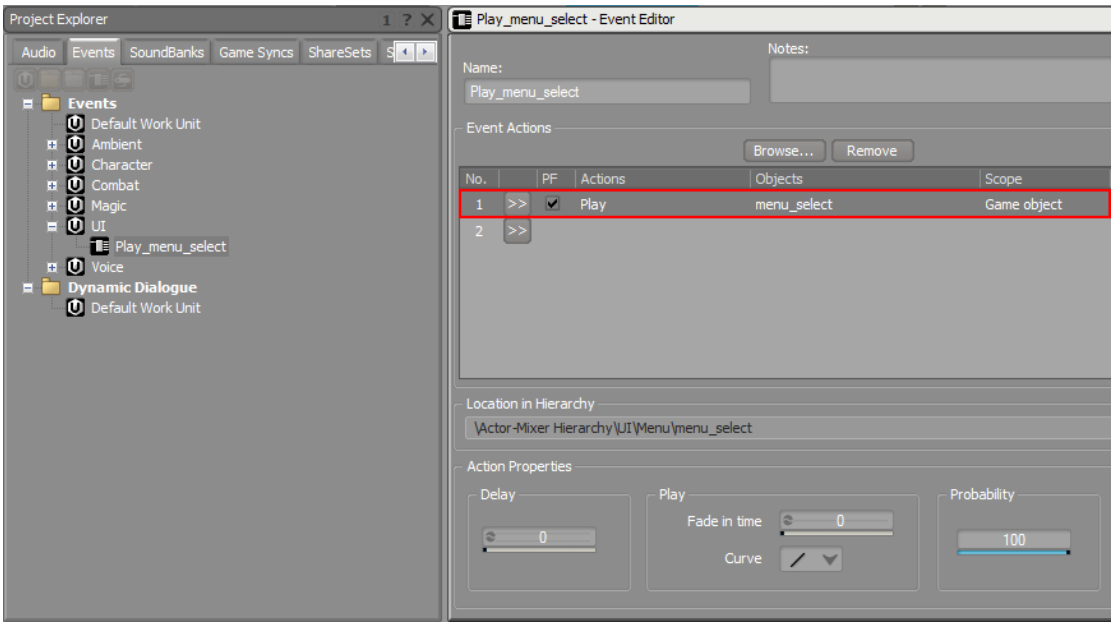
簡単なMenu Select（メニュー選択）イベントの作成

Sound SFXを選択してコンテキストメニューを表示すれば、すぐにこのサウンドのPlayイベントが作れます。



サウンドオブジェクトのコンテキストメニューからPlayイベントを作成

これで、以下のイベントが作られました。



イベント“Play_menu_select”

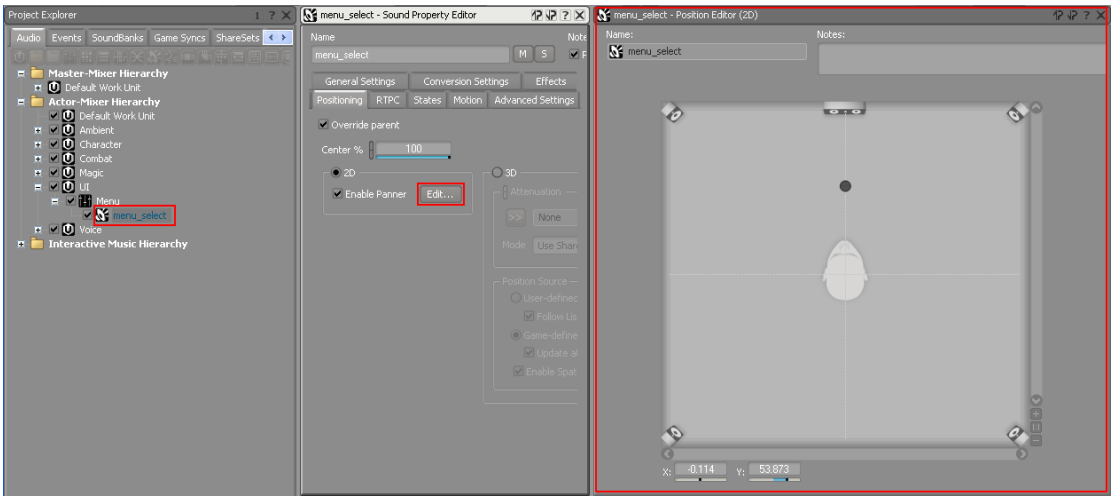
イベント名称“Play_menu_select”をプログラマーに伝え、プログラマーはメニュー項目が選択される度にイベントをPost（送出）します。

Programmer Note

イベントをトリガーするには、AK::SoundEngine::PostEvent()を呼びます。

2Dサウンドポジショニングの定義

メニューのビジュアル的な要素は一般的に平面表示であるため、サウンドオブジェクトの3Dポジションは考慮せず、2Dポジションをマニュアルでコントロールすることが一般的です。WwiesのProject Explorer画面にあるPositioningタブを開くと、サウンドオブジェクトの2Dポジションを直接コントロールすることができます。



2D用Position Editor画面

2Dポジションに切り替えると、2Dパンインターフェースでサウンドのポジションをコントロールできるほか、センタースピーカーで出すサウンドの割合を、センターチャンネルのパーセンテージ設定で変えられます。

Programmer Note

通常、UI（ユーザーインターフェース）のイベントは、ポジション情報を持たないグローバルゲームオブジェクトにトリガーされます。

ユーザーエクスペリエンスを高める上で特に意識しなくてはならないのが、サウンド自体の雰囲気と美しさです。特別なエフェクトを達成するために高度なテクニックを使うこともできますが、ユーザーへのフィードバックを支えるサウンドのデザインは、実装作業よりも難しいことが多いです。

一時停止で発生する複雑なやりとり

最終的な個別サウンドの再生は単純なものであっても、ゲームプレイからメニューへ移る時には、複雑な操作を続けて実行する必要がよくあります。プレイヤーはゲームプレイ中にいつでも一時停止ボタンを押して、すぐにゲームプレイからメニュー操作に移れることを求めます。

メニューへの移行がサウンドの観点からシームレスに行われるためには、以下が実行されます。

- インゲームで再生中のミュージックは、ミュート、一時停止、またはメニュー中も再生
- インゲームのサウンドエフェクトは、一時停止、停止、またはミュート
- インゲームのアンビエントサウンドは、ミュート、一時停止、またはメニュー中も継続
- クリティカルでないダイアログは一時停止、または停止
- クリティカルなダイアログは一時停止、または再スタート
- メニューミュージックやアンビエントサウンドを再生してもよい

以上のアクションは、サウンドデザイナーに許容される操作の範囲外と思われるかもしれませんが、イベントシステムでは、このような判断項目をオーサリングアプリケーション上でコントロールし編集することができるのです。Wwiseのイベントがプロジェクト階層の中にある様々な構成に対してアクションを適用します。各イベントは1つまたは一連のアクションを含みます。選んだアクションによって、Wwiseオブジェクトの再生、一時停止、停止などが決まります。

イベントアクションの一覧が、以下のHelpドキュメンテーションにあります。

Wwise Help > Wwise Reference > Events > **Event Editor**

Wwise Help > Where to Begin? > Wwise Fundamentals > Understanding Events > **Action Events**

Wwise Help > Interacting with the Game > Managing Events > Overview > **Types of Event Actions**

一時停止とシナリオの定義

例として、以下を実行するシナリオを検討します。

- ミュージック、アンビエント、サウンド、ボイスを全て一時停止
- トランジション用サウンドエフェクトを再生
- メニュー専用のミュージックをループ

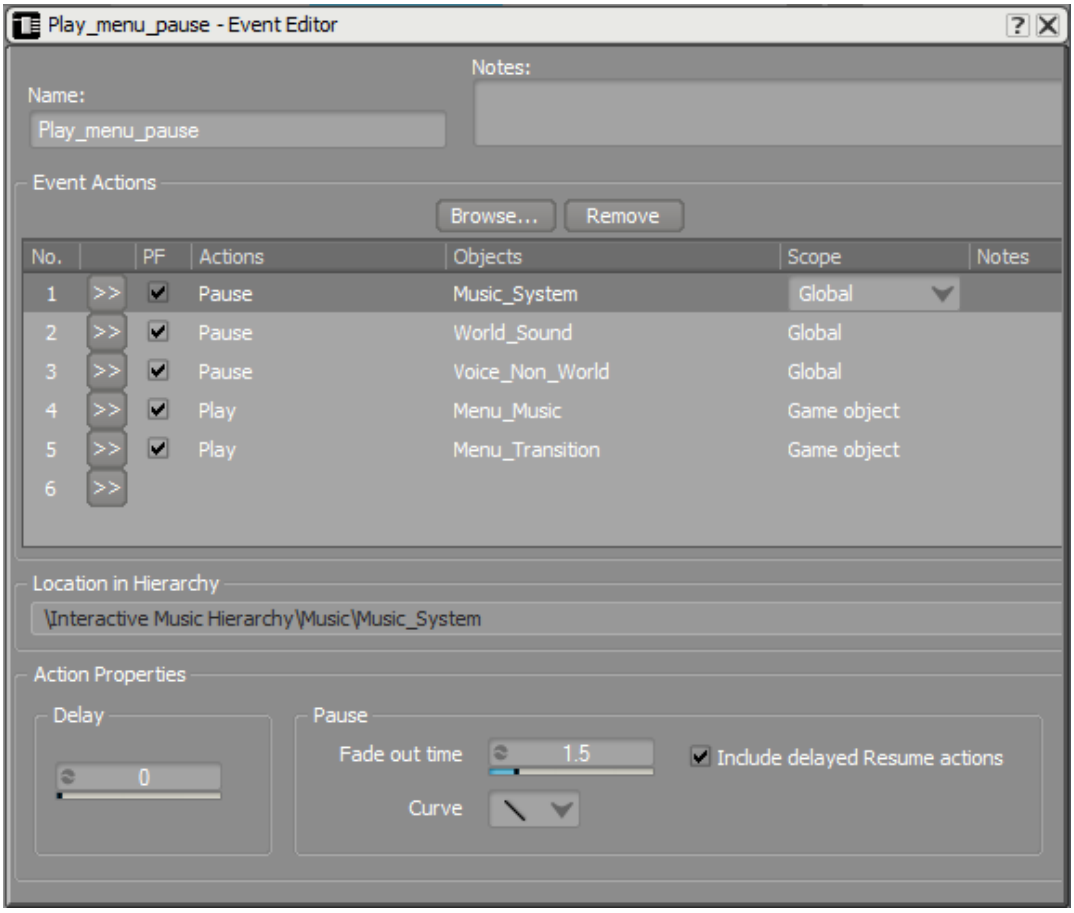
Designer Note

1人のユーザーが一時停止している間も継続されるマルチプレイヤーのゲームや展開し続けるワールドの場合は、オーディオをミュートにできます。

理想的には、プログラマー的なシンプルな考え方では、2つのイベント、つまりPause Audio（オーディオ一時停止）とResume Audio（オーディオ再開）を伝達すれば充分です。プログラマーがこれらのイベントを送出した時のサウンド側の動作は、サウンドデザイナーが具体的に決めることができます。今回は例を分かりやすくするため、マスターミキサー階層のバスに基づき一時停止します。

ゲームの一時停止

下記のイベントでは、ミキシング階層の各レベルで一時停止のアクションを使い、各オブジェクトに1.5秒のフェードアウトをかけます。



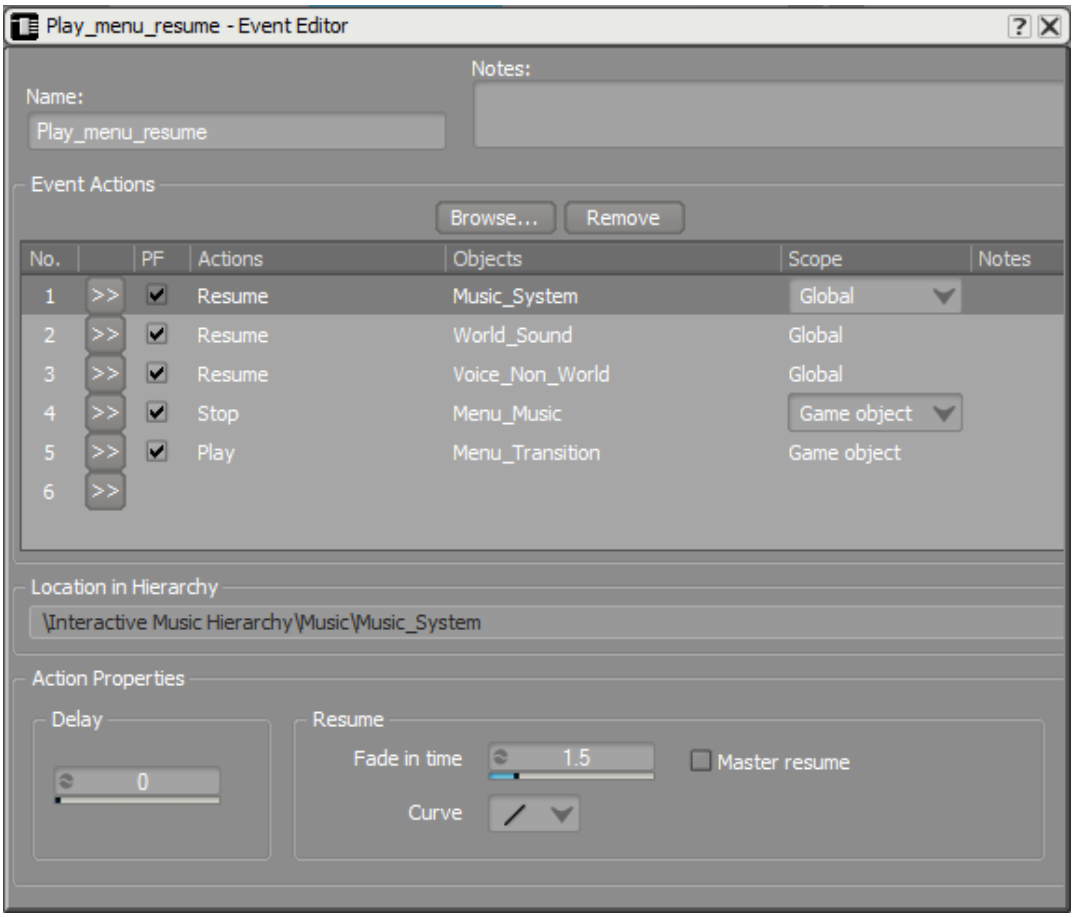
イベントPauseのイベントアクション

イベントアクション	オブジェクト名	場所
Pause	Music_System	(Interactive Music)
Pause	World_Sound	(Master Audio Bus)
Pause	Voice_Non_World	(Master Audio Bus)
Play	Menu_Music	(Interactive Music)
Play	Menu_Transition	(Actor-Mixer)

なお、“menu_music”と“menu_transition”がアクションPlayを同時に使います。

ゲームの再開

ゲームを再開するには、マスターミキサー階層の一時停止されたバスに対してアクションResume（再開）を使うだけで、全てを一時停止された時点から再開できます。同時に、“menu_music”に対してアクションStopを1.5秒のフェードアウトをかけて使い、“menu_transition”サウンドをまた再生します。



イベントResume（再開） のイベントアクション

イベントアクション	オブジェクト名	場所
Resume	Music_System	(Interactive Music)
Resume	World_Sound	(Master Audio Bus)
Resume	Voice_Non_World	(Master Audio Bus)
Stop	Menu_Music	(Interactive Music)
Play	Menu_Transition	(Actor-Mixer)

この例で肝心なのは、たった2つのWwiseイベントでゲームのサウンド全てをコントロールできるということです。イベントアクションを使えば、ゲームプレイから一時停止してまた再開するというトランジションがゲーム側で起きた時、サウンド側の設定がサウンドデザイナーの希望通りにオーサリングできます。

ユーザーインターフェースのまとめ

ユーザーインターフェースに対するサウンドの実装は、ゲームの雰囲気や美観をサポートするために、必要なだけシンプルに、あるいは複雑に計画することができます。プレイヤーがゲームプレイやメニューシステムを操作している時も、両者間を行き来している時も、全体の芸術観をサポートするのがサウンドの目的、一言で言えばサウンドはエクスペリエンスの延長です。インタラクションの詳細をイベントシステムで管理することで、Wwiseを使うサウンドデザイナーが決定権を有します。

本章では、以下を説明しました。

- サウンドファイルのインポートの復習
- ゲームにおける美観をサポートするためのUser Interfaceサウンドの役割の検証
- 一時停止や再開に特有の問題の確認

プロセスの説明

- メニュー選択の単純なイベントの作成
- Position Editorを使った2Dサウンドの設定
- 一時停止と再開のイベントの作成

詳細設定の説明

- 様々な種類のイベントアクション

オブジェクトの作成方法

- 単純なメニュー選択のSound SFXとイベント
- 一時停止と再開のイベント

参考ドキュメントとチュートリアル

Wwise Help > Interacting with the Game > Managing Events > Overview > **Types of Event Actions**

Wwise Help > Using Sounds and Motion to Enhance Gameplay > **Defining Positioning for Sound and Motion**

Video Tutorial - Creating Events

第7章 音楽のアドベンチャー

概要

ゲームのヒーローが勇敢に前進する時に鳴り響くラッパやファンファーレが、何時間も延々とループするブラス音では疲れてしまいます。ゲームに使われるインタラクティブでダイナミックな音楽は、それ自体が芸術作品として進化しています。シンクポイントのオーサリング、拍子記号の設定、音楽のエントリーポイントとエグジットポイントの維持といった作曲の複雑なテクニックを、オーサリングアプリケーションで実行できるようになり、ゲームという媒体専用の複雑に織り成された作品が可能となりました。このような能力が備わったアプリケーションを使いこなせば、ゲームとプレイヤーの体験をサポートするカスタム化された音楽を提供できます。

音楽システムの実装戦略を決めるにあたり最初に検討するのは、ゲーム中の感情の変化やゲームの目的に対して、適した音楽をどう組み合わせるかです。システムとは本質的に柔軟で多様なものですが、根本的な目標としてゲームとミュージックシステムをシームレスに結びつける必要があります。ミュージックシステムの方針が確立されれば、ゲームプレイをサポートする音楽について検討した上で、作曲の要素を組み合わせて、ゲーム全体を流れる感情の河を構築します。

本節では以下の操作を説明します。

- インタラクティブミュージック階層からのスタート
- コンテンツの準備
- 横型のアプローチ
- 縦型のアプローチ
- ミュージックタイプの切り替え
- インタラクティブミュージックのトランジションの定義

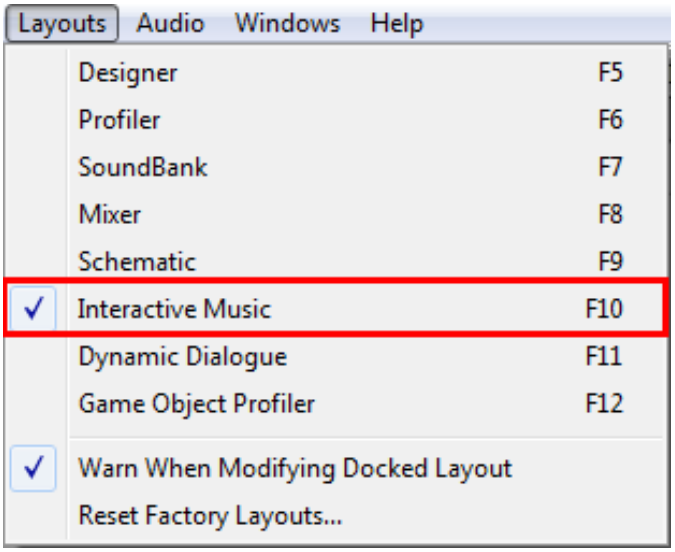
インタラクティブミュージック階層からスタートする

ミュージックエンジンとは？

ミュージックエンジンは、インタラクティブミュージック階層にあるミュージックセグメントの複雑でハイレベルな順番を操作するハイレベルエンジンです。インタラクティブミュージック階層とアクターミキサー階層の根本的な違いは、様々なミュージックコンテナの動作を管理する時のテンポや拍子記号の使い方です。

インタラクティブミュージック階層に追加したオーディオファイルは、ミュージックセグメントとしてインポートされ、さらに他のミュージックセグメントにトラックとして入れることもできます。またセグメントをミュージックプレイリストコンテナやミュージックスイッチコンテナに入れると、機能が広がります。

インタラクティブミュージック階層のコンテナが提供する音楽専用の機能を見るには、レイアウトをInteractive Musicに変更します。



レイアウトをInteractive Musicレイアウトに変更

コンテンツの準備

作業中のプロジェクトにミュージックコンテンツをインテグレートする前に、使うシステムの種類に合わせてコンテンツを処理します。本節ではダイナミックミュージックを生み出す2種類の方式を説明した後、両者を一緒に活用してゲームで探索を続けるヒーローの心理的状況を表現します。2つの方式、つまり横型のアプローチ（タイムシンクしたミュージックレイヤーのボリュームをゲームプレイに反応させる方法）と縦型のアプローチ（ループ再生する音楽をランダム化させ、変化させる方法）を駆使して、プレイヤーのゲーム内の行動に合わせて変化するシンプルな音楽システムを作成します。

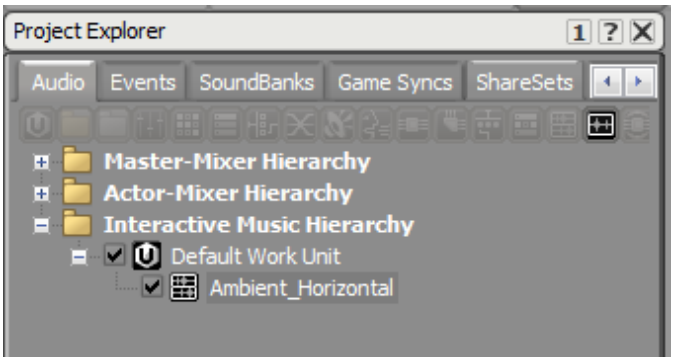
横型のアプローチ

ゲームで森の中を進みながら未知の冒険や隠れた危険を覚悟するプレイヤーに対して、怪しい雲行きを音楽で表現できます。常に新鮮で変化に富むダイナミックな音楽を提供するには、プレイヤーが危険に近づくにつれ変化する横型レイヤーのアプローチを導入します。ゲームエンジンが送出するプレイヤーから敵や場所までの距離のパラメータを利用して、プレイ中の音楽の激しさを変えます。

ここで横型アプローチ用に準備したミュージックコンテンツは、重ねられたトラック10本のレイヤーで、1つのミュージックセグメント内で同時に再生させます。

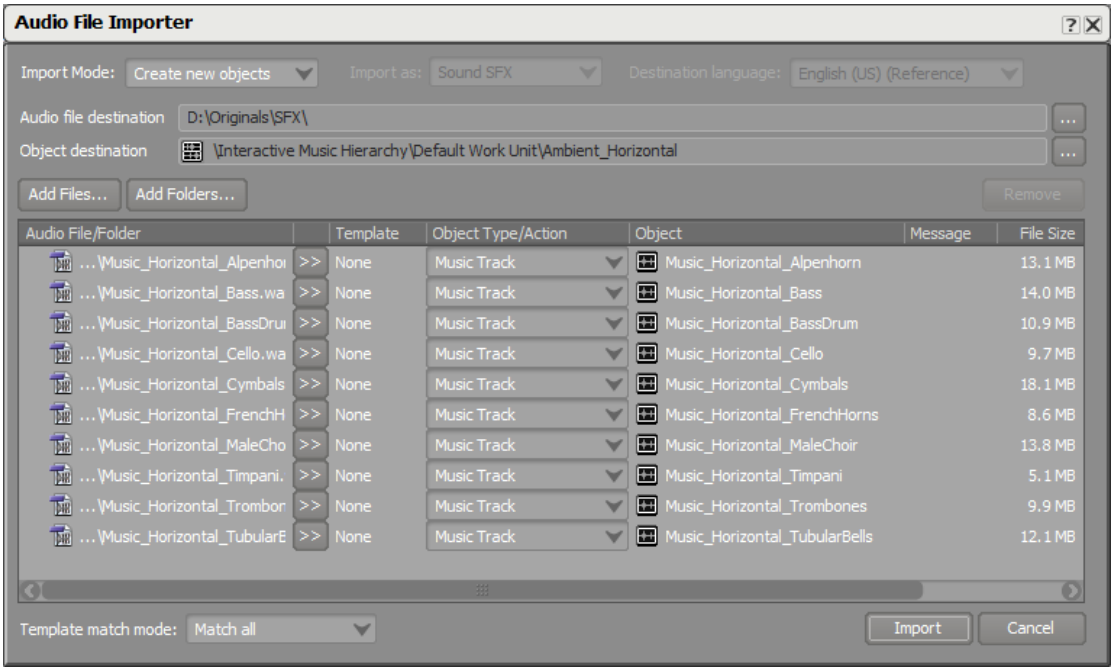
アンビエントミュージックセグメントの作成

インタラクティブミュージック階層のデフォルトワークユニットに、ミュージックセグメント“Ambient_Horizontal”を子として追加して、アンビエントミュージックシステムの構築を開始します。



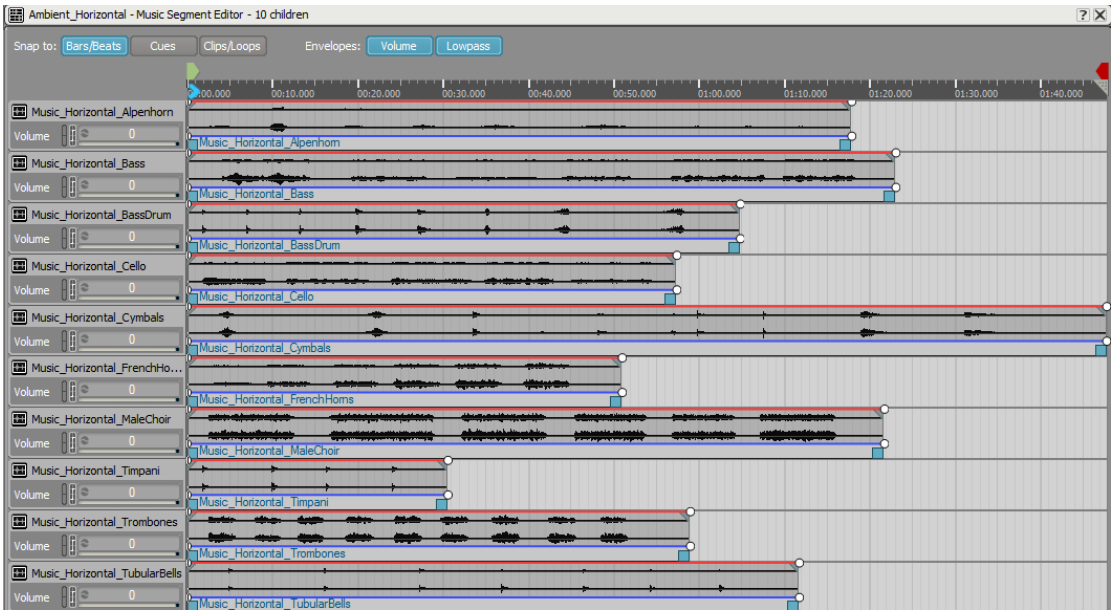
横型ミュージックシステムで、ミュージックセグメント“Ambient Horizontal”を作成

次に、このセグメントのトラックとなるミュージックファイルを、Audio File Importer機能を使うか、セグメントにドラッグ&ドロップしてインポートします。



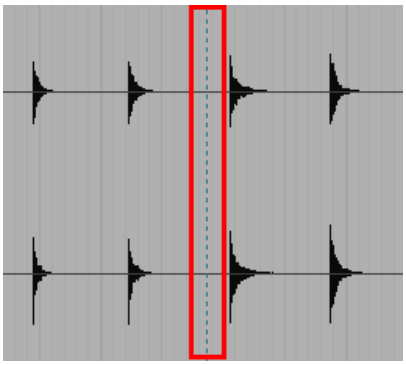
Audio File Importer機能を使って複数のミュージックトラックをセグメントにインポートする

ミュージックファイルをトラックとしてセグメントに追加したら、Music Segment Editor画面で確認して、ミュージックセグメントの中でトラックを並べ替えたり編集できます。



Music Segment Editor画面でトラックを編集

トラックの基本要素がクリップで、単一のwavファイルを表す長方形のエリアとして表されます。クリップをトラック上で左右にドラッグすることで、いつ再生されるのかを調整します。別のトラックに移動するには、クリップを移動先のトラックにドラッグします。また、トラックの中で複数のクリップをオーバーラップさせることも可能です。クリップの長さは、クリップのハンドルを内側に移動すると短くなり、外側に移動すると延長できます。延長すると、クリップは反復されます。1回の繰り返しを1ループと呼びます。クリップの中のループポイントには縦の点線が入ります。



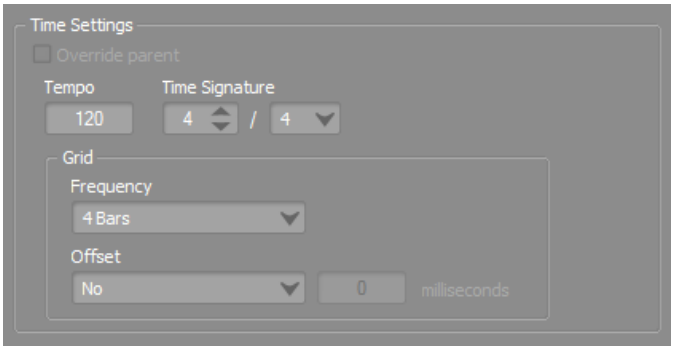
クリップのループポイントを示す点線

Music Segment Editor画面では、Cue（キュー）やカーソルが表示されます。キューはセグメントの重要ポイントを表すマーカーで、エントリーポイントやエグジットポイントなどに使います。

Music Segment Editor画面でEntry CueやExit Cueを調整

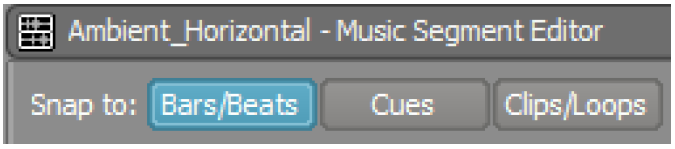
キューをカスタム設定して、プロパティの変更、トランジションの実行、スティンガーの再生などのポイントを示すこともできます。セグメントを再生するとPlay Cursor（再生カーソル）が再生箇所を追いますが、カーソルを動かして再生の開始点をコントロールすることもできます。End Cursor（終了カーソル）はセグメントの終了を示します。

セグメントのTime Settings（時間設定）はGeneral Settingsタブで設定でき、そのセグメント内にあるトラックに継承されます。セグメントは親コンテナの時間設定を継承するか、オーバーライドしてさらに細かく設定されます。



Tempo、Time Signature（拍子記号）、Grid単位の設定

マルチレイヤーで構成された今回の音楽のテンポは120BPM、拍子記号は4/4です。セグメントとその中にあるトラックの時間設定が決まれば、Music Segment Editor画面のSnap to機能を使って、クリップ、カーソル、キューマーカーなどをより正確に調整できます。

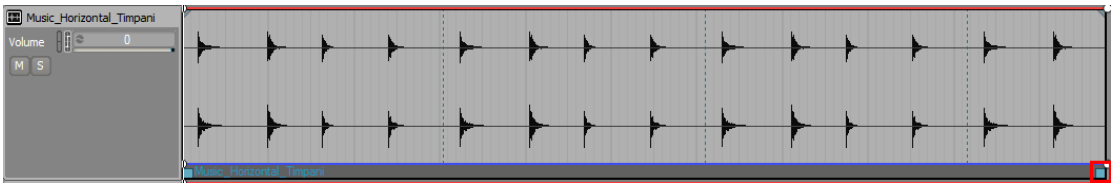


Music Segment Editor画面でSnap to機能を設定

トラックを整える

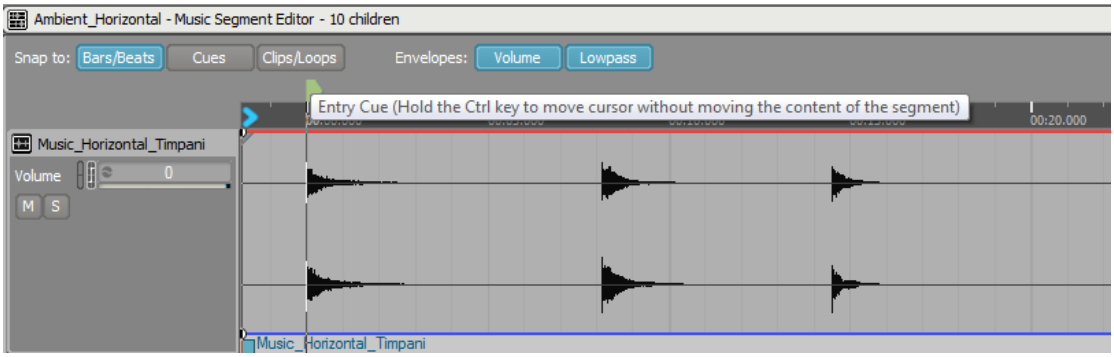
それでは、このアンビエントセグメントを音楽システム全体で使えるように、キューやクリップハンドルの追加や調整を行います。

まず、複数のパーカッションのトラック（ティンパニー、バスドラム、チューブラーベル）をシンバルのトラックの長さに合わせて延長（ループ）します。各トラックの右下にある青いクリップハンドルをクリックして、基準トラックの長さに合わせて大体の位置までドラッグします。



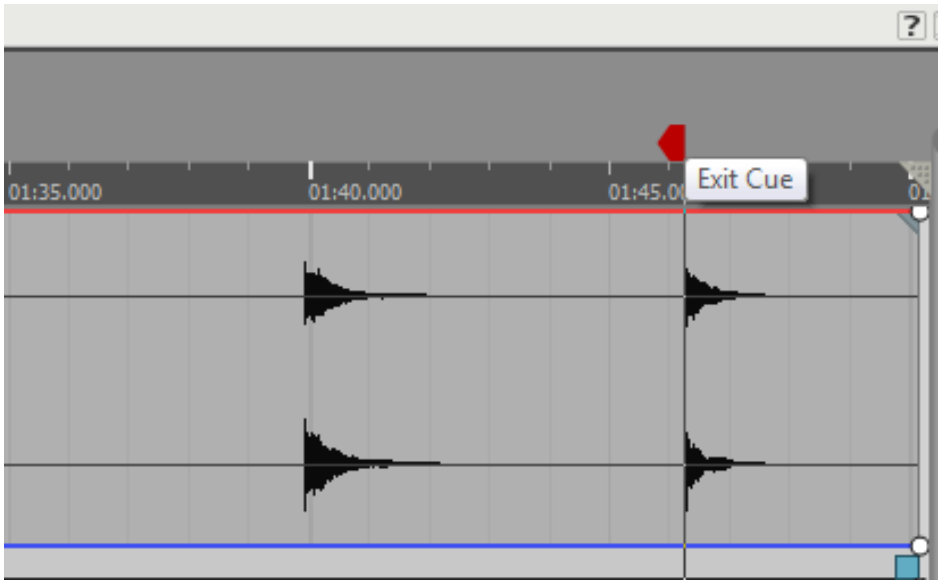
クリップハンドルでトラックを延長してループさせる

次にティンパニーの第1ビートにEntry Cueカーソルを合わせるために、Ctrlキーを押しながらセグメントを動かさずにカーソルだけを移動します。これでセグメントのエントリーキューが決まり、アンビエントミュージックセグメントへとトランジションするミュージックが必ずビートで始まるように設定できました。Entry Cueカーソルの左側はセグメントのプレエントリー（エントリー前）領域です。プレエントリー領域をゲーム中に再生するかどうかは、トランジション設定で決めます。



Entry Cueカーソルをティンパニーの第1ビートまで移動

次にExit Cueカーソルをティンパニーの最終ビートまでドラッグします。これにより、セグメントのエグジットキューの位置を決めます。Exit Cueカーソルの右側はセグメントのポストエグジット（エグジット後）領域です。ポストエグジット領域をゲーム中に再生するかどうかは、トランジション設定で決めます。

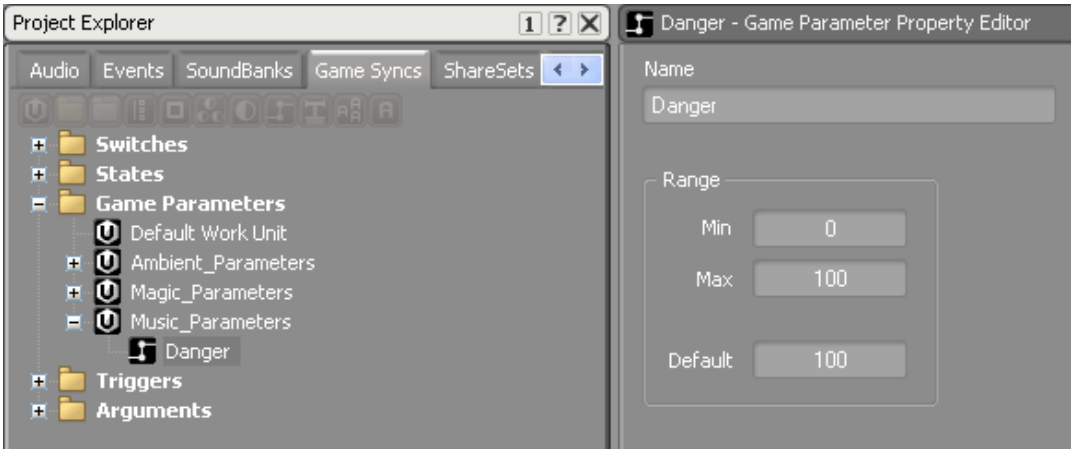


Exit Cueカーソルの調整

ダイナミックな「危険」の表現

アンビエント音楽のエントリーポイントとエグジットポイントが決まり、最終的な長さも決定されたところで、危険性に応じてダイナミックに変化するようにボリュームを設定します。ゲーム側で森に潜む怪物に危険値を1体ずつ付与して危険を表現して、オーディオエンジンにその値を送信する実装でも、ボリューム変化は直接オーサリングアプリケーションで試聴できます。

まず最初に、Project Explorer画面のGame Syncsタブで、“Danger”（危険）というゲームパラメータを作ります。



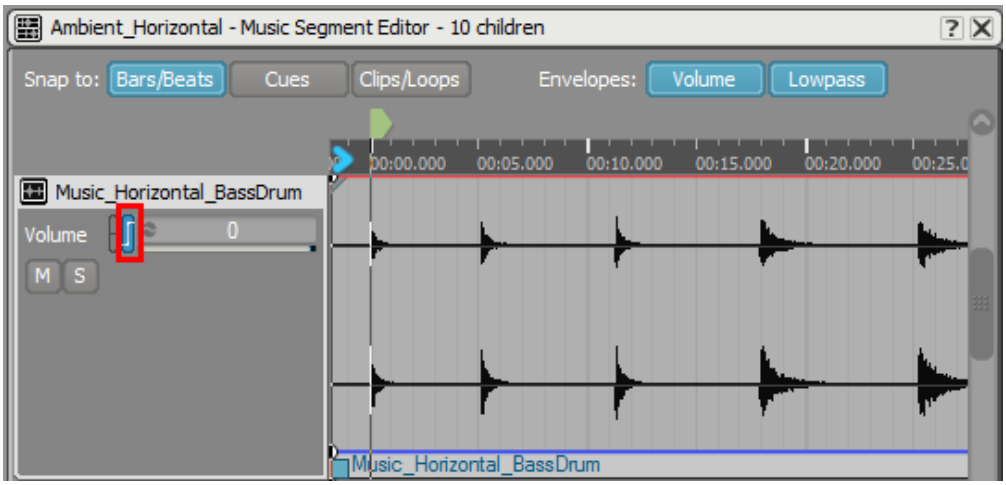
アンビエントミュージックシステムで使う“Danger”パラメータの作成

“Danger”パラメータをボリューム用RTPCと組み合わせて、アンビエントミュージックシステムの複数のトラックに対して適用します。RTPCを使って設定できるトラックのプロパティは他にもあり、ステート、エフェクト、ポジショニングなどの設定や、バス、プライオリティ、リミッターなどの設定にも使えます。

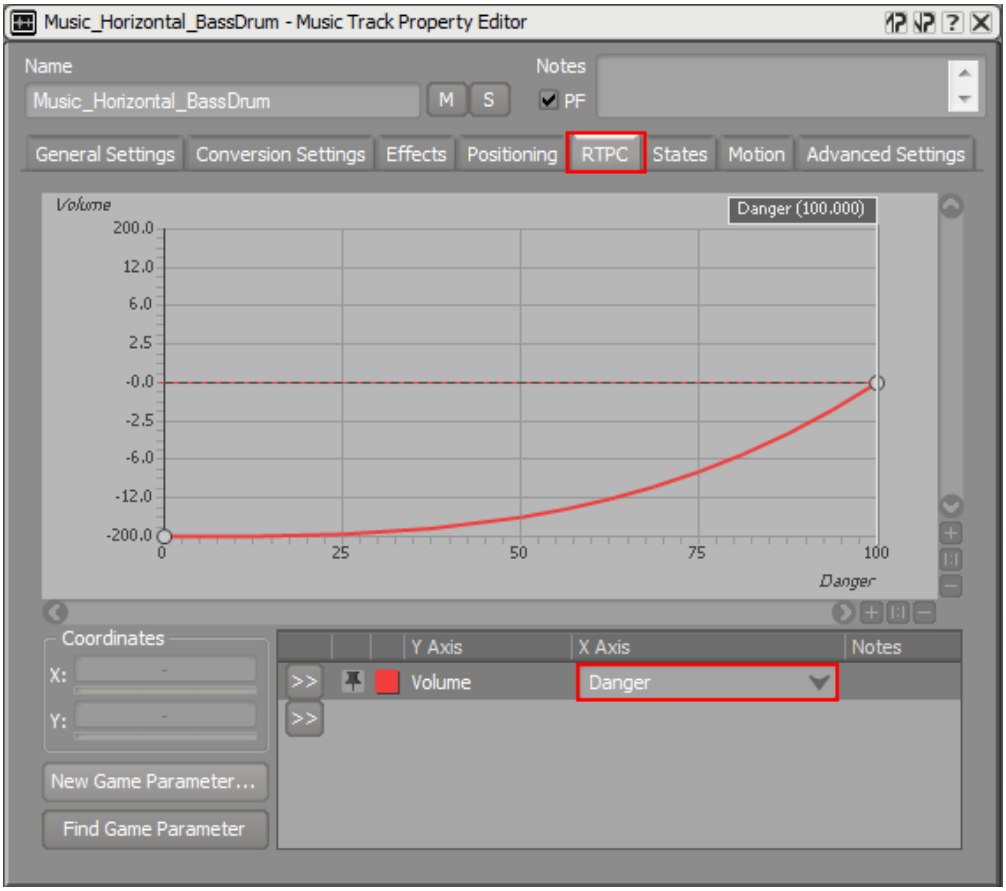
トラックにRTPCを追加

アンビエントミュージックシステムをマルチレイヤーにする目的は、セグメントの中の主要トラック、具体的にはバスドラム、シンバル、フレンチホルン、男性コーラス、そしてトロンボーンなどを選んで、そのボリュームを変化させて雰囲気を変えることです。

Bass DrumトラックのRTPCアイコンをダブルクリックして、トラックのRTPCタブを開き、ここでVolume設定を追加して、ゲームパラメータ“Danger”に結びつけます。

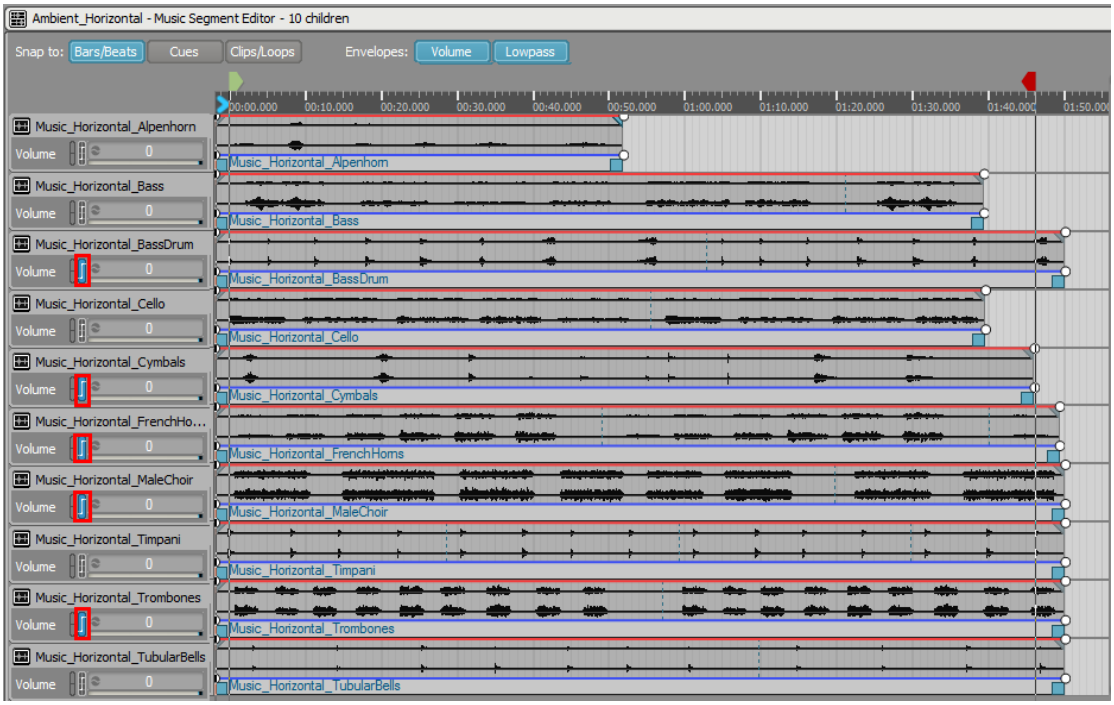


RTPCアイコンをクリックしてProperty Editor画面を表示



Bass Drumトラックのボリューム設定を追加してゲームパラメータ“Danger”に結びつける

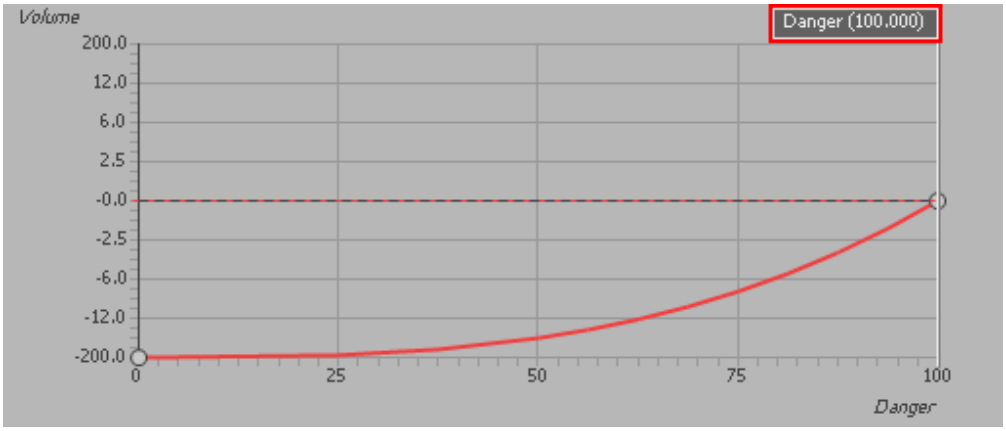
この設定変更を他のトラックでも採用すれば、“Danger”のパラメータ値が増加すると、他の楽器もフェードアップして音楽の激しさが何層にもわたって表現できます。



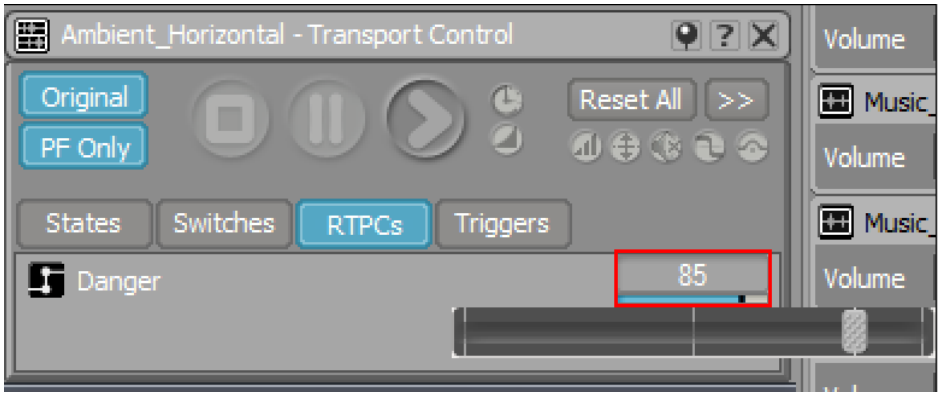
RTPCを使ったトラックを含む、完成したアンビエントミュージックシステム

RTPCの試聴

ここで加えた変更を試聴するには、RTPCビューのゲームパラメータカーソルを使うか、Transport Control機能のRTPCタブを開きます。



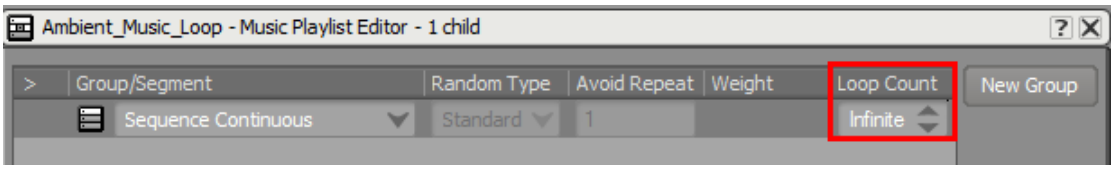
ゲームパラメータカーソルを使ってRTPCを試聴



Transport Control機能を使ってRTPCを試聴

ミュージックセグメントのループ設定

インタラクティブミュージック階層の中でミュージックセグメントをループするには、プレイリストコンテナに子セグメントとして入れ、グループの中でループのプロパティを管理します。セグメントを追加すると、デフォルトグループのLoop Count（ループ回数）にループのプロパティが表示されます。矢印を下にクリックすると回数がInfinite（無限）になり、オーディオシステムによって停止されるまでグループはセグメントを再生しつづけます。



無限にループさせるため、Loop CountをInfinateに設定

Designer Note

セグメントのループと、セグメント内のオーディオクリップのループを混同しないように、気をつけて下さい。両者のアプローチの違いについては、Wwise Knowledge Baseをご参照下さい。 Looping and streaming of audio clips and interactive music

本節のまとめ

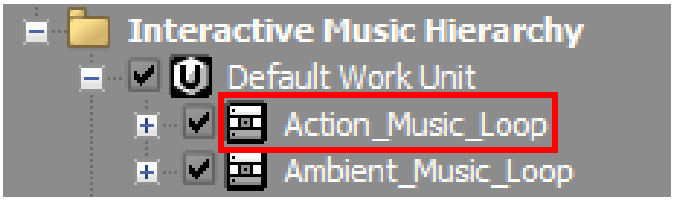
ゲーム内で発生する危険性のレベルに合わせて、ダイナミックにループする音楽ができました。複数レイヤーを設定してボリューム調整にRTPCを使うことで、ゲーム中のヒーローの冒険を反映する探検音楽のループが完成しました。

縦型のアプローチ

ゲームプレイの探検段階で音楽を再生するダイナミックな横型システムができたところで、バトル開始の音楽も準備します。ゲームで勝利を達成したヒーローのためには縦型アプローチを採用し、アクション専用の音楽を使って激しさのレベルを増減させます。ループに多少のランダム性を導入するためにアクション音楽を2つに分け、それぞれのバリエーションを作り、ブリッジ役のセグメントもプレイリストに入れます。これらをグループ機能を使ってミュージックプレイリストコンテナの中で編成します。

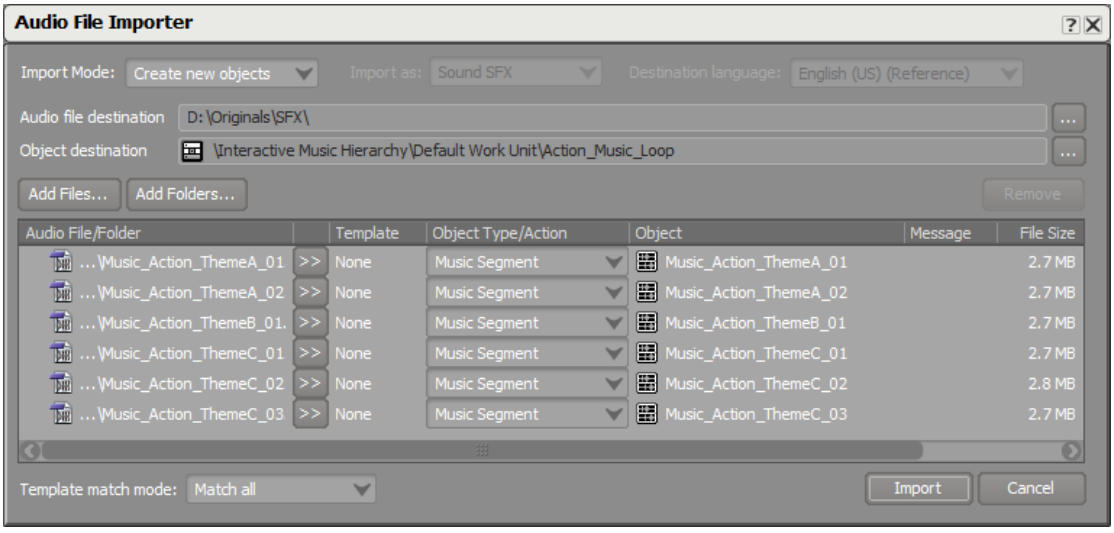
グループと動作設定

まずミュージックプレイリストコンテナを作成して、ミュージックシステムでループさせるアクションミュージックをここにまとめます。



プレイリスト“Action_Music_Loop”を作成

次に、使用するwavファイルをインポートします。デフォルトで、ファイルはセグメントとしてミュージックプレイリストコンテナに追加され、後でMusic Playlist Editor画面で並べ直すことができます。



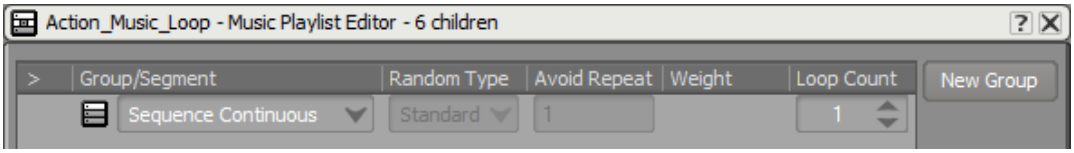
ミュージックプレイリストコンテナにアクションミュージックのwavファイルを追加

アクションミュージックセグメントをプロジェクトに追加できたところで、ミュージックプレイリストコンテナにドラッグ＆ドロップします。追加した後、グループ分けして整理することもできます。プレイリストは、プレイリストコンテナがゲームエンジンに呼び出された時に、どのオブジェクトをどの順番で再生するかを決めます。

グループやセグメントの4種類の再生方法：

- **Sequence continuous（連続再生）**：グループが再生されると、グループ内の全てのミュージックオブジェクトを順番に再生する。
- **Sequence step（ステップ再生）**：グループが再生されると、グループにあるミュージックオブジェクトを1つだけ、再生する。もう一度グループが再生されると、グループ内にある次のミュージックオブジェクトを再生する。
- **Random continuous（ランダム連続再生）**：グループが再生されると、グループ内の全てのミュージックオブジェクトを、ランダムな順番で連続再生する。
- **Random step（ランダムステップ再生）**：グループが再生されると、グループ内からランダムに選んだミュージックオブジェクトを1つだけ、再生する。

これ以外にもランダム再生の種類、ウェイト付け設定、繰り返し再生の変数などがあり、Music Playlist Editor画面から設定できます。



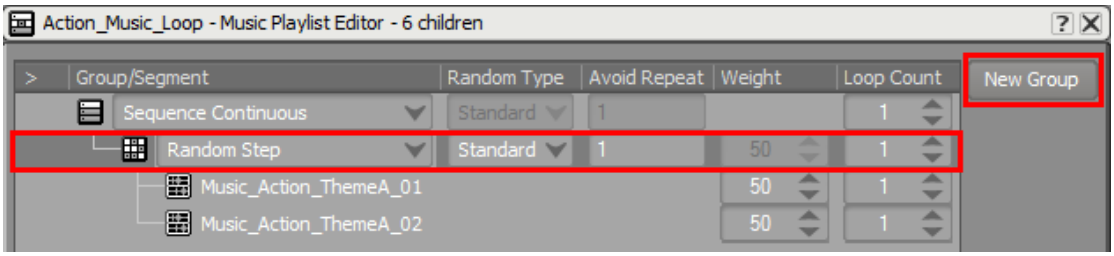
Music Playlist Editor画面でセグメントグループや再生プロパティを設定

Music Playlist Editor画面でグループを並べる

アクションミュージックファイルはテーマ別にA、B、Cの3つに分かれています。

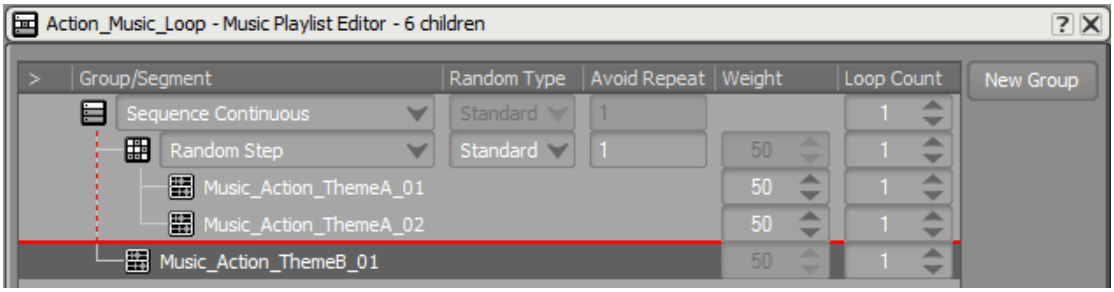
ミュージックプレイリストコンテナの中にグループを作り、テーマごとに分けます。今回はミュージックファイルとして“Action Theme A”を2種類、“Action Theme B”を1種類、“Action Theme C”を3種類使い、変化するアクションループを作るために順番に並べます。

まず新規グループをRandom Step再生に設定して、2種類の“Action Theme A”をドラッグします。



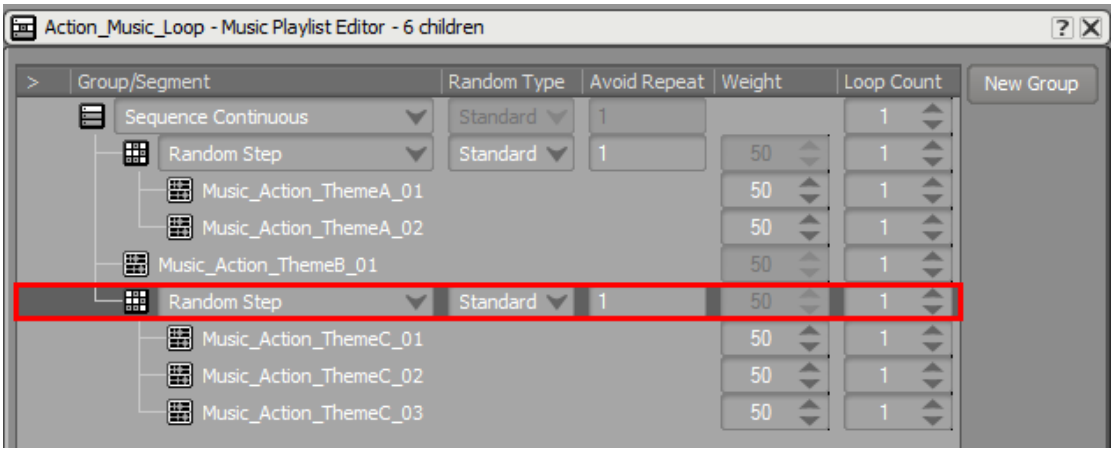
“Action Theme A”用のContinuous Randomグループを新規作成

次に、“Action Theme A”と“Action Theme C”のセグメントで作った2つのランダムグループをつなげるために、“Action Theme B”のセグメントをブリッジとして配置します。“Action Theme B”を挿入するには親グループSequence Continuousにドラッグし、赤いインジケーターを目安に位置を決めます。



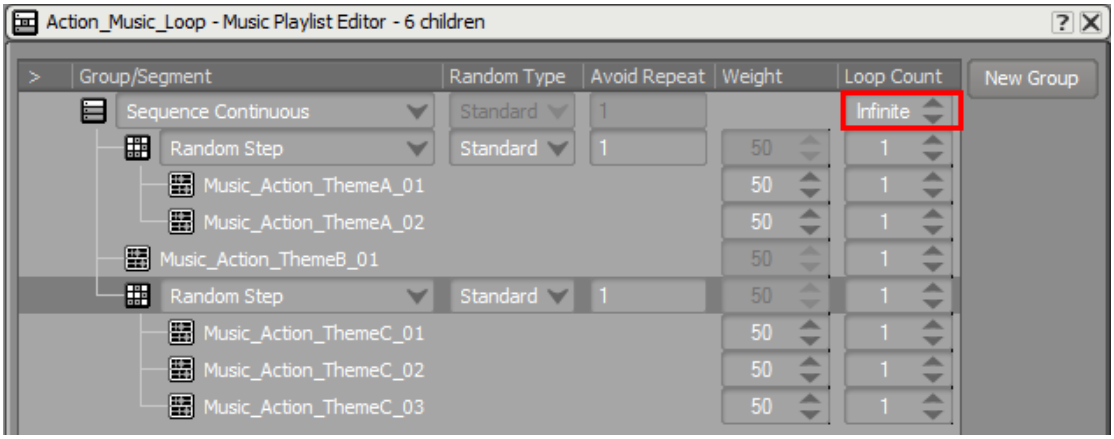
ランダムグループ“Action Theme A”の後に“Action Theme B”を配置

最後に、“Action Theme C”はRandom Step再生にしてグループ内の3つのバリエーションから1つを選び再生します。



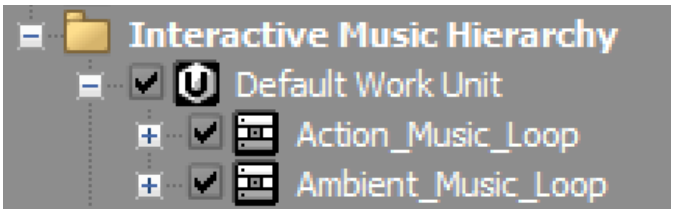
“Action Theme C”用のRandom Stepグループを新規作成し、配置する

親のシーケンスコンテナのループ回数をInfiniteにすれば、オーディオエンジンから変更要求がない限り、繰り返しループします。



プレイリストコンテナのLoop CountをInfiniteに設定

これで、アンビエントミュージックとアクションミュージックの2種類のプレイリストコンテナが完成しました。



“Action”と“Ambient”の2種類のプレイリスト用コンテナ

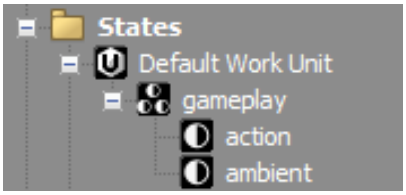
本節のまとめ

セグメントのプレイリストを使ってランダムに順番が変わるアクションミュージックのループを設定できました。セグメントの種類に合わせて再生方法を変えることで、展開する音楽の役割や進展、ストーリーがはっきりとします。

音楽の種類をステートを使って切り替える

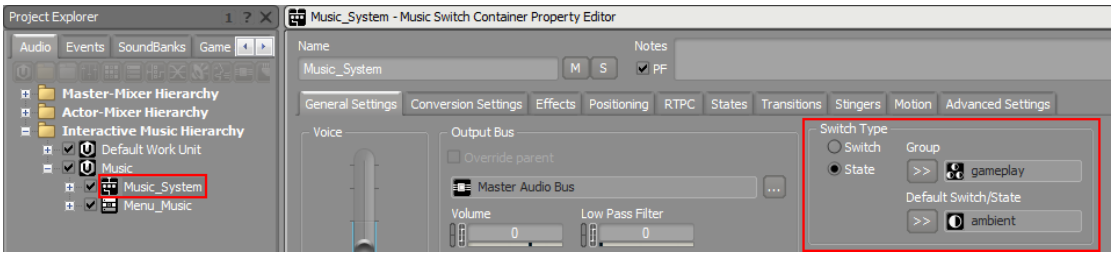
シナリオによって違う種類の音楽を設定した後は、ゲームプレイの変化に音楽が反応できるように実装します。これはステート（またはゲームステート）を使って行い、ミュージック用コンテナを切り替えたり、サウンドや音楽の他の面を変えます。

ミュージックスイッチシステムを作るには、まず音楽の種類が切り替わる時に使う、ゲームから伝達されるステートを定義します。今回の例ではアクションミュージックとアンビエントミュージックを切り替えます。ステートは、現在のサウンド、ミュージック、またはモーションのプロパティをグローバルレベルで変えるゲーム側の状況を表します。ミュージックスイッチコンテナはアクターミキサー階層のスイッチコンテナと同じ方針で動き、時間とテンポの機能が加わることでコンテナの切り替えに合わせて音楽的トランジションが適用されます。



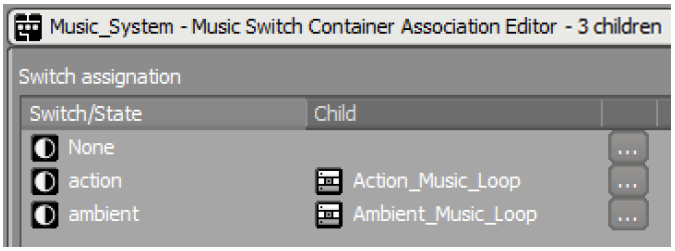
ゲームプレイの状況によって音楽の種類を切り替えるためのステートを作成

既存の“ambient”ミュージックと“action”ミュージックのループリストを、ミュージックスイッチコンテナの子とすることで、新規ステートグループ“gameplay”を適用できます。



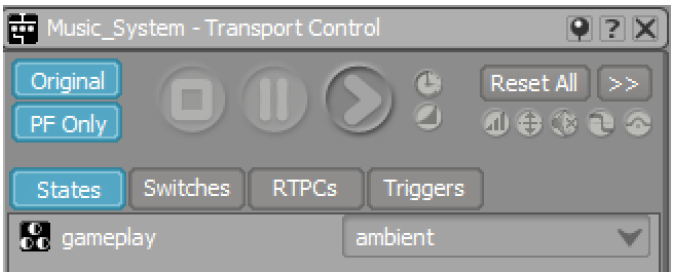
Music Switch Containerにステートグループ“gameplay”を設定し、デフォルトのステートとして“ambient”を設定

これで階層からドラッグ＆ドロップするか、選定ボタンをクリックして、Music Switch Container Association Editor画面で、各ステートに該当するミュージックプレイリストコンテナをアサインできます。



**Music Switch Container Association Editor画面で、
ミュージックプレイリストコンテナをステートにアサインする**

ミュージックプレイリストコンテナを該当するステートにアサインしたので、Transport Control機能のStatesボタンを使って、音楽が変わるときのデフォルトのトランジションを試聴できます。



ステートに合わせて変わる音楽を試聴

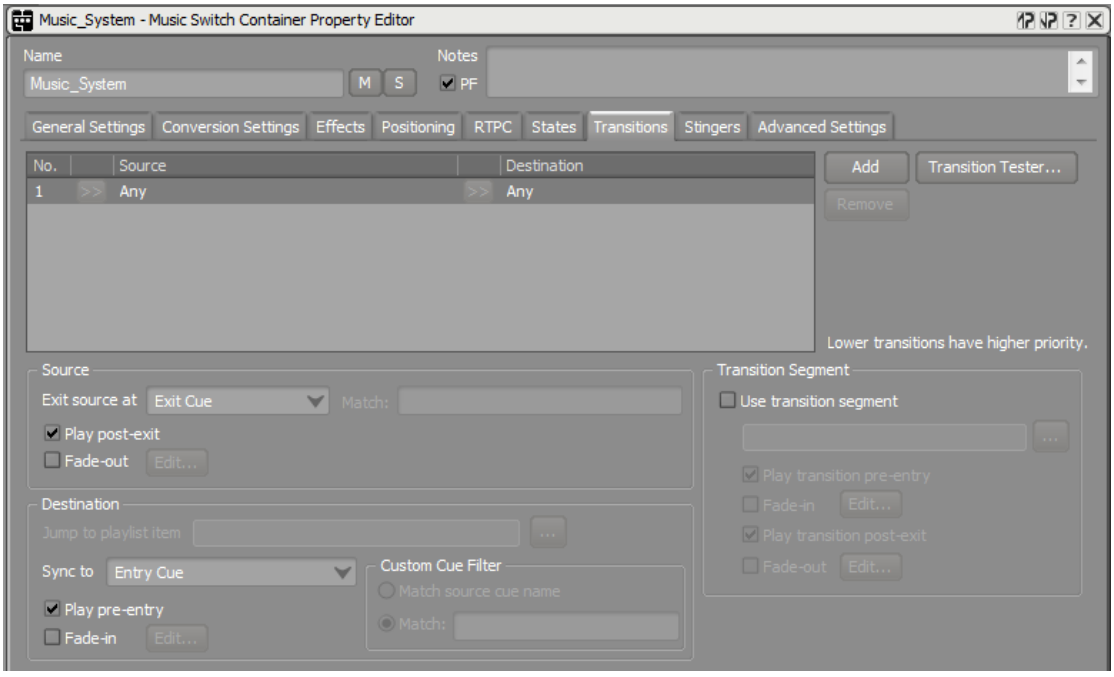
インタラクティブミュージックのトランジションの定義

ミュージックセグメントやミュージックプレイリストコンテナが切り替わる時の、トランジションをより細かくコントロールするには、ミュージックスイッチコンテナを利用し、具体的なトランジション動作をオーサリングします。

ミュージックプレイリストコンテナやミュージックスイッチコンテナの中にあるミュージックオブジェクトを切り替える時のトランジションは、Transitionsタブで設定します。トランジションとは、再生中のミュージックオブジェクトから別のものに変わる時に使う音楽的動作のことです。トランジションにはソース（出所）とデスティネーション（行き先）が必ずあります。ソースとデスティネーションの間をつなぐ音楽的ブリッジとして、トランジションセグメントと呼ばれるセグメントを使います。

Transitionsタブを開くとマトリックスが表示され、ここでコンテナ内の各オブジェクトが他のオブジェクトに切り替わる時のルールを設定します。オブジェクトごとに適用するルールや、複数のオブジェクトに適用する一般的なルールを作成できます。トランジション時のデフォルトのルールとして“Any to Any”（どれからどれにでも）があり、定義されていない残りのトランジションにデフォルトで適用されます。

トランジションの一覧がTransitionsタブに降順で表示されます。トランジションが必要な状況になるとWwiseはこの一覧を下から検索し、今の状況に合うトランジションを見つけるまで順に探します。合致するトランジションがなければ、デフォルトのトランジション“Any to Any”が使われます。

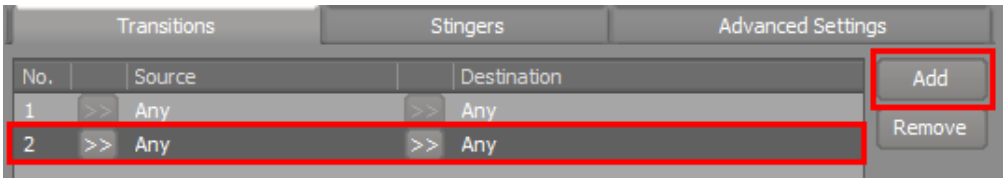


Transitionsタブとトランジションマトリックス

トランジションのオーサリング

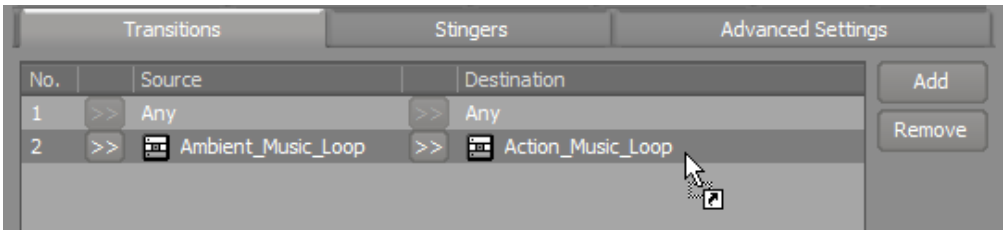
それでは、アンビエントループからアクションループへ音楽的に移行し、再びアンビエントへ戻るまでの専用のトランジションセグメントを追加します。まず最初に“Ambient Music Loop”から“Action Music Loop”へのトランジションを定義し、トランジションマトリックスに新規トランジションを追加して、動作を定義します。

Addボタンで新規トランジションを追加します。



トランジションマトリックスに新規トランジションを追加

次に、プレイリスト“Ambient Music Loop”をソース、プレイリスト“Action Music Loop”をデスティネーションとして、ソースとデスティネーションのミュージックオブジェクトをドラッグ＆ドロップします。



トランジションマトリックスにミュージックオブジェクトをドラッグ＆ドロップ

トランジション動作の定義

それでは、ソースを出る時とデスティネーションに入る時の動作、そしてトランジションセグメントがあればそれを設定して、ソースとデスティネーションの間のトランジション動作をオーサリングします。

トランジションは1つのミュージックオブジェクトから別のものへの単純な切り替えです。トランジション機能の威力が発揮されるのは、ソースとデスティネーションをカスタム設定してユニークな音楽の推移を作成した時です。ソースとデスティネーションのプロパティを設定すれば、オブジェクト間の切り替えをスムーズで音楽的に違和感のないものにできます。

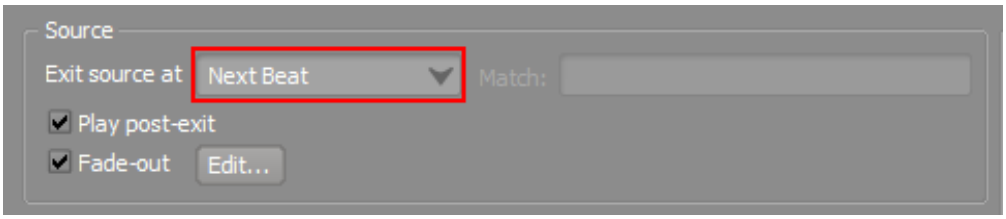
オブジェクト間のトランジションを自由に細かく操作できるように、ソースからのエグジット位置とデスティネーションへのエントリー位置を、いくつかの選択肢から別々に選べます。

ソースからエグジットする（出る）方法として、選択肢が多数あります。

- **Immediate（即時）**：ソースの再生をすぐに止める。
- **Next Grid（次のグリッド）**：ソースの再生を次のグリッドで止める。グリッドとは、ミュージックオブジェクトを任意の位置でバーチャルに区切る方法。
- **Next Bar（次の小節）**：ソースの再生を次の小節で止める。
- **Next Beat（次のビート）**：ソースの再生を次のビートで止める。
- **Next Cue（次のキュー）**：カスタム設定のキュー、またはエグジットキューのいずれであっても、ソースの再生を次のキューで止める。
- **Next Custom Cue（次のカスタムキュー）**：ソースの再生を次のカスタム設定したキューで止める。再生中のミュージックセグメントにカスタムキューがなければ、次のセグメントに入り、カスタムキューを見つけるまで再生する。
- **Exit Cue（エグジットキュー）**：ソースの再生をエグジットキューで止める。

アンビエントからアクションへのトランジション

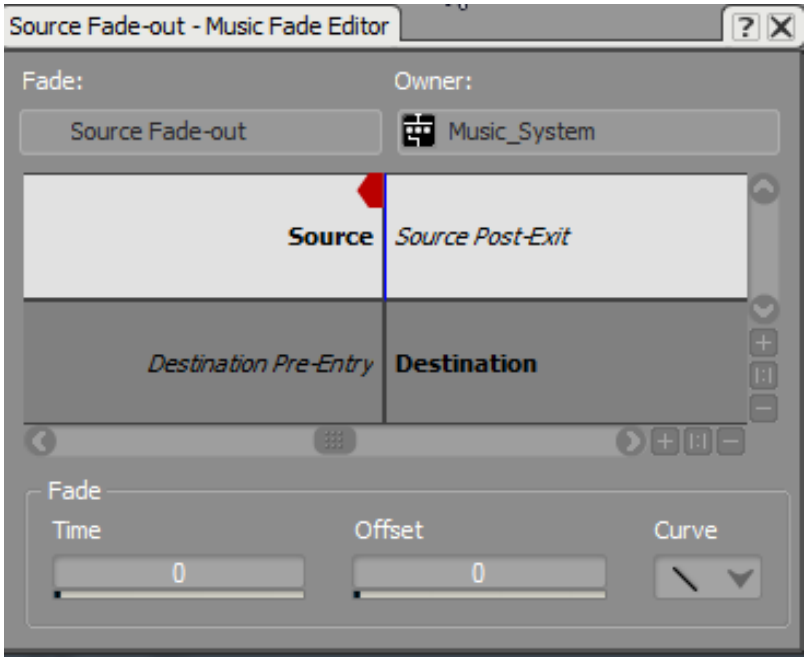
次のビートで切り替わるトランジションは、アンビエントミュージックからアクションミュージックに素早く移行するのに適しているNext Beatを、Exit Sourceのウィンドウで指定します。



ソースのExit source at（ソースをエグジットするタイミング）の設定

デフォルトで、トランジション時にソースのPlay post-exit（ポストエグジット部分を再生）が設定されていますが、ソースのポストエグジットを再生するのは、エグジットキューでソースがエグジットするか、エグジットキュー以降でフェードアウトする場合だけです。それ以外の場合はトランジション中にポストエグジットは再生されません。

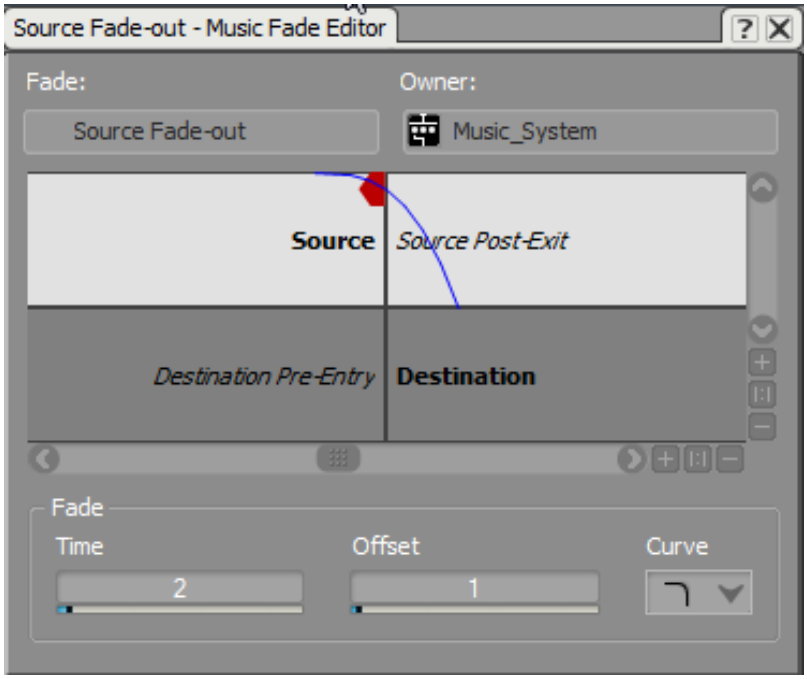
今回のトランジションでは再生をフェードアウトで終了させたいと思います。Fade-outを選び、Editをクリックしてmusic Fade Editor画面を開きます。



Music Fade Editor画面でSource Fade-outを設定

Music Fade Editor画面で、ミュージックオブジェクトが切り替わる時に使うフェードのプロパティを個別に設定できます。フェードのプロパティ画面で設定できるのは、デスティネーションオブジェクトのフェードイン設定、ソースオブジェクトのフェードアウト設定、トランジションセグメントントのフェードイン、フェードアウト設定です。また、フェードの長さやオフセットを設定したり、カーブの形状を変更することで、トランジション部分を細かくカスタム化できます。

今回の例では、アンビエント音楽に対して2秒間のフェードを、1秒オフセットして設定し、急降下するカーブとしたので、アンビエント音楽が素早くフェードアウトして、アクション音楽のプレイリストへの移行が自然に聞こえます。



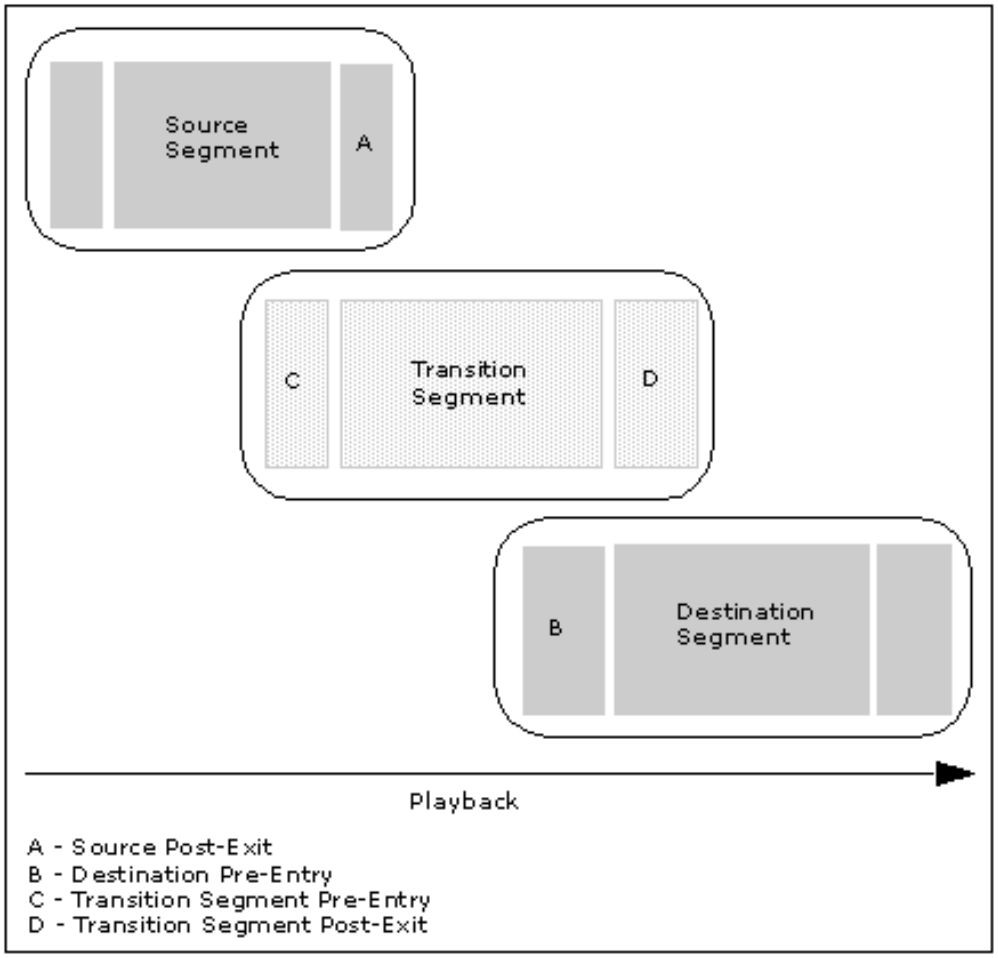
アンビエントからアクションへトランジションする際の、フェードの詳細設定

設定したトランジションは、前述の方法で試聴できます。

アクションからアンビエントへのトランジション

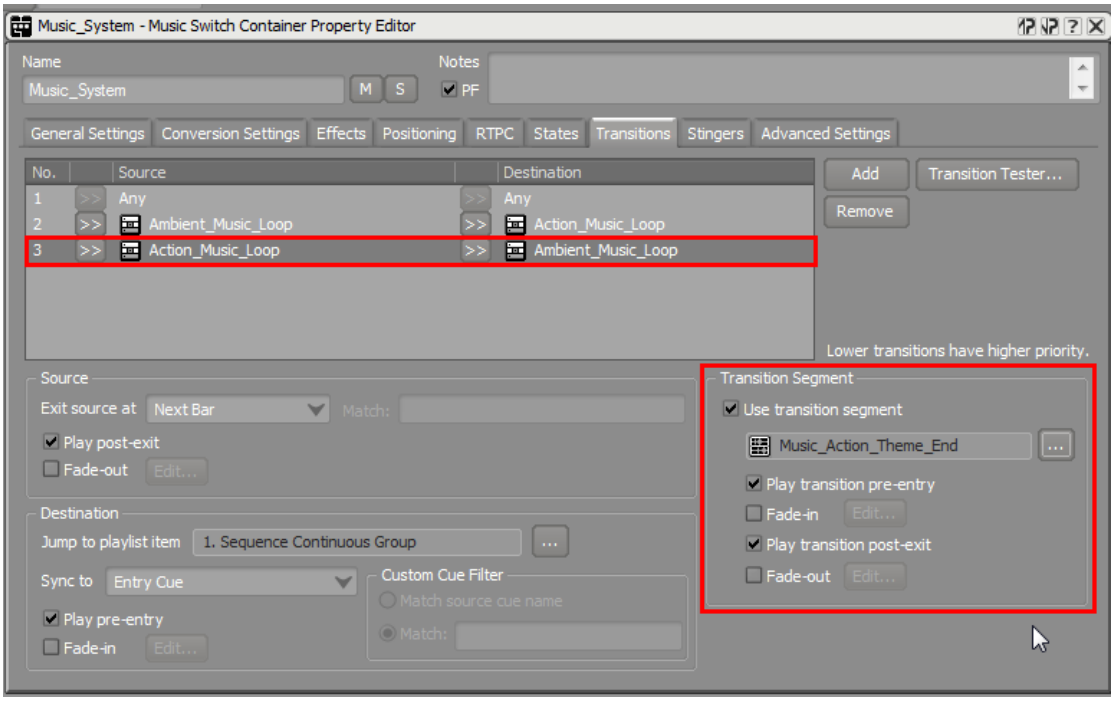
アクションミュージックからアンビエントミュージックへのトランジションは、アクションミュージックのプレイリストがループしているために特別な配慮をする必要がありますし、アクションミュージックの終了が意味するゲーム側のドラマチックな変化を充分に訴えるためにも工夫が必要です。

場合によって、トランジション時のソース終了とデスティネーション開始の上に別の音楽を重ねて再生した方が、聞こえがよくなります。この音楽的ブリッジをトランジションセグメントと呼び、全てのトランジションに使えます。下図は、音楽のトランジションにおいてソースとデスティネーションの間でトランジションセグメントが再生される様子を示します。



ソース、デスティネーション、そしてトランジションセグメントのプレエントリー部分やポストエグジット部分を自由に組み合わせて、よりシームレスなトランジションを作成できます。

ソースのプレイリスト“Action Music Loop”からデスティネーションのプレイリスト“Ambient Music Loop”に戻る時に両者をつなげるために、トランジションマトリックスで新規トランジションを作成して、トランジションセグメント“Action Theme End”（アクションテーマ終了）を追加します。



ソースのプレイリスト“Action Music Loop”とデスティネーションのプレイリスト“Ambient Music Loop”の間に、Transition Segmentを設定

本節のまとめ

プレイヤーにシームレスな音楽的エクスペリエンスを提供するためには、ミュージックプレイリストコンテナやセグメントの間の、微妙な関係やトランジションが重要です。ゲームステートに応じるスイッチ機能の導入と、アクションループやアンビエントループをつなげるトランジションセグメントの設定で、自然な展開で音楽がゲームに反応しながら変化します。

ミュージックのまとめ

本章では、ゲームの激しさに応じて横型と縦型に同時に機能するミュージックシステム、そしてゲームのステートに応じて縦型に機能するミュージックシステムと、ゲームの状況に反応できるシンプルなミュージックシステムを紹介しました。また“Danger”のレベルによって、アンビエントミュージックを構成する複数のトラックのボリュームをダイナミックにコントロールできる一方で、テンポを利用したトランジションはステートが変化する時に音楽をシンクさせるのに役立ちます。これらのテクニックを組み合わせると、プレイヤーのエクスペリエンスにカスタム化されたミュージックサウンドトラックが生まれます。

本章では、以下を説明しました。

- インタラクティブミュージックやダイナミックミュージックの、ゲームにおける役割
- 実装方法の種類と目的
- 使用するミュージックコンテンツの事前処理
- インタラクティブミュージック階層とアクターミキサー階層の比較
- ミュージックセグメント、ミュージックプレイリストコンテナ、ミュージックスイッチコンテナの使用目的

プロセスの説明

- アンビエントミュージックをダイナミックにコントロールするRTPCを利用した横型アプローチ
  - ミュージックセグメントの作成
  - トラックの長さの調整
  - ミュージックトラックのループ
  - ミュージックプレイリストコンテナへのミュージックセグメントの追加
  - プレイリストのループ
  - ゲームパラメータの作成
  - RTPC情報による、ミュージックトラックのボリューム変更
- アクションミュージックのセグメントをランダムに組み合わせるプレイリストを利用した、縦型アプローチ
- ゲームのステートに応じた音楽の切り替え
  - ソースとデスティネーション間のトランジションマトリックスの定義
  - トランジション時の動作の定義
  - ソースをエグジットする時の変数の設定
  - ミュージックフェードエディターの利用
  - トランジションセグメントの定義

詳細設定の説明

- Interactive Music Layout画面の活用

オブジェクトの作成方法

- ゲームのステートに合わせてアンビエントミュージックとアクションミュージックを切り替える、音楽システムのスイッチコンテナ
- アンビエントミュージックのセグメントとミュージックトラック10本が入って、ゲームの危険パラメータ値に応じてボリュームが変化するRTPCを設定した、ループするミュージックプレイリストコンテナ
- 危険をシミュレーションするために使うゲームパラメータ
- シームレスにループするアクションミュージックを作成するために、ミュージックセグメントをランダム再生するプレイリストコンテナ
- アクションからアンビエントへとステートが切り替わる時にトランジションセグメントとして使う、アクションミュージックのエンディングテーマ

参考ドキュメントとチュートリアル

Wwise Knowledge Base - Advanced settings playback limits and interactive music

Wwise Knowledge Base - Looping and streaming of audio clips and interactive music

Wwise Knowledge Base - Understanding the Interactive Music Engine

Wwise Knowledge Base - Interleaved Streaming in Interactive Music

Wwise Knowledge Base - Using States with the Interactive Music Hierarchy

Video Tutorial - Creating Interactive Music Structures

Video Tutorial - Defining Interactive Music Transitions

Video Tutorial - Creating Stingers

Game Sound Design - Making Music Interactive: Elaboration of the Feature Set in Wwise

Game Sound Design - Dynamic Music Creation Using Wwise

第8章 MIDIのアドベンチャー

概要

風に曝された戦場で、敵に有意な状況が変るのを待っているヒーロー。彼は、歴史をすべて知り尽くした目を持ち、昔の話を知り尽くしている神官に知識を求める。すでに忘れ去られた錬金術や戦略が、正当な王位継承者である彼の立場を取り戻すのに役に立つのでは考えていたのだ。古い手法の理解を武器にすれば、失敗はあり得ない。道具を手に、そして過去のパワーをもって究極の理想を創造する。

合成とサウンドサンプルと同時にMIDI（Musical Instrument Digital Interface）を利用した包括的なパイプラインは非常にパワフルな組み合わせです。標準化されたMIDIフォーマットは、最新式のDAW（Digital Audio Workstation）の中心であり、音、速度、その他再生用の制御情報をやり取りする補正可能なフォーマットです。このデータは Wwise 内でサウンドサンプルもしくはシンセサイザーにマップすることができ、音楽やサウンドデザインのための効果的なワークフローを作り上げます。MIDIの実力と柔軟性は、DAWワークフローの一部として良く知られており、その利点をWwiseでも利用することができます。

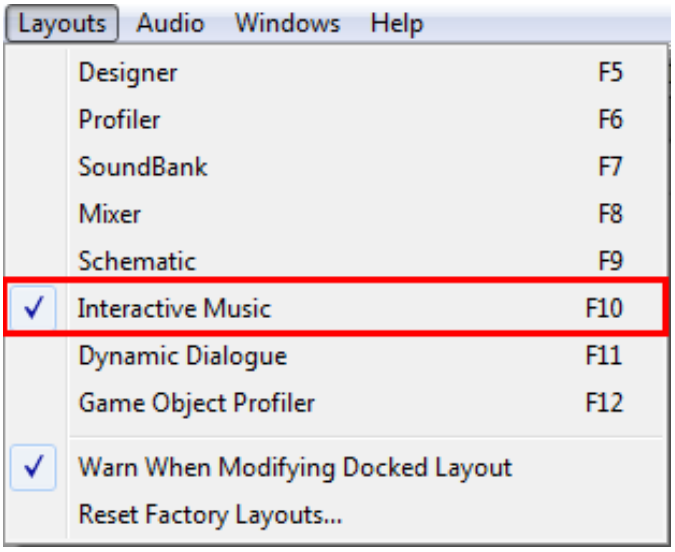
本章では、以下の操作を説明します。

- MIDIファイルのインポート
- MIDIファイルをミュージックトラックに追加
- MIDIターゲットを修正
- MIDIクリップテンポを調整
- Wwise Synth Oneの調整 - 特徴と機能

MIDIファイルのインポート

MIDIファイルは、第1章（アンビエントの設定: 基盤の設定 - オーディオファイルのインポート）で説明したのと同じ方法でインポートできます。インタラクティブミュージック階層に追加したMIDIファイルは、ミュージックセグメントとしてインポートされ、さらに他のミュージックセグメントにトラックとして追加することもできます。MIDIファイルは、ノーテーション、ピッチ、速度、その他の制御情報を指定するすべてのメッセージを含みます。WwiseプロジェクトでMIDIファイルを設定すると、他のオーディオファイル同様に置き換えや編集ができます。

インタラクティブミュージック階層のコンテナが提供する音楽専用の機能を見るには、レイアウトをInteractive Musicに変更します。

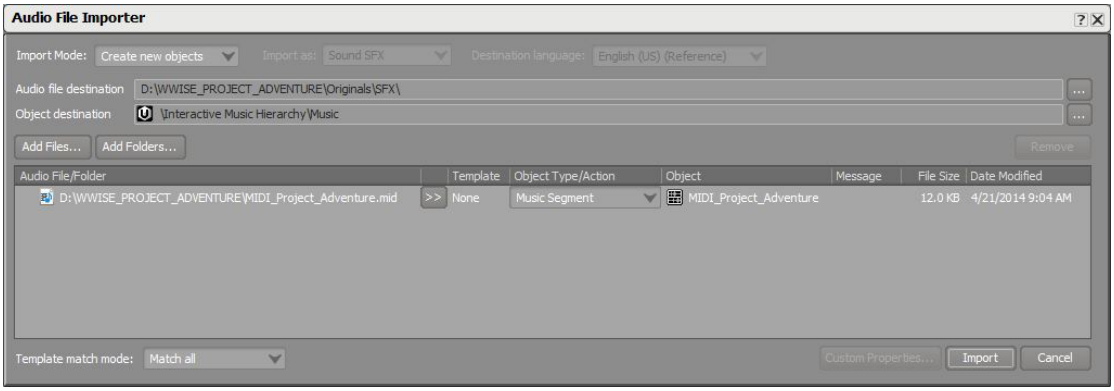


Interactive Music レイアウトへの切り替え

MIDIファイルは次のような設定でインポートできます。

- すべてのトラックを含み、各トラックが固有のMIDIチャンネル（1-16）に割り当てられている1つのMIDIファイル
- 各トラックに1つのMIDIファイルがあり、各トラックは固有のMIDIチャンネル（1-16）に割り当てられている
- 各トラックに1つのMIDIファイルがあり、各トラックが同じMIDIチャンネル（1）に割り当てられている

MIDIファイルは、Audio File Importerを使用、またはインタラクティブミュージック階層へドラッグ&ドロップして、ミュージックセグメントとしても、ミュージックセグメント内のトラックとしてインポートできます。セグメントにトラックとしてファイルを追加した後は、他のミュージックトラックと同様、Music Segment Editor画面からファイルにアクセスします。

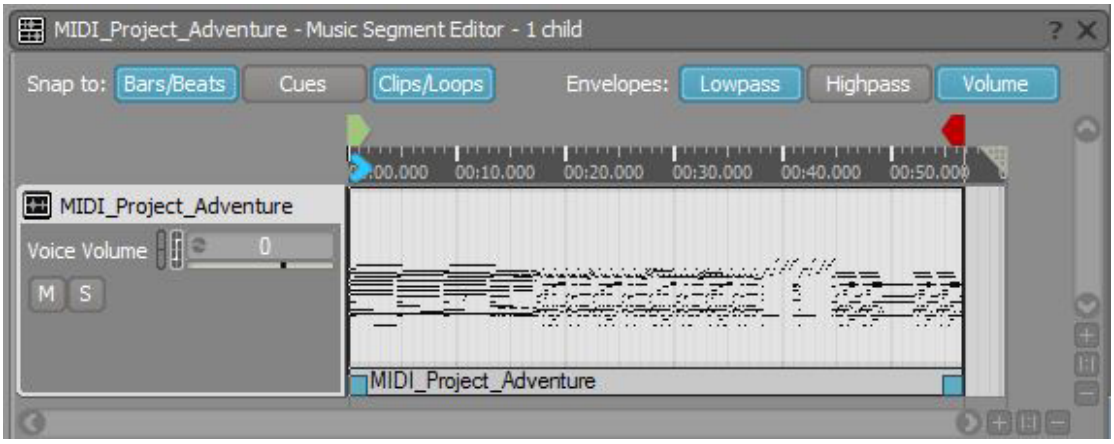


Audio File Importerは、ミュージックセグメントとしてインポートされる単一のMIDIファイルを表示します。

また、フォルダーからMIDIファイルを直接Wwiseインタラクティブミュージック階層へドラッグ&ドロップすることも可能です。

インポートされたMIDI_Project_Adventure .midファイル

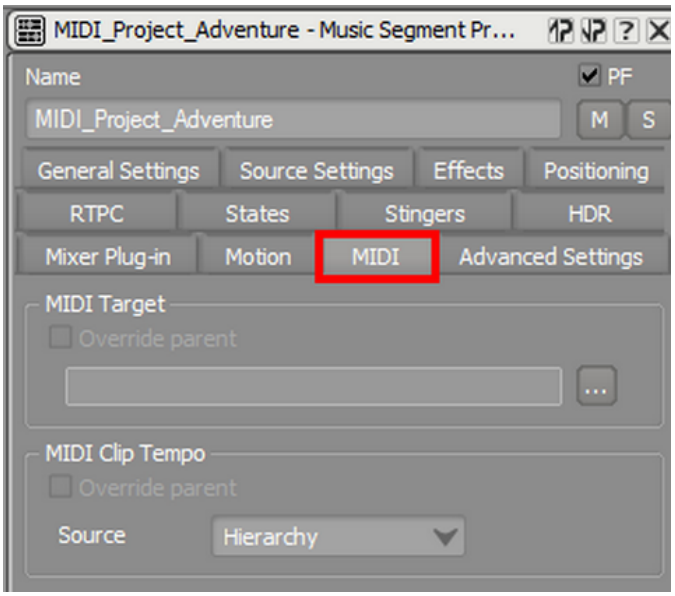
MIDIファイルをセグメントにトラックとして追加した後は、Music Segment Editor画面からアクセスし、ここで様々な組み合わせのオーディオ、MIDIトラックをミュージックセグメント内で並べ替え、編集ができます。



Music Segment Editor画面でトラックを編集

ミュージックセグメント内のMIDIトラックは、オーディオファイルと同様の動作や機能を示しますが、1つ例外なのは、オーディオプロパティはミュージックセグメントで修正できません。MIDIはデータ表現のみなので、オーディオプロパティはMIDIターゲットまたは"Instrument"（楽器）の一部となります。

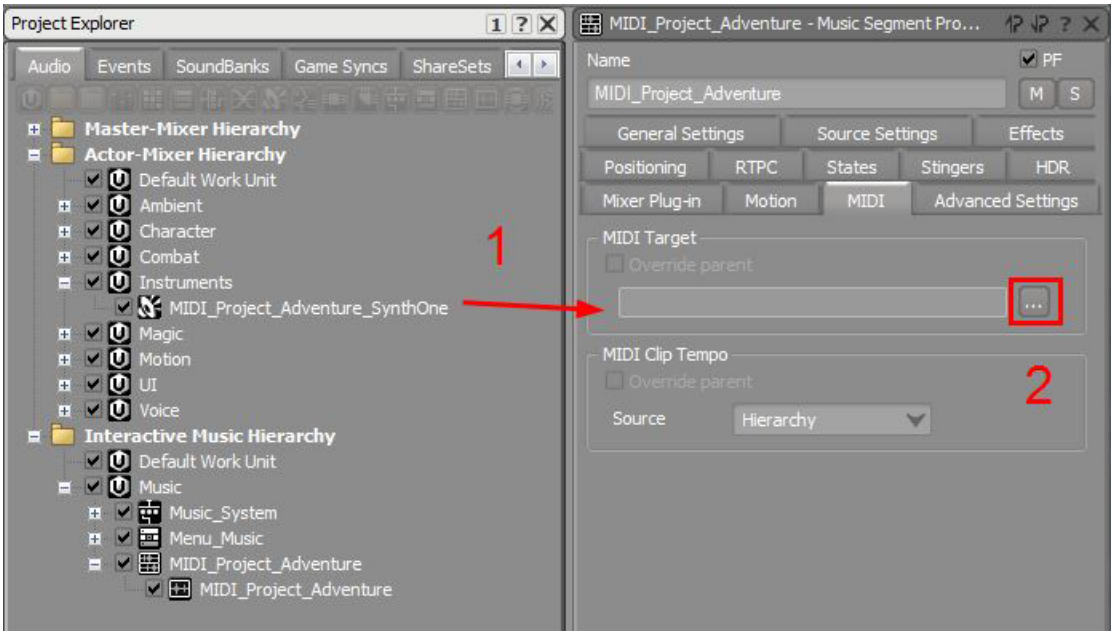
MIDIターゲットとMIDIクリップテンポは、 Music Segment Property Editor画面またはMusic Track Editor画面のMIDIタブで、ミュージックセグメント（親）、ミュージックトラック（子）のいずれにも設定することが可能です。



Music Segment Property EditorのMIDIタブ

MIDIターゲットはサウンドオブジェクトとして特定され、これを通じてすべてのMIDIイベントがルートされます。このプロパティは、各々が固有のMIDIプロパティを持つ複数のサウンドオブジェクトを含む、特定のサウンドオブジェクト、もしくはブレンドコンテナをターゲットにできます。

MIDIターゲットをサウンドセグメントまたはミュージックトラックにアサインするには、Project Explorer - Browser画面でサウンドオブジェクトを選ぶか、サウンドオブジェクトをアクターミキサー階層からMIDIターゲットフィールドにドラッグ＆ドロップするだけです。



ドラッグ&ドロップ（1）、またはProject Explorer - Browser（2）でMIDIターゲットをアサインする

MIDIクリップは、テンポを制御するためにMIDIファイルに指定されたTempo、またはミュージックセグメントのテンポセットのいずれも利用できます。

MIDIクリップテンポには、ミュージックトラック階層の一般設定にあるテンポ設定、または（MIDI）ファイルのテンポのいずれも利用できます。

MIDIクリップテンポソースを割り当てる

WwiseミュージックシステムでMIDIテンポ情報を使用するにはいくつか注意点があります。

- セグメント間のトランジションスケジューリングは、常にセグメントのテンポセットを利用して行いますが、これはMIDIクリップ／ファイルに示されたテンポイベントに影響されません。
- トリガーは、ビート、バー、ミュージックセグメントのグリッドでの再生をスケジュールしますが、MIDIクリップ／ファイルのテンポに基づくものではありません。
- MIDIファイル／クリップは、一回以上のテンポ変更でエクスポートされ、'file'（ファイル）オプションが選択されている場合、そのクリップはテンポ変更をリッスンして、MIDIノートを再生してタイミング通りにCCします。

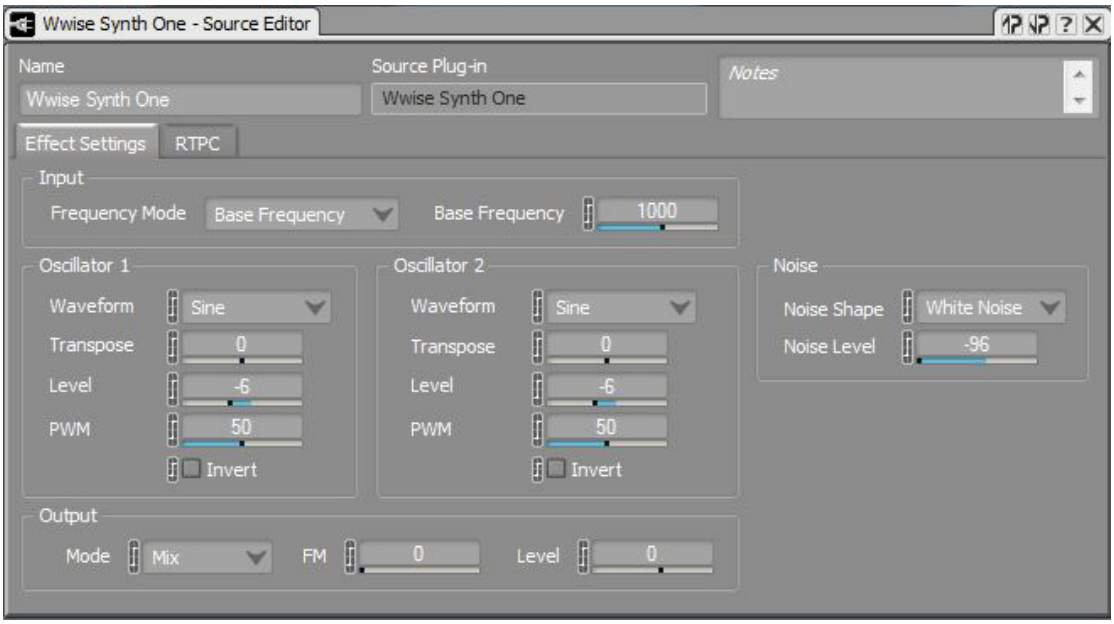
- これらのMIDIファイル／クリップベースのテンポ変更は、そのセグメントのテンポには影響を与えません。よって、MIDIクリップとセグメントの移行は簡単に同期しなくなります。

本節のまとめ

本節では、MIDIファイルをインポートして、トラックに特化したMIDIの機能について説明しました。MIDIファイルは、DAWで作成された構成とゲームでの最終的な表現の間のコミュニケーションの流れです。

Wwise Synth Oneの設定

インタラクティブミュージック階層でMIDIファイルを使っての作業を始める前にWwise Synth Oneソースプラグインのシンセサイザー機能を見てみましょう。



Wwise Synth One の Source Editor画面

- **Input（入力）**
  - **Frequency Mode** - 周波数モード。オシレーターが使用する入力周波数のソース。
    - Base Frequency - 基底周波数。Base Frequency プロパティから得た周波数。
    - Midi Note - MIDIノート。受信したMIDIノートイベントから得た周波数。
  - **Base Frequency（基底周波数）**
    - 20～20000 Hz - オシレーターへの入力周波数（Hz）。
- **2つのオシレーター**
  - **4つの波形タイプ** - 各オシレーターで利用できる波形は次の通りです。
    - サイン
    - 三角形
    - 矩形
    - のこぎり形
  - **Transpose:** トランスポーズ。オシレーターのピッチ
    - -3600 ～ +3600 セント - 入力周波数のトランスポジション（セント）。
  - **Level** - オシレーターの出力レベル（dB）。オシレーターが統合される前に適用する。

    - **PWM**（Pulse Width Modulation）- パルス幅変調。モジュレーター信号にパスル幅を一致させるテクニック。
    - **Invert** - インバート。オシレーターの出力の位相を反転する。
- **Output** - 出力。オシレーター出力の統合方法。
    - **Mode（モード）**
        - Mix: サンプルを追加
        - Ring: サンプルを乗算
    - **FM**（Frequency Modulation）- 周波数モジュレーション。この値はオシレーター１の出力を生成するためにオシレーター２の出力をどれだけ使用するかを定義する。
        - 20～20000 Hz
    - **Level**（Volume）- ボリュームのレベル。最終信号に適用するレベル（オシレーター出力とノイズジェネレーターの出力を合わせて）。
- **Noise（ノイズ）**
    - **Noise Shape** - ノイズシェイプ。生成されるノイズの種類。
        - White Noise（ホワイトノイズ）
        - Pink Noise（ピンクノイズ）
    - **Noise Level**（Volume）：ボリュームに関するノイズレベル。ノイズモジュールの出力に適用するレベル（dB）。ノイズモジュールの出力をオシレーター出力の値と合わせる前に、このレベルを適用する。

合成のアドベンチャー

MIDI プロジェクトアドベンチャーでは、Wwise Synth One プラグインの機能で再生した4つの楽器を使いました。その楽器とは、２つのリード矩形波のシンセ、サイン波とのこぎり波を組み合わせたベースシンセ、そしてホワイトノイズを使ったパーカッションシンセです。合成の作業する時に重要なのは実験することであり、最高のシンセサウンドは、往々にして機能セットの制限内で創造的な探求をした結果手に入れることができるものです。

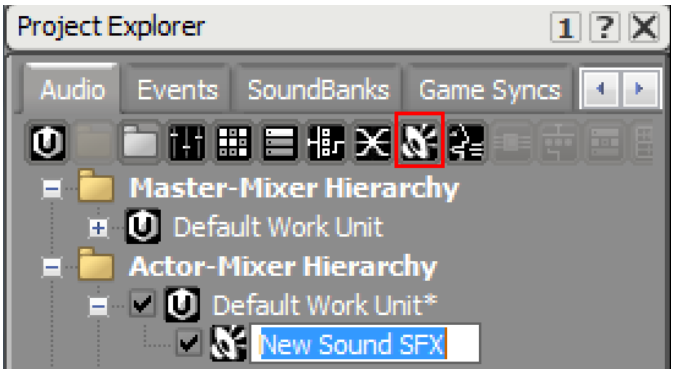
シンセサウンドをデザインする際に気をつけたい点は次のようになります

- **RTPC:** Real Time Parameter Control は、ゲームが設定した、もしくはゲームパラメーターを出力するためにWwise メーターエフェクトプラグインの一部としてWwiseが内部的に設定したゲームパラメーターと共に使用でき、シンセのプロパティを変化させます。
- **モジュレーターエンベロープ:** モジュレーターエンベロープは、Attack（アタック）、Decay（減衰）、Sustain（サステイン）、Release（リリース）およびRTPCを使って変化させることのできる他のプロパティのエンベロープの動作をコントロールします。
- **モジュレーターLFO:** モジュレーターLFOは、幅広く変化するダイナミックスを生み出すRTPCとして、Wwise Synth One のプロパティを変化させることに使われます。

Designer Note

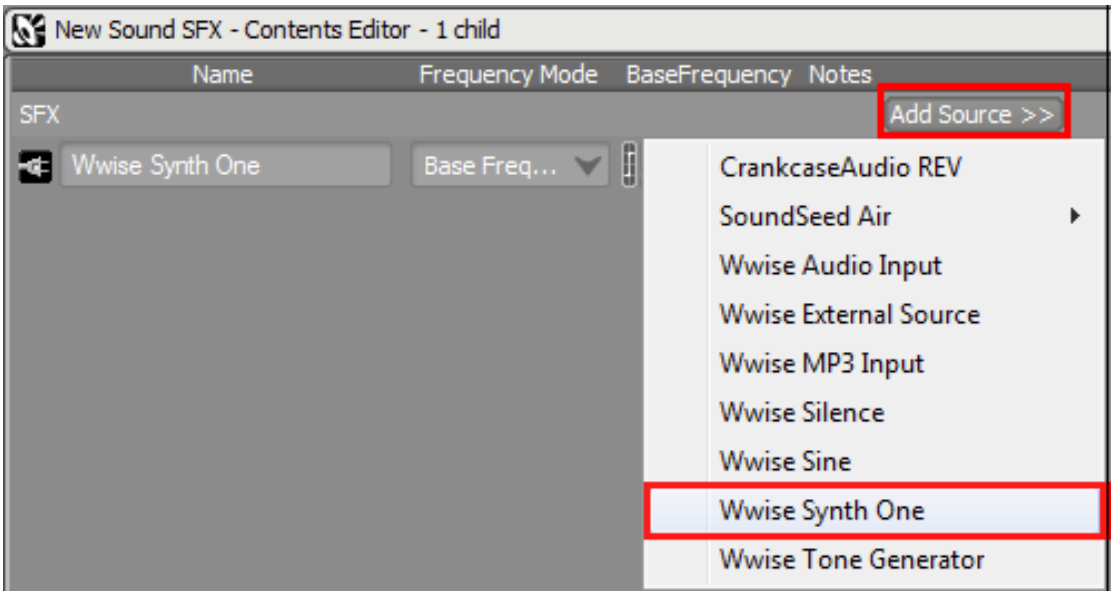
モジュレーターLFOとモジュレーターエンベロープについての詳細は「マジックの表現」の章で「モジュレーター」についてご覧ください。

第 1 章でみたSilenceプラグインの追加と同様に、Wwise Synth One プラグインをサウンドSFXへの入力ソースとして追加できます。Project Explorer ツールバーのサウンドSFXアイコンをクリックし、デフォルトワークユニットにサウンドSFXを追加すると、新しいサウンドSFXが作られます。また、コンテキストメニューやショートカットキーからサウンドSFXを作ることもできます。



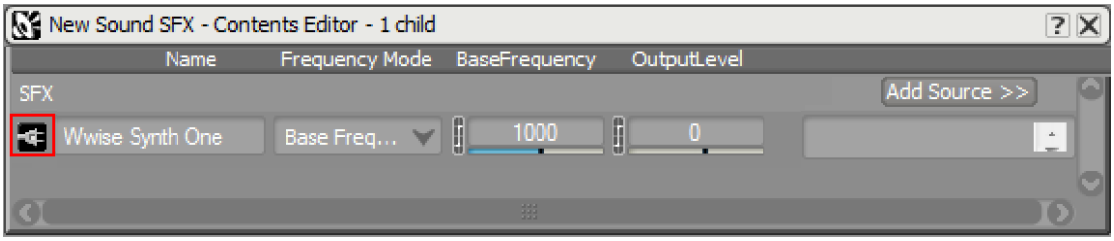
Project ExplorerツールバーからNew Sound SFX（新規サウンドSFX）を作成

ダブルクリックで新しいサウンドを選び、Contents Editor画面にあるAdd Sourceメニューから Wwise Synth Oneプラグインを追加します。

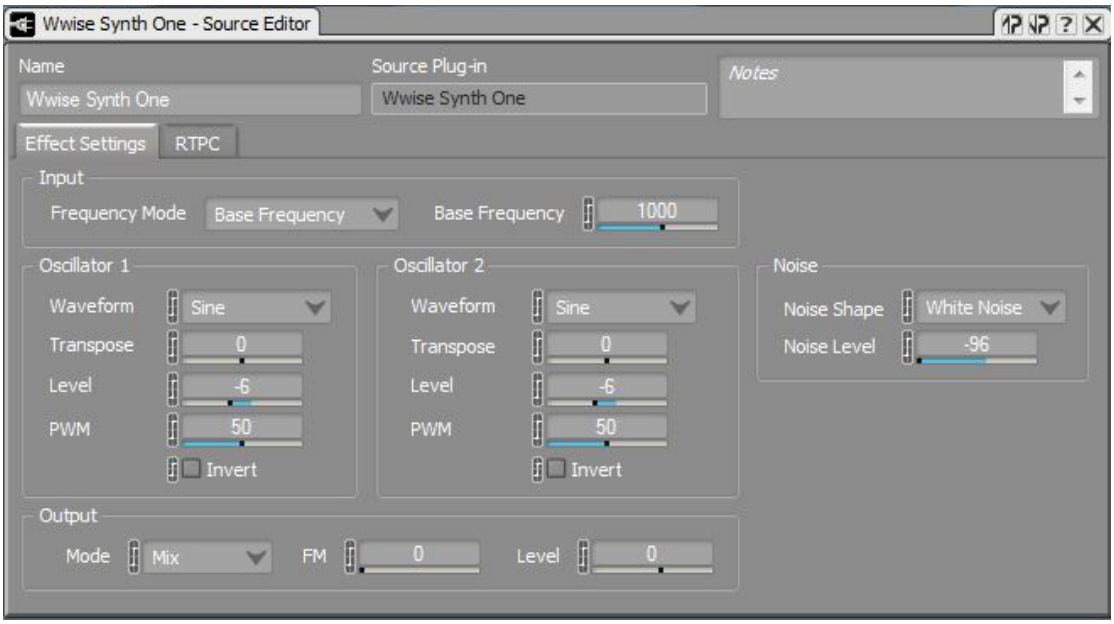


サウンドオブジェクトのWwise Synth Oneのソースプラグインを作成

Contents Editor画面にあるプラグインアイコンをダブルクリックしてSource Editor画面を表示します。

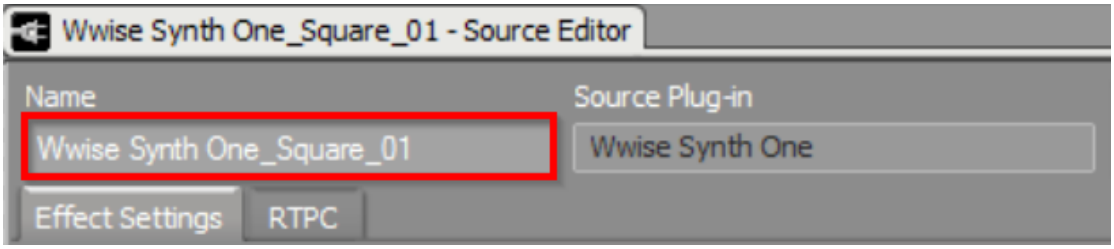


Wwise Synth OneのSourceプラグインアイコン



Wwise Synth One - Source Editor画面

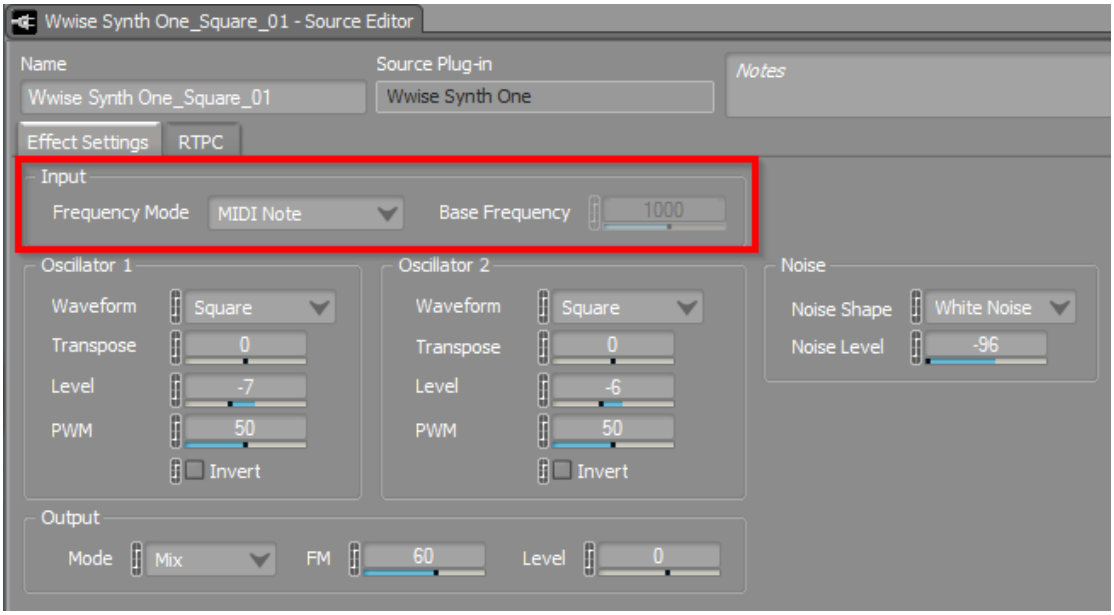
Wwise Synth One source プラグインの名前を、Source Editor画面でその意図する利用法を反映した名前に変更します。



Source Editor画面で、Wwise Synth Oneプラグインの名前を変更

このシンセはアクターミキサー階層のサウンドオブジェクトを選択して、'Play'（プレイ）（またはスペースバー）を押すと、試聴ができます。

デフォルトではInput Frequencyモードは、Base Frequencyに設定され、Base Frequency プロパティ値（Hz）をオシレーターの入力とし使用します。Wwise Synth OneプラグインをMIDIファイルと同時に使用するためには、入力をMIDIノートに設定する必要があります。



Source Editor画面でInput Frequency モードをMIDIノートに設定

Wwise Synth Oneプラグインの基本インスタンスをMIDIファイル（複数可）と同時に使用するには何の努力も必要ありません。

本節のまとめ

合成のプロセスと方法論は、創造的な作曲活動の中でもしっかりと確立された分野です。Wwise オーディオエンジンのランタイム機能の一部としてWwise Synth Oneを加えると、リアルタイムサウンド合成のパワーを用いるための幅広い可能性を開いてくれます。ダイナミックにプロパティの修正できるということは、間違いなくインタラクティブな可能性の宝の山への道を切り開くことになるのでしょう。

MIDIとサウンドを接続する

音を構成するタスクには、その基盤に音のグラフィック表現と動作があります。五線譜や「ピアノロール」、もしくはDAWの下にMIDIノートとして存在しようと、ノーテーションは、意図したパフォーマンスを伝えるメカニズムとして機能し、演奏者もしくは機械的/デジタル的な演奏によって再生されます。最もコンピューター化されたサウンド創作ツールの下で動作しながらも、MIDIは一貫した、標準的な方法で作曲者の音楽的意図を表現します。Wwiseは、この意図の解釈者としての役割を果たし、再生に使用され、サウンドのプロパティをリアルタイムで変化させながら、サウンドと「演奏者としてのゲーム」の間の相互作用に関与します。

DAWのMIDIファイルをWwise オーサリングアプリケーションで使用するには、特別の配慮をしてエクスポートする必要があります。特に、テンポとMIDIチャンネル設定は、それらに期待する使用法に則して割り当てる必要があります。Wwiseはデフォルトとして、MIDIファイルでエクスポートされたテンポとMIDIチャンネルの割り当てを、次のような制御に利用します。

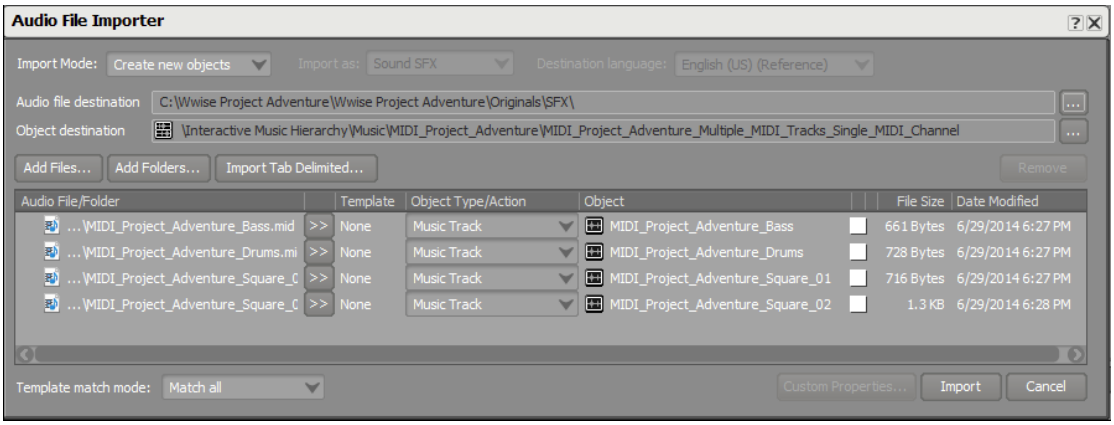
個々のMIDIトラックのインポート

DAWから、異なる楽器を表す4つの個々のファイルとして楽器トラックがエクスポートされました。

- MIDI_Project_Adventure_Square_01.mid
- MIDI_Project_Adventure_Square_02.mid
- MIDI_Project_Adventure_Bass.mid
- MIDI_Project_Adventure_Drums.mid

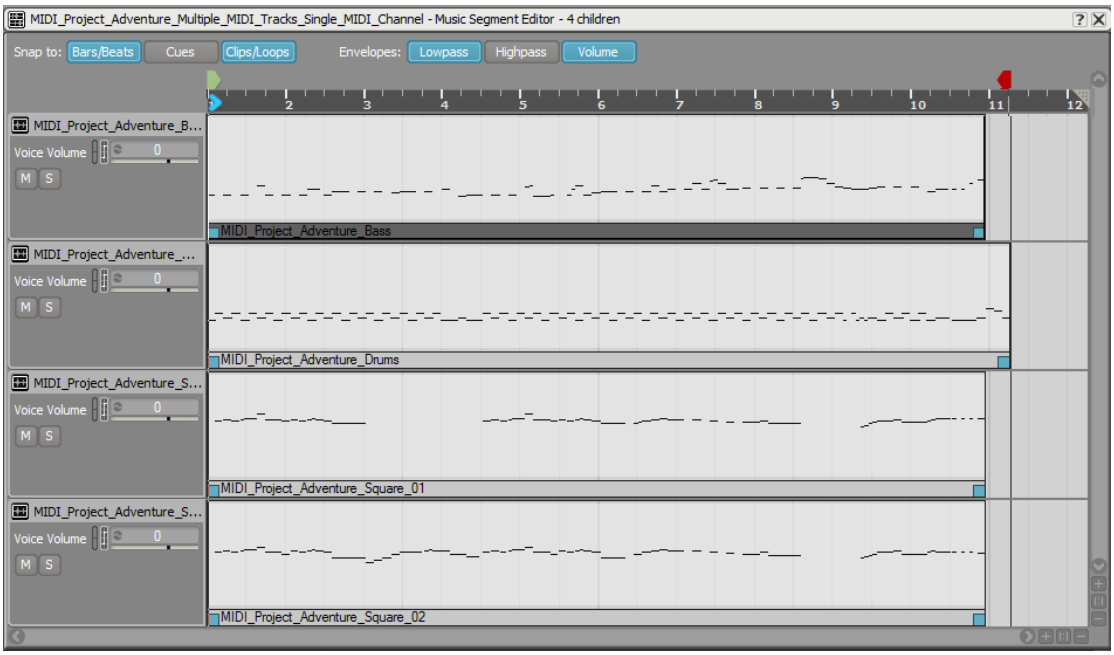
各ファイルは、140BPMのテンポ、すべてのチャンネル（オムニ）、および完全なベロシティで出力されました。

MIDIファイルは、Audio File Importerを使用、またはインタラクティブミュージック階層へドラッグ&ドロップして、ミュージックセグメントとしても、ミュージックセグメント内のトラックとしてインポートできます。セグメントにトラックとしてファイルを追加した後は、他のミュージックトラックと同様、Music Segment Editor画面からファイルにアクセスします。



Audio File Importerは、ミュージックトラックとしてインポートされる複数のMIDIファイルを表示

MIDIファイルをセグメントにトラックとして追加した後は、Music Segment Editor画面からアクセスし、ここで様々な組み合わせのオーディオ、MIDIトラックをミュージックセグメント内で並べ替え、編集ができます。



Music Segment Editor画面でトラックを編集

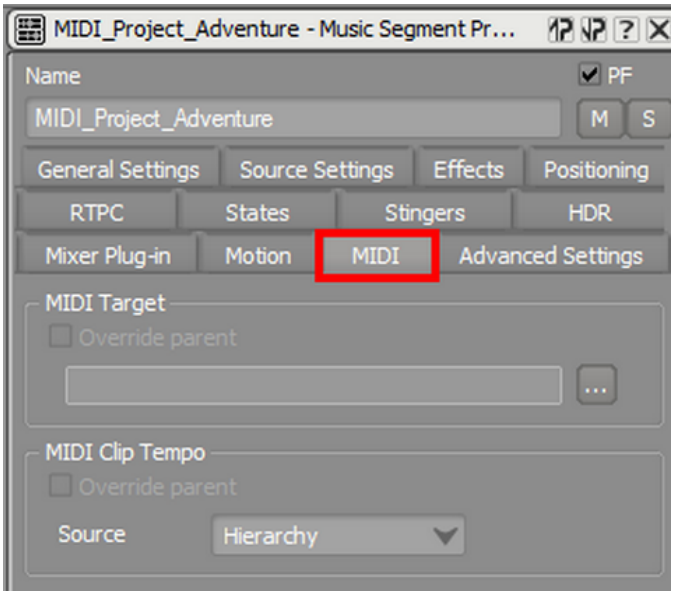
ミュージックセグメント内のMIDIトラックは、オーディオファイルと同様の操作と機能を示します。 （第7章「音楽のアドベンチャー - 横型のアプローチ - アンビエントミュージックセグメントの作成」を参照してください）

各トラックは、MIDIファイルから伝えられた情報を再生するために使うサウンドオブジェクトに関連させる必要があります。

ミュージックセグメントMIDIプロパティ

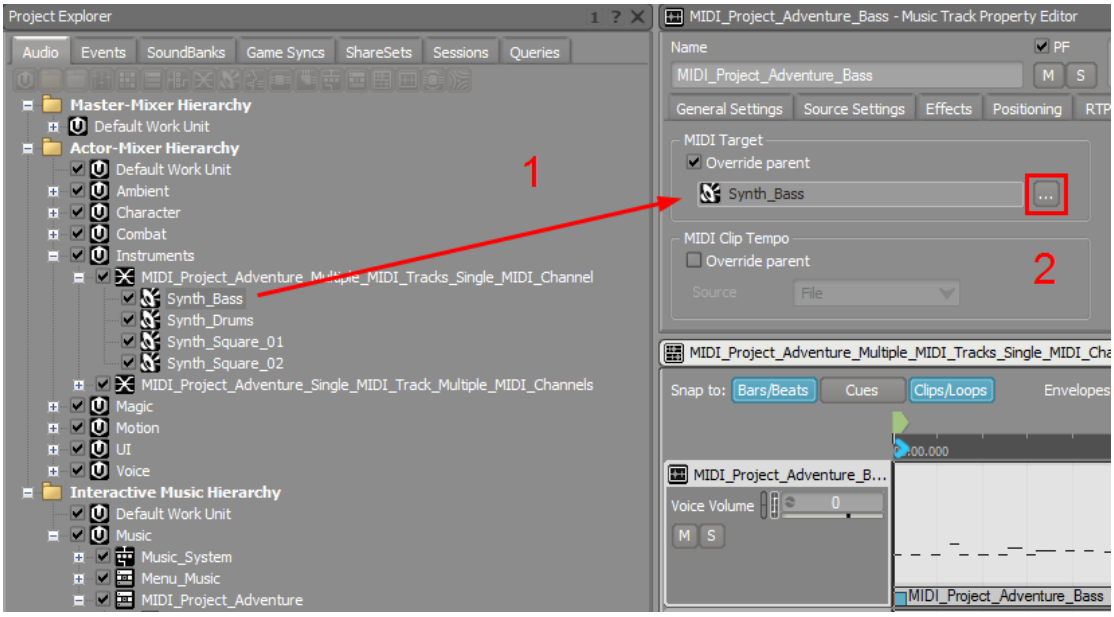
MIDIターゲットとMIDIクリップテンポは、 Music Segment Property Editor画面またはMusic Track Editor画面のMIDIタブで、ミュージックセグメント（親）、ミュージックトラック（子）のいずれにも設定することが可能です。

MIDIターゲットはサウンドオブジェクトとして特定され、これを通じてすべてのMIDIイベントがルートされます。このプロパティは、各々が固有のMIDIプロパティを持つ複数のサウンドオブジェクトを含む、特定のサウンドオブジェクト、もしくはブレンドコンテナをターゲットにできます。



Music Segment Property EditorのMIDIタブ

MIDIターゲットをサウンドセグメントまたはミュージックトラックにアサインするには、Project Explorer - Browser画面でサウンドオブジェクトを選ぶか、サウンドオブジェクトをアクターミキサー階層からMIDIターゲットフィールドにドラッグ＆ドロップするだけです。



ドラッグ&ドロップ（1）、またはProject Explorer - Browser（2）でMIDIターゲットをアサインする

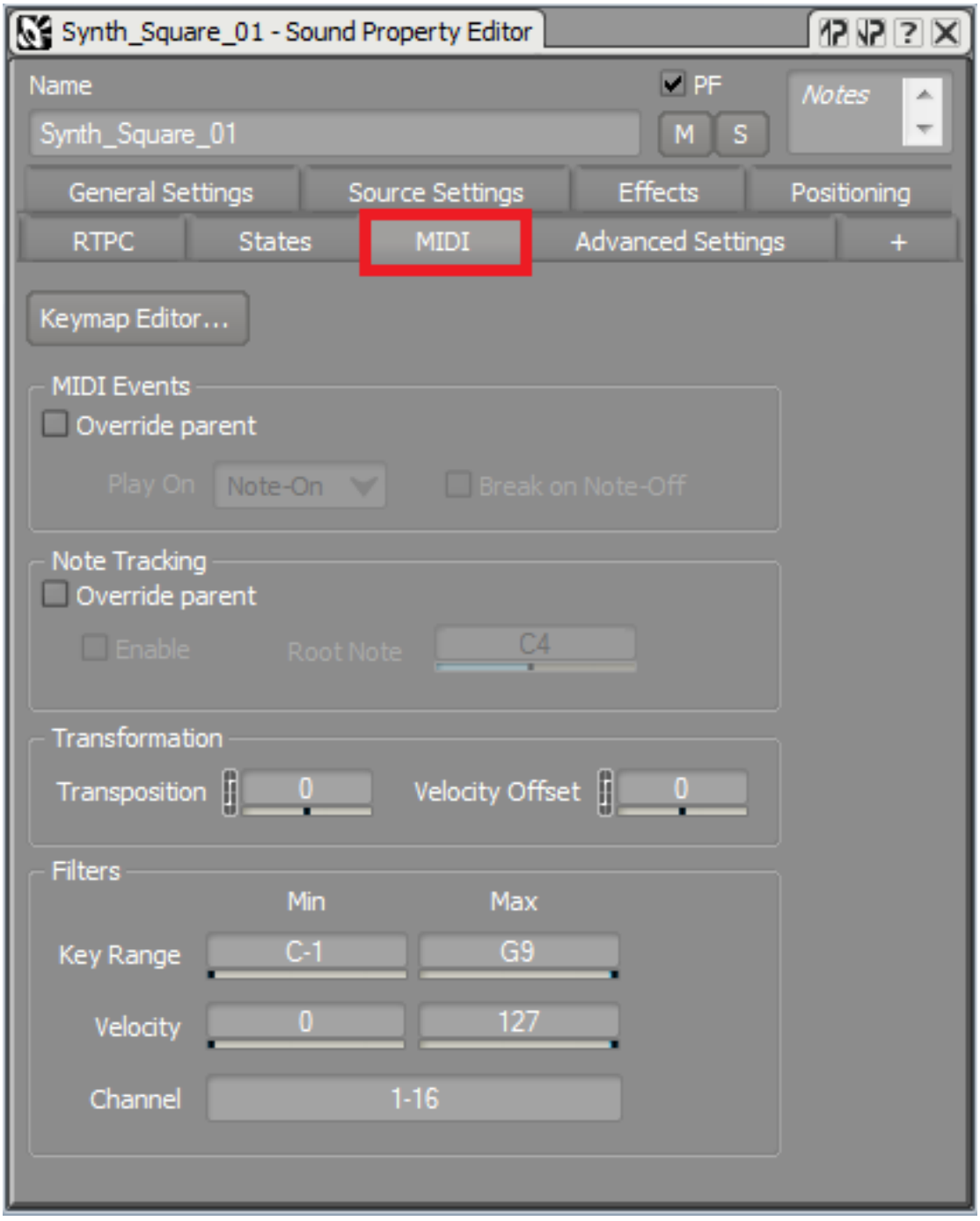
WwiseはTempoマップを使って（MIDIファイルの一部としてエクスポートした）テンポをコントロール、もしくはインタラクティブミュージック階層でサウンドオブジェクトに設定したテンポの仕様を使うこともできます。

インポートしたMIDIトラックは、（MIDI）ファイルからのMIDIクリップテンポ情報を使って、テンポをコントロールします。

このコンポジションはインタラクティブミュージック階層でミュージックセグメントを選択して、'Play'（プレイ）（またはスペースバー ）を押すと、試聴ができます。

サウンドオブジェクトMIDIプロパティ

追加のMIDIプロパティは、親サウンドオブジェクトのためにMIDIタブにあります。これらのプロパティは次の用途に利用されます: 入ってくるMIDIイベントの動作を制御、実行前にMIDIデータトランスポジションもしくはベロシティオフセットを変化させる、または入ってくるMIDIデーターをフィルターすること。これらのテクニックは、複数のサンプル、シンセサイザー、もしくはMIDIファイルがターゲットとしているいずれの組み合わせを使って楽器を作る際、特に便利です。



Sound Property Editor画面にあるMIDIタブのサウンドオブジェクトプロパティ

サウンドオブジェクトMIDIプロパティには次が含まれます。

- **Keymap Editor** - キーマップエディター。MIDI Keymap Editorで、オブジェクトを検査。
- **MIDIイベント**

- **Override parent** - オーバーライドペアレント。MIDIイベントコントロールを親から継承するか、または階層の現在のレベルで定義するかを判断する。このオプションが選択されていない場合、MIDIイベントコントロールは使用できません。

  オブジェクトがトップレベルオブジェクトの場合、このオプションを使用することはできません。
- **Play On** - どの種類のMIDIノートイベントで、オブジェクトに再生をさせるかを判断します。
  - **Note-On** - オブジェクトを Note-Onで再生。
  - **Note-Off** - オブジェクトを Note-Off で再生。
- **Break On Note-Off** - Note-OnにPlay Onが設定されている場合、このプロパティで、ノートオフを受信した時、再生オブジェクトがループを停止するかどうかを決定する。その場合、現行のオブジェクト（複数可）は終了するまで再生を許す一方、ループサウンドの再生または継続したコンテナは停止します。

- **Note Tracking**（ノートトラッキング）:
  - **Override parent** - オーバーライドペアレント。MIDIイベントコントロールを親から継承するか、または階層の現在のレベルで定義するかを判断する。このオプションが選択されていない場合、Note Trackingコントロールは使用できません。オブジェクトがトップレベルオブジェクトの場合、このオプションを使用することはできません。
  - **Enable** - 有効。これが選択されていると、ノードの再生はピッチをシフトさせて行われます。ピッチのシフトの量は、受け取ったMIDIイベントとルートノートの値で判断します。
    - デフォルト値: False
  - **Root Note** - ルートノート。ノードサウンドのルートノート。この値は受け取ったMIDIノートと共に利用し、ノードサウンドのピッチシフトを判断する。
    - デフォルト値: C4
- **Transformation**（トランスフォーメーション）:
  - **Transposition** - トランスポジション。MIDIイベントのノートに適用したオフセット。このトランスポジションはKey Rangeフィルター前に適用する。
    - デフォルト値: 0
  - **Velocity Offset** - ベロシティオフセット。MIDIイベントのノートベロシティに適用するオフセット。これはMIDIノートイベントのみに適用します。このオフセットはVelocityフィルター前に適用します。
    - デフォルト値: 0
- **フィルター**
  - **Key Range** - キーレンジ。受信したMIDIノートイベントのノートに適用するフィルター。そのノートが最小〜最大の範囲内でなければ、受信したMIDIノートイベントは無視されます。
    - デフォルト最小値: C-1 デフォルト最大値: G9

注意: 数的なMIDIノートのオクターブへのマッピングは、User Preferences を経由して指定します; 以下を参照してください: Wwise Help > Wwise Reference > Projects > User Preferences for more details.

- **Velocity** - ベロシティ。受け取ったMIDIノートイベントのベロシティに適用するフィルター。そのベロシティが最小〜最大値の範囲内でなければ、受信したMIDIノートイベントは無視されます。
  - デフォルト最小値: 0 デフォルト最大値: 127
- **Channel** - チャンネル。受け取ったMIDIノートイベントチャンネルに適用するフィルター。チャンネルがフィルターにないと、受け取ったMIDIノートイベントは無視されます。
  - デフォルト値: 1〜16

本節のまとめ

拡大するWwiseオーサリングアプリケーションの新しい機能を利用すれば、MIDIを適したところに、適した技術として利用することができます。 小さなサンプルセット、合成、そしてMIDIのパワーと柔軟性を活用すれば、常駐メモリ（ディスクやRAMのサイズ）と処理（CPU）のリソース利用のバランスを維持するのを助けます。ターゲットプラットフォームの制限と実現したい効果によっては、MIDIは強力な味方となって、リソースを最大限に活かし、ダイナミックなオーディオ体験を作り上げてくれます。

MIDIのまとめ

地に落ちたヒーローのように、過去の知識と且つての姿を思い出しながら持って帰ってきたMIDI。この知識の一部をすぐに利用して、MIDIにとっての新しい時代を切り開きます。サンプルセットの作成、MIDIチャンネルの割り当て、ベロシティ、そしてMIDIイベントでこれを体験してください。MIDIが継承した有効性、ダイナミズム、そして柔軟性を上手く活用することで、十分にその機能が試された方法論の復活を予告します。しかし、MIDIのいくつかの部分はまだネガティブな印象もあり、これらは最小限にする必要があります。つまりそれらには音質の悪さ、困難なワークフロー、そしてMIDIで利用可能な基本構造に対する一般的な偏見があげられます。

本章では、以下を説明しました。

- MIDIファイルのインポート
- MIDIファイルをミュージックトラックに追加
- MIDIターゲットの修正
- MIDIクリップテンポの調整
- Wwise Synth Oneのシンセサイザー機能
- 使用するミュージックコンテンツの事前処理

プロセスの説明

- MIDIファイルのインポート
- Wwise Synth Oneプラグインの設定
- 次からなるマルチトラックコンポジションを作成する:
  - 異なる楽器を表す4つのMIDIトラック
  - 異なる楽器を表す4つのWwise Synth One シンセ

詳細設定の説明

- MIDIフォーマットの高度な機能

オブジェクトの作成方法

- 個々のチャンネルと持つMIDIファイルを含むミュージックセグメントで、異なる楽器を表すWwise Synth One のインスタンスをそれぞれターゲットにしています。
- 4つ個々のMIDIファイルを含むミュージックセグメントで、異なる楽器を表すWwise Synth Oneのインスタンスをそれぞれターゲットにしています。

参考ドキュメントとチュートリアル

Wwise Knowledge Base - Advanced settings playback limits and interactive music

Wwise Knowledge Base - Looping and streaming of audio clips and interactive music

Wwise Knowledge Base - Understanding the Interactive Music Engine

Wwise Knowledge Base - Interleaved Streaming in Interactive Music

Wwise Knowledge Base - Using States with the Interactive Music Hierarchy

Video Tutorial - Creating Interactive Music Structures

Video Tutorial - Defining Interactive Music Transitions

Video Tutorial - Creating Stingers

Game Sound Design - Making Music Interactive: Elaboration of the Feature Set in Wwise

Game Sound Design - Dynamic Music Creation Using Wwise

第9章 ミックスをマスターする

概要

ファイナルミックスにとりかかるのは、火を吹くドラゴンとのバトルに立ち向かうのと似ています。ゲームで殺気だった生物と最後の戦いに挑むシーンに到達する頃には、化け物を降伏させるだけのスキルとウェポンを取り揃えておきたいものです。信号がルーティングオプションの待ち受ける迷宮に流れていくのを見ても、オーディオパスを成形する柔軟性と、最終ミックスを主導するパワーが手元にあれば、迷いはないはずです。

たった1つのインスタンスサウンドから、1フレームで発生した膨大な量のサウンドの積み上げに至るまで、あらゆる場面で、聞こえるべきサウンドが聞こえるようにゲームサウンドを仕組まなければなりません。音楽や映画の業界で使われる「ミックス」が時間の決まった非インタラクティブなサウンドを表現するのに対し、ゲームオーディオで行うミキシングは不透明な部分があります。ダイナミックなミキシング方式やその他のテクニックを使い、最終的なサウンド出力のバランスを整えることが、現行世代のゲームにおける最大の課題です。

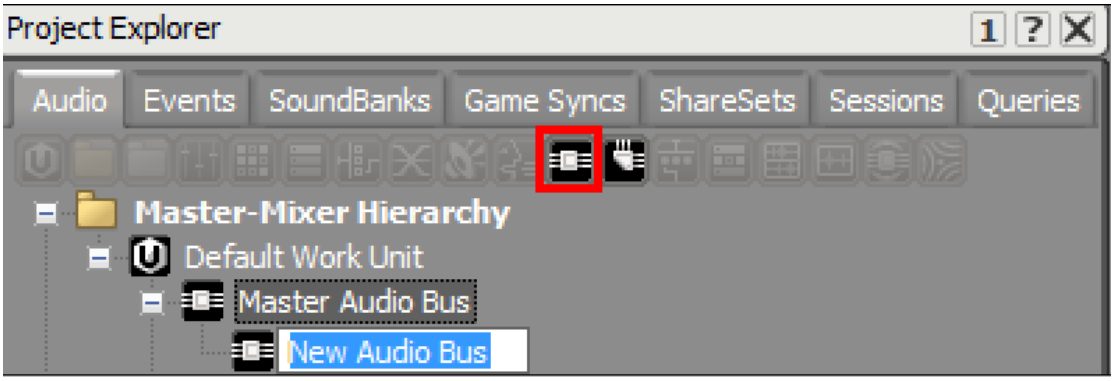
本章では、以下の操作を説明します。

- オーディオバスのルーティング
- AUXバスのルーティング
- AUXセンド
- ステートとミックススナップショット
- オートダッキングvs.サイドチェイン
- RTPCを使ったミキシング
- マスターミキサーでエフェクトを使用
- 環境リバーブ
- マスターミキサーの整理
- ミキシングデスクのビジュアル化
- 減衰のミキシングテクニック

オーディオバスのルーティング

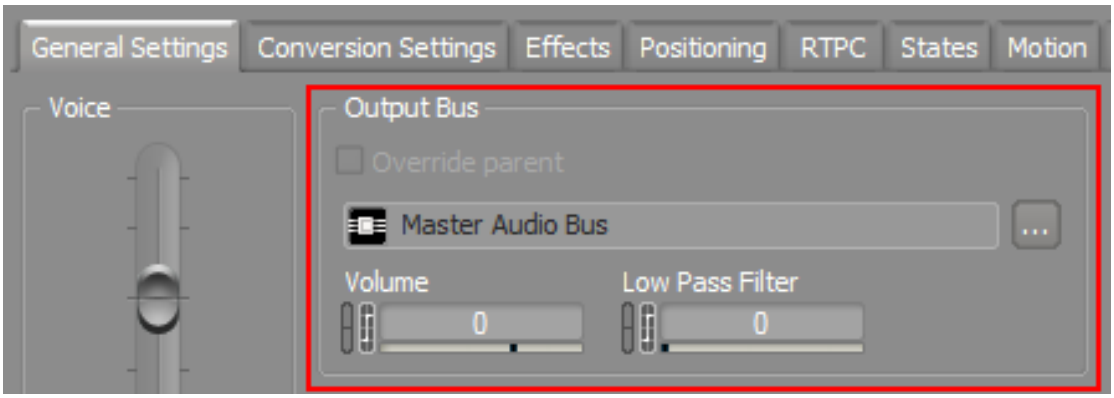
ハードやソフトのミキサーを使った経験があれば、Wwiseのマスターミキサー階層のカスタマイズできるインターフェースで、サウンドオブジェクトのオーディオ信号をルーティングできる仕組みに好感を持つでしょう。オーディオバスやAUXバスを追加し整理することで、信号フローを柔軟で統合された形で表現できます。

基本的にオーディオバスは、ボリュームのバランス、エフェクトの追加、RTPCの適用、ステートに合わせた変更などに使います。オーディオバスをマスターミキサー階層に追加するには、既存オーディオバスを選択して、Project Explorerツールバーのオーディオバスアイコンをクリックします。選択したオーディオバスの新しい子オーディオバスが作成されます。



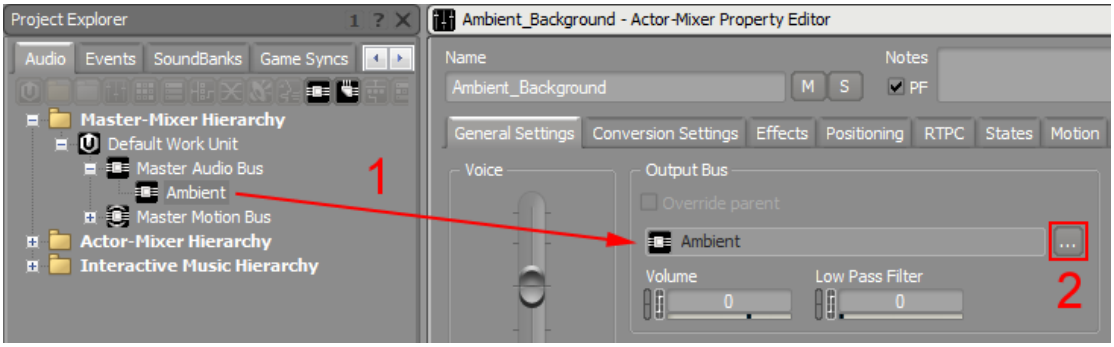
Project Explorerツールバーでオーディオバスを作成

作成されたオーディオバスは、どのサウンドオブジェクトに対してもProperty Editor画面のGeneral Settingsタブでアサインできます。オーディオバスをサウンドオブジェクトにアサインするには、“Project Explorer - Browser”画面でバスを選ぶか、オーディオバスをマスターミキサー階層からAudio Output Busフィールドにドラッグ＆ドロップするだけです



Property Editor画面でAudio Output Busの詳細を設定

オーディオバスの構成は、開発プロジェクトのニーズに合わせて繰り返し変更できます。また、マスターミキサー内でオーディオバスをドラッグ＆ドロップしても、サウンドオブジェクトにアサインされたルーティング設定は維持されます。



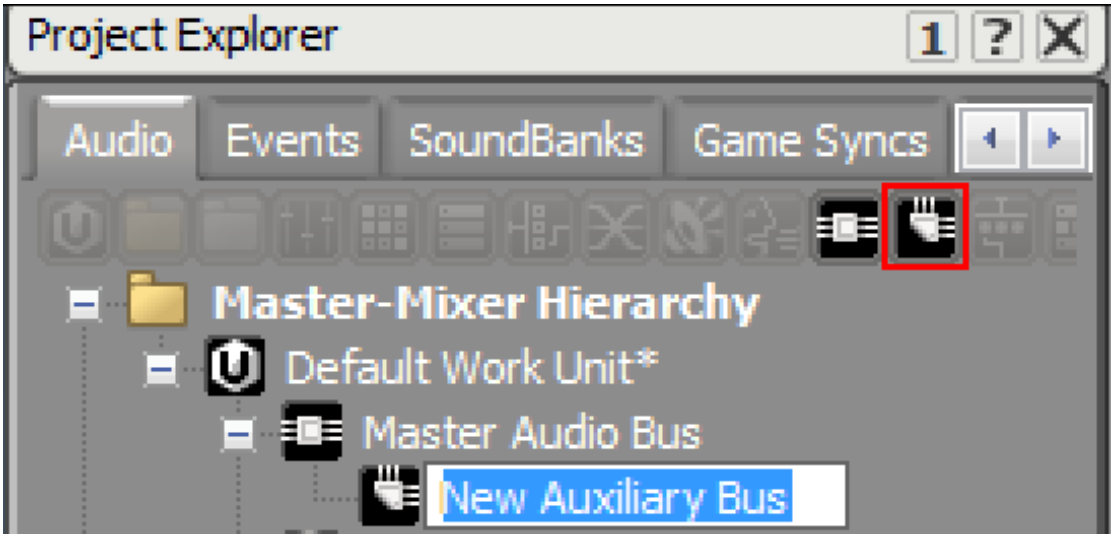
サウンドオブジェクトのAudio Output Busをアサインするには、ドラッグ＆ドロップする（1）か、Project Explorer - Browser画面を使う（2）

前図の例では、オーディオ出力バス“Ambient”をアサインしたので、アクターミキサー“Ambient_Background”から出力されるサウンドが全てオーディオバス“Ambient”にルーティングされます。Output Bus Volume（出力バスボリューム）とOutput Bus Low Pass Filter（出力バスローパスフィルター）を設定して、オーディオ出力バスに送られるサウンドオブジェクトのボリュームやローパスフィルターの量をコントロールします。さらに、マスターミキサー階層で設定したオーディオバスのプロパティが最終的な出力を管理します。

AUXバスのルーティング

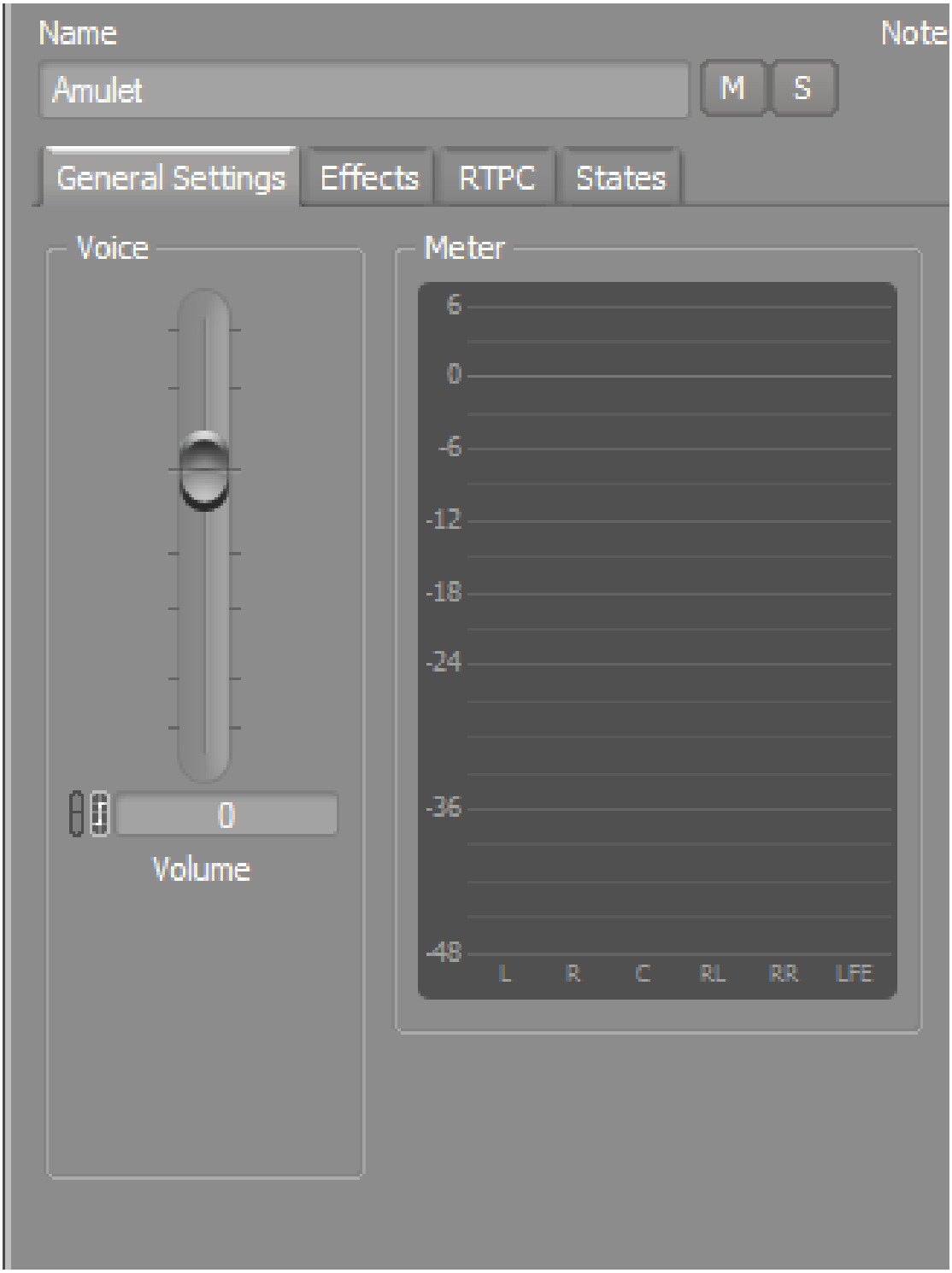
AUXバスは標準オーディオバスと同様にマスターミキサー階層内に配置します。各AUXバスで最大4つのエフェクトを適用でき、エフェクトの有効化と無効化はプログラムやゲームパラメータで管理できます。マスターミキサー階層内の配置で他のオーディオバスやAUXバスの子とすることで、一度に4つ以上のエフェクトが可能となります。

まずAUXバスをマスターミキサー階層に追加するには、既存オーディオバスまたはAUXバスを選択して、Project ExplorerツールバーのAUXバスアイコンをクリックします。選択したオーディオバスまたはAUXバスの、新しい子AUXバスが作成されます。



Project ExplorerツールバーでAUXバスを作成

AUXバスのMeter機能は、ボリュームの設定、エフェクトの追加、RTPCの適用、ステートに合わせた変更などに使います。



Property EditorにあるAUXバスのGeneral Settingsタブ

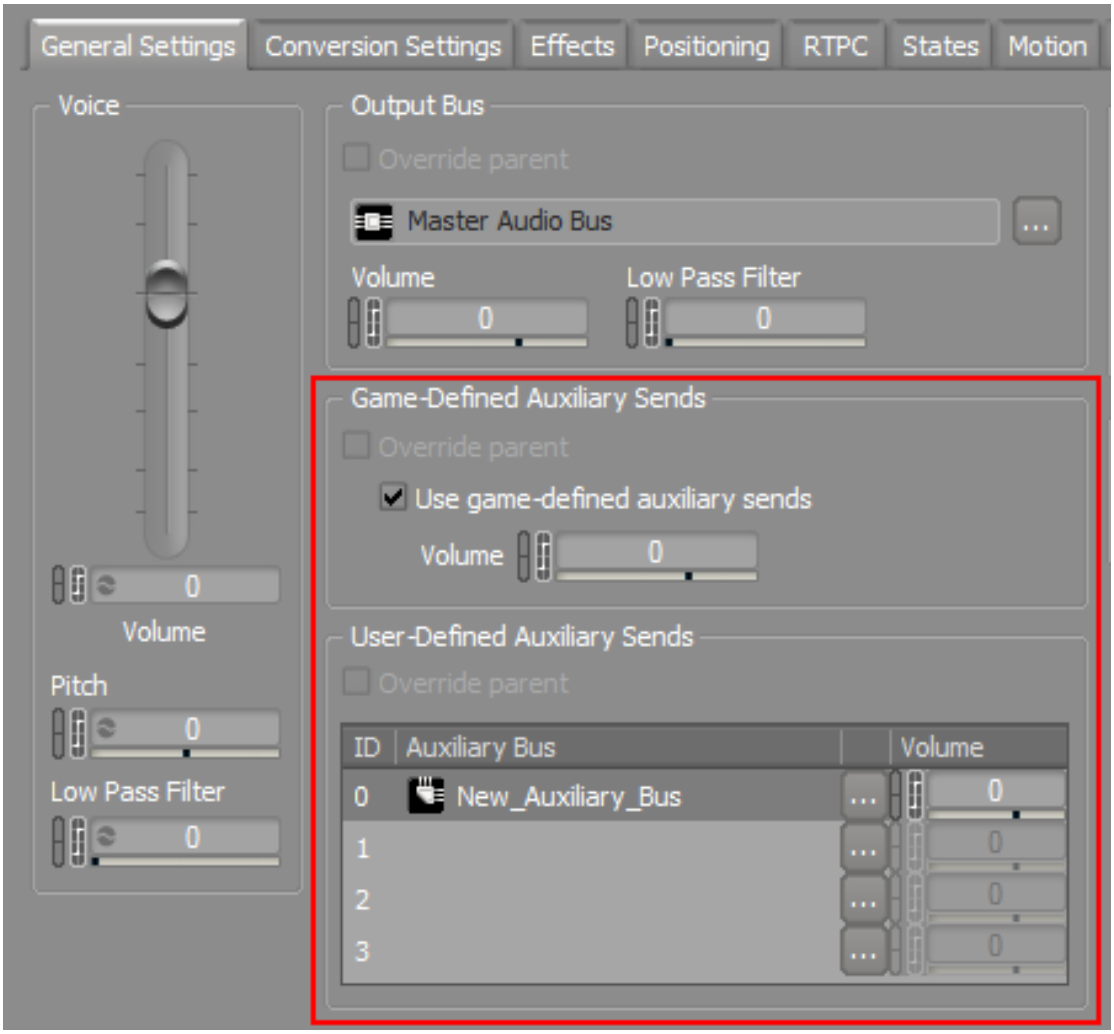
AUXセンド

AUXセンドでオーディオ信号を複数のエフェクトのセットに送り、オーディオ信号全体にエフェクトを適用するのが、従来の使い方です。つまり、アクターミキサー階層にあるサウンドオブジェクトの出力を、AUXセンドでマスターミキサー階層にあるAUXバスにルーティングすることができます。

WwiseのAUXセンド機能には、以下の2種類があります。

- ユーザー定義によるAUXセンド
- ゲーム定義によるAUXセンド

Property Editor画面のGeneral Settingsタブで、サウンドオブジェクトのAUXセンドを1つまたは2つ、選択できます。どちらのAUXセンドも全く同じ動作をします。1つのサウンドオブジェクトから送られる信号に対して、2種類のセンドをオーサリングアプリケーション内で組み合わせて使えます。



Property Editor画面のGeneral SettingsタブでAUXセンドを選択

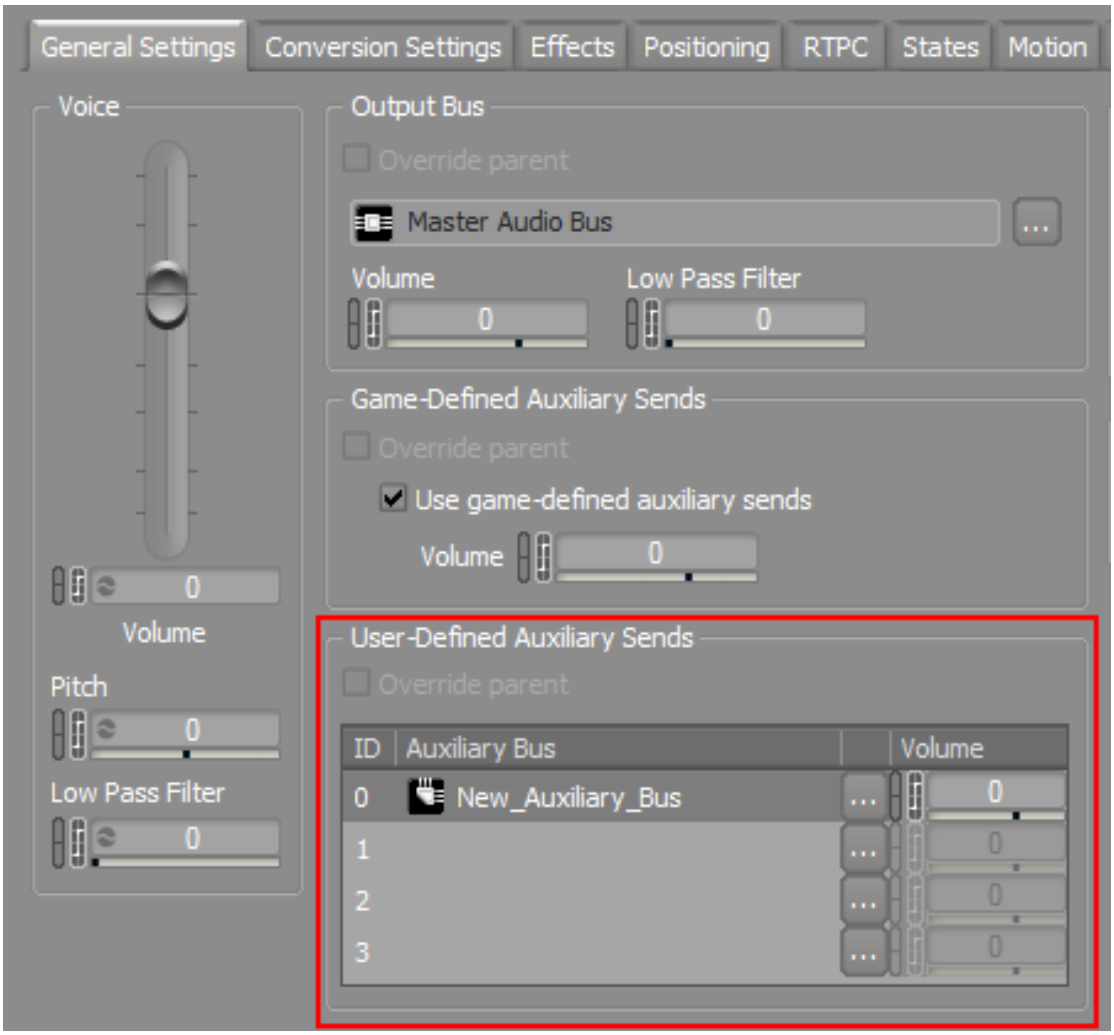
2種類のAUXセンドは、制御方法が以下の通り異なります。

- ユーザー定義：Wwiseのサウンドオブジェクトごと
- ゲーム定義：ゲームオブジェクトごとにSDK APIを使って

マスターミキサー階層にあるAUXバスに関連したセンドボリュームは、オーディオ出力バスのボリュームやルーティングとは独立しています。このため、エフェクトのクリエイティブな適用方法など、インタラクティブミキシングのテクニックの応用がききます。

ユーザー定義のAUXセンド

最大4つのユーザー定義AUXバスを、アクターミキサー階層のどのサウンドオブジェクトにアサインすることもできます。センドボリュームはAUXバスに送るオーディオ信号のレベルまたは振幅のことで、RTPCを使ってパラメータ化することもできます。ユーザー定義AUXバスは、Property Editor画面のGeneral Settingsタブで設定します。

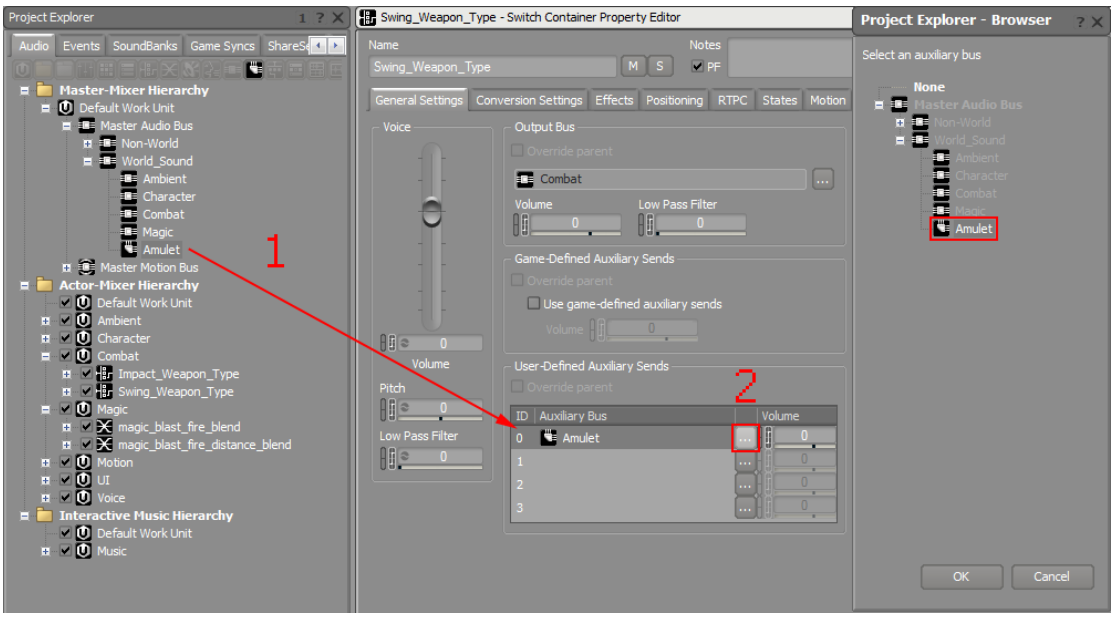


Property Editor画面のUser-Defined（ユーザー定義） Auxiliary Sends

それでは、ユーザー定義AUXセンドを使い、ヒーローが危険にさらされている場面で特定のサウンドタイプにマジカルエフェクトを適用する例を説明します。AUXバスに入ったエフェクト一式は、ゲームパラメータ“Danger”を利用しながら複数のプロパティを調整します。ウェポンのスイング音やマジックサウンドを変化させて、迫る危険を表しながらダイナミックな戦闘シーンを盛り上げていきます。

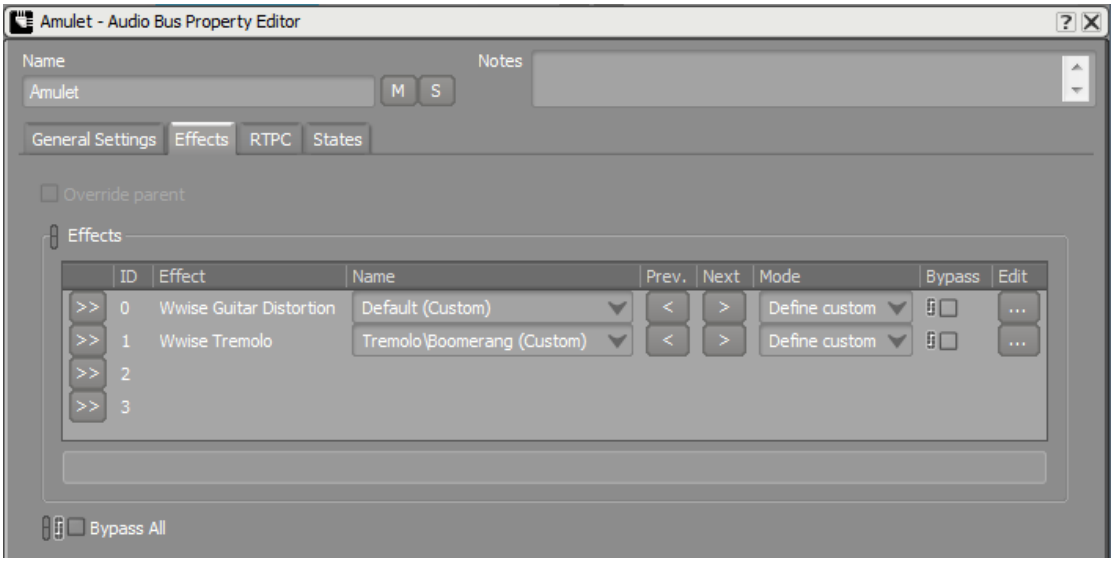
古いアムレットがゲームのヒーローのパワー源なので、これにちなんで“Amulet”という新規AUXバスを作ります。今回の例では、スイッチコンテナ“Swing_Weapon_Type”とブレンドコンテナ“magic_blast_fire_distance_blend”の出力をAUXバス“Amulet”にルーティングします。

既存のAUXバスをサウンドオブジェクトにアサインするには、Project Explorer - Browser画面でバスを選ぶか、AUXバスをマスターミキサー階層からProperty Editor画面のUser-Defined Auxiliary SendsにあるAuxiliary Busフィールドにドラッグ＆ドロップします。



サウンドオブジェクトにAUXバスをアサインするには、ドラッグ＆ドロップする（1）か、Project Explorer - Browser画面を使う（2）

サウンドオブジェクトから信号をAUXバスに送る設定をした後は、センドボリュームを調整します。次にAUXバス“Amulet”でエフェクトの追加と調整を行い、ヒーローが危険にさらされている時にマジカルな変化が起きる様子を表現します。今回の例では、カスタム設定した“Wwise Guitar Distortion” を1番目のエフェクトとして追加して、ゲームパラメータ“Danger”に基づいてウェット・ドライミックスを増加するように設定します。さらに2番目のエフェクトとして“Wwise Tremolo”を追加して、LFOのデプスをゲームパラメータ“Danger”でコントロールします。



AUXバス“Amulet”で適用する一連のエフェクト

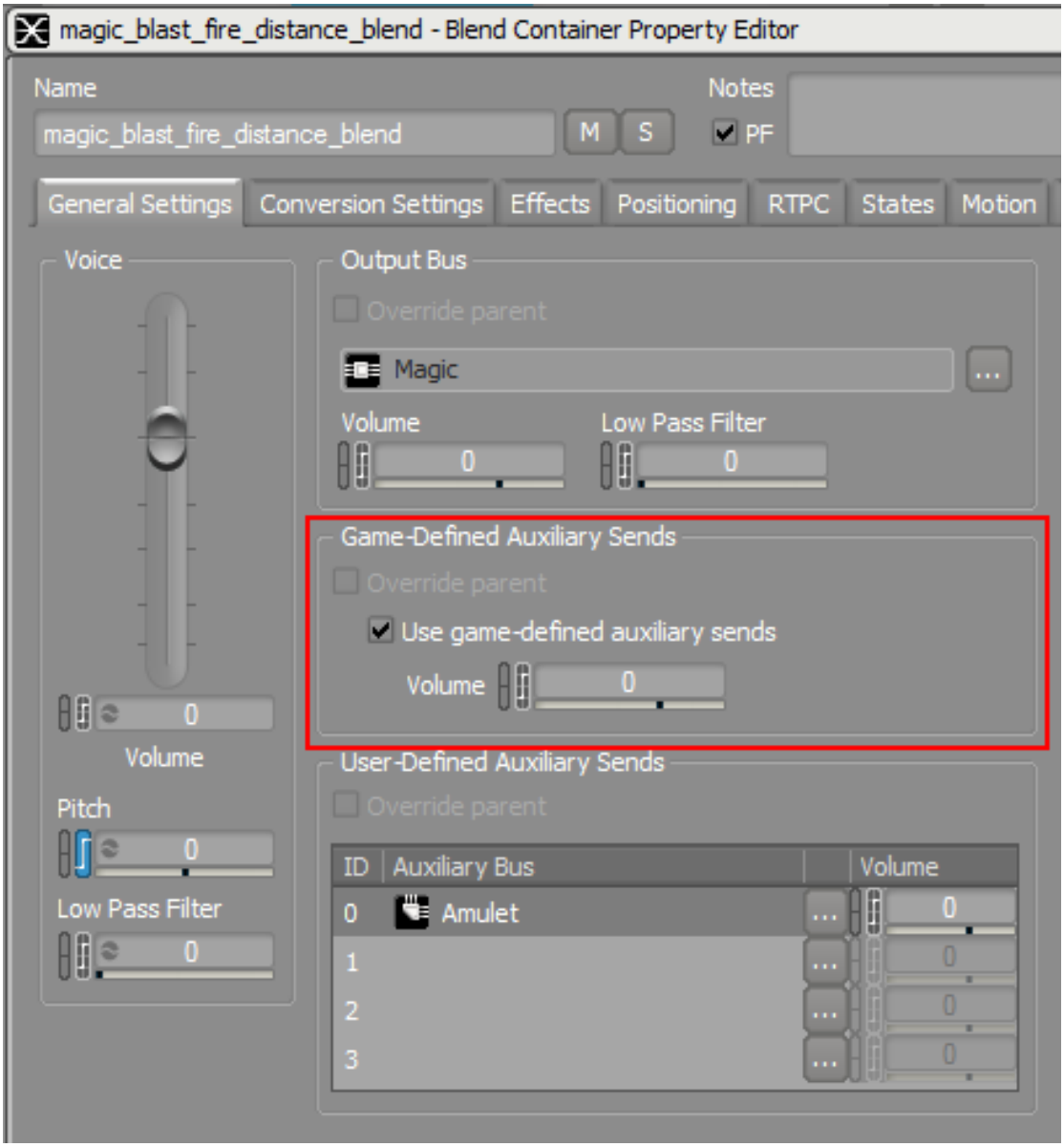
Designer Note

ユーザー定義AUXバスのルーティングをランタイムに変えることはできませんが、ゲームで“SetBusEffect()”や“SetActorMixerEffect()”を使い、AUXバスに異なるエフェクトシェアセットを設定することができます。この方法で環境アコースティックを細かく設定できます（オーディオオブジェクトごと、そしてゲームオブジェクトごとに）。

AUXセンドをコントロールして、クリエイティブでランタイムを意識したスペシャルエフェクトをプロジェクト全体で適用できます。

ゲーム定義のAUXセンド

ユーザー定義AUXバス以外に、1つのゲームに対して最大4つのゲーム定義AUXバスを、ゲームエンジンによって設定、またはオーサリングアプリケーション外のプログラムで管理することができます。ゲーム定義のAUXバスは、リバーブ、ゲームステート依存のエフェクト、インタラクティブミキシングなど、ゲームの状況に基づく使い方ができます。また、サウンドデザイナーはユーザーインターフェースからProperty Editor画面のGeneral Settingsタブを開いて、ゲームが決めるAUXセンドボリュームの有効化やオフセットをサウンドごとにできます。



Property Editor画面のGame-Defined （ゲーム定義）Auxiliary Sendsを適用

ゲームオブジェクトは、ゲーム定義とユーザー定義のAUXバスを組み合わせて、最大8つの独立したAUXターゲット（ゲーム定義4、ユーザー定義4）にルーティングできます。

Designer Note

最終的なセンドボリュームは、UIのゲーム定義AUXセンドボリュームと、プログラマーが設定した SetGameObjectAuxSendValues() SDKの組み合わせとなります。

ステートとミックススナップショット

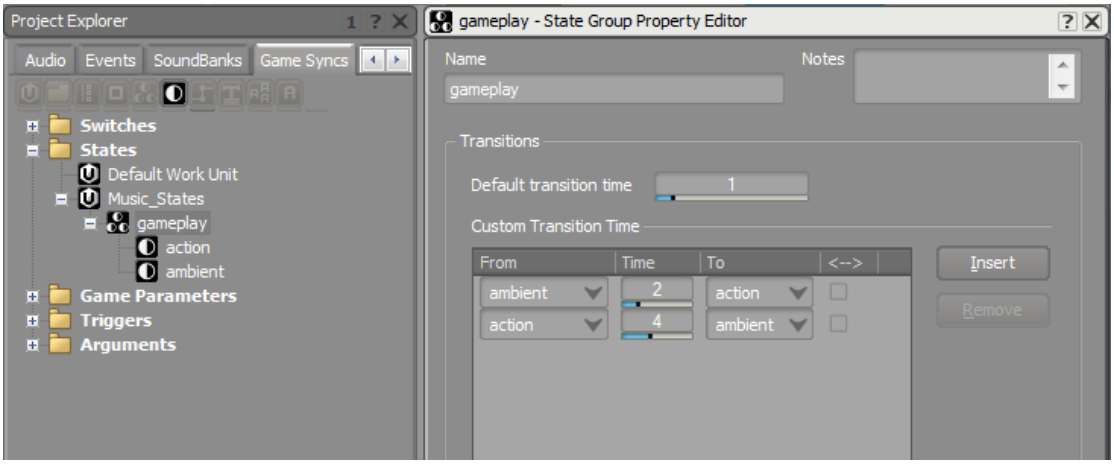
インタラクティブミックスを操作するもう1つの強力な切り口がステート（状態）、つまりこの場合はミックスのスナップショットです。ステートはコンバット、ステルス、アイドルなどのゲームステートと直接関係していることがあり、呼び名も同じことがよくあります。他にも森林、廊下、ダンジョンなどの空間を定義するためにも使われ、ゲームのサウンドを変更することが予想されるあらゆる条件を定義できるように、抽象化することができます。ステートは一般的にゲームエンジンに定義されWwiseでトリガーされますが、ここで複数のステートを同時に組み合わせることも可能です。

Designer Note

1つのオブジェクトが複数のステートに登録されていると、1つのプロパティが複数の変数値の影響を受けることがあります。この場合、値の増減の合計が使われます。例えば2つの異なるステートグループにある2つのステートで、それぞれ-6DBのボリューム変更が設定され、両者が同時にアクティブになると、最終的なボリュームは-12dBです。

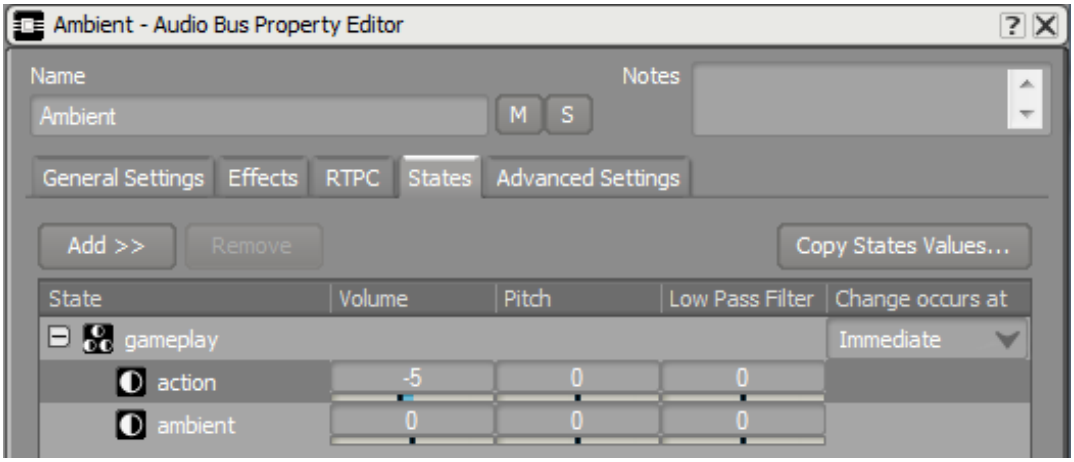
多くのゲームは非線形に進行し潜在的にランダムな性質を有するので、ゲームで起きるイベントにダイナミックに対応できるミックスを設計する方が、ゲーム内の状況に応じて変化することのない静的なミックスよりも推奨されることが多いです。多様なミックスバスのプロパティを調整するためにステート変更を利用すれば、ゲームをダイナミックにミックスするシステムをつくれます。これはある意味、サウンドデザイナーのルールに従いミックスを変化させる人口知能のプログラムとも言えます。

ステートの定義はProject Explorer画面のGame Syncsタブで行い、デフォルトのトランジション時間の設定や、ステート間の切り替わりに使うトランジションのカスタム設定ができます。



ステートの定義と、デフォルトやカスタム設定のトランジション時間

ステートが確立されれば、どのオーディオオブジェクトやオーディオバスでも、Statesタブを開いてそのステートを追加すれば、ステート変更で影響を受けるプロパティを利用できます。



“Ambient”バスのステートが “action”にある時の、ボリューム減少の設定

ステートの詳細情報：

Wwise Help > Interacting with the Game > **Working with States**

Wwise Knowledge Base - Using Multiple States to Affect Sounds

Wwise Knowledge base - Creating a Temporary Loss of Hearing Effect

オートダッキングvs.サイドチェイン

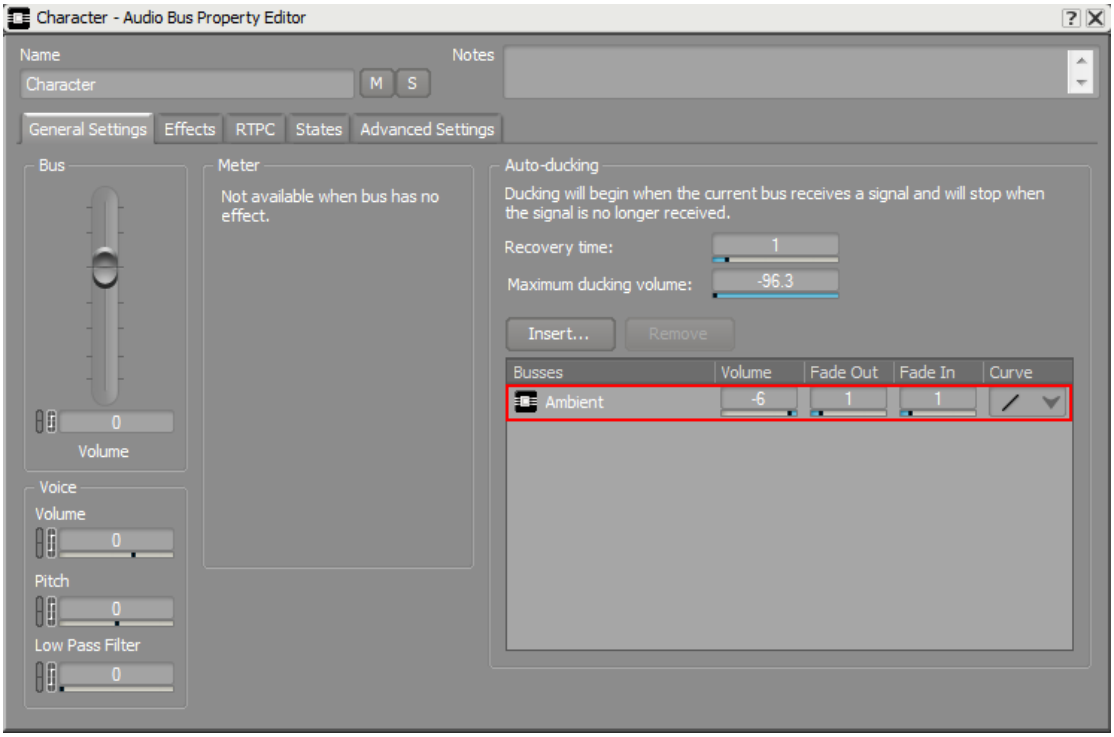
ミックスのステートでプロパティの変化を定義しますが、他のオーディオバスから送信された信号に基づいて、ボリュームを自動的に下げることもできます。これが、よくダッキングと呼ばれる処理です。

Designer Note

ダッキングとは、あるバスを通る全てのオブジェクトのボリュームを自動的に下げ、他のバスを目立たせる手段です。

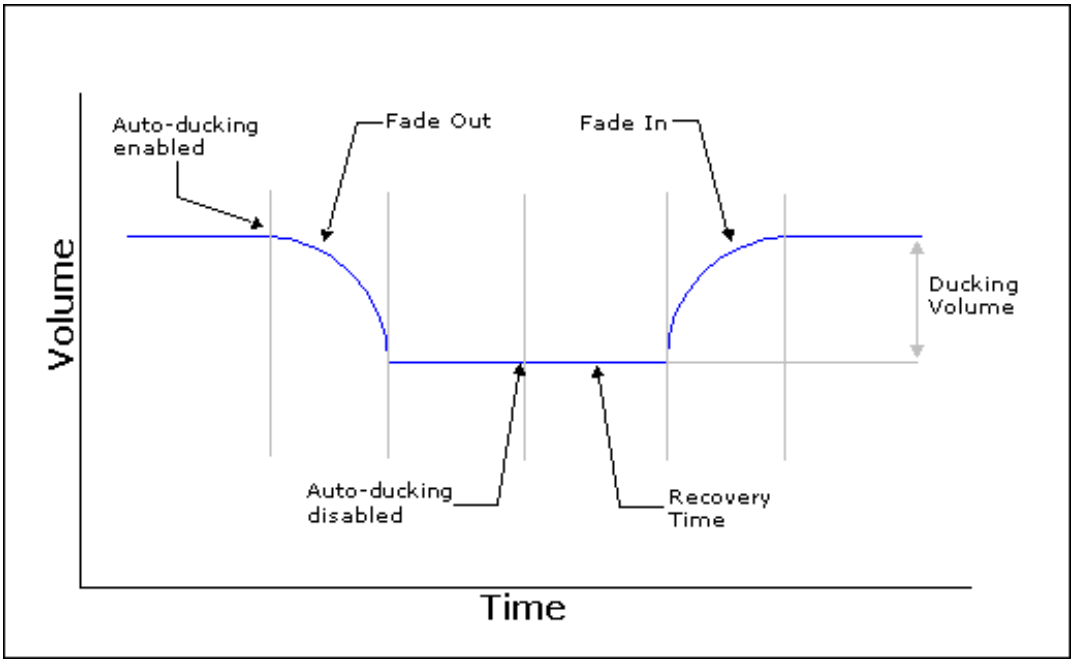
オートダッキング

Wwiseのどのオーディオバスでも、General Settingsタブを開くとオートダッキングが設定できます。オートダッキングのウィンドウに他のバスを入れて、ボリューム減衰やフェードイン・フェードアウトのプロパティを設定します。



オーディオバス “Character”がオーディオバス “Ambient”を、1秒かけて直線カーブに従って-6dBだけオートダッキングさせる設定

選択されたバスが何らかの信号を受信した時点で、オートダッキングが始まり、インサートされているバスは、その信号（サイレンスを含む）が継続する間は減衰設定が適用されます。



サイドチェイン

サイドチェインは、ダッキングの別のアプローチです。サイドチェインではオーディオ信号のレベルをモニタリングして、他のオーディオ信号を操作するのに利用します。例えば、ラジオでDJの声で音楽のボリュームが自動的にダッキングされるのは、サイドチェインの典型的な例です。

基本的に、ミキシングにサイドチェインを取り入れると、Wise Meterエフェクトで入力信号のボリューム情報を使い、ミックスの他の部分に作用することになります。

Designer Note

サイドチェインの詳細情報：

http://www.audiokinetic.com/download/documents/Wwise_SideChaining_Tutorial.pdf

RTPCを使ったミキシング

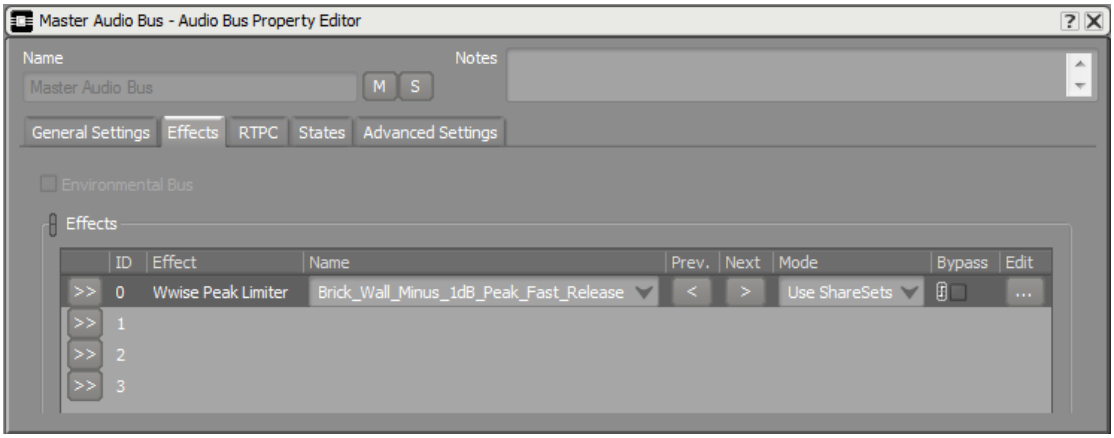
ボリューム変更を起こすためにゲームが送信するパラメータを使ってダイナミックなミキシングを行うために、リアルタイムパラメータコントロール（RTPC）を活用することもできます。パラメータを使った一般的なミキシング手法として、プレイヤーが弱体化するにつれゲームの環境ミックスにローパスフィルターを適用することができます。単数または複数のオーディオバスに対するRTPCに健康状態のゲームパラメータを使えば、簡単に達成できます。

横型ミュージックシステムで激しさを増加させるためにゲームパラメータ“Danger”を使いました。攻撃の場面で注目しなくてもよいサウンドやオーディオバスのボリュームを下げるためにも、“Danger”を使えます。例えば、アンビエントサウンドやキャラクターの動作サウンドのボリュームをRTPCカーブで下げ、ミックスに入っている攻撃のサウンドが聞こえる空間を作ります。

ミックスの様々なやりとりを達成する方法が複数あると、問題が発生してもいくつかの解決策が考えられます。RTPCのミキシングは、インタラクティブでダイナミックなミキシングを提供するツールボックスに入った、有効なツールの1つです。

マスターミキサーでエフェクトを使用

DSPは、リニア音楽やサウンド制作の現場で既に何十年も使用されてきました。中でもイコライザー、ディレイ、ディストーション、コンプレッションなど多くは、主要なDAW（デジタルオーディオワークステーション）に数十年前から取り込まれていますが、ランタイムに使用してミキシングに適用できるレベルに改善されたのは、最近（ここ10年ほど）のことです。成長中のこの分野が注目を集めるのは、事前に定義された設定だけでなく、ゲームから送られるパラメータを使ってDSPを操作して、他のテクニックでは得られないスペシャルエフェクトを生み出す、ダイナミックでインタラクティブな性質を有するからです。



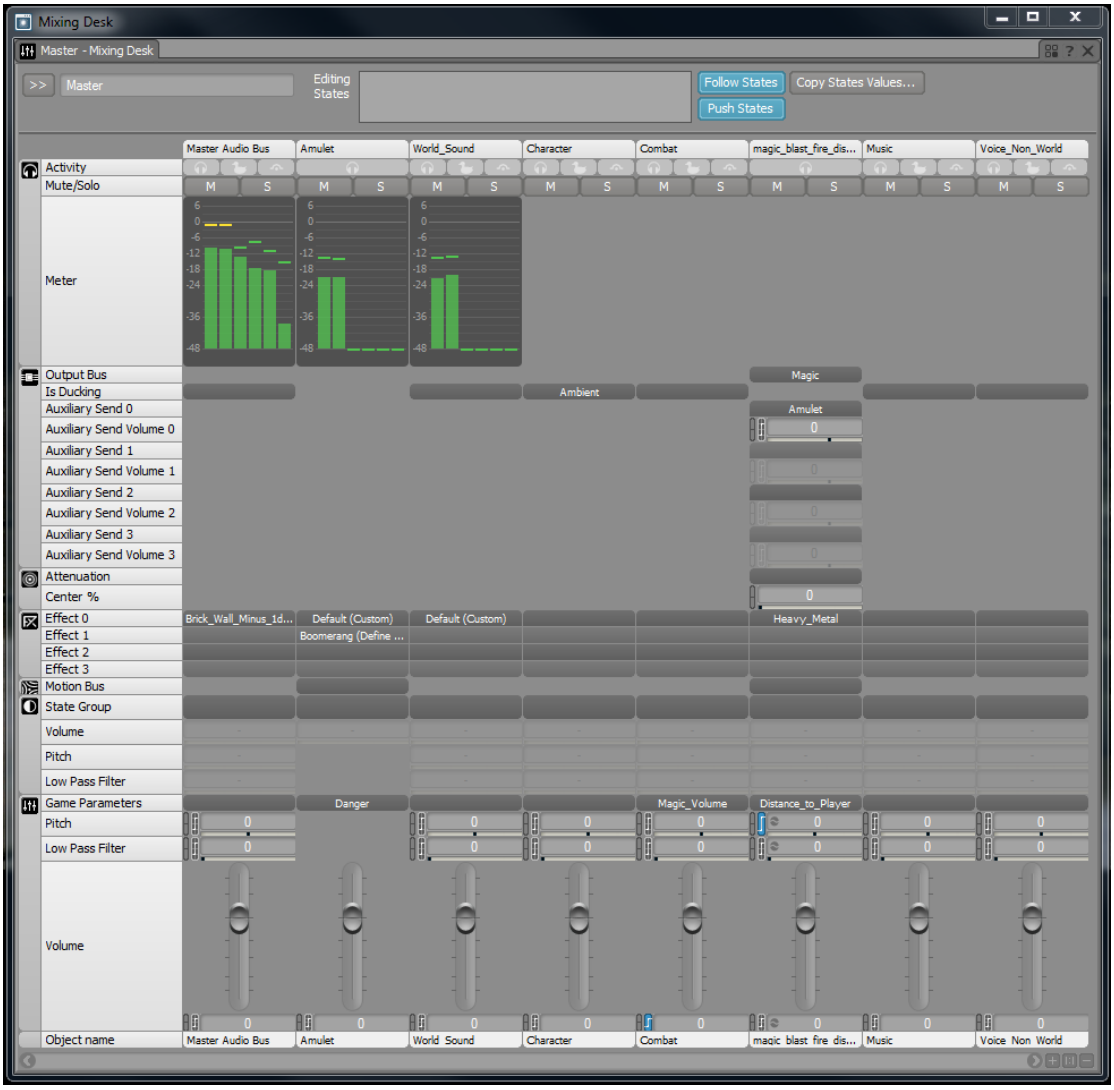
マスターオーディオバスにリミッターを設定

ランタイムでエフェクトを適用すればCPUを消費しますが、負荷の違いを知っていれば効率的な使い方を選べます。例えば、サウンドオブジェクトのモノインスタンス数個にエフェクト（単数または複数）を適用する時は、アクターミキサーのレベルで適用した方が効率的です。一方、同時に起きる多数のサウンドをエフェクトで処理するには、そのエフェクト（単数または複数）をオーディオバスに対して設定して、ミキシング後にエフェクトを適用させた方が効率的です（スピーカー設定によって、ステレオまたは5.1ch）。

ミキシングデスクのビジュアル化

Mixing Desk画面は、様々なプロパティをまとめて表示でき、リアルタイムでオーディオミックスの微調整ができる、柔軟で強力なインターフェースです。サウンドオブジェクト、オーディオバス、AUXバスをMixing Desk画面に自由に追加して、ルーティングの設定、エフェクトや減衰のシェアセットの適用、ステート設定の変更、プロパティの調整などをオブジェクトやバスごとに実行できます。

ゲームにオーディオとモーションを実装すれば、Wwiseをゲームに接続してゲームのプレイ中にリアルタイムでプロファイル情報を確認できます。ゲーム中の調整やミキシング変更を、リアルタイムでオーサリングアプリケーションから実行できます。ミキシングデスク内の各オブジェクトのアクティビティも確認でき、ボイスの再生中にダッキングされるバスがあるのか、あるいはマニュアル操作やプログラム設定によりエフェクトがダッキングされているのかも、観察できます。オーディオオブジェクトをミュートまたはソロにして、オーディオミックス内のオブジェクトを個別に調整することもできます。



Mixing Desk画面でミックスをビジュアル化して表示

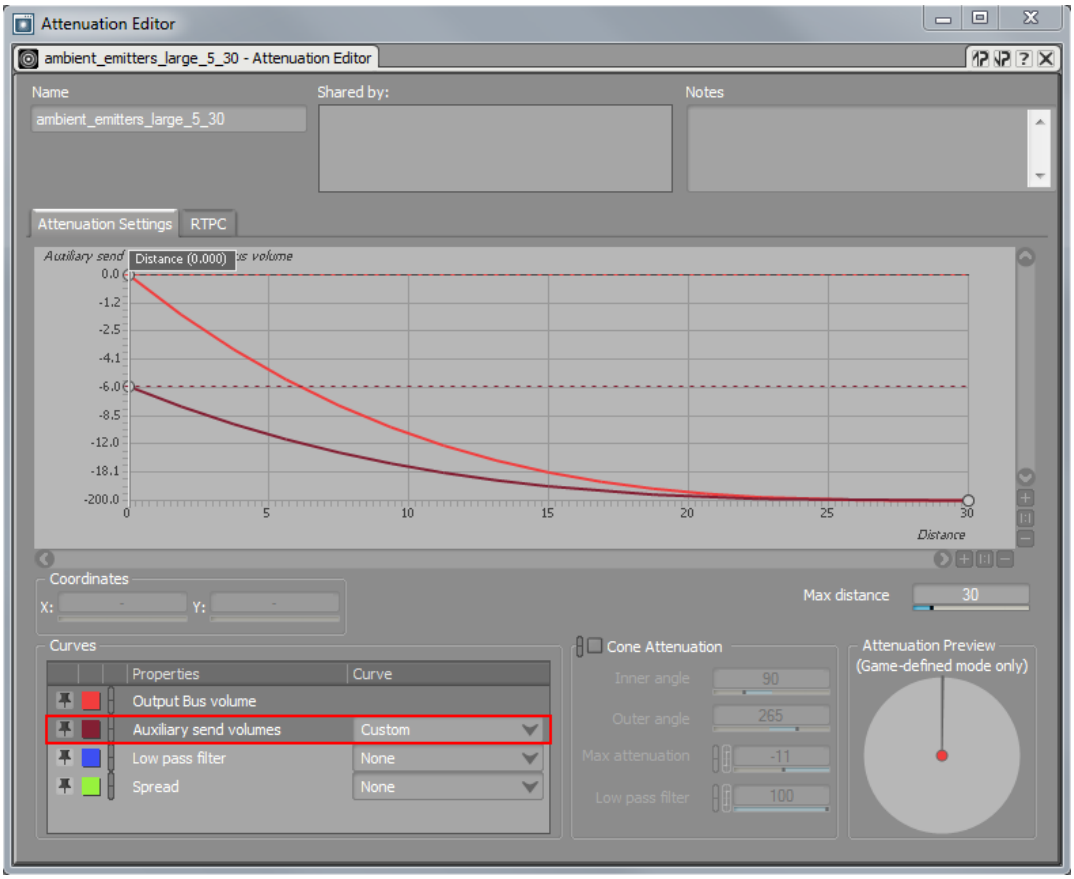
ミキサーストリップにあるプロパティ欄には、それぞれコンテキストメニューが付いています。このメニューからプロパティを設定するための専用コマンドを選べます。例えば、Effect 0からEffect 3までの4つの欄のいずれかを右クリックすると、インサートされているエフェクトのプロパティ編集、新規エフェクトの設定、エフェクトのバイパスなどを実行できます。コンテキストメニューをアクセスするには、ミキサーのストリップ上で編集したいプロパティを右クリックします。

セットアップされたミキシングデスクは、ミキシングセッションのシェアセットとして保存されます。この機能を使い、ゲームの様々な要素ごとに専用のミキシングセッションを作成できます。また、一度セットアップしたミキシングセッションを後から呼び出して、細かい調整を再開することもできます。

減衰のミキシングテクニック

ミキシング結果に減衰を適用して全体のバランスを保つ手法は、ゲームのサウンドを自然に仕上げる最良の方法の1つです。減衰の最大距離を延長すると同時にカーブの深さや形状を変更するだけで、著しい効果を出せることもあり、サウンドがより遠くまで、より低いボリュームで聞こえるようになります。逆に最大距離を短くすれば、込み合ったミックスをすっきりさせてサウンド同士が区別できるようになります。

さらに、減衰のAUXセンドのボリュームはゲーム定義のバスとユーザー定義のバスの両方に送られる信号の量と、直接関係します。1つのサウンドやサウンドグループの出力バスボリュームとゲームまたはユーザー定義のセンドボリュームをバランス良くすると、ミックスの詳細が明確になります。



カスタム化したAUXセンドのボリュームカーブを、減衰設定に追加

ミックスのまとめ

総合的でダイナミックなミキシングを実現するツールが手元にあれば、最終ミックスを完成する段階でサウンドデザイナーが焦ることはありません。リニアな音楽業界で一般的に使われるツールを元に設計され、インタラクティブに反応する能力を内蔵したツールがあれば、ダイナミックミキシングの次のレベルへの扉が開かれます。

本章では、以下を説明しました。

- オーディオバスを使った信号送信とルーティング
- ユーザー定義AUXセンドの使い方
- ゲーム定義AUXセンドの使い方
- ミックススナップショットとしてのステートの使い方
- オートダッキングとサイドチェインの比較
- RTPCを使ったミキシング
- マスターミキサーでエフェクトを用いるメリット
- 画面を使ったミックスのビジュアル化
- 減衰を使ったミキシングのテクニック

プロセスの説明

- オーディオバスの作成
- AUXバスの作成
- ステートの設定
- マスターミキサーのオーディオバスに体するオートダッキングの適用

詳細設定の説明

- 減衰エディターにおける、ウェットとドライのリバーブボリュームの関係

オブジェクトの作成方法

- ゲームパラメータ“Danger”に基づいてマジックやウェポンスイングのサウンドを変化させるAUXバス “Amulet”

参考ドキュメントとチュートリアル

Video Tutorial - Mixing Desk

Video Tutorial - Dynamic mixing using States

Video Tutorial - Wwise Side-Chaining

Wwise Knowledge Base - Advanced settings: usage and dynamic mixing techniques

Wwise Knowledge Base - Wwise Side-Chaining Tutorial

Wwise Knowledge Base - Playback Limit and Priority: Use Case Scenarios

Wwise Knowledge Base - Using Multiple States to Affect Sounds

Wwise Knowledge base - Creating a Temporary Loss of Hearing Effect

Wwise Help > Interacting with the Game > **Working with States**

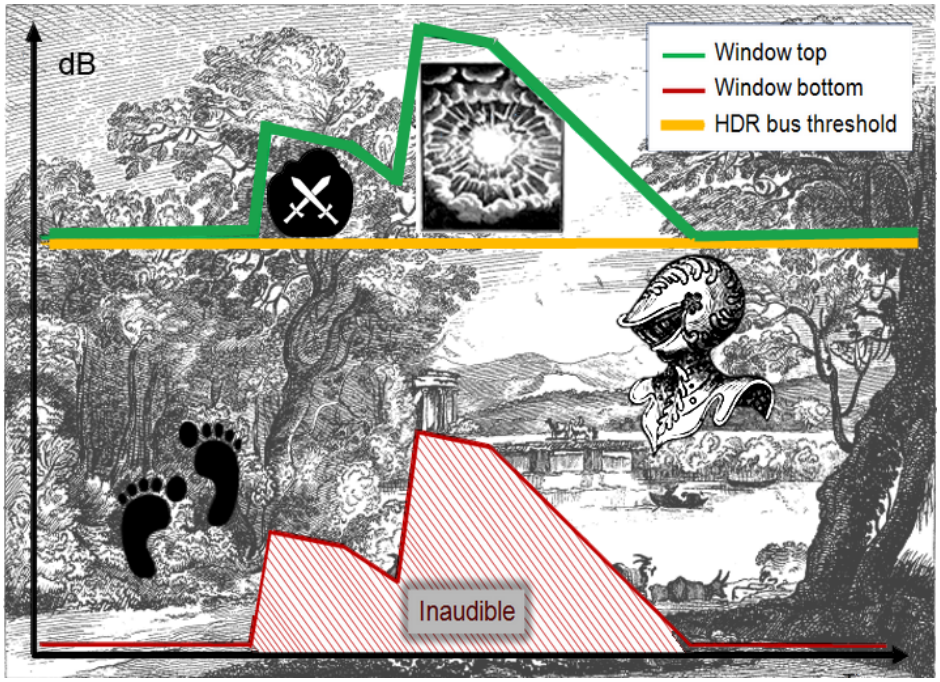
第10章 HDRオーディオWwizard

概要

霧のかかった山の上から、年老いたWwizardが、下の谷で繰り広げられるバトルをじっと見つめます。混乱の中、剣戟の冷たい金属音や、マジカルパワーのブラスト、投石の突撃音など、大きなサウンドが次々と上まで聞こえます。優先順位の高いサウンドが発生する度に、サウンドスケープにスペースがつくられ、繰り広げられる狂乱を強化する音が加わります。プレイヤーに気付かれないところで高度に調整されたメカニズムが作動して、耳に聞こえるラウドネスの優先順位が決まり、プレイヤーが感じ取る周囲の環境がサウンドで形成されます。それでは、大きいサウンドを優先するという考え方は、今日のゲームにどのように取り込まれているのでしょうか。

HDR（ハイダイナミックレンジ）は、Wwiseの様々なダイナミックミキシングツールを強化するために導入された、新しいミキシングパラダイムです。HDRを使うことで、大きいボリュームでオーサリングされたサウンドが優先され、ラウドネスのダイナミック性に対応するシステムを作成できます。音の相対関係を反映するミキシングシステムによって、オーサリング時の意図を維持して、重要なサウンドが聞こえるようにミックスの中に充分なスペースを確保できます。

先ほどのシーンに戻りましょう。バトルのサウンドが続く中、プレイヤーは無意識に最大サウンド（コンバット音）にフォーカスして、小さいサウンド（足音）を無視します。HDRシステムでは、大きいサウンドが、ユーザーの設定したウィンドウの上端（Window top）位置を決定し、このウィンドウが、最大サウンドが目立つようにダイナミックに移動します。ウィンドウが上に移動するにつれ、ウィンドウの下端（Window bottom）よりも下に位置するサウンドは、ミックスから外されます。谷間で繰り広げられるバトルで一方が勝利をおさめ、人々が霧に包まれた山の上を見上げると、大地を走る馬の蹄サウンドの振幅が強まり、最も音の大きい、脅威のサウンドとなります。HDRウィンドウのスレッショルド（Threshold）が元のポジションに戻ると、馬のサウンドの元々の振幅が表現されます。



**サウンドのラウドネスが徐々に増加する
と、HDRウィンドウも反応してシフトする**

本章では、以下の操作を説明します。

- WwiseのHDRオーディオの活用
- HDRオーディオミックスの設定
- HDRオーディオの、ダイナミックレンジウィンドウの設定
- マスターミキサー階層における、HDRオーディオの有効化
- HDRオーディオの、ダイナミックプロパティの設定
- アクターミキサー階層におけるHDRオーディオの利用
- エンベロープトラッキングの有効化
- 波形エンベロープの編集
- ソースノーマライゼーションの有効化
- メイクアップゲインの利用
- HDRオーディオを理解するための、Voice Monitorの利用
- Voice Monitorビューの設定
- Soundcasterを使ったサウンドの試聴
- Wwiseのデータキャプチャー

WwiseのHDRオーディオの活用

WwiseのHDRは、マスターミキサー階層のどの親オーディオバスに対しても、有効にできます。HDRを有効にしたオーディオバスは、HDRバスのインプット側にあるサウンドオブジェクトのボリュームと、同じバスのアウトプット側のフル（デバイス）スケールの間の、コンバーターとして機能します。このHDRバスにルーティングされた全てのサウンドが、相互関係を考慮してバス内で扱われ、音の小さいサウンドのアウトプットは、システムのプロパティ設定に合わせて、常に修正されます。

HDRバスのコントロール設定は、オーディオコンプレッサーのものと似ています。スレッショルド（閾値）、レシオ（比率）、リリースタイムのプロパティを使って、プロジェクトに設定されたダイナミックレンジウィンドウの動作を変更します。ランタイムに、オーサリングしたシステムがこの広い範囲に及ぶ様々なレベルを、使用するサウンドシステムのアウトプットに適したボリュームレンジにダイナミックにマッピングします。

可聴ダイナミックレンジは、最も大きい音や人間の聴力の限界によって決まり、実際には、ゲームプレイ向けのスピーカーで提供できるダイナミックレンジよりも何倍も広い範囲です。Wwise HDRシステムの役割は、現実世界のダイナミックレンジを、約40dB（テレビや音楽視聴向けレベルの70dBSPLから、室内ノイズレベルの30dBSPLを引いた数値）まで畳み込む、つまり圧縮することです。

この処理は、一種の動作圧縮です。大きいサウンドが再生された瞬間に小さいサウンドが聞こえなくなり、また小さいサウンドだけが再生されると、再び聞こえるようになるので、サウンドミックスに影響します。サウンド同士の相対的なレベルを維持して、再生するサウンド数を減らすことで、ミックスが鮮明になります。

Designer Note

これまでの文献で、HDRオーディオシステムは、サウンドごとにSPL値を直接アサインすると説明されています。Wwiseでは、SPLという概念を排除し、代わりに相対的ミキシングにフォーカスをあてています。このため、WwiseにはSPLスライダがどこにもなく、相対的デシベル値だけが使われます。現実世界のSPL値をこのシステムで採用したい場合は、基準値とする値を決め、必要な引き算をして該当する相対的dBレベルを計算します。例えば、100dBSPLを基準値、つまり0dBとして定義します。そして、80dBSPLのサウンドのボリュームスライダは-20dBに、130dB SPLのサウンドのボリュームスライダは+30dBに設定します。

HDRオーディオミックスの設定

オーサリングアプリケーション内にある様々な機能を使いながら、HDR機能を活用することで、最も大きいサウンドをミックスの前面中央に配置できます。各種サウンドタイプの重要性を決定してから、変化するミックスの中に、これらのサウンドタイプを制作中に配置することで、HDRシステムを使用している時のサウンドやサウンドタイプ同士の重要性が決まります。ここから生まれるミックスは、サウンドごとに設定されたボリュームに従い、推移するダイナミックレンジウィンドウの作用で常にバランスが調整されます。

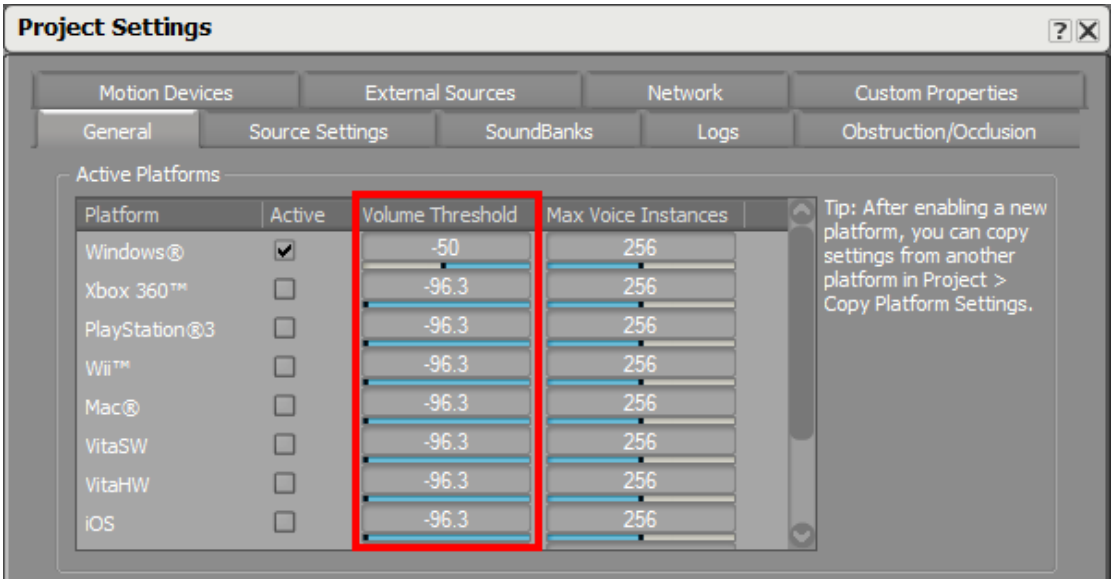
まず最初に、プロジェクトのダイナミックレンジ（ボリュームスレッショルド）を決めます。次に、1つのオーディオバスのHDR処理を有効にして、ダイナミック関連のプロパティを設定しながら、システム内のシグナルにウィンドウがどう反応するかを調整します。

全てのAUXバスにメーターがあり、ボリューム設定、エフェクト追加、RTPC、ステートに応じた変化などの機能が備わっています。

HDRオーディオのダイナミックレンジウィンドウの設定

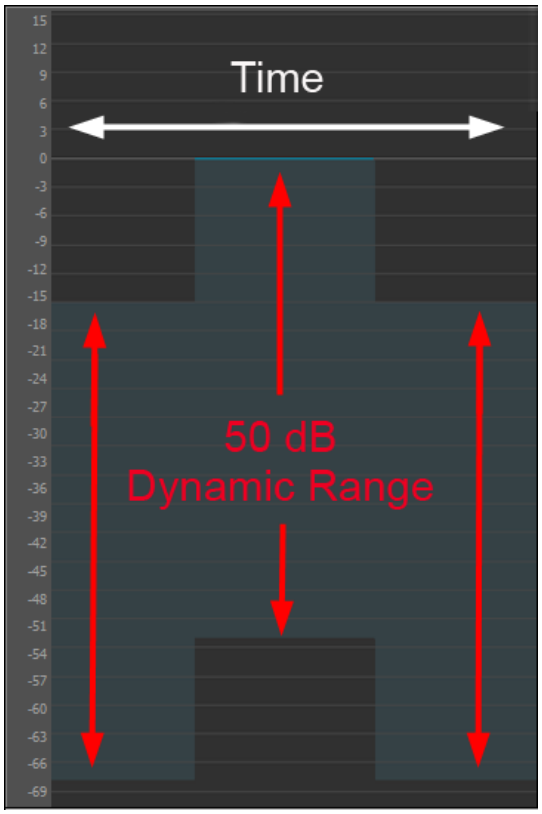
WwiseのHDRを準備するには、まずウィンドウの縦幅となるボリュームスレッショルド（Volume Threshold） を設定しますが、ここに対象プラットフォームの予想されるダイナミックレンジを収めることになります。このボリュームスレッショルドは、HDRシステムのアウトプットのダイナミックレンジに適用されます。

提供中のWwise Project Adventureを使ってHDRシステムを準備するには、下図の通り、Project Settingsのボリュームスレッショルドを-50dBに設定してください。



Project SettingsのVolume Thresholdの設定

Project Settingsのボリュームスレッショルド値によって、ウィンドウの上端と下端の差が決まります。つまり、最大サウンドの振幅がHDRバスボリュームで再生される時に、このプラットフォーム用に設定したボリュームスレッショルド（ウィンドウの下端）より下にある全てのサウンドは、サウンド構造のオブジェクトプロパティで設定した詳細設定に従って、キルされる（停止される）か、 バーチャルボイスとなります。ウィンドウは、このウィンドウに設定されたダイナミックレンジを維持したまま、HDRバスボリュームが最大である再生中のサウンドに合わせて、上下します。

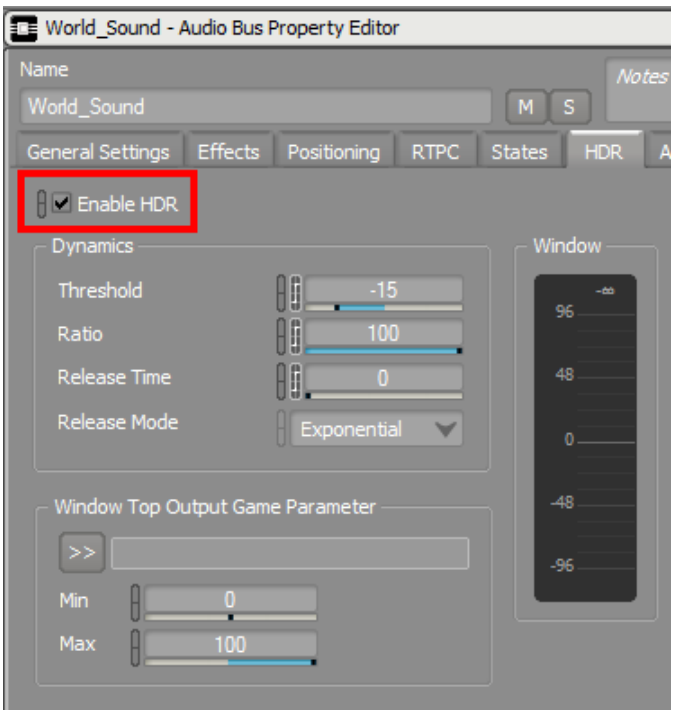


50dBのダイナミックレンジのボリュームスレッショルドウィンドウが、移動していく様子

ボリュームスレッショルドによって、指定したボリュームレンジ内のサウンドが、アウトプット側で聞こえます。ウィンドウが移動すると、再生中の各種サウンドのボリュームが、最大サウンドの値によって変化します。この処理によって、最も大きいボリュームとしてオーサリングされたサウンドは聞くことができ、最も小さいサウンド（ウィンドウ下端よりも下のサウンド）は、ミックスから除かれます。

マスターミキサー階層で、HDRオーディオを有効にする

HDRは、親レベルのどのオーディオバスに対しても、有効にでき、そのバスの階層中の子バスにルーティングされる全てのサウンドが、対象となります。ある階層でHDRオーディオバスとして設定できるのは1つのバスだけですが、HDRを有効にしたオーディオバスの外の、他のオーディオバスは、追加のHDRバスとして利用できます。

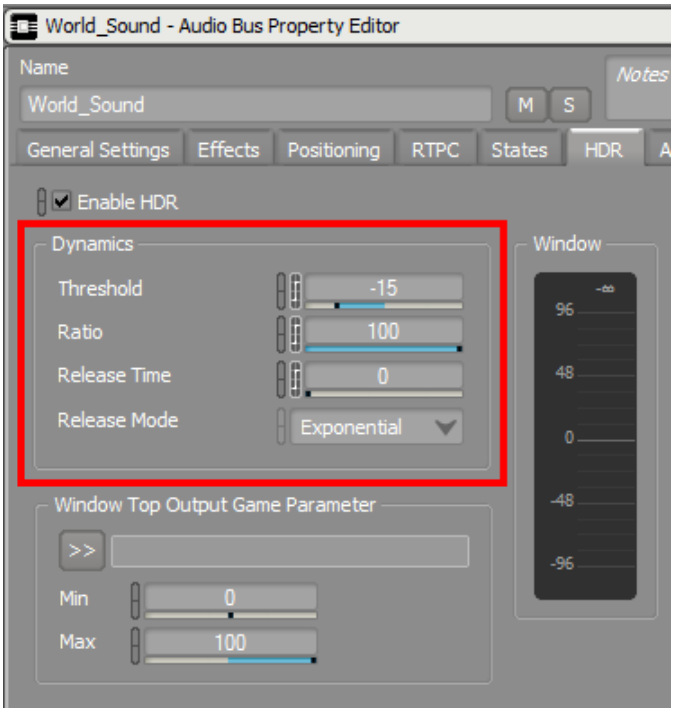


マスターミキサー階層で、HDRを有効にする

HDRオーディオの、ダイナミックプロパティの設定

ダイナミックプロパティを設定することで、HDRオーディオバスにルーティングされるサウンドが互いに作用し合った時のHDRシステムの反応を規定します。様々なボリュームのサウンドがHDRオーディオバスを経由すると、これらのサウンドの振幅が、スレッショルド、レーシオ、リリースタイムなどのなじみあるプロパティに基づき、グローバル（全体）ボリュームをダイナミックに調整します。

Wwise Project Adventureを使ってHDRシステムを試してみるには、ミックスバス“World_Sound”でHDRを有効にして、プロパティThresholdを-15dBに設定し、他のDynamics設定はデフォルト値のまま残します。



HDRを有効にしたバスの、Dynamics設定

Threshold（スレッショルド）

スレッショルドを設定して、HDRウィンドウの上端が反応するための、最低インプットレベル（単位：dB）を指定します。

Ratio（レシオ、比率）

このコントロール機能は、オーディオコンプレッサーのレシオコントロールと似た動作をします。HDRウィンドウの上端は、レシオに従ってスレッショルドを超過したピーク部分を減衰させ、同時に、小さいサウンドのボリュームを下げる役割を担います。例えば、2つのサウンドがあり、1つのピーク値がスレッショルドより20dB上に、もう1つがスレッショルドより40dB上にある場合は、同時に再生されない限り、両者とも0dBFSという同じレベルで出てきます。この2つのサウンドの違いは、スレッショルドより低いサウンドが、前者では20dB減衰するのに対して、後者は40dB減衰するという点です。

より低いレシオ、例えば4では、+20dBでピークするサウンドは+5dBで出て、+40dBでピークするサウンドは+10dBで出てきます。その結果、スレッショルドよりも下のサウンドの減衰が、前者は-15dB、後者は-30dBとなります。 スレッショルドよりも上にあるサウンドの場合、HDRシステムによって全体的なダイナミックレンジが失われてしまうことがあり、これを回復するのに低いレシオは便利です。欠点は、

スレッショルドより上でピークに達するサウンドもあることで、クリッピングを回避するには、HDRバスの後で充分なヘッドルームを確保する必要があります。これは、HDRバスのボリュームを0dBより低い値（例えば、-10dB）に設定することで対応できます。

Release Time（リリースタイム）

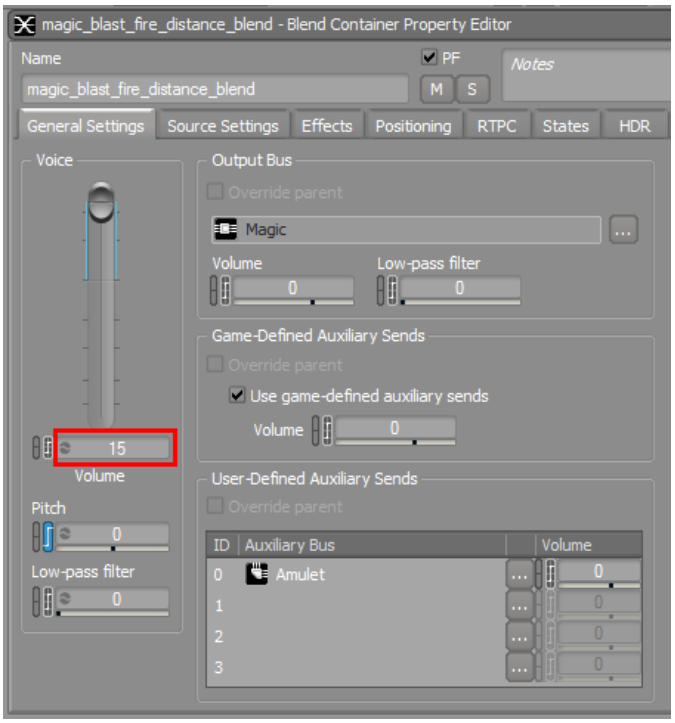
リリースタイムを設定して、ターゲット値が現在の値よりも低い時、HDRウィンドウが下がって安定するまでの速度を指定します。Linear Mode（線形モード）では、約10dB下がるまでにかかる時間を、秒単位で指定します。Exponential Mode（指数モード）では、ターゲットと現在値の差の、約0.37（1/e）の位置に達するまでにかかる時間を、秒単位で指定します。どちらのモードを選択するかは、ソース素材やゲームの種類を考慮して聞こえの良い方に決めます。

アクターミキサー階層におけるHDRオーディオの使用

アクターミキサー階層の各サウンドに対して、追加のHDR設定が可能です。Property Editorの、HDRタブを開くと、HDRシステムで処理されるサウンドの動作に影響する詳細を、改めて検討できます。

Wwise Project Adventureで、アクターミキサー階層にある既存ワークユニットごとに、サウンドグループ別のボリューム値を、以下の通り設定します。

- Ambient: -20 dB
- Character: - 10 dB
- Combat: -5 dB
- Magic: 15 dB



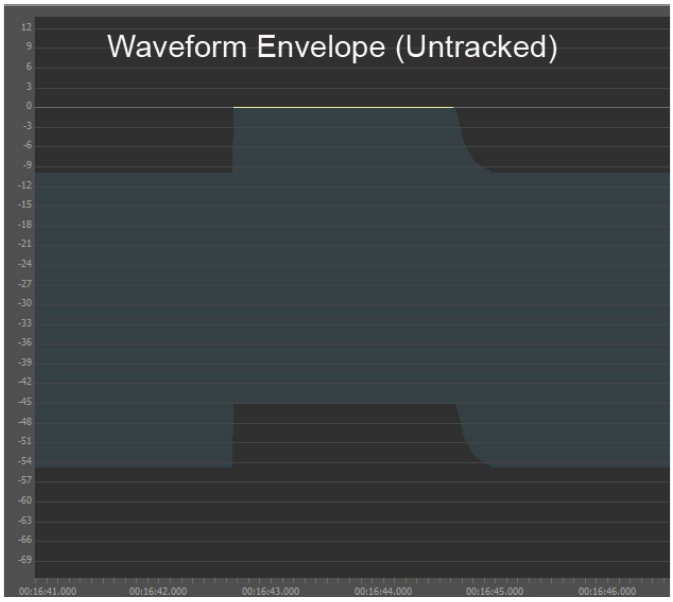
ブレンドコンテナ“magic_blast_fire_distance_blend”のボリューム設定

上記の数値は相対論的なミックスを表し、マジックサウンドが最大となり、次に、コンバット、キャラクター、アンビエントの順に、振幅が減少します。このようにグループ同士のミックスをシンプルに表すことで、HDRシステム内で、異なるサウンドタイプを試聴でき、再生中に互いにどのような影響を与えるのかを確認できます。

エンベロープトラッキングを有効にする

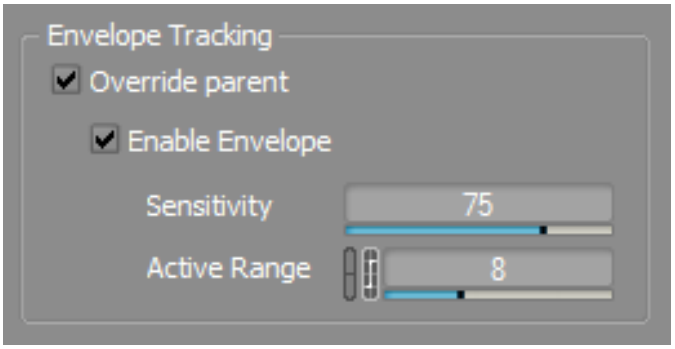
マジックブラストが初めて戦場の隅々まで響き渡ると、山のはるか上の方から、スレッショルドがHDRシステムを作動させます。マジックブラストのボリュームは、プロジェクトの最大サウンドとしてオーサリングされているので、ウィンドウ上端を素早く引き上げ、足音サウンドが、ミックスから外されます。戦場のどこにいようと、マジックブラストの振動が危険を知らせ、周囲の自然、足音、倒れた戦士達など全てのサウンドは遠ざかり、絶大なサウンドに注目せざるをえません。

ウィンドウ上端がマジックブラストのエンベロープの位置を追いかける間、剣戟の冷たい金属音が、疲れ切った戦士達の間で次々と広がります。剣戟のインパクトサウンドはマジックブラストの相対ボリュームよりも下にオーサリングされたもので、鋭い攻撃のように鳴り響き、ウィンドウ上端を引き続きピークさせます。スレッショルドより下にあるサウンドの減衰の様子は、レシオで設定したプロパティ値の影響を直接受けます。このため、次の攻撃に近づき戦いの騒ぎが大きくなれば、アンビエントはミックスから外れたままとなります。

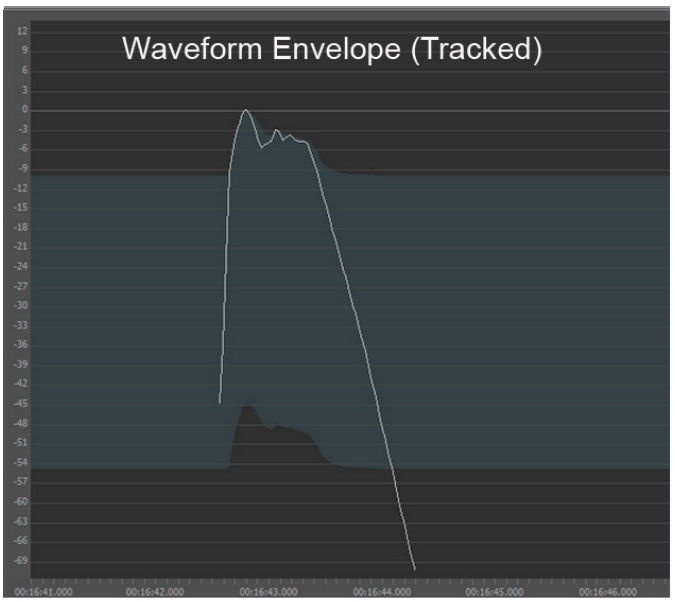


トラッキングなしの波形エンベロープ
が、HDRウィンドウ上端を上に移動させる様子

エンベロープトラッキング（Envelope Tracking）を有効にすると、Wwiseがオフラインで自動的に解析した計算結果を使い、波形のエンベロープに合わせてウィンドウ上端を調節します。これは、ウィンドウ上端をボリュームスレッショルドより上に動かすような大きいサウンドで使うと、非常に便利です。波形エンベロープをトラッキングする処理は、デフォルトエンベロープの動きと比べて多少のメモリコストがあるので、大きいサウンドや、ボリューム変動のある、長いエンベロープのサウンドに使用することが推奨されます。

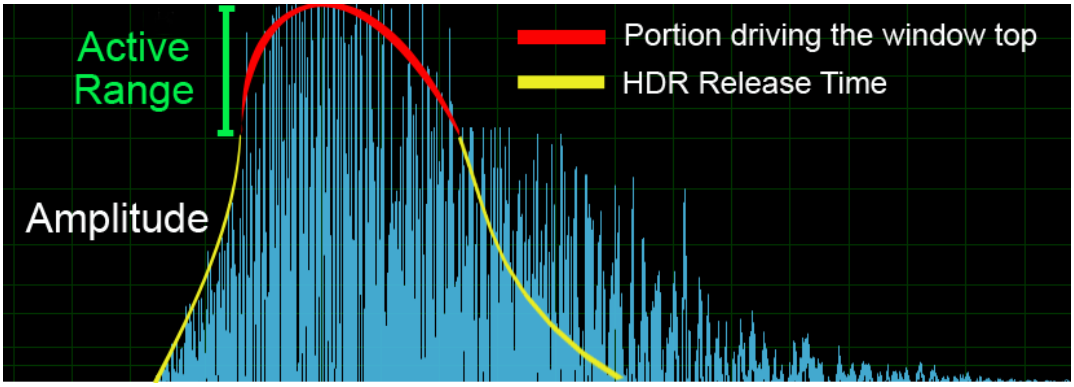


Envelope Trackingのプロパティの、SensitivityとActive Range



**トラッキングしているWaveform Envelope
が、HDRウィンドウ上端を動かす様子**

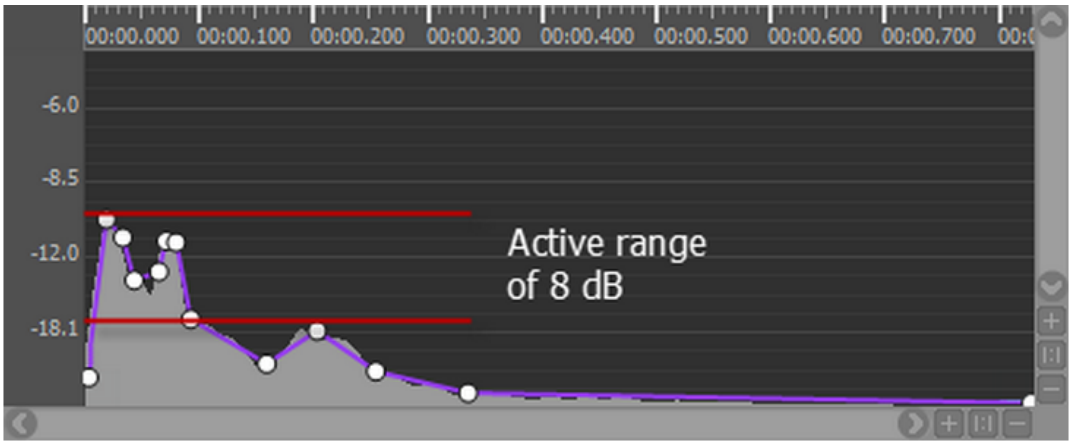
アクティブレンジ（Active Range）を設定して、ボイスのピーク値からHDRが作用するエリアまでの範囲を指定します（dB単位）。このROI（Region of Interest、関心領域）は、解析したエンベロープに基づき、現在のエンベロープレベルが「ピークレベル」から「アクティブレベル」を引いた位置よりも上であれば、アクティブレンジとされます。アクティブレンジ外の場合は、HDRがサウンドの内容を無視するので、HDRバスのリリースタイムが適用されます。



**あるサウンドのActive RangeとHDR Release Time
の関係を、サウンドのエンベロープを元に示したグラフ**

波形エンベロープを編集する

Source Editor画面で、サウンドファイルの波形エンベロープ（Waveform envelope）を個別に調整することができます。

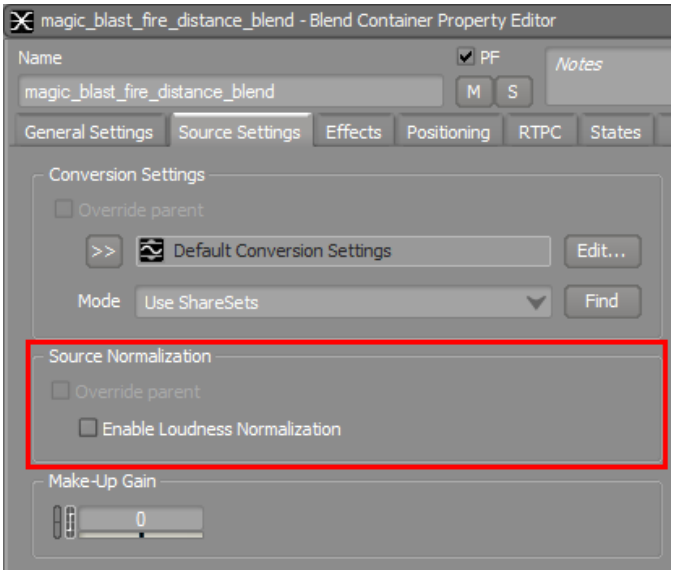


オーディオソースエンベロープを編集することで、オーディオシグナルの重要部分を隔離して、Object properties のHDRタブで指定した8dBのアクティブレンジに含める

エンベロープ編集のセンシティビティ（Sensitivity、感度）の設定は、エンベロープトラッキングを有効にした時に、エディタに表示されてトラッキングされるデータポイントの数に直接、関係します。センシティビティを下げると、ランタイムにHDRシステムがトラッキングする情報の精度が下がります。データポイントは、Source Editorでデータポイントの位置を変更することで編集できます。

ソースノーマライゼーションを有効にする

非破壊的であるソースノーマライゼーションは、アクターミキサー階層にある親レベルで有効にすることも、個別のサウンドやサウンドのグループを対象に、子レベルでオーバーライドすることもできます。ラウドネスノーマライゼーションを有効にすると、ソースレコーディングのラウドネス測定値から計算した自動ゲインが適用され、どのようなソースもノーマライズできます。



サウンドオブジェクトのSource Settingsタブで、Enable Loudness Normalizationを選択して有効にする

ラウドネスノーマライゼーションの効果を聞くには、以下のワークユニットの中のサウンドオブジェクトを開き、Source Settings画面でラウドネスノーマライゼーションを有効にします。

- Ambient
- Character
- Combat
- Magic

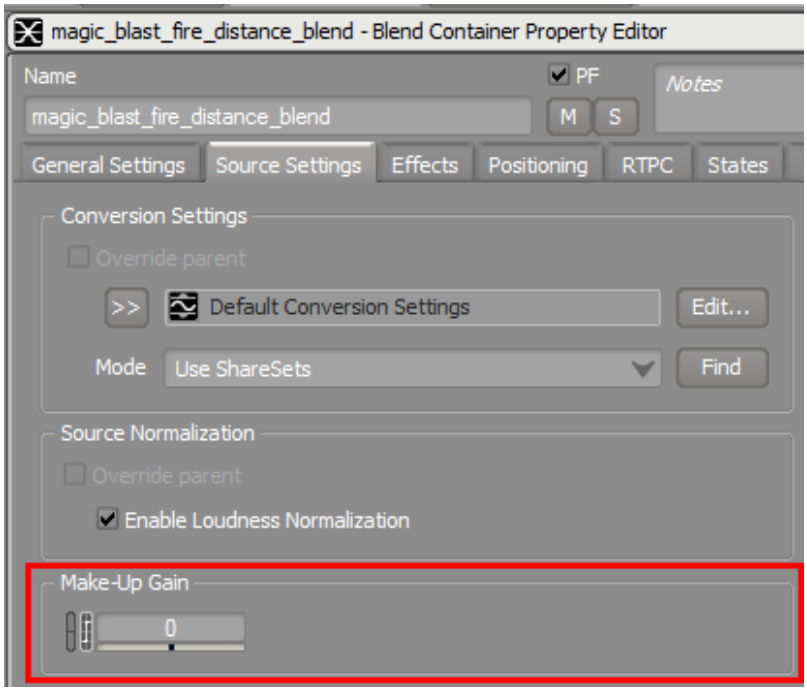
ノーマライゼーションの実施は、従来、使用する開発パイプラインやオーディオエンジンの方式によって大きな差がありました。ミキシング戦略の一貫として採用できるラウドネスノーマライゼーションは、同じボリューム（例えば-10dB）で再生される2つのサウンドが必ず、全く同じレベルで再生されたように聞こえる、安定した方式です。サウンドコンテンツの均一性を確保できるので、安心してミキシングできます。

Designer Note

ソースファイルに適用するラウドネスノーマライゼーション処理のターゲット値は、-23dBです。Wwiseが波形データを解析して、この波形のRMS値に基きラウドネス測定値を保存し、ランタイムに、このサウンドのラウドネスが-23dBに等しくなるようにゲインを適用します。例えば、解析したサウンドのラウドネスが-35dBであれば、ランタイムに+12dBのゲインを適用して「ノーマル」にします。

メイクアップゲインを使う

メイクアップゲイン（Make-Up Gain）を設定して、HDR処理の後に適用されるゲインの量を指定します。このゲインはHDR処理に影響しないので、標準的なボリュームスライダによる調整と異なります。メイクアップゲインは、HDRウィンドウとの関係において、サウンドレベルをオフセットする時に使います。また、ラウドネスノーマライゼーションの調節にも使えます。



Make-Up Gainを使い、サウンドオブジェクトのボリュームレベルをオフセットする

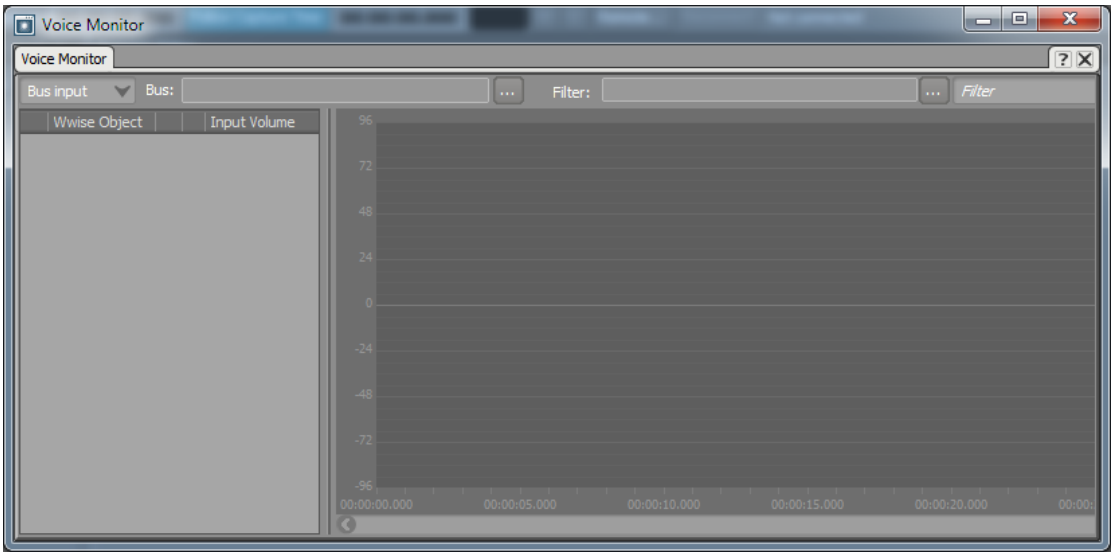
HDRの処理に影響することなく、ソースのボリュームをコントロールする機能は、ミックスを正確にコントロールできる強力なツールとなります。ボリューム値の抽象化は、制作過程でミキシングを繰り返すうちに、自然な作業となります。この追加のボリュームコントロール機能があれば、HDRシステム内でサウンドボリュームを設定したあとでも、全体の構成を壊すことなく、ミックスの最終調整ができます。

Audio Voice Monitorを使ってHDRを理解する

HDRバスにルーティングしたサウンドが、移動するウィンドウの影響を受ける様子を見ることで、イメージがわき、その過程が理解しやすくなります。Voice Monitorビューには、ウィンドウの動き、サウンドのボリューム、そして最大サウンドのエンベロープがHDRシステムに与える影響が表示されます。

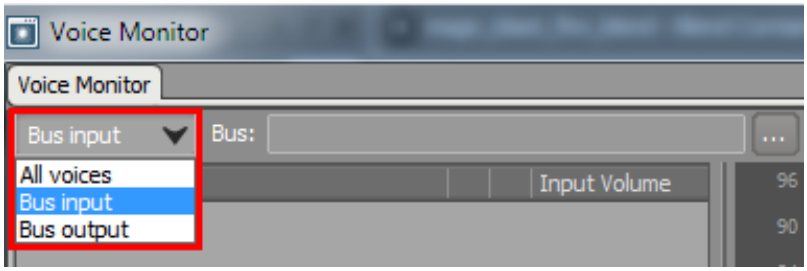
Voice Monitorビューを開く

まずメニューバーの、Viewsメニューから、Voice Monitorビューを開きます。



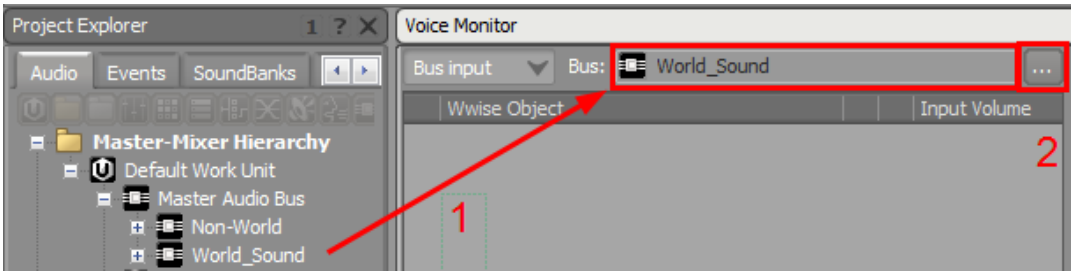
Voice Monitorビューを開く

作動中のHDRを見て様々な役割を理解するには、モニターのモードを、Bus Input、Bus Output、またはAll voicesにします。



Voice Monitorで、モニターのモードを選択する

次に、Voice MonitorにHDRバスをアサインするために、ドラッグ＆ドロップするか、Project Explorerブラウザから選択します。

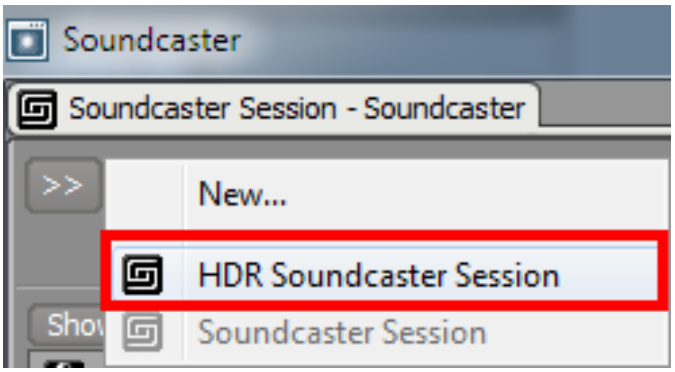


ドラッグ＆ドロップ（1）、またはProject Explorer ブラウザ（2）で、ミキシングバスをアサインする

Voice Monitorビューでこれらのプロパティをアサインできたら、オーサリングアプリケーション内でキャプチャーをしている時、または開発対象のプラットフォームにリモート接続してゲームからキャプチャーしている時に、指定したバスを経由する全てのオーディオがモニター画面に表示されます。

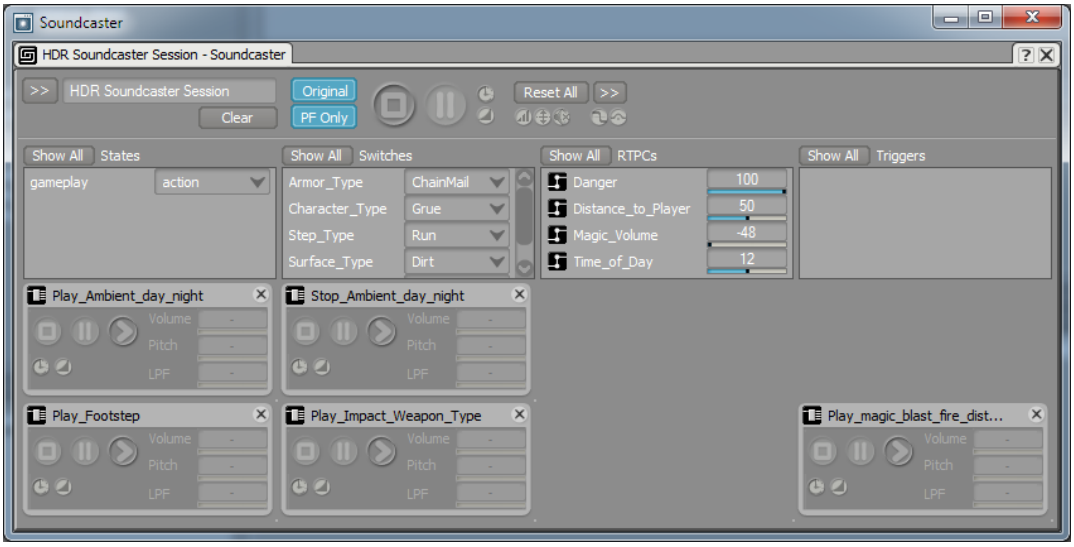
Soundcasterでサウンドを試聴する

以下の通り、Wwise Project Adventureプロジェクトで、Viewsメニューから Soundcasterビューを開き、セッションのリストからデフォルトのSoundcaster Sessionを選択します。



Soundcaster画面で、デフォルトのSoundcaster Sessionを選択する

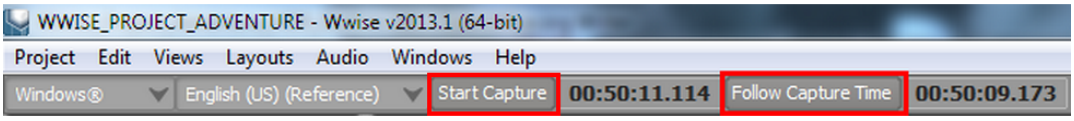
Soundcaster Sessionには、プロジェクト内のサウンドを試聴できるように、数種類のイベントを入れてあり、この他のイベントも追加して、様々なサウンドのカテゴリーを試せます。



Wwise Project Adventureに含まれている、デフォルトのSoundcaster Session

Wwiseからデータをキャプチャーする

Soundcasterを使って、HDRシステム内のサウンドを試聴する準備ができたところで、プロジェクトからくるデータを、Wwiseでキャプチャーします。オーサリングアプリケーションでデータキャプチャーを行うには、Start Captureをクリックします。Follow Capture Timeボタンを使えば、新しいキャプチャーデータが自動的に表示され、何が起きているのかをリアルタイムで見られます。

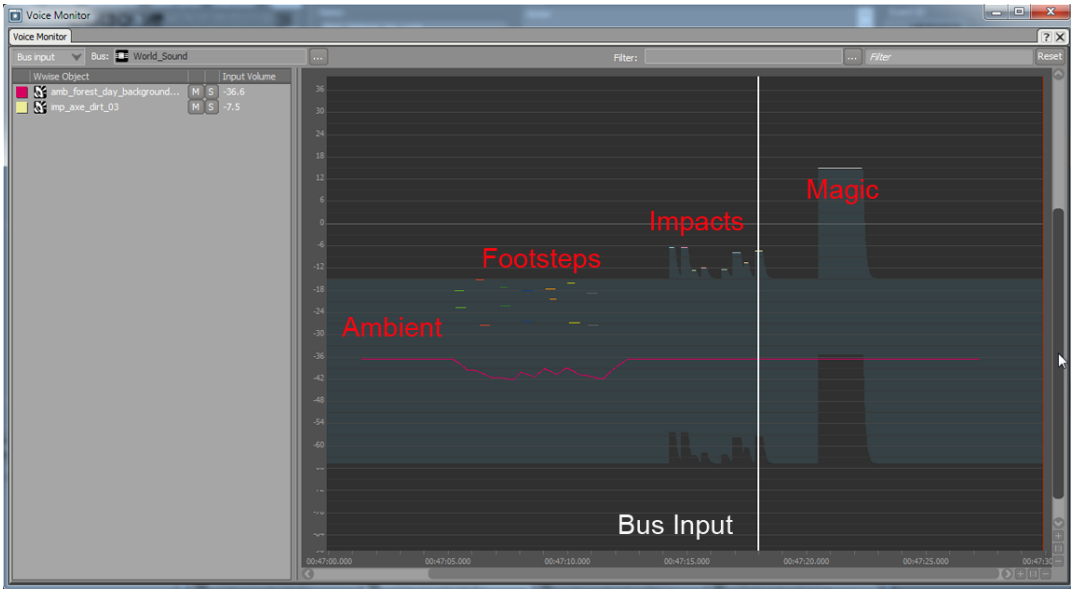


Voice Monitorで、Start Capture、Follow Capture Timeを選択

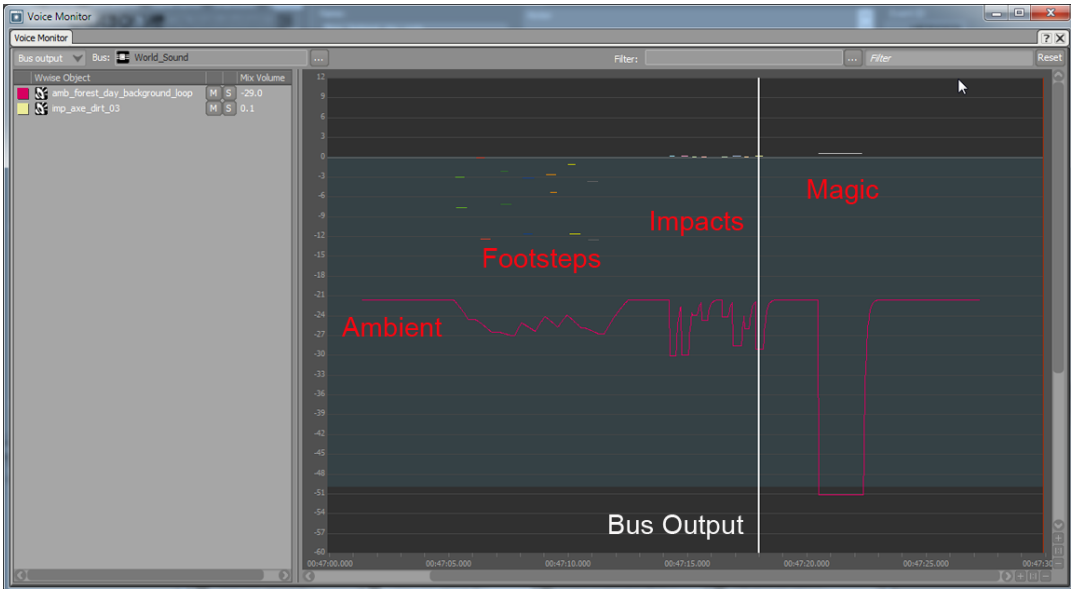
キャプチャーを開始すると、選択中のバスを経由する全てのサウンドが、現在のボリュームに基づいて表示されます。サウンドを試聴するには、TransportやSoundcasterを使うか、開発対象のプラットフォームで稼働するゲームにローカル接続またはリモート接続します。

イベント“Play_Ambient_day_night”のPlayアイコンを選択すると、試聴が開始されます。続けて他のイベントも試聴していくと、HDRウィンドウへの影響が、Voice Monitorで表示されます。マジックサウンドのエンベロープを有効にして、設定によって何が変わるかを確認すると良いでしょう。HDR スレッショルドやレシオを変更すれば、HDRシステムによってダイナミックレンジがどう調整されるのかが、さらに詳しく分かります。

一連のイベントをVoice Monitorでキャプチャーした様子を、以下に示します。



Voice Monitorを使って、バスへのインプットのキャプチャーを、モニターする



Voice Monitorを使って、バスからのアウトプットのキャプチャーを、モニターする

WwiseのHDRシステムはオーサリングから試聴に至るまで対応でき、その効果を、Voice Monitorで見ることができます。オーサリングアプリケーション上で、様々な設定値をオフラインでテストしながら、ミックスへの効果をすぐに目で確認できるので、作成したシステム内の仕組みが今までになく明瞭に分かります。HDR処理後のミックスを聞けるだけでなく、サウンド同士が影響し合う様子が見られることは、最適なミキシング判断につながり、最終的なアウトプットに与える影響を知る上で鍵となります。

HDRオーディオのまとめ

HDRは、マスターミキサー階層の1つの親バスに限定して適用するもので、ゲームのダイナミックミキシングに用いる唯一のテクニックではありません。HDRは6つの内の1つに過ぎず、Wwiseオーサリングアプリケーションには、6種類のミキシングテクニックがあり、具体的には、ボリューム設定によるミキシング、ステートに基づいた（スナップショット）ミキシング、オートダッキング、RTPC、サイドチェイン、そしてHDR（ハイダイナミックレンジ）ミキシングです。複数のミキシングシステムをバランスよく使いこなすことで、インタラクティブミキシングをクリエイティブに編成できるだけでなく、ミキシング関連の問題に対処したり、カスタマイズしたミキシングソリューションを作成したり、制作中に発生するプロジェクトのニーズに対応したりする時にも、役立ちます。

基本の説明

- WwiseのHDR（ハイダイナミックレンジ）オーディオの理論
- Wwise Project Adventureを例にした、HDR適用の基本操作
- HDRオーディオの原理を分かりやすく説明する、シナリオ事例

プロセスの説明

- HDRオーディオの、ダイナミックレンジウィンドウの設定
- マスターミキサー階層における、HDRオーディオの有効化
- HDRオーディオのダイナミックプロパティの設定
- アクターミキサー階層における、HDRオーディオの有効化
- エンベロープトラッキングの有効化
- 波形エンベロープの編集
- ソースノーマライゼーションの有効化
- メイクアップゲインの利用
- HDRオーディオを理解するための、Voice Monitor
- Wwiseでデータキャプチャーして、Voice Monitor画面でHDRオーディオを確認

その他の説明

- HDRオーディオの原理を分かりやすく説明する、シナリオ事例

オブジェクトの作成方法

- プロジェクトのボリュームスレッショルド設定
- マスターミキサー階層のオーディオバス“World_Sound”
- 以下のワークユニット内にあるサウンドオブジェクトの、ボリュームと、ラウドネスノーマライゼーション
  - Ambient
  - Character
  - Combat
  - Magic

第11章 アドベンチャーに向けたセットアップ

概要

ゲームでこれから直面する試練や苦難への対策は、プレイヤーの心身の準備から始まります。熟考された攻撃プランがあれば、展開する冒険にスムーズに挑めます。

本章では、以下の操作を説明します。

- ワークユニットの管理
- アクターミキサー階層におけるオブジェクトグループの形成
- プロジェクトの詳細設定
- ワークグループの設定
- オーディオファイルの保存場所の選定
- デフォルトのコンバージョン設定
- オブストラクションとオクルージョン
- ランゲージマネージャー
- プラットフォームセレクター
- レイアウトのカスタム設定
- モーション機能
- サウンドバンクの内容と生成方法
- コンバージョン設定
- インテグリティレポート
- ファイルパッケージャーの使い方
- DLC（ダウンロードコンテンツ）の事前準備

階層を構築するにあたり、検討すべき点がいくつかあります。

ワークユニットの管理

ネーミングルールの確立

各種アセットの明確なネーミングルール（名称設定のルール）を最初に確立すれば、作成するワークユニット、サウンドオブジェクト、オーディオファイルの内容や、場合によってはプロジェクト内でのオリエンテーションを、一目で判断することができます。大きいプロジェクトにおいては、複数あるアセットの種類を名前で区別できるだけで作業効率が、大幅に改善されることがあります。今回のサンプルプロジェクトにおいても、一貫性のある分かりやすいネーミングルールを用いる努力をしました。

ワークユニットのロジカルグルーピング

アクターミキサー階層にwavファイルをドラッグ＆ドロップするのは簡単ですが、計画的に取り込んだ方が大きなプロジェクトでは特に有利です。シンプルな階層であれば、全てのゲームオブジェクトを単一のワークユニットに入れることもできます。あるいは、アンビエンス、キャラクター、戦闘シーン、UI、ボイスなどの種類に分けるなど、サウンドの種類別にワークユニットを作成することもできます。ワークユニットの中にどの種類のゲームオブジェクトを入れてもよく、ワークユニットをフォルダやアクターミキサーに入れてさらに整理することもできます。

共有を念頭にワークユニットを作成

ワークユニットは、Wwiseのプロジェクトフォルダにある個別ファイルを表しています。このファイはテキストエディターで読め、Wwiseで設定した情報、プロパティ、他ファイルとの関係設定などが含まれます。ワークユニットをネスト化させた階層をWwiseで作り、物理的なフォルダやサブフォルダに入れて整理することも可能です。特定カテゴリー用（アクターミキサー階層など）のサブフォルダの中に複数のワークユニットが存在している場合、これらのワークユニットはプロジェクトと一緒にロードされます。その結果、ソースコントロールの対象となるファイルがより細かく分別され、マルチユーザー環境での開発を効率化できます。

Designer Note

人数の多いオーディオチームを編成した場合、担当するワークユニットの作業を各自が一定期間、他人に邪魔されることなく続けられるように、それなりのワークユニット数を用意する必要があります。Wwiseが提供するソースコントロールソリューションを使用する場合は、プログラマーが確立したファイル承認システムと併行して使えるような階層の構成を設計することが重要です。

アクターミキサー階層でオブジェクトのグループ化

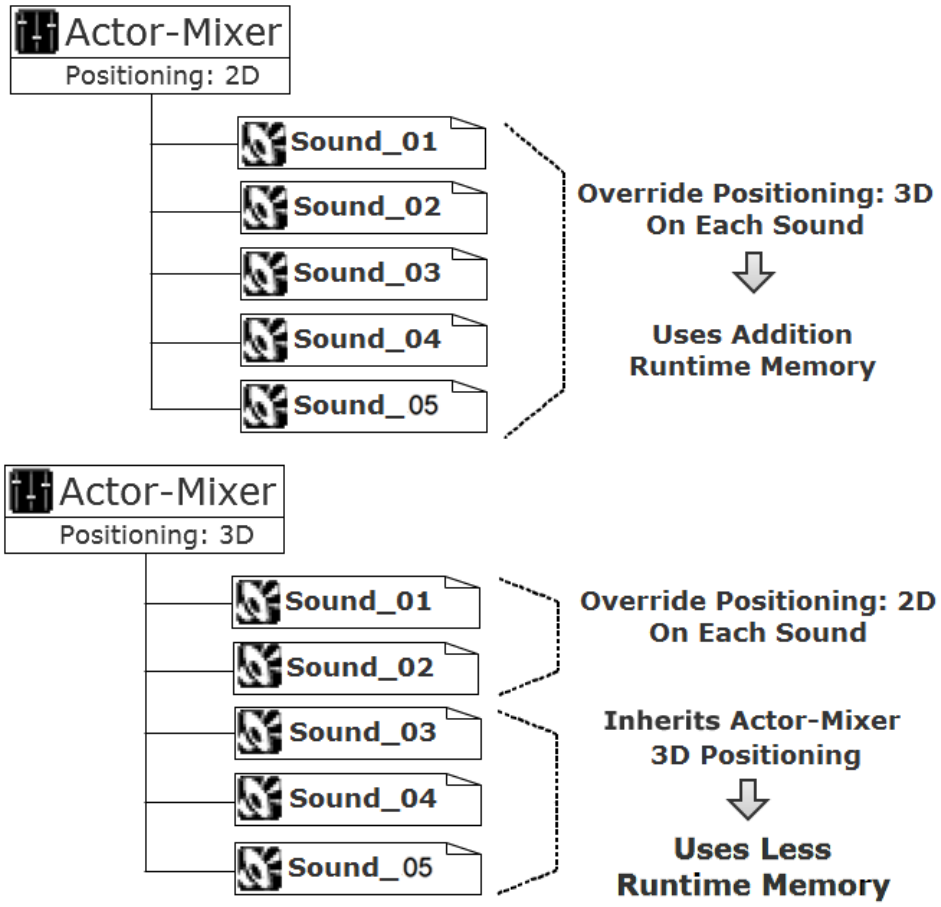
アクターミキサー階層で設定したポジショニングやRTPCなどのプロパティは、全ての子オブジェクトに承継されるので、メモリやCPUを節約する究極の手段ともいえます。多数のサウンドオブジェクトを整理する時は、アクターミキサー内のグループ分けについて、以下の点を検討して下さい。

- プロパティ設定の処理が一回で済むように、共通設定をシェアさせる
- オーバーライド設定によってオブジェクトごとに処理する状況をできるだけ回避するために、オーバーライド数を制限する

メモリーを効率的に使用するには、状況に応じてアクターミキサー内でオブジェクトをグループにまとめて、下記のプロパティをシェアさせます。

- ポジショニング
- RTPC
- ステート
- ランダマイザー

例えば、サウンドが10個入ったアクターミキサーにおいて、サウンドポジションの設定を3Dにする場合を考えます。サウンドを1つ1つ開いてOverride parent（親の設定を無視）のオプションを選択して、3D設定とすることができます。しかしこれでは、アクターミキサー自体を3Dポジショニングに設定した時と比べてランタイムのメモリ使用量が10倍になります。また、アクターミキサーに入ったサウンドのいくつかを2Dに設定する場合も、アクターミキサー自体をゲーム定義による3Dポジショニングに設定した上で、その内のいくつかのサウンドを2Dポジションにした方が、メモリ使用量が最適化されます。後から設定した2Dサウンドによるメモリ負担増はないので、2Dポジションのサウンドだけをオーバーライド設定とします。



階層プロパティの承継によって、メモリの節約が可能

共通するプロパティをアクターミキサー全体に対して設定するのが一般的には最良ですが、状況によってはコンテナに対して共通プロパティを設定した方がメモリ消費を抑えられます。例えば、ランダムコンテナに入った足音のように特定のコンテナのオブジェクトだけにポジショニングを設定したい状況では、ポジショニングのプロパティ設定をコンテナに対して設定した方が、親のアクターミキサーに対して設定するよりもメモリの節約になります。ただしアクターミキサーの構成に入った全オブジェクトにポジショニングのプロパティを共有させたいのであれば、アクターミキサーに対して設定します。

オーディオチャンネルの詳細設定

デフォルトのチャンネル設定

- メニューバーより、**Audio > System Default Channel Configuration**
- Wwiseはデフォルトで、Windowsのコントロールパネルのスピーカー設定を使用します。Windowsのコントロールパネルで選択した値を選ぶには、本オプションを選択します。

ステレオチャンネル設定（スピーカー）

- メニューバーより、 **Audio > Stereo Channel Configuration (Speakers)**
- パンのルール（スピーカーやヘッドフォン）については、次の「スピーカー対ヘッドフォンのパン」で説明します。

ステレオチャンネル設定（ヘッドフォン）

- メニューバーより、 **Audio > Stereo Channel Configuration (Headphones)**
- パンのルール（スピーカーやヘッドフォン）については、次の「スピーカー対ヘッドフォンのパン」で説明します。

5.1チャンネル設定

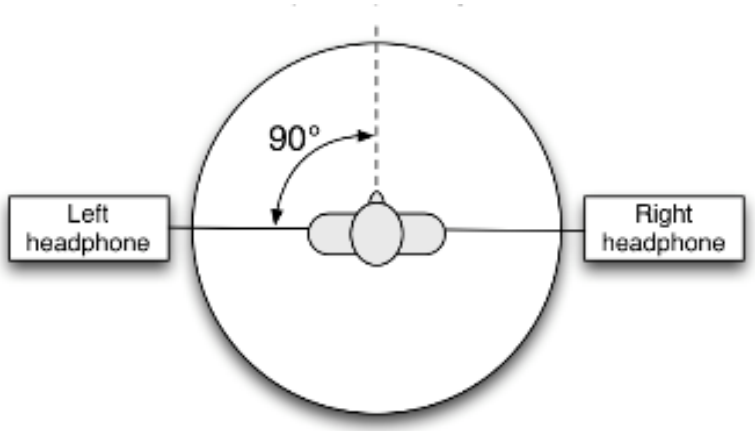
- メニューバーより、 **Audio> 5.1 Channel Configuration**

Designer Note

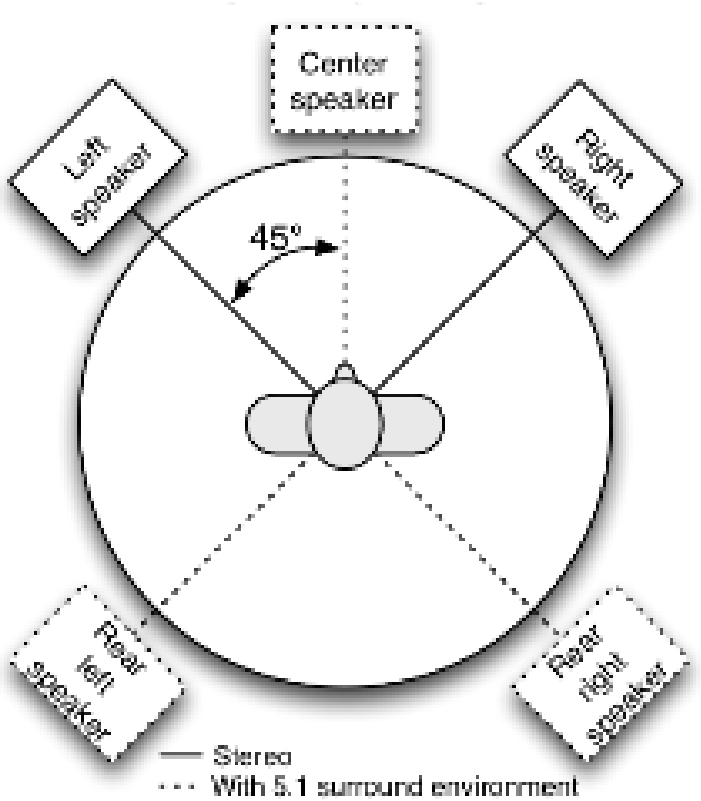
Windowsコントロールパネルでステレオ設定にした状態でも、5.1 Channel Configurationを選択することができます。ただしダイレクトサウンドが強制的に5.1chからステレオにダウンミックスされることもあります。

スピーカー対ヘッドフォンのパン

Wwiseにはヘッドフォン用とスピーカー用の2種類のパンルールがあります。デフォルトとして全プラットフォームでスピーカー用パンルールを使用しますが、例外としてハンドヘルドのゲーム機ではデフォルトがヘッドフォン用パンルールとなります。2つのモードは微妙に違い、聞く環境に応じてリアルで正確なオーディオを提供するために設定されています。パンルールの設定はWwise上で試聴できる他、ランタイムにゲーム内で設定することも可能です。



ヘッドフォン用パンルール



ラウドスピーカー用パンルール

2つのモードを視聴するには

- メニューバーより、 **Audio > Stereo Channel Configuration (Speakers)**
- メニューバーより、 **Audio > Stereo Channel Configuration (Headphones)**

Programmer Note

ゲーム内でパンルールを設定することも可能で、詳細はサウンドエンジンのドキュメンテーションにある“AK::SoundEngine::SetPanningRule”をご参照ください。

Soundcasterを使ったシミュレーション

開発の過程で、作成中のオブジェクトやイベントを使ってWwiseでシミュレーションを行うこともできます。Wwiseのシミュレーション環境Soundcasterを使ってサウンド構成、音楽構成、モーション構成を非同期で再生できます。つまり何をいつ再生するのかをコントロールできます。ですからイベントのテストやリアルタイムのミキシングなどに非常に便利です。強力なツールであるSoundcasterは、以下の場面で活用できます。

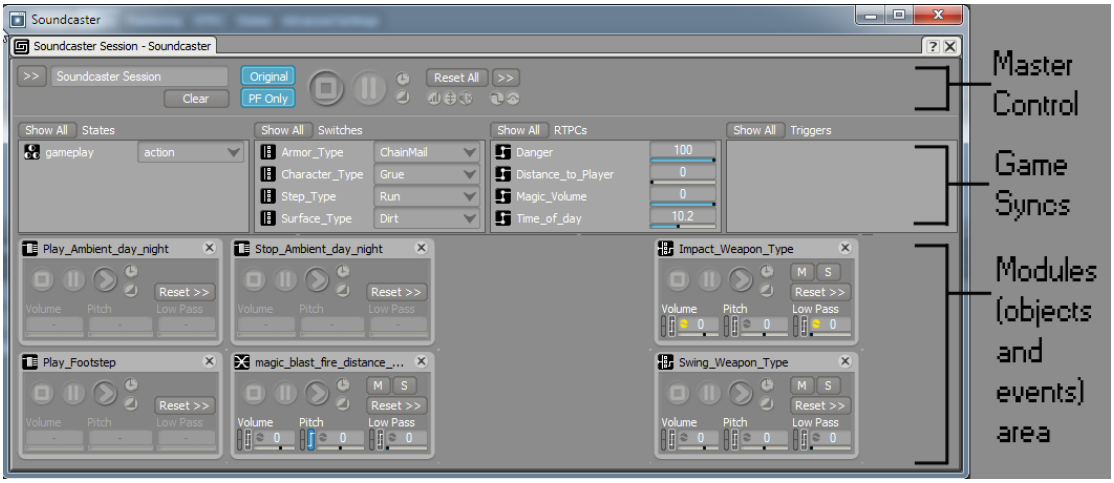
- プロトタイプ制作や実験
- プルーフオブコンセプト（概念実証）の開発
- サウンド、音楽、モーションの同時試聴

Soundcasterを使ったシミュレーションはWwise単体でも、ゲームにリモート接続した状態でも可能なので、様々な使い方ができます。また、シミュレーション中に以下を実行することができます。

- 特定プラットフォーム向けのオーディオやモーションの視聴
- コンバージョン後のオーディオファイルの試聴
- オーディオやモーションを再生しながら、プロファイルの確認
- マニュアル操作でゲーム中のアクションをシミュレートしながら、オーディオやモーションのWwise上でのミキシングやテスト
- ゲーム中やWwise上で、オーディオやモーションのプロファイルの確認
- 1つのゲームオブジェクトに関連するサウンドオブジェクト、ミュージックオブジェクト、モーションオブジェクトを使った実験
- ゲーム中のサウンド、ミュージック、モーションを確認しながらの、ミキシングやテスト

Soundcaster画面は以下の3つに分かれています。

- マスターコントロール
- ゲームシンク
- オブジェクトやイベント



Soundcaster画面のシミュレーション用の主要3機能

Soundcasterの3つの部分を使い、シミュレーション中にミキシングや再生動作を操作できます。

Soundcasterの詳細情報

Wwise Help > Finishing Your Project > **Creating Simulations**

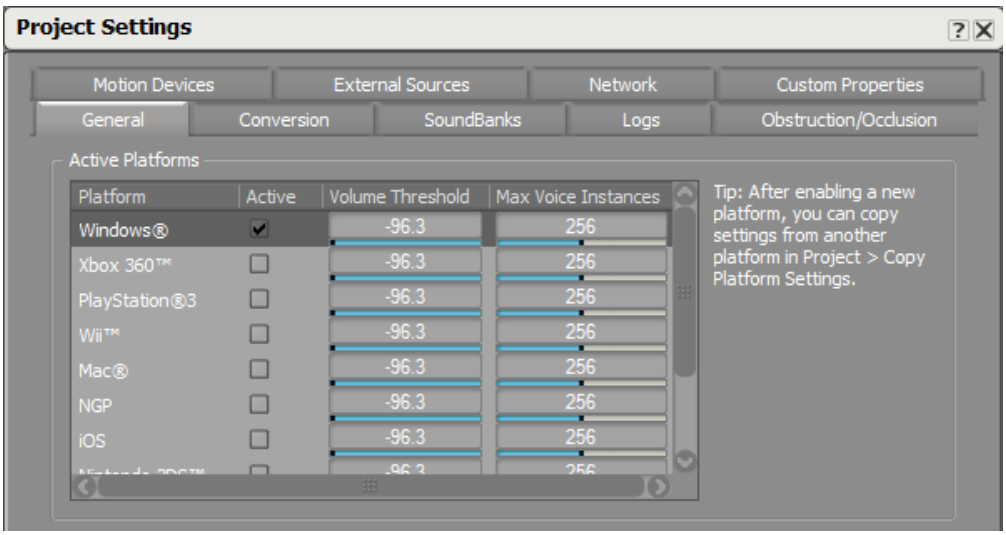
Video Tutorial - Wwise Quick Tip - Transport and Soundcaster

プロジェクトセッティング

プロジェクトを通してサウンドに適用させるデフォルト動作は、Project Settings画面で設定できます。

プロジェクトセッティング - Generalタブ

Generalタブを開いて開発対象となるプラットフォームをアクティブにしてから、各プラットフォームのVolume Threshold（ボリューム閾値）とMax Voice Instances（最大ボイスインスタンス数）を設定します。



Activeなプラットフォームの選択と、各種設定の管理

Volume Threshold（ボリューム閾値）：

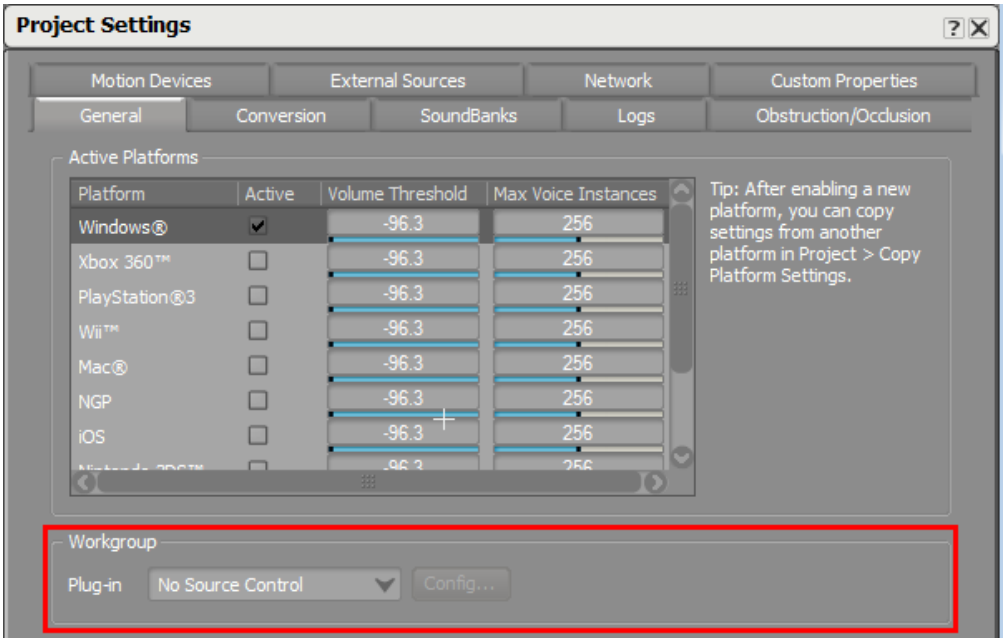
ボリューム閾値として設定したデフォルトボリューム値以下のボイスはProperty Editor画面のAdvanced Settingsタブで定義した動作をとり、具体的には、ボイスはキルされる（停止される）か、バーチャルボイスとなります。

Max Voice Instances（最大ボイスインスタンス数）：

プロジェクト全体で同時にアクティブ状態となれる最大の同時ボイス数を示します。なお、バーチャルボイスはアクティブボイスとみなされません。この最大数を超えると、プライオリティが最も低いボイスに対し、そのボイスのバーチャル動作が適用されます。複数のサウンドのプライオリティの順位が等しければ、古いサウンドが優先されます。

ワークグループのプラグイン構成

プラグインWorkgroupを使えば、ソースコントロールのソリューションを指定して詳細を設定できます。なお、プロジェクト全体のアセットやその他のプロジェクトファイルを効率的に管理するために、開発環境で既にPerforceやSubversionなどのソースコントロールシステムが採用されているかもしれません。



Wwiseでプロジェクトファイルを管理するための、ソースコントロール用プラグインの指定

ソースコントロールシステムを使って、プロジェクト内の以下のファイルを管理できます。

- **Wwiseプロジェクトファイル** - 拡張子「.wproj」
- **Wwiseワークユニット** - デフォルトのワークユニットを含む、拡張子「.wwu」
- **[Originals]フォルダ** - Wwiseにインポートしたオリジナルのサウンドファイルが保存されたフォルダ
- **生成されたサウンドバンク** - 各プラットフォーム用に言語ごとに生成されたサウンドバンクファイル

Designer Note

プロジェクトディレクトリーにある[.cache]フォルダは、Wwise用のローカルのワーキングフォルダです。[.cache]フォルダの中身をソースコントロールシステムに追加するとWwise側で予期せぬ動作を引き起こすことがあるので、避けて下さい。

ゲーム開発中にいつでも、プロジェクトファイル（.wproj）、ワークユニットのファイル（.wwu）、オーディオファイルのそれぞれのステータスをFile Manager画面で確認できます。PerforceやSubversionなどのワークグループ用プラグインを使用していれば、Wwiseから直接ソースコントロール機能を使えます。ワークユニットを含めたWwiseのプロジェクトファイルは全てXMLベースなので、ソースコントロールシステムでこれらのファイルも簡単に管理できます。

Designer Note

Wwiseにおけるワークグループ用プラグインの使い方は、ヘルプの「Managing Project Files Using a Workgroup Plug-in」をご参照下さい。

チームの一員としてソースコントロールシステムを使ってプロジェクトのファイルを管理する場合、同じプロジェクトで複数の人が作業していること、そしてファイルのマージでコンフリクトが発生する可能性があることを常に念頭に置いて下さい。作業中は適宜シンクやマージを行い、頻繁に他のチームメンバーと進捗状況を確認し合う必要があります。

Designer Note

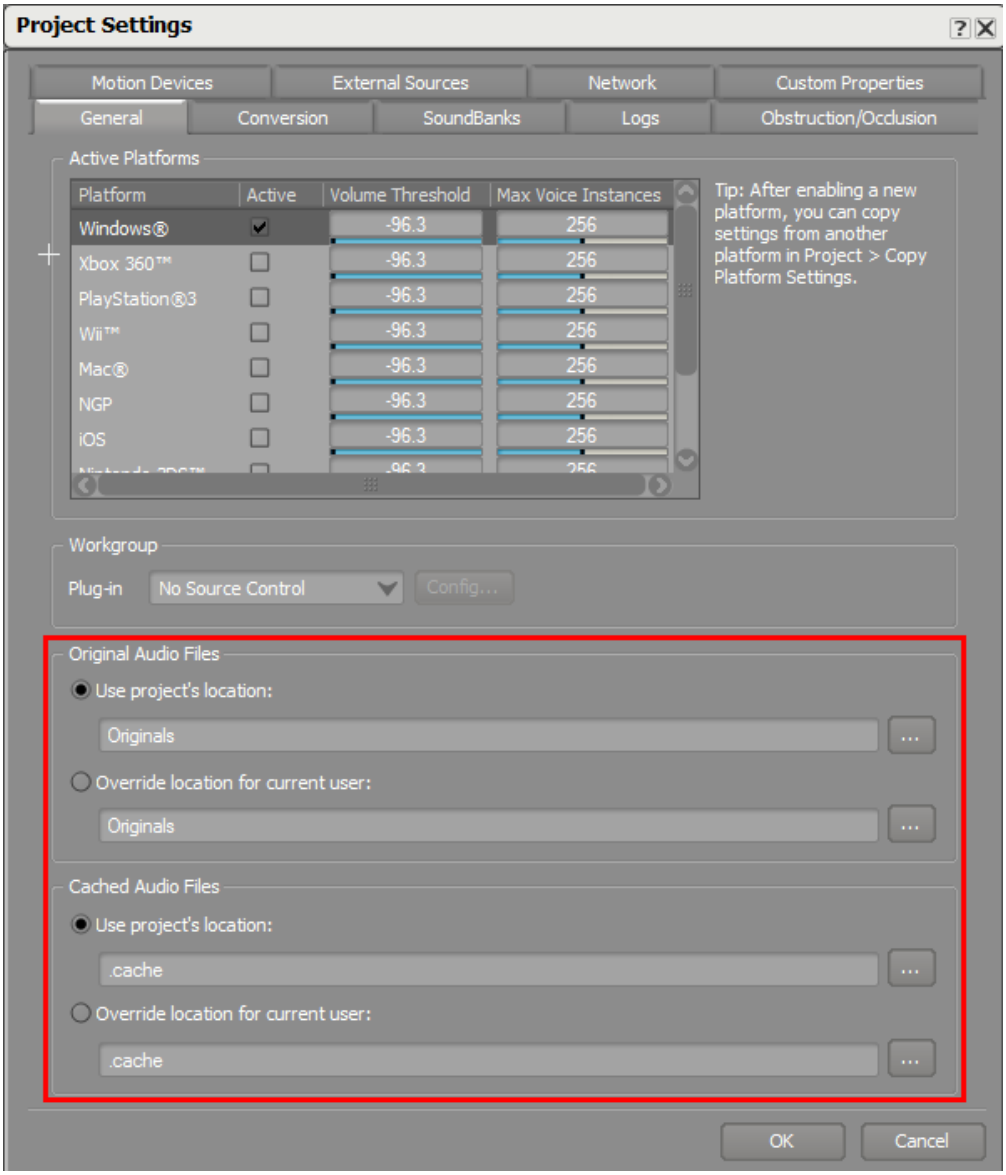
ベストプラクティスの一覧がヘルプの「Workgroup Tips & Best Practices」に掲載されていますので、ご参照下さい。

ワークグループの詳細情報

Video Tutorial - Workgroup management in Wwise using Perforce

オーディオファイルの保存場所

オーディオファイルのオリジナルファイルとキャッシュファイルを保存する場所を、プロジェクト用にカスタム設定でき、また、Override location for current user（設定を無視した、現在のユーザー用の保存場所）を設定することもできます。

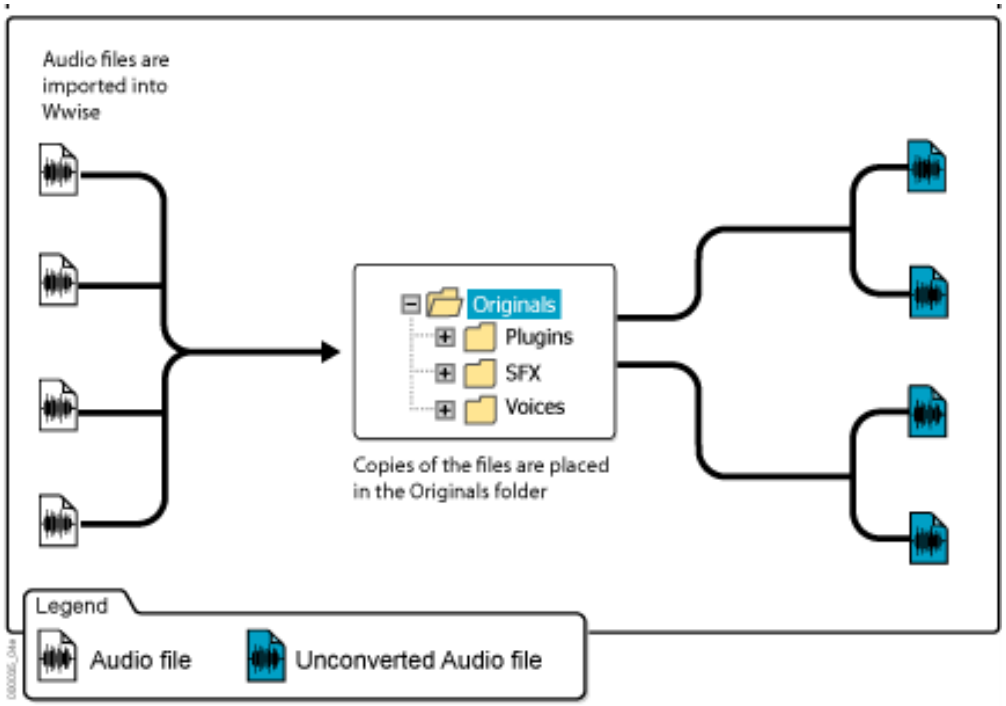


プロジェクトで使うオーディオファイルの、オリジナルの保存場所の変更

以下のような場合は、設定を無視したユーザー用の保存場所を設定すると便利です。

- [Originals]または[Cached]フォルダへ、一時的にアクセスできない
- [Originals]または[Cached]フォルダの中身を編集する権限がない
- プロジェクトの[Originals]または[Cached]フォルダの場所を変えずに、一時的に[Originals]または[Cached]フォルダの場所を作成する必要がある

プロジェクトにファイルをインポートすると、元のアセットのコピーがプロジェクトの[Originals]フォルダに保存されます。このアセットは複数のチームメンバーによって使われることが多いため、フォルダをネットワーク上のどこに置くかは自由に決められ、ソースコントロールシステムで簡単に管理できます。



異なるプラットフォーム用に作成されたアセットの各バージョンは、ユーザーのローカルのプロジェクト用[Cache]フォルダに保存されます。これによってユーザーが各自のプラットフォーム用バージョンを自分で管理して様々なコンバージョン設定を試すことができます。

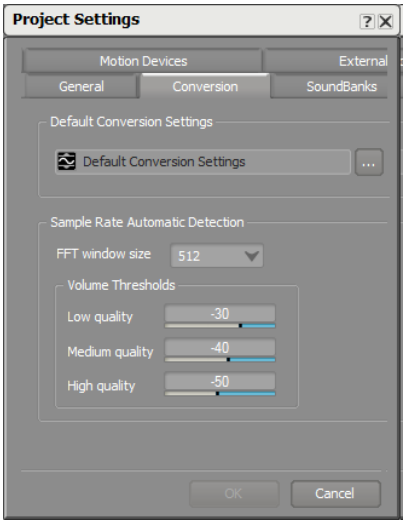
プロジェクトの[Cache]フォルダには、オーディオファイルのコンバージョンやサウンドバンク生成の過程でWwiseが生成した中間データが保存されます。新規プロジェクトが作成されると、プロジェクトのダイレクトリー内に[.cache/]として初期の場所が設定されます。

どちらのタイプのファイルも、Wwiseで再生できます。コンバージョン前の元ファイルは、Transport ControlやSoundcasterでOriginalコントロールを有効にすると再生されます。Originalコントロールが有効でない場合、コンバージョン後のファイルが存在すればWwiseはその再生を試みます。

Designer Note

Wwiseでコンバージョン後のファイルを再生する際、多少の制限があります。オーディオファイルをあるプラットフォーム向けに変換すると、そのプラットフォームの具体的なハードウェア要件に対応した変換が行われます。その結果、Windows以外の他のプラットフォームを選択している時は、コンバージョン後のファイルを再生できない場合があります。

デフォルトのコンバージョン設定



デフォルトのコンバージョン設定

Project Settings画面のDefault Conversion 設定で、プロジェクトでデフォルトで使用するシェアセットの名前を決定します。新規オブジェクトが作成されると、それが最高位の親オブジェクトである場合に限り、デフォルトのコンバージョン設定が適用されます。新規オブジェクトが他のオブジェクトの子であれば、親に設定されたコンバージョン設定を継承します。

サンプルレート自動検知設定の定義

Project Settings画面で、FFTアルゴリズムで使用するハニング窓のサイズや、3つの品質基準（High、Medium、Low）のそれぞれの閾値を決定できます。この閾値は、サンプルレートのコンバージョン方式としてAuto High、Auto Medium、またはAuto Lowを選択したときに適用されます。

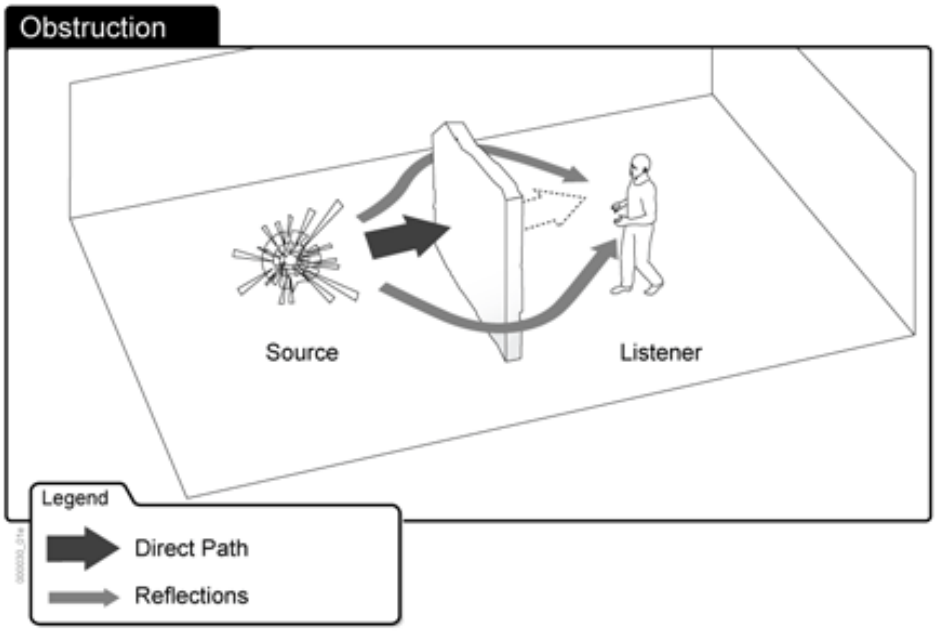
コンバージョン設定の詳細情報

Wwise Help > Setting Up Your Projects > Working with Projects > Defining your Project Settings > **Defining the Conversion Settings for Your Project**

オブストラクションとオクルージョン

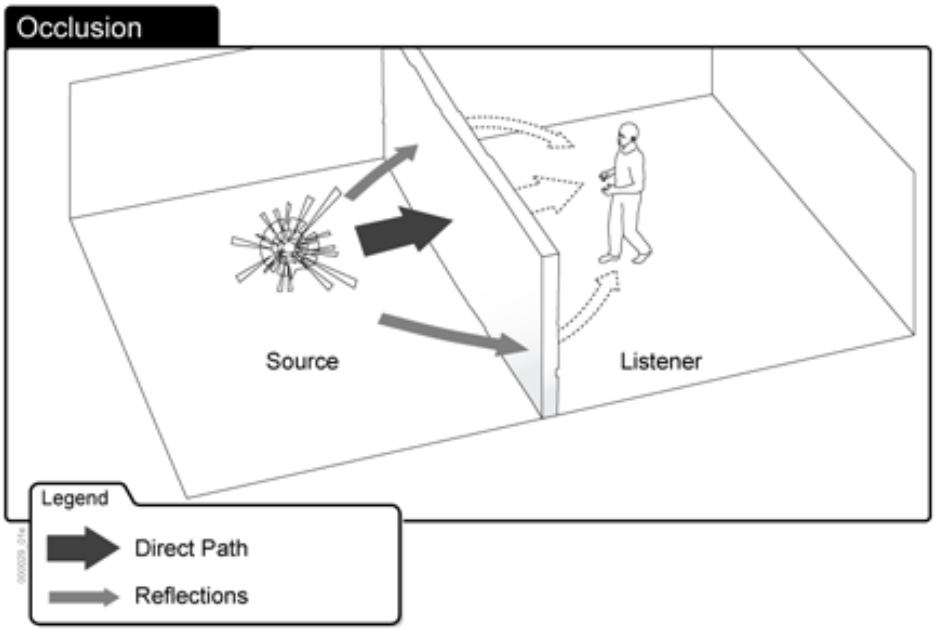
ゲームの典型的な状況として、ゲームオブジェクトが他のオブジェクト（壁や梁など）によってオブストラクション（妨害）を受けたり、隣の部屋の音が壁越しに漏れてくるようなオクルージョンが起きる環境がよくあります。

オクルージョンやオブストラクションの例を下図に示します。



オブストラクションの例

オブストラクションのモデルでは、信号のDirect Pathだけにボリュームコントロールやローパスフィルターを適用します。環境によるReflectionsには適用しません。



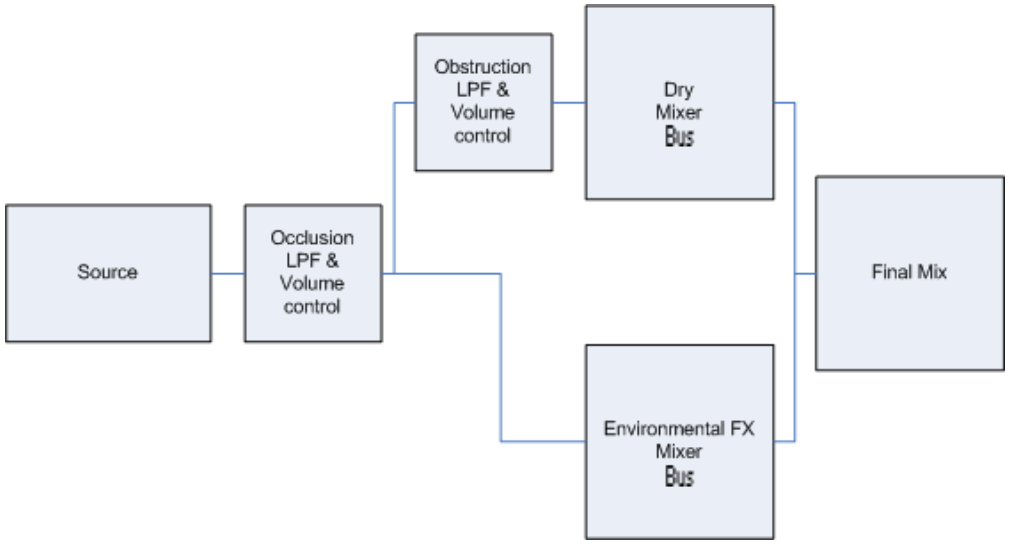
オクルージョンの例

オクルージョンのモデルでは、信号のDirect Pathと環境によるReflectionsの両方にボリュームコントロールやローパスフィルターを適用します。

Designer Note

オブストラクションとオクルージョンを同時に発生させるには、ダイレクトパスに対して、オブストラクションとオクルージョンの両方の値を適用します。ただし、リフレクションパスはオクルージョン値の影響のみを受けます。

オブストラクションやオクルージョンを処理するサウンドエンジンのパイプライン内の流れを、下図に示します。



オクルージョンを処理するパイプライン

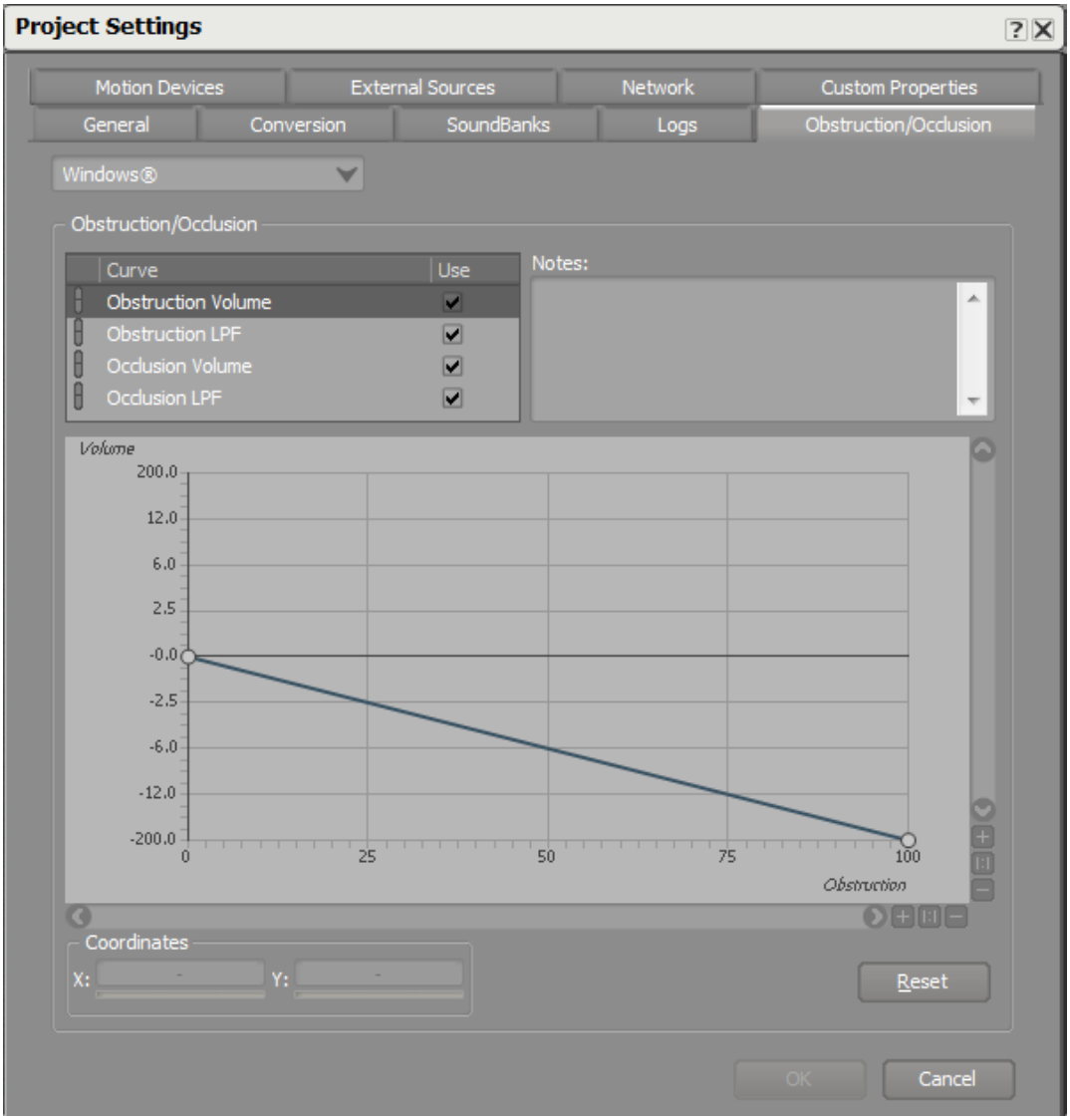
オブストラクションとオクルージョンの設定

ゲームエンジンは、オブジェクトやリスナーのゲームにおける物理的な位置情報を使い、オブストラクション値やオクルージョン値を決定します。

Programmer Note

リスナーに影響するゲームオブジェクトのオブストラクション値やオクルージョン値を、ゲームがサウンドエンジンに対してプログラム的に送出する必要があります。

Project Settings画面のObstruction/Occlusionタブを開くと、プロジェクトでアクティブなプラットフォームごとに、オブストラクションやオクルージョンのデフォルトボリュームやローパスフィルターのCurve（グラフ中のカーブ）を、それぞれ有効化して設定できます。カーブのUse（使用する）設定は、ゲームのパフォーマンス要件や描写レベルに応じて、サウンドデザイナーが自由に有効化または無効化できます。



Project Settings画面でObstructionやOcclusionのカーブをカスタム設定

上図のスナップショットで設定したグラフのカーブを使うと、オブストラクション値を1.0f（100%）とした時、ソースオブジェクトで-50dBのボリューム変化が発生します。

Designer Note

グラフのカーブを直線の組み合わせにすると、全体のCPU負荷やメモリ使用量を抑えることができます。カーブはできるだけシンプルな形にして、やむを得ない場合のみカスタム化します。

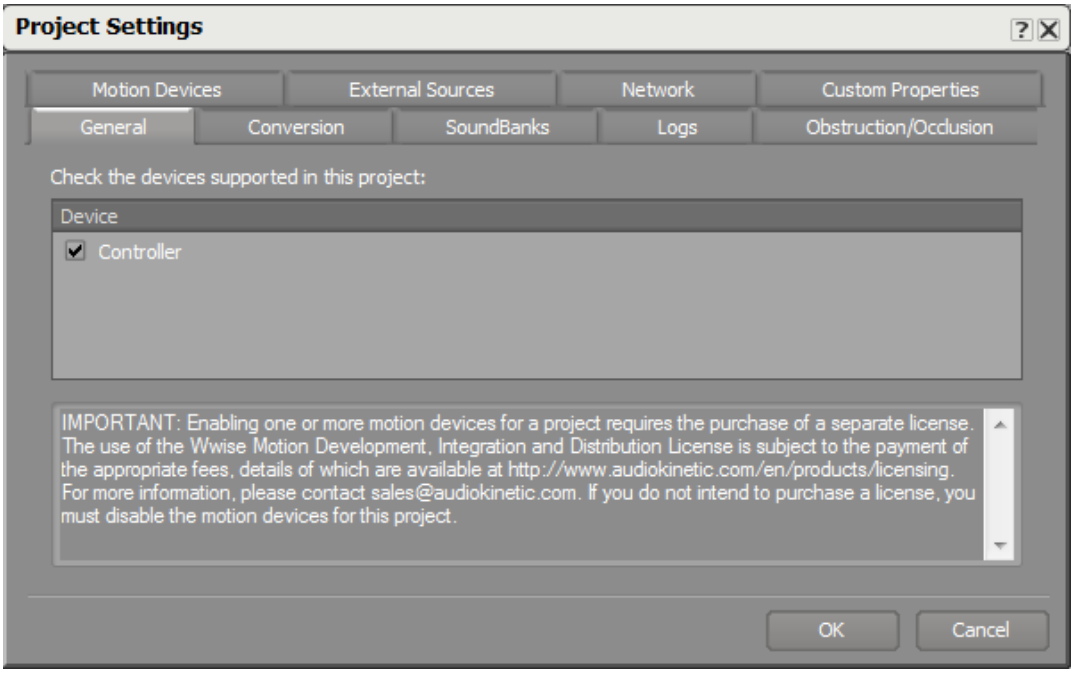
モーション

ドラゴンが地下の巣から現れる場面では、まず地面を揺るがすとどろきが響き渡り、そして最後の対決に向けて突進してくる大音量の足音が続きます。この荒々しい状況に見合うサウンドトラックを提供する時、ゲーム機のバイブレーションを起動するのにサウンドを利用する方法があります。

Wwiseは、モーション効果の制作とゲームへの実装のための完成されたパイプラインを提供します。オーディオ用のソリューションと似たWwiseの総合的なモーション用パイプラインを活用することで、以下が実現します。

- 短期間で習得可能な機能で、洗練されたリアルなモーションエフェクト（バイブレーションエフェクト）を作成
- ゲームやサウンドエンジンのパフォーマンスに大きな影響を与えることなく簡単にモーション機能をゲームに実装
- オーディオと同様の機能を使った、モーションの作成と実装
- 異なるプラットフォームの類似デバイス用に、追加作業なしでモーションエフェクトを作成
- ゲーム要件に従ってモーション部分を簡単に追加または削除

プロジェクトでモーションエフェクトの作成を始める前に、Project Settings画面で対応するモーションデバイスを有効化します。



Project Settings画面でMotion Deviceを有効化

モーションデバイスを有効にすると、以下が可能となります。

- メディアファイルやモーションジェネレーターを使った、モーションエフェクトオブジェクト用のソースの作成

- 選択されたデバイス用のモーションデータの、サウンドバンクへの投入

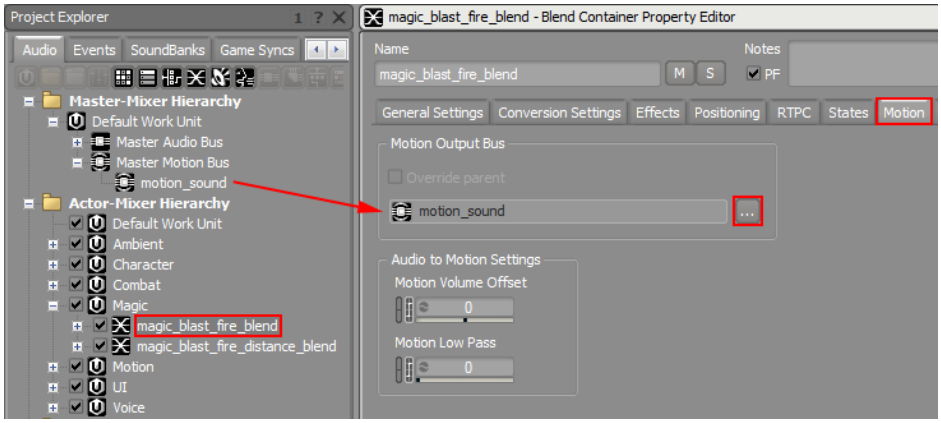
最も基本的なレベルでは、モーションデータをWwise内でソースから生成します。ソースとは、既存のメディアファイル、新規メディアファイル、またはモーションジェネレーターのことです。対応するモーションデバイスを決定した時点で、モーションエフェクトの生成方法を選ぶ必要があります。

Wwiseでモーションソースを作成するには、以下の2種類の方法があります。

- 既存のオーディオ信号を使用
- Motion FXオブジェクトを使用

既存オーディオシグナルからモーションソースを生成

既にあるオーディオ信号をモーションソースに変換する場合、ランタイムにRTPCやエフェクトなどを適用してからオーディオ信号を2つに分割します。この時、元のサウンドに影響がないように分割します。また、オーディオの方がモーションよりもはるかに広いスペクトルを有するので、高周波はローパスフィルターを使ってフィルターにかけます。次にサンプルレートを大幅に下げて再度サンプルを取り、モーションソースを作成します。



既存のサウンドオブジェクトからモーションバスへのルーティングを設定する

Designer Note

既存のサウンドオブジェクトからモーションを生成する場合、LFEチャンネルは無視されます。

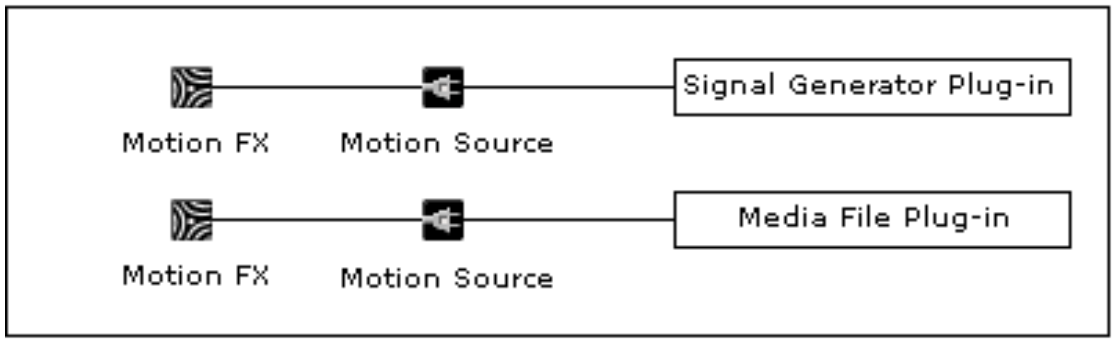
モーションのソースは既存オーディオソースから生成されるため、モーションはゲーム中のオーディオ再生と結びついています。つまり、ゲーム中にモーションソースをトリガーするための別のイベントを設定する必要はありません。また、このモーションソースはオーディオオブジェクトと同じプロパティ、動作、ゲームシンクなどの影響を受けることになります。

Designer Note

Wiiプラットフォームでは、既存オーディオソースを使ったモーションソースの生成ができません。Wiiでモーションを生成するには、Motion FXオブジェクトやプラグインのMotion Generatorを使う必要があります。

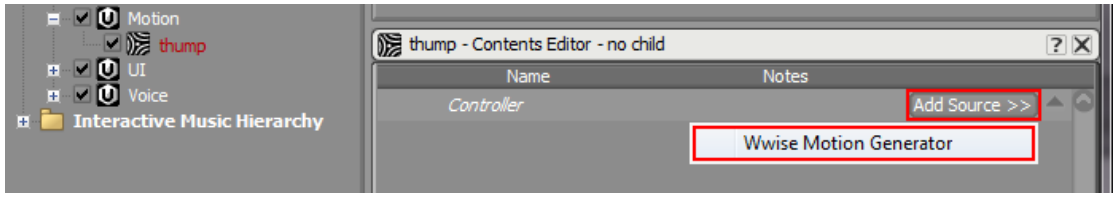
モーションFXオブジェクトを使ったモーション生成

モーションを生成するには、Motion FX（モーションエフェクト）オブジェクトという特別なWwiseオブジェクトを作成する方法もあります。このモーションエフェクトオブジェクトの中に、サウンドオブジェクトと同様にソースが入っています。プラグインのMotion Generatorを使ってモーションソースを作成します。



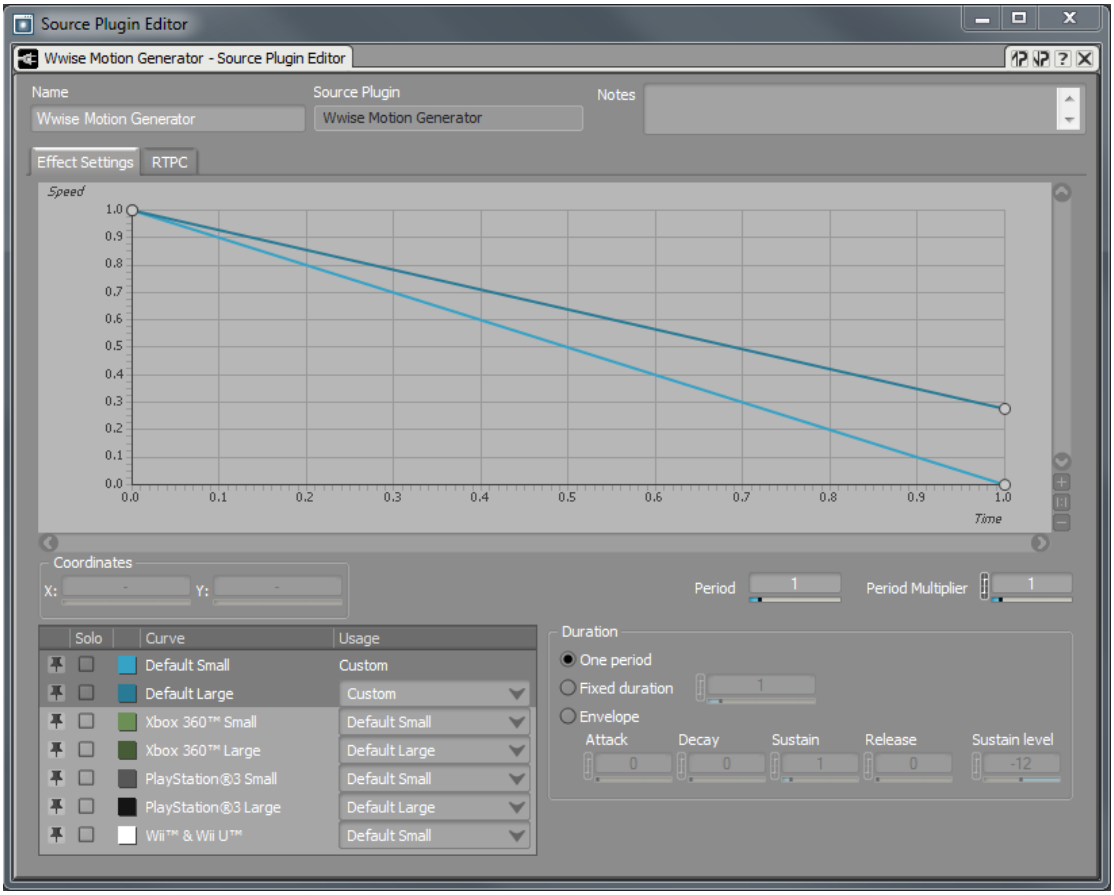
モーションエフェクトオブジェクトを使って、高度なモーション構造を作ることも可能で、モーションエフェクトのプロパティや動作を定義するために、コンテナやアクターミキサーを使います。この場合、モーションエフェクトは必ずしもゲーム内でオーディオに結びついていないので、単独のイベントを使いゲーム中にいつでもトリガーできます。

Project Settings画面でモーションエフェクトの使用を有効にした後は、Motion Generatorソースを使ってモーションエフェクトオブジェクトを追加することができます。



モーションエフェクトオブジェクトにMotion Generatorソースを追加

プラグインのMotion Generatorソースがあると、エンベロープ設定以外にも、時間やスピードに基づいたモーションエフェクトのオーサリング機能が使えます。モーションエフェクトの大小のカーブを定義するためのデフォルト設定を作成すれば、全てのデュアルモーター型コントローラーに継承させることができます。さらにコントローラー別にデフォルト設定をオーバーライドさせてモーションの効果を微調整できます。



Motion Generator機能のSource Plugin Editor画面

当然、一方の方式が他方よりも有利な状況もあります。両者のそれぞれのメリットについては、ヘルプの“Creating Motion for Your Game”をご参照下さい。

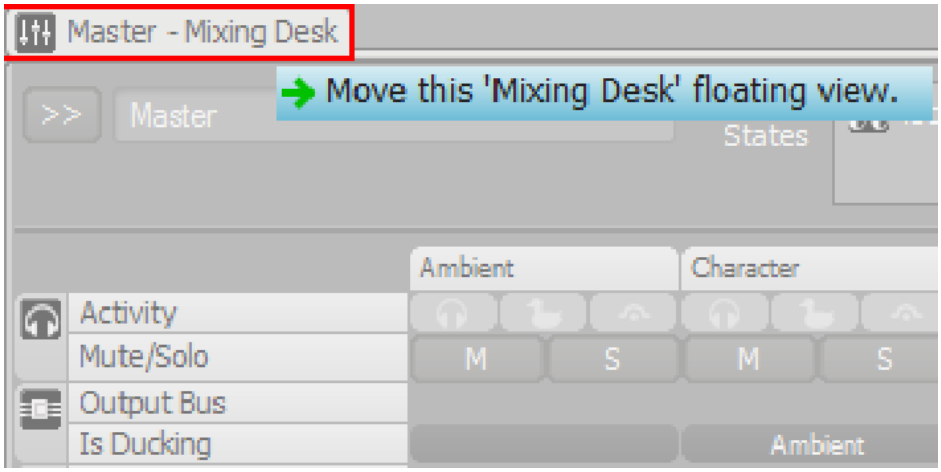
モーションの詳細情報

Video Tutorial - Wwise Motion

レイアウトのカスタマイズ

Wwiseのワークフローで発生する様々な作業に対応するために、メニューバーから最適なレイアウトを選ぶことができます。またユーザーのワークフローに合わせて、レイアウト中に配置されるビューを自由に追加したり削除することもできます。

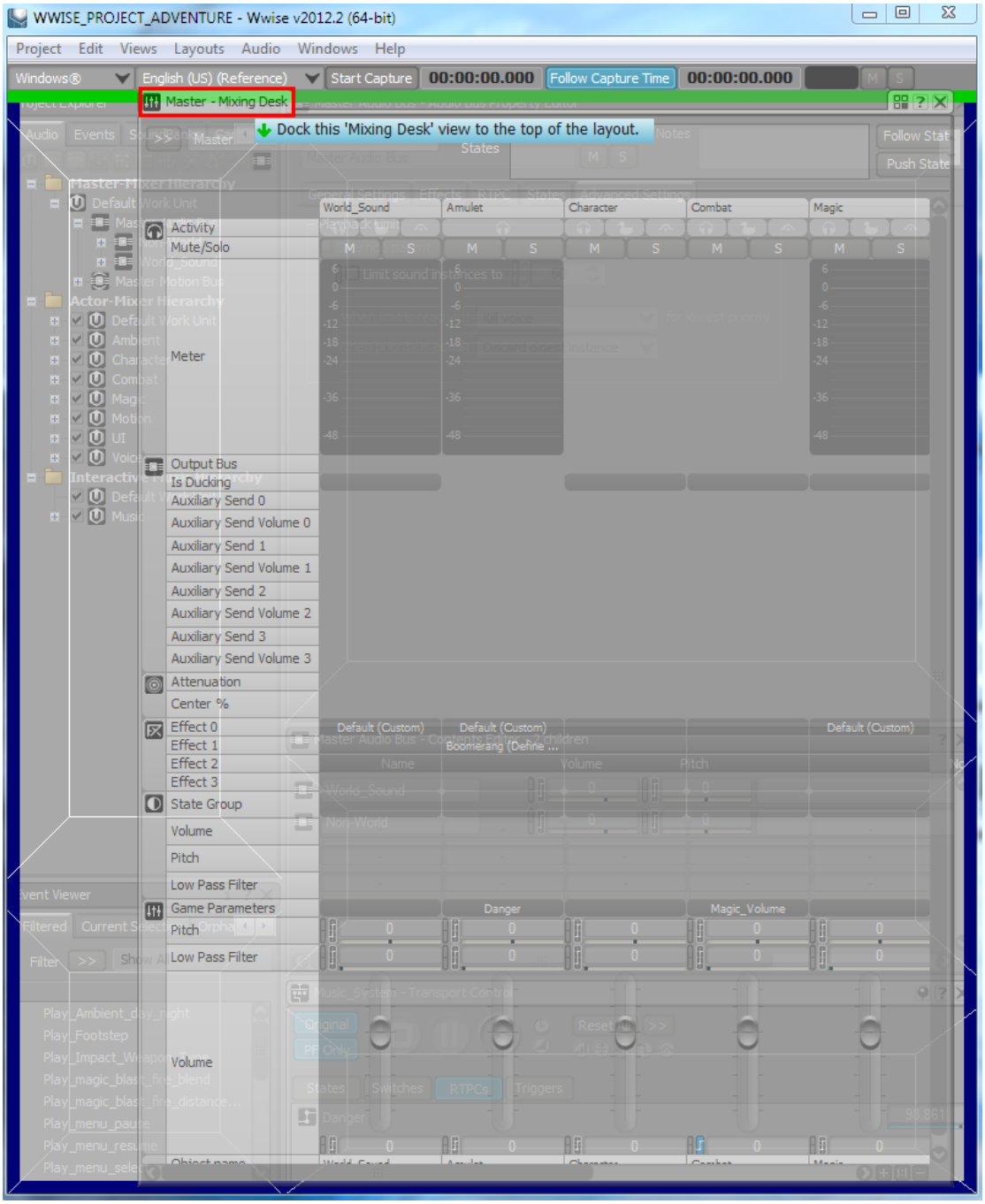
ビューのタイトルバーをクリックしてドラッグするとフロートビューになり、さらに、ドッキング可能な位置も表示されます。



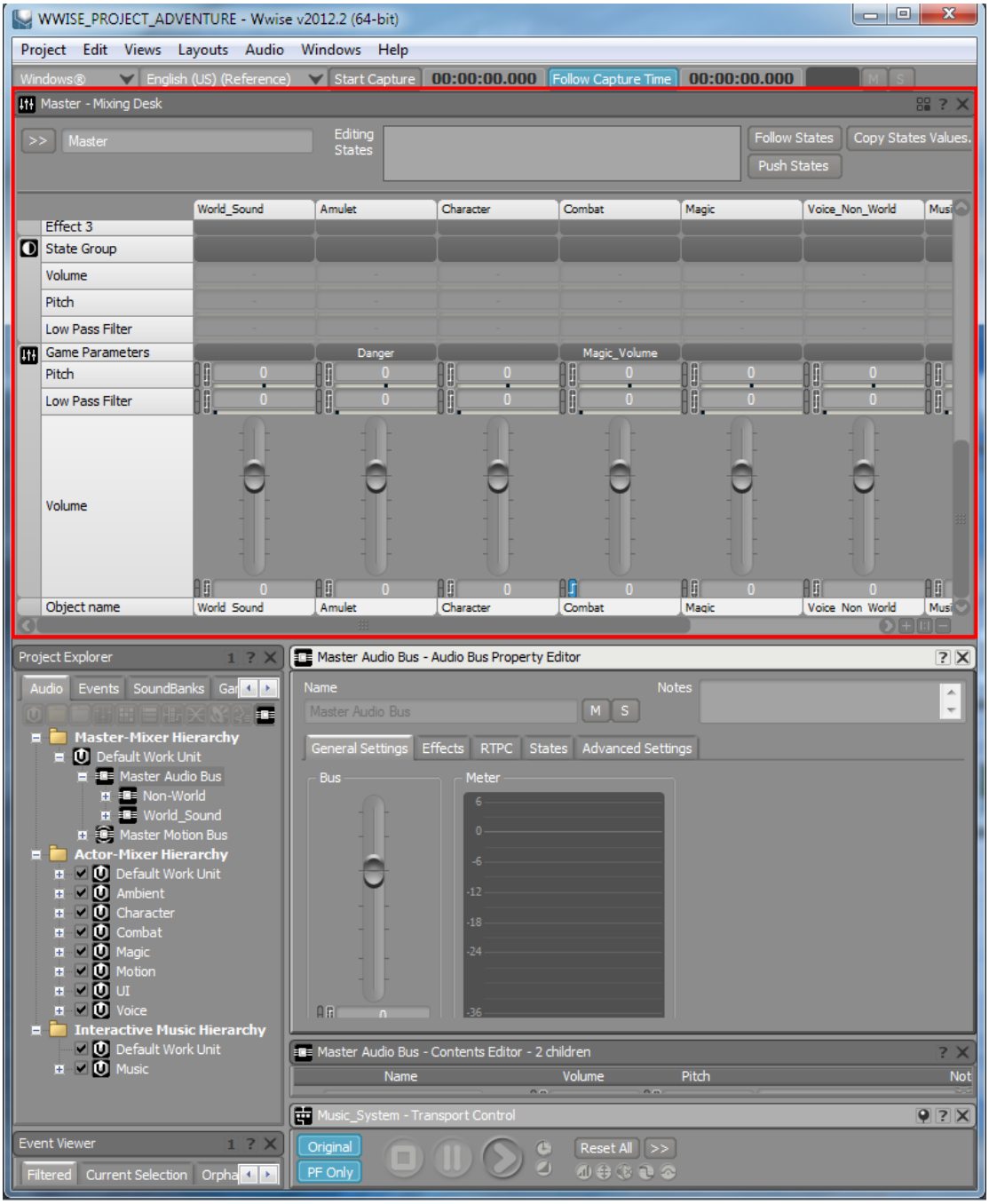
タイトルバーをクリックしてドラッグすることで移動

レイアウトやドッキングされたビューの上下左右に、別のビューをドッキングできます。

レイアウトのドッキング

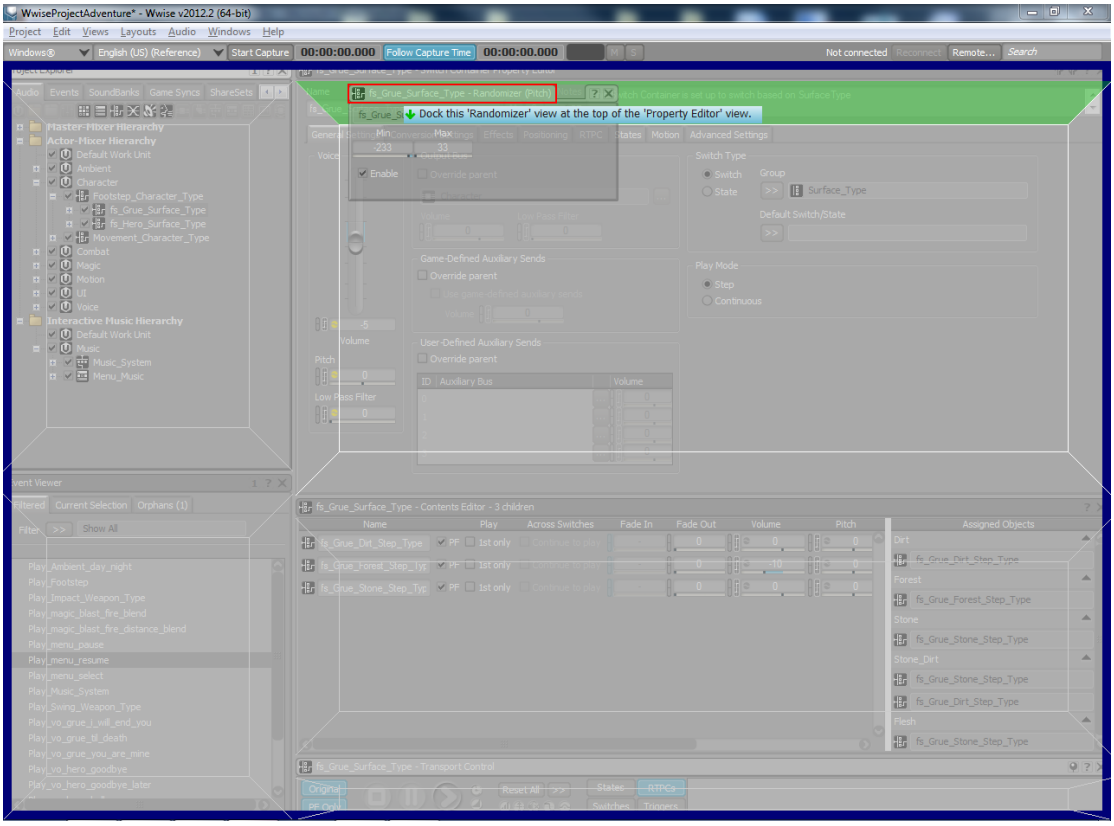


レイアウトウィンドー周りのドッキング可能な位置が青で表示され、カーソルを移動すると、緑に変わる

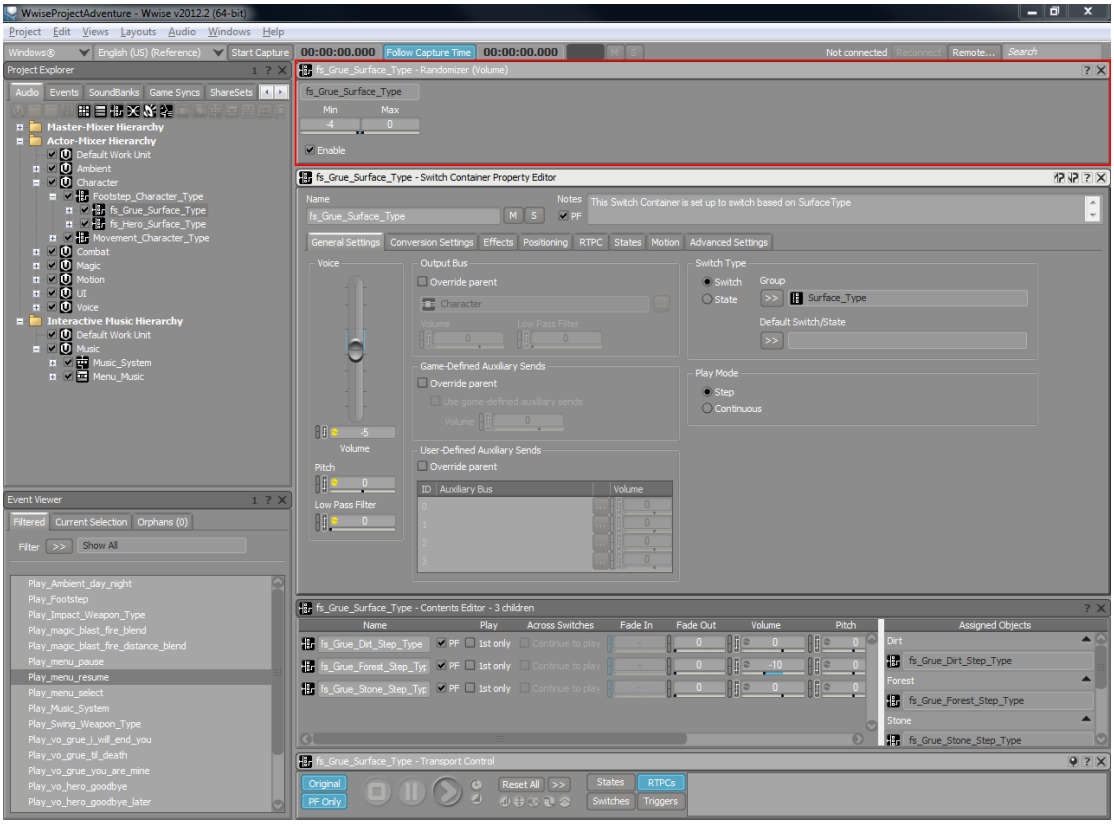


Mixing Deskビューをレイアウトの上部にドッキングさせた例

ビューのドッキング

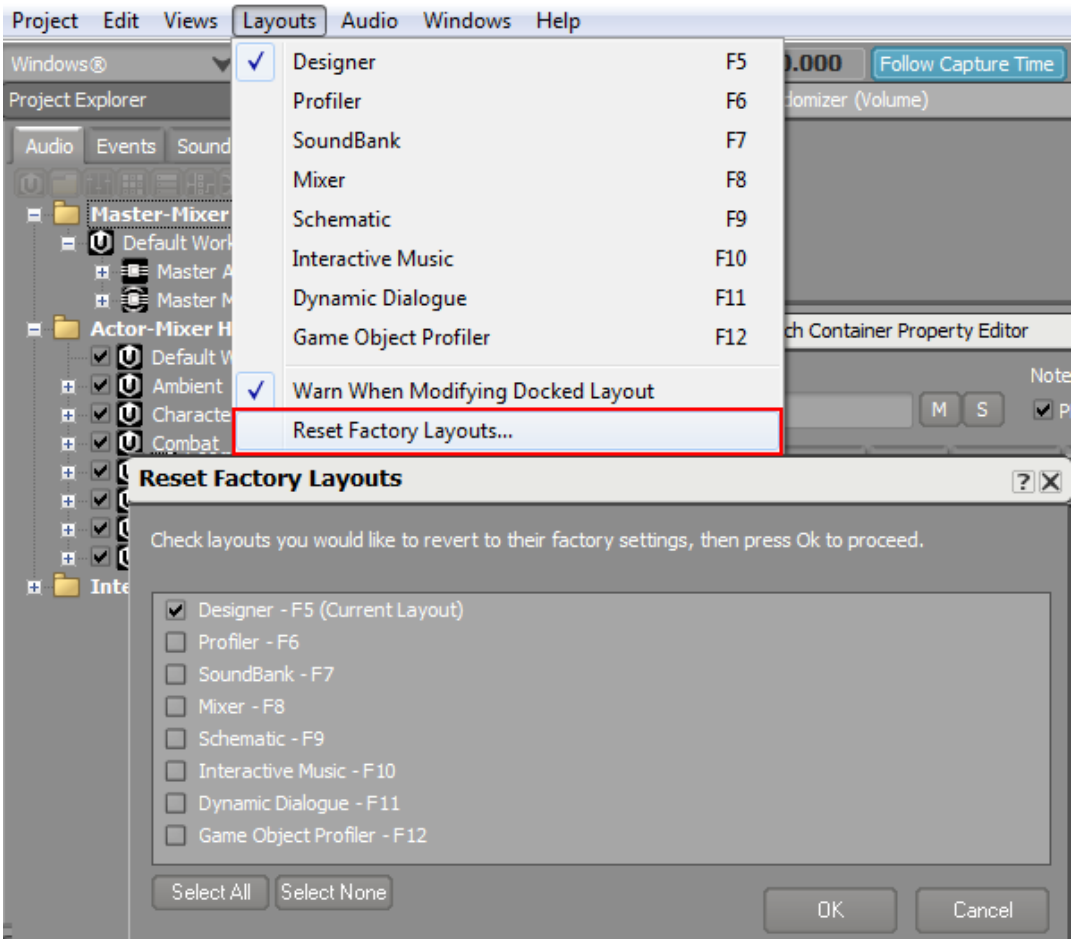


ドッキングされたビューの上下左右のドッキング可能な位置にカーソルを移動すると、緑に変わる



Property Editorビューの上にRandomizerビューをドッキングさせた例

デフォルトのレイアウト設定に戻すには、メニューバーのLayoutsメニューからリセットを選択します。



DesignerレイアウトのFactory Layouts設定をリセット

モニターを数台使い、レイアウトをカスタム設定してディスプレイ面積を最大限に利用すれば、作業内容に合った効率的なビューの配置が可能です。

レイアウトの詳細情報

Video Tutorial - Managing Layouts

エクスターナルソース

エクスターナルソース（外部ソース）とは、Wwiseのサウンドオブジェクトに入れることができる、特別なオーディオソースのことです。実際のサウンドデータはランタイムに提供されます。例えばダイアログが多数ある場合に、それぞれのサウンドとイベントを作成して全てをサウンドバンクに入れる方法の代わりに使うと、便利です。またAIを使う音声用生成システムなど他のシステムでダイアログを既に管理している時にも、非常に便利です。

プロジェクト中のダイアログ管理の方法にもよりますが、プラグインExternal Sourceを使うとメモリに多量のボイスアセットをロードせずにダイアログを再生できるので、ランタイムのさらなるメモリ節約につながる可能性があります。

プラグインExternal Sourceは、以下の処理を行います。

- Wwise上で、プラグインExternal Sourceを使いテンプレート（ひな形）Sound Voiceが作られます。このテンプレートは共通のプロパティを有する複数のオーディオファイルを表します。
- プラグインExternal Sourceを、コンテナ、アクターミキサー、ステート、RTPC、エフェクトなどの中に配置すれば、プロジェクト階層の機能や柔軟性を最大限に利用できます。
- 外部ソースを呼び出す再生イベントが作成されます。
- プラグインExternal Sourceが使用する、外部オーディオアセットをプールしてある保存場所や、コンバージョンの設定は、External Source Listファイルで定義されます。このファイルは単純なXMLファイルで、コンバージョンが必要な外部オーディオアセットの場所や、適用すべきコンバージョン設定の情報が含まれます。このファイルの保存場所は、Project Settingsダイアログボックスで定義されます。
- ランタイムにゲームがプラグインExternal Sourceを呼び出し、テンプレートを1つの外部オーディオファイルと組み合わせます。実際に再生されるオーディオファイルはプログラマーが自由に決めることができます。なお、ソースのオーディオファイルはWwiseサウンドエンジンの外で管理されます。作業量が増えますが、柔軟性が向上します。

Designer Note

外部ソース機能の作業の大部分は、SDK内でオーディオプログラマーが行います。詳しくはWwise SDKドキュメンテーションをご参照ください。

サウンドバンクとサウンドバンク生成

サウンドバンクの中に、オーディオファイルやその再生方法の情報が、ゲームからの要求に応じて簡単にロードしたりアンロードできるように、ファイルにまとめた状態で入っています。ゲーム側でメモリやリソースを管理するするために様々なシステムを採用しますが、Wwiseのサウンドバンクは大部分の実装方法をサポートできる形で生成されます。

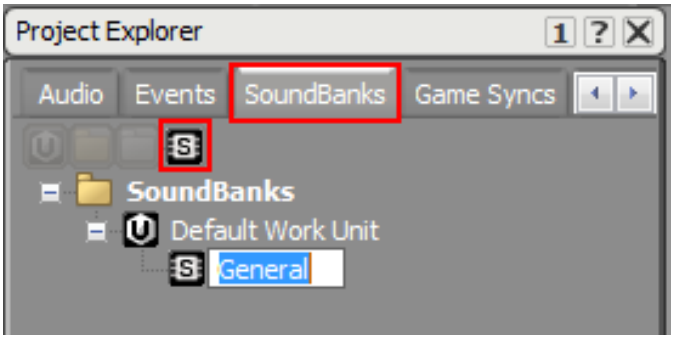
Designer Note

サウンドバンクには、イベント、Wwiseオブジェクト、コンバージョン後のメディアファイルなどを無制限に入れることができます。ゲーム中の特定ポイントで、これからトリガーされる可能性のあるサウンドやモーションオブジェクトに備えて、プロジェクトを構成するコンポーネントが1つ以上、ゲームのプラットフォームメモリにロードされます。

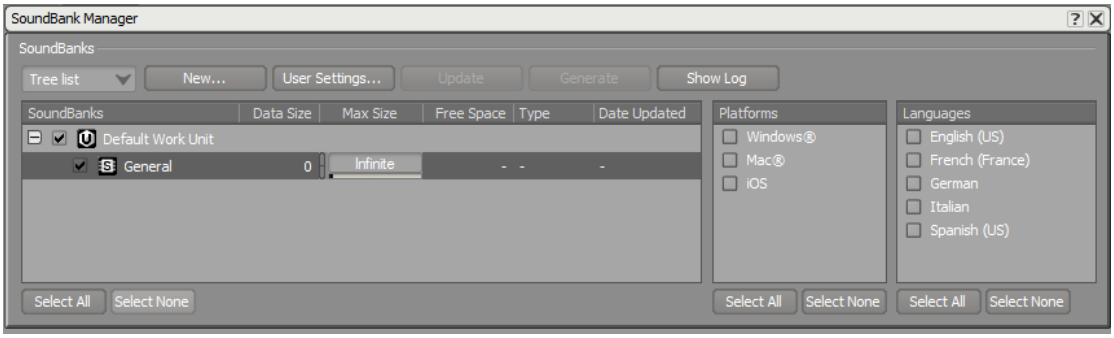
サウンドバンクを作成するには、専用のサウンドバンクレイアウト（F7）を使うと便利で、SoundBank Manager画面やSoundBank Editor画面へのアクセスが表示されます。

サウンドバンクの作成

まず、Generalという新規サウンドバンクを作成します。Project Explorer画面のSoundBanksタブを開き、デフォルトワークユニットを選択します。次にProject Explorer画面のツールバーにあるサウンドバンクのアイコンをクリックすると、新規サウンドバンクが追加されます。また、サウンドバンクのワークユニットのコンテキストメニューや、ショートカットキーから、サウンドバンクを作成することもできます。



サウンドバンクGeneralの作成

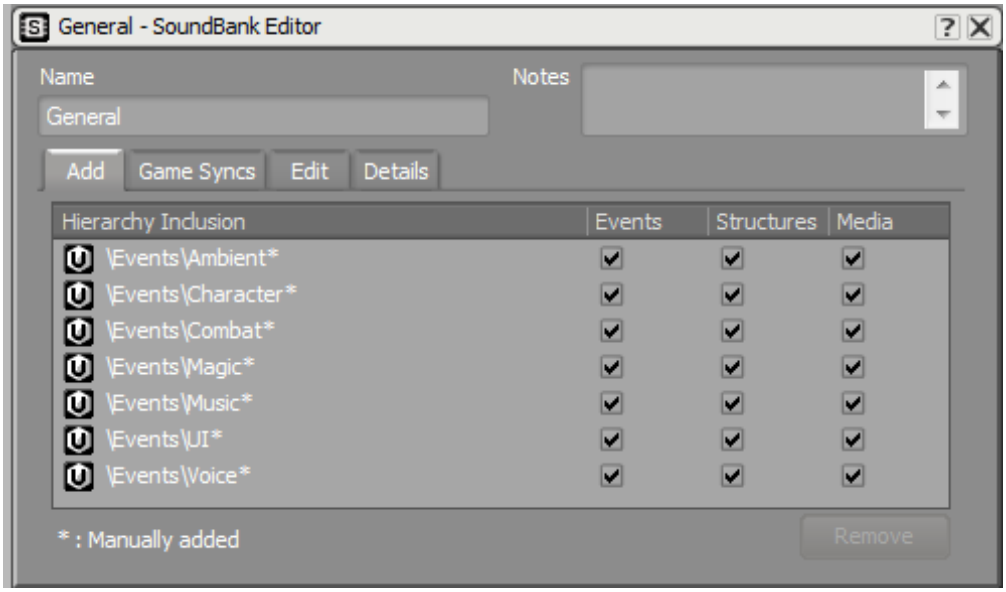


SoundBanks Manager画面に表示された、サウンドバンクの詳細

SoundBank Manager画面に作成したサウンドバンクの一覧が表示されます。それぞれのData Size（サイズ）、Free Space（未使用のスペース）、Type（種類）が表示されます。ここで、サウンドバンクを生成する前に、サウンドバンクのアップデート、ユーザー設定のカスタム化、生成するサウンドバンクが対応すべきプラットフォームやランゲージの指定などができます。

サウンドバンクのコンテンツの投入と管理

SoundBank Manager画面でサウンドバンクをダブルクリックすると、自動的にSoundBank Editor画面に関連情報が表示されます。



SoundBank Editor画面にマニュアルで追加したイベント

SoundBank Editor画面には以下の4つのタブがあり、サウンドバンクのコンテンツの追加と管理を行います。

SoundBank Editor - Add - この追加タブは、サウンドバンクに追加した実際のイベント、階層、ワークユニット、フォルダだけを表示。これらに関連してサウンドバンクに自動的に追加された子オブジェクトは、Editタブにのみ表示される。Addタブで、サウンドバンクの階層エレメントごとに入れる情報やメディアの種類を決定する。

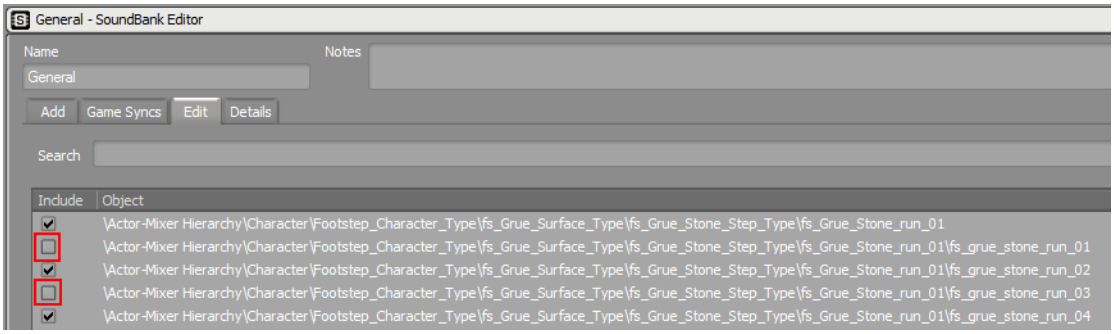
SoundBank Editor - Game Syncs - このゲームシンク用タブには、Addタブで入れたイベントやサウンド構成が参照するゲームシンクの一覧を、ゲームパラメータや引数を除き表示する。このタブで特定ゲームシンクとの関係に基づくフィルターを使って、サウンド構成、イベント、メディアファイルを除くことができる。

SoundBank Editor - Edit - この編集タブには、イベント、オブジェクト、メディアファイルの詳細情報の一覧を表示し、これにはAddタブの階層的なプロジェクト要素に結びつけられた子オブジェクトも全て含まれる。フィルターを使ってランゲージやオブジェクトタイプで整理して、サウンドバンクから除く構成要素を選択できる。

SoundBank Editor - Details - この詳細タブには、メモリサイズ、ファイルサイズ、サウンドエフェクト対ボイスのサイズ情報、欠如しているファイル数や置き換えられたファイル数を含むサウンドバンクの詳細情報が表示される。

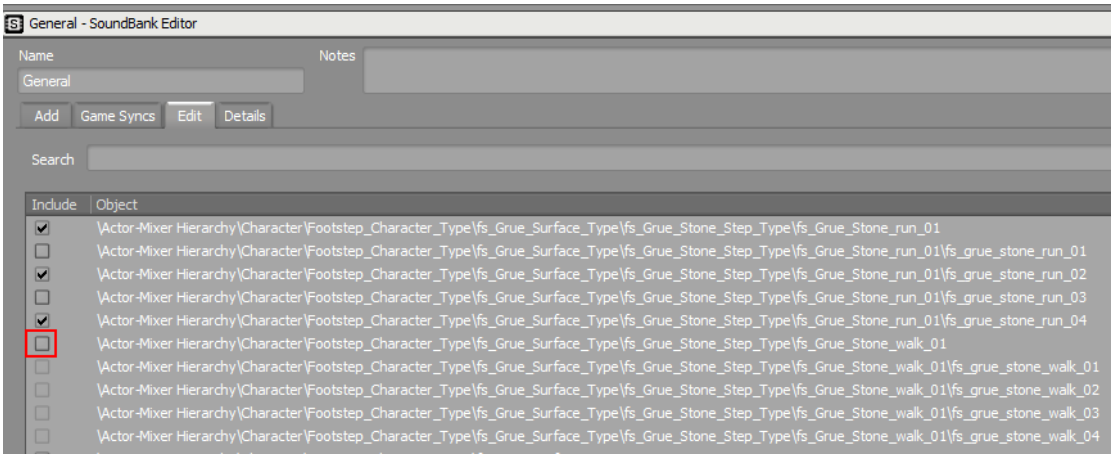
サウンドバンクからエレメントを排除

サウンドバンクに入っている構成要素のチェックボックスを外せば、サウンドバンクから除くことができます。



サウンドオブジェクトをマニュアル選択して、サウンドバンクから除く

サウンドバンクの中の親オブジェクトを取り除くことで、子サウンドオブジェクトを取り除くことができます。



サウンドオブジェクトの親と子をマニュアル選択して、サウンドバンクから除く

必要なサウンドオブジェクトを入れてサウンドバンクを生成すれば、ゲームで使用できる状態になります。

サウンドバンク管理の新しいアプローチ

様々なゲーム要件に柔軟に対応するために、Wwiseではゲームのサウンドバンク管理に、新たな考え方を導入しました。従来の方法（後述）で提供する利便性を維持したまま、ゲームの要件管理を改善するために、コントロール範囲と柔軟性を広げました。

この新しいアプローチは従来の方法に比べ、主に以下の3つの改善点があります。

- イベント以外にも、構成されたコンテンツ（サウンド、コンテナ、アクターミキサーなど）、ワークユニット、フォルダをサウンドバンクに入れることが可能。
- サウンドバンクに入れる情報の種類を限定できる。サウンドバンクにメディア、構成データ、イベントコンテンツのうち1種類だけを入れることも、組み合わせていれることも可能。例えば、特定のイベントに関連するメディアのみを入れたサウンドバンクを作成することもできる。
- サウンドバンクに特定のアイテムを入れたり削除することが可能。

この新しいアプローチの最大のメリットは、メディアコンテンツを複数のメモリバンクに分けられることです。例えば、ゲーム全体の音楽を1つのイベントでスタートさせるとします。従来の方式では、このイベントをバンクに追加すると、対応するメモリ内サウンドやプレフェッチ用のメディアが自動的に追加され、ゲームの最後だけに再生される歌のプレフェッチ部分まで含まれてしまいます。この場合、ゲームの最後までメモリ内に全てのメディアが保存され続け、非常に効率が悪いことがあります。新しいアプローチでは音楽用のメディアを分割して、再生される可能性のあるサウンドのみをロードさせることができるので、メモリ管理が改善されます。

メディアを複数のバンクに分割することで、ロードするメディアの優先順位も決められます。例えば、メモリ量が制限された環境では、最も大事なメディアのみをロードします。クリティカルでないメディアは別のバンクに保存し、メモリに余裕がある場合だけロードします。これまでは、クリティカルなメディアファイルとクリティカルでないファイルを同じバンクに入れていました。このため、バンクがメモリにロードしきれない程に大きい場合はサウンドは再生されず、クリティカルなメディアも再生されない結果となりました。

サウンドバンクの生成とゲームへの実装方法はいくつかあり、それぞれの方式はWwise SDK Bank Management Samplesで詳しく説明されています。なお、同じゲームで2種類以上の方式を組み合わせて使うこともできます。ゲームによって要件が異なるので、選択する方式（単一または複数）はゲームごとに判断します。どのソリューションを選んでも機能しますが、メモリ使用量、I/Oアクセス、ゲームへの実装方法などを考慮した上で判断すると良いでしょう。それぞれに長所と短所があるので、多くの場合では、メモリ使用量と実装のしやすさのバランスをみながら選択することになります。

サウンドバンクのインテグレーションについての詳細情報

Wwise SDK - Windows > Sound Engine Integration Walkthrough > Integrate Wwise Elements into Your Game > Integrating Banks > **Integration Details - Banks**

コンバージョン設定

コンバージョン設定はシェアセットとして管理され、プロジェクトのニーズや対応プラットフォームの要件に基づき作成されます。コンバージョン設定で決定した設定の多くはオーディオ全体のパフォーマンスや品質に大きく影響します。オブジェクトにコンバージョン設定のシェアセットを適用した後でも、使用するプラットフォームやゲーム自体の制約範囲内で最良の品質を達成するために、いつでもシェアセットを編集しなおすことができます。また、オーディオファイルをインポートする時にシェアセットを再利用すれば、より早く処理できます。

シェアセットの詳細情報

Wwise Help > Finishing Your Project > Managing Platform and Language Versions > Authoring Across Platforms > Converting Audio Files > **Creating Audio Conversion Settings ShareSets**

Video Tutorial - Conversion Settings ShareSets

サウンドバンク定義ファイル

サウンドバンクをマニュアル操作で生成する方式以外に、SoundBanks Definitionファイルを使ってゲーム中のレベル、キャラクター、オブジェクト、材質などの情報をゲームから受け取り、自動的にサウンドバンクを生成する方式も一般的となっています。ここで使う定義ファイルは、ゲームレベルエディターなど外部アプリケーションで自動的に生成できるほか、サウンドバンクで分類するゲーム内の全イベントを一覧にした、タブ区切りテキストファイルをマニュアルで作成するこ

ともできます。サウンドバンクが生成されると、各サウンドバンクの中にあるイベントが使用するアクターミキサー、コンテナ、サウンドなどを、Wwiseがパッケージ化します。またサウンドバンクにSound Voicesのサウンドが入っている場合は、Wwiseで対応したランゲージごとにサウンドバンクが生成されます。

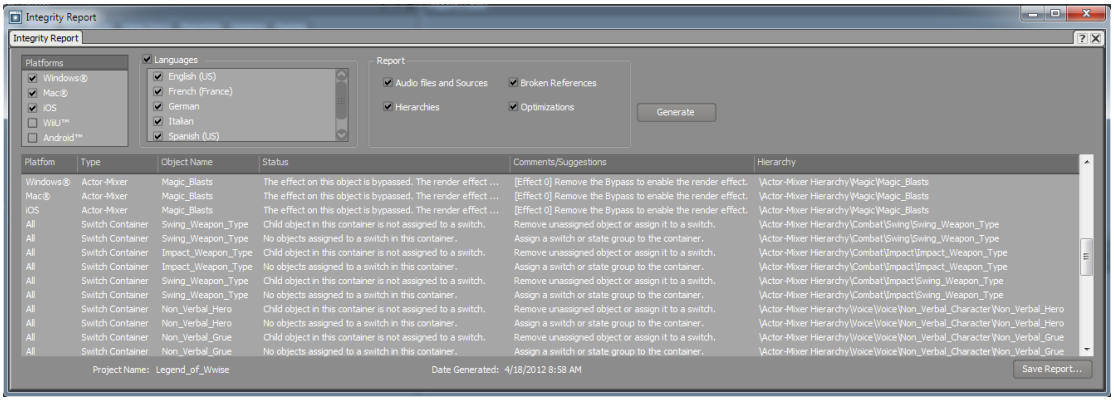
サウンドバンクの詳細情報

Video Tutorial - Creating and Managing SoundBanks

Video Tutorial - Building SoundBanks

インテグレティーレポートの使い方

ゲームの闇世界にトレジャーが隠され敵が潜んでいるように、プロジェクトの裏側に隠れているエラーや矛盾を示す地図となり得るのが、インテグリティレポートです。WwiseのIntegrity Report画面上でプロジェクトのエラーとその解決方法など、様々な情報を含むインテグリティレポートを生成できます。



各種情報、エラーの可能性、解決方法などが示されたIntegrity Report画面

インテグリティレポートには、以下のようなエラーが表示されます。

- 欠如しているメディアファイル
- 欠如しているオーディオソースやモーションソース
- プラグイン問題
- サウンドバンクに欠如しているイベント

エラーリスト中のエラーをダブルクリックすると、対応するWwiseダイアログボックスが開くので、ここでエラーを解決したり対処方法の詳細情報を閲覧できます。

インテグリティレポートにフィルターを設定して、以下の項目に関して表示する情報の種類を選ぶこともできます。

- プラットフォーム
- ランゲージ
- オーディオファイルとオーディオソース
- 階層
- レファレンス
- 最適化

プロジェクト中でさらなる検討が必要な部分や、理解を深める必要がある部分を、インテグリティレポートで確認することができます。

ファイルパッケージャーの使い方

ファイルパッケージャーは、Wwiseプロジェクトのサウンドバンクやストリーミング用メディアファイルをグループに分け、指定したプラットフォームで使えるようなファイル（単数または複数）にパッケージする、独立したツールです。ファイルパッケージャーは、各言語用のランゲージバージョンやリリース後に提供するダウンロード用コンテンツを管理するのにも便利です。

以下のような内容のファイルパッケージを作成できます。

- サウンドバンクのファイルのみ
- ストリーミング用メディアファイルのみ
- サウンドバンクのファイルとストリーミング用メディアファイル

Wwiseプロジェクトのサウンドバンクやストリーミング用メディアファイルの情報は、SoundBanksInfo.xmlファイルをファイルパッケージャーにインポートすることで全て入手できます。なお、SoundBanksInfo.xmlファイルはサウンドバンクを生成した時に毎回、Wwiseが自動的に作成します。

ファイルパッケージはファイルパッケージャーを使ってマニュアルで作成できるほか、サウンドバンク生成の処理の一貫としてファイルパッケージャーを起動するコマンドラインで自動化することもできます。このコマンドラインはプロジェクトレベルで定義するか、サウンドバンクのユーザーによるカスタム設定とすることができます。

ファイルパッケージャーの詳細情報

Wwise Help > Finishing Your Project > **Managing File Packages**

Wwise Knowledge Base - Using file packages

Wwise Knowledge Base - How do LoadBank/UnloadBank and PrepareEvent work together?

Wwise Knowledge Base - How to avoid duplication of source files when a sound exists in multiple SoundBanks

Video Tutorial - File Packager

ダウンロードコンテンツ（DLC）

ゲームのリリース後にコンテンツを提供することは、ゲームの開発戦略として一般的になりました。ゲーム制作のスケジュールに最初から戦略として含まれることも多く、適切に計画することでコンテンツを追加する段取りをシームレスに行えます。

リリース後に配布するコンテンツを開発する場合、必ずメインリリースの開発に使用したWwiseプロジェクトを使ってダウンロードコンテンツを作成します。また互換性の確保のために、メインリリースとダウンロードコンテンツはWwiseの同じバージョンを使う必要があります。ダウンロードコンテンツの基本的な管理方法の説明がWwise Knowledge Baseにあり、プロジェクトの準備における検討事項や制限事項も記載されています。

ダウンロードコンテンツの準備に関するプロセスを事前に研究して、よく理解しておくことが大事です。適切なアプローチをとることで、実装されたオーディオを変更して追加オーディオを提供する能力を発揮して、初回リリース以降もクリエイティブな開発が推し進められていくでしょう。

ダウンロードコンテンツの詳細情報

Wwise Knowledge Base - Downloadable Content Overview

セットアップのまとめ

セットアップのオプションや検討事項が無数にあるようで、最初は圧倒されるかもしれませんが、特定の開発方式のニーズに対応できるようにプロジェクトを準備し修正することで、必要な機能を利用できる状態が確保されます。プロジェクト全体の基盤となるゲームコンセプトを理解することで、目の前にある課題をこなす環境が整います。道を切り開くためのツールが揃えば、可能性は無限に広がるでしょう。

本章では、以下を説明しました。

- メモリ
- 初期段階に確立するネーミングルール
- ワークユニットの論理的なグループ分け
- 共有を前提としたワークユニットの作成
- アクターミキサー階層内のオブジェクトのグループ分け
- Soundcasterを使ったシミュレーションの作成
- プロジェクトセッティング
- ワークグループ用プラグインの設定
- オーディオファイルの保存場所
- デフォルトのコンバージョン設定
- サンプルレートの自動検知設定の定義
- オブストラクションとオクルージョン
- オブストラクション設定とオクルージョン設定
- モーション
- レイアウトのカスタム設定
- レイアウトのドッキング
- ビューのドッキング
- 外部ソース
- サウンドバンクとその生成
- サウンドバンク管理への新しいアプローチ
- コンバージョン設定
- サウンドバンクの定義ファイル
- インテグリティレポートの活用
- ファイルパッケージャーの活用
- ダウンロードコンテンツ（DLC)）

参考ドキュメントとチュートリアル

Video Tutorial - Workgroup Management in Wwise Using Perforce

Video Tutorial - Wwise Motion

Video Tutorial - Managing Layouts

Wwise Help > Finishing Your Project > Managing Platform and Language Versions > Authoring Across Platforms > Converting Audio Files > **Creating Audio Conversion Settings ShareSets**

Video Tutorial - Conversion Settings ShareSets

Video Tutorial - Creating and Managing SoundBanks

Video Tutorial - Building SoundBanks

Wwise Knowledge Base - Using file packages

Wwise Knowledge Base - How do LoadBank/UnloadBank and PrepareEvent work together?

Wwise Knowledge Base - How to avoid duplication of source files when a sound exists in multiple SoundBanks

Video Tutorial - File Packager

Wwise Knowledge Base - Downloadable Content Overview

第12章 ワークフローの最適化

概要

ゲームのジャンルやプラットフォームに関わらず、どのプロジェクトでも、割り当てられたメモリーやプロセッサーの容量制限を必ず気にかけなくてはなりません。モバイルプラットフォームの厳しい制約条件への対応、CPU負荷の最適化、一番大事なサウンドの多様性を維持するための最低限のメモリー容量の確保など、最適化への取り組みは開発が終わるまで、有機的に展開し続けます。ゲームが常に変化している間は目標値を達成するのも困難ですが、一定の秩序を保つために活用できる手法がいくつかあります。

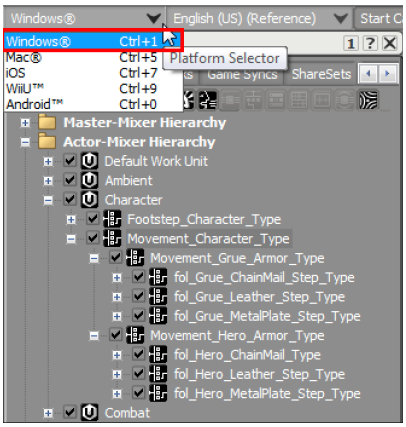
本章では、以下の操作を説明します。

- プラットフォームごとの採用と除外の設定
- プロパティのリンクとアンリンク
- エフェクトのレンダリング
- Wwiseのプロファイリング機能の役割
- ゲームへの接続
- プロファイラを使ったデータキャプチャ
- インゲームのサウンドのプロファイルを確認
- SoundFrameを使ったゲームエンジンとのインテグレーション
- インテグレティーレポート

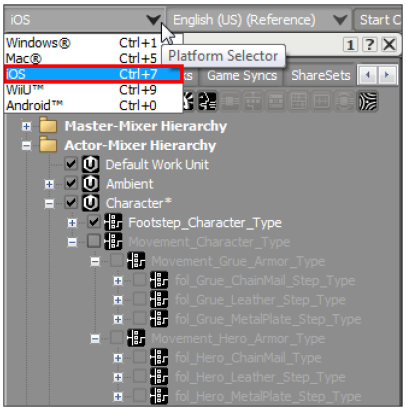
プラットフォームごとの採用と除外の設定

サウンドコンテンツを削って正常な状態を維持する戦略の1つが、サウンドオブジェクトを破壊せずに、プラットフォームごとに採用したり除外する方法です。サウンドオブジェクトの隣のチェックボックスを外すだけで、選択されたプラットフォームのビルド処理から、この親オブジェクトと子オブジェクトがはずされます。

例えば、メモリー領域の小さいプラットフォームに限り全てのMovement（動作）サウンドを除外することでメモリー制限を守りたい場合は、下図のように階層中の親サウンドオブジェクトのチェックを外すだけで完了します。



WindowsプラットフォームではMovementサウンドを維持



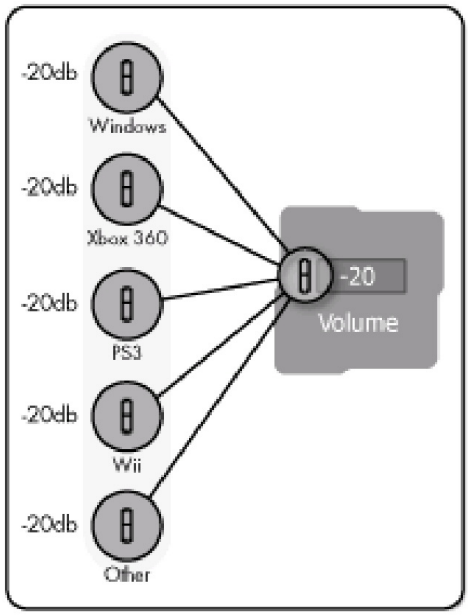
iOSプラットフォームではMovementサウンドを除外

複数のプラットフォームを対象としたプロジェクトでは、この手法で特定のサウンドオブジェクトやサウンドオブジェクトの別バージョンなどを除外できます。プラットフォームセレクターを使い、プラットフォームごとに維持する要件を設定できます。

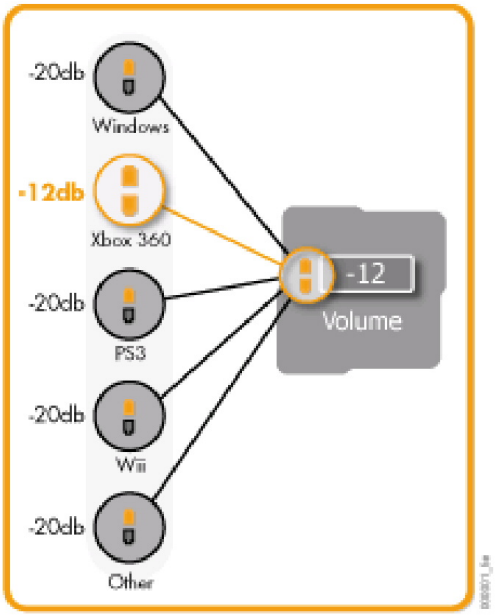
プロパティのリンクとアンリンク

デフォルトとして、全ての対応プラットフォームにおいて、同じプロパティのサウンドやモーションが設定されています。このプロパティは全プラットフォームでリンクされていますが、特定のプロパティを特定のプラットフォームでアンリンクする（リンクを解く）ことも可能です。アンリンク機能を活用すればプロジェクト全体で一貫性を保ちつつ、効率的に複数のプラットフォームに対応できます。コンテンツを破壊せずに、あるプラットフォームでプロパティをアンリンクしてミキシング設定を変えたり、他のプラットフォームでDSPエフェクトを無効化できる機能は、マルチプラットフォーム対応ゲームの開発ワークフローにおいて必ず役に立ちます。

あるプラットフォームでプロパティをアンリンクすると、リンク表示がオレンジ色に変わります。



Properties are linked across all active platforms.

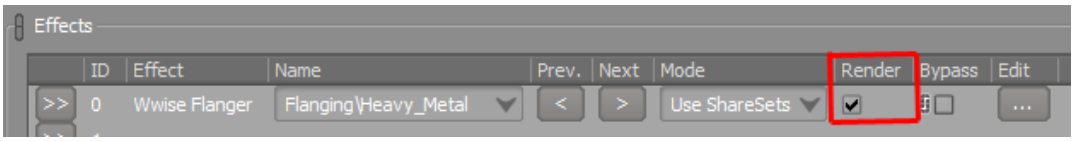


Volume is unlinked and customized for “XBox 360” platform, and the remaining platforms are partially unlinked.

複数のプラットフォームのプロパティを設定する時に、他のプラットフォームでアンリンクされているかを簡単に確認できます。リンク表示が半分オレンジ色であれば、そのプロパティが一部アンリンクされていることを意味します。

エフェクトのレンダリング

貴重なリソースを有効利用したい時は、RTPCで動的に変化するエフェクト以外のエフェクトをレンダリング処理して、節約できる場合もあります。エフェクトのレンダリングを有効にすると、オーディオファイルのエフェクト設定がサウンドバンク生成中に適用されます。結果的にサウンドバンク全体のサイズが大きくなりますが、CPUが不足しているときに役立つ手法です。



エフェクトのRender設定を適用

Designer Note

時間に関係するエフェクト、例えばディレイ、リバーブ、タイムストレッチなどは、ファイルの長さが延長する可能性があり、これをレンダリングするとサウンドバンクでより大きいファイルが生成されますが、メモリー使用量が増える代わりにランタイムのCPU負荷を減らせます。

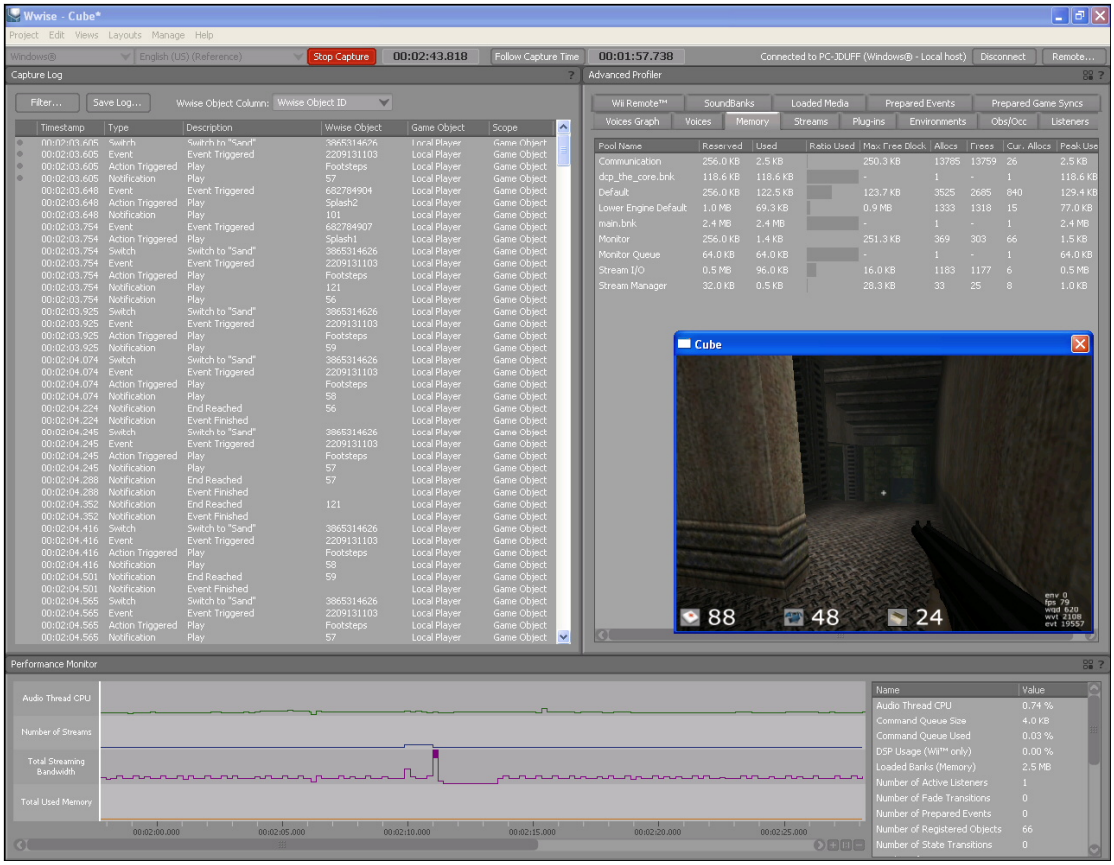
Wwiseのプロファイリング機能の役割

リソースの問題を解決する時に活躍するのが、Wwiseのプロファイラ機能です。オーサリングアプリケーションであるWwiseを、起動中のゲームに接続してリアルタイムでキーパフォーマンス情報をキャプチャーすれば、オーディオエンジン関連の現象をほぼ全て解明できます。プロファイラ機能はインゲームのオーディオ状態を測るための確実なバロメーターであり、デバッグする際は深いレベルまでアクセスできます。

Wwiseは以下の2種類のプロファイリング機能を提供します。

- ゲームのプロファイリング
- ゲームオブジェクトのプロファイリング

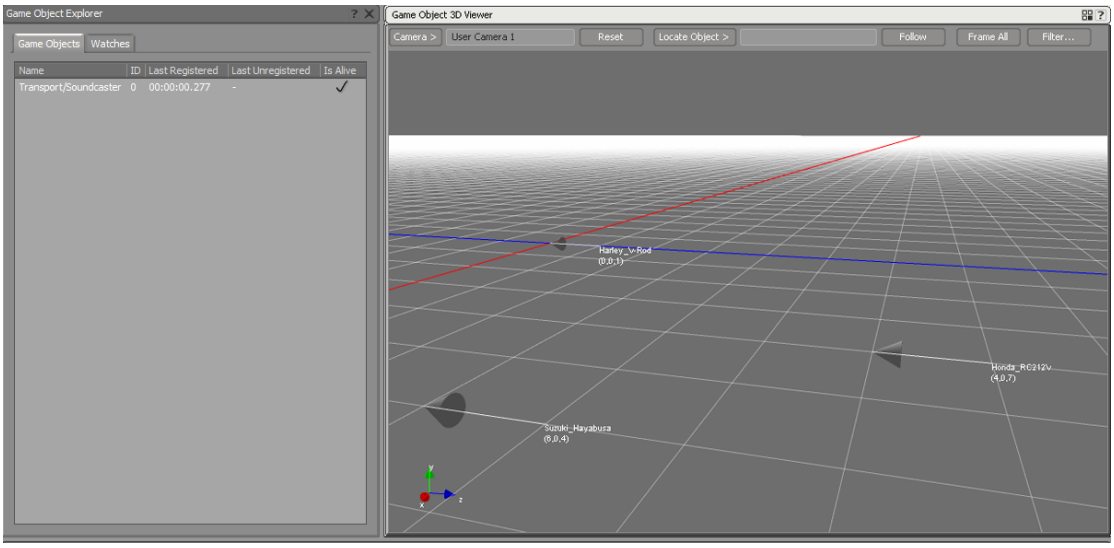
ゲームのプロファイリングを行うと、サウンドエンジンやプロジェクト構成の様々な観点から、パフォーマンス要件や要求に関する情報が表示されます。サウンドやモーションのエフェクトがプラットフォームのパフォーマンスに与える影響を累積値としてリアルタイムで表示できるので、個別のボイスの影響を調査できます。



実行中のゲームに接続して、ゲームプロファイラ機能でパフォーマンスを確認

一方Game Object Explorer画面は、ゲームオブジェクトやリスナーを観察するためのスタート地点です。ゲームに登録されたゲームオブジェクトを全て表示した一覧から、Wwiseで監視するゲームオブジェクトやリスナーを決定します。あるゲームオブジェクトを監視対象として選択すると、Game Sync Monitor画面に表示され、ゲームオブジェクトとリスナーの両者がGame Object 3D Viewer画面に表示されます。

ゲームオブジェクトのプロファイラ機能でもサウンドエンジンのアウトプットを分析しますが、個別のゲームオブジェクトの観点から結果を表示します。ゲームオブジェクトをトラッキングするので、リアルタイムでゲームオブジェクトの行動や動作傾向を観察することができます。これは問題を起こしているゲームオブジェクトなどの検出に使えます。



Game Object Explorer 3D Viewer画面で個別のゲームオブジェクトをトラッキング

ゲームオブジェクトは、ゲーム内に存在する個別のアクターや物体のことです。例えばプレイヤーキャラクター、ノンプレイヤーキャラクター（NPC）、ウェポン、ビークルなど、サウンドを出すゲーム内の全てのオブジェクトやエレメントに、オーディオプログラマーがゲームオブジェクトを登録または作成します。ゲームオブジェクト用のプロファイラ機能（Game Object Explorer、Game Object 3D Viewer、Game Sync Monitor）は相互に補完し合いながら、ゲーム中やシミュレーション中のゲームオブジェクトとリスナーを観察します。

ゲームへの接続

特定のプラットフォームでゲーム中のサウンドやモーションエフェクトを観察するために、プロファイラやシミュレーションを開始するには、まず対象のPCやゲーム機に接続します。LANでアクセスできる稼働中のWwiseサウンドエンジンがあれば、接続できます。

Designer Note

デバッグコンフィギュレーションのパフォーマンスが最適化されていないので、プロファイラ機能を使う時は、ゲームのデバッグ用ビルドを接続する場合でも、WwiseサウンドエンジンのProfileビルド設定に接続することを推奨します。

プロファイラ機能で接続するPCまたはゲーム機を見つけやすくするために、Wwiseはネットワーク上の同じサブネットでWwiseサウンドエンジンが稼働している全てのPCとゲーム機を自動的に検出します。サブネット外にあるゲーム機やPCに接続したい時は、プラットフォームのIPアドレスを直接入力します。

ゲーム接続に関する詳細情報：

Wwise Help > Finishing Your Project > Profiling > **Connecting to a Local/Remote PC or Game Console**

プロファイラを使ったデータキャプチャ

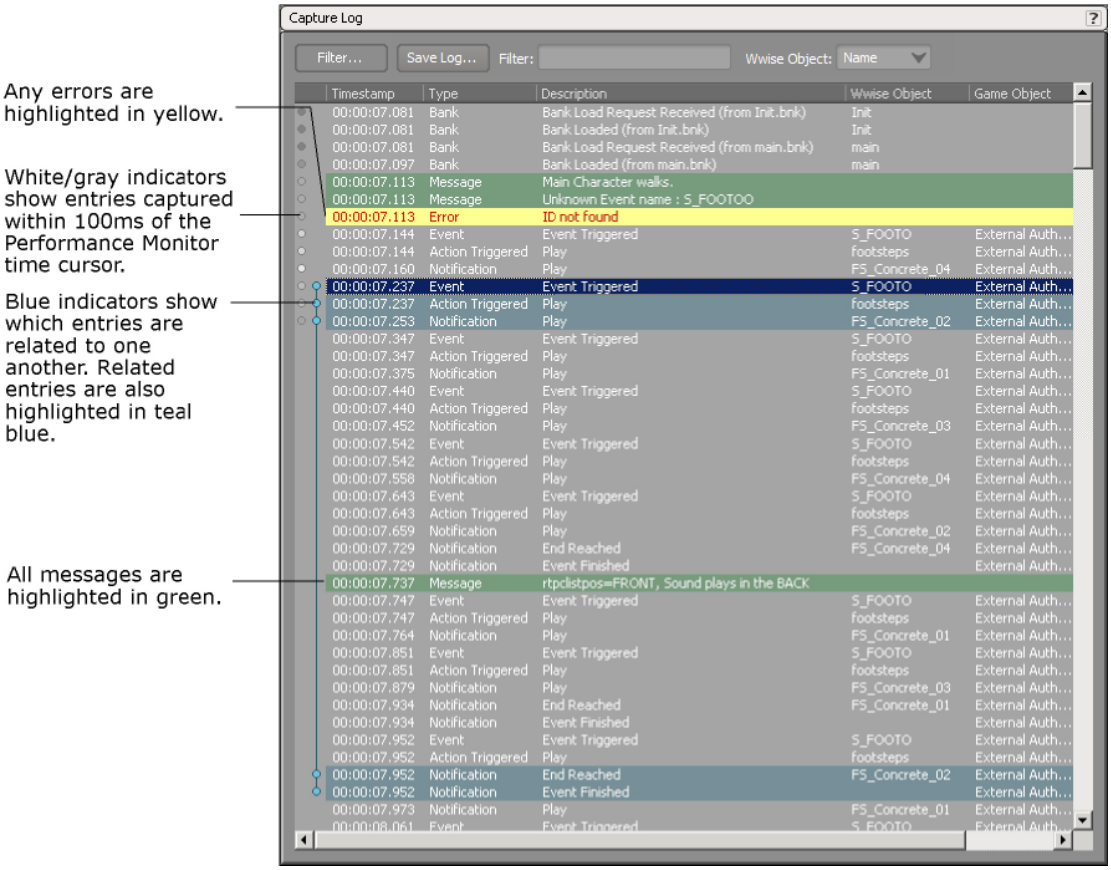
PCまたはゲーム機に接続した後、サウンドエンジンから送られてくるデータを直接キャプチャーして、ゲームオーディオやモーションエフェクトのプロファイルを確認できます。サウンドエンジンから送られてくる情報は、全てCapture Log画面に表示されます。

Capture Log画面では、以下の情報のエントリーが記録されます。

- ノーティフィケーション
- プロパティ
- バンク
- マーカー
- ステート
- エラー
- イベント
- スイッチ
- メッセージ
- アクション
- 待機中のイベント

上記のエントリーはPerformance Monitor画面やAdvanced Profiler画面で監視できます。詳細情報として、メモリー、ボイス、エフェクト使用、ストリーミング、サウンドバンク、プラグインなどの状況が表示されます。

Capture Log画面に多数のエントリーが表示されますが、素早く識別できるように下図の通り色分けされています。下図では、Capture Log画面でどのように異なるインジケーターおよび色が使われているかを示しています。



Capture Log画面の色表示の内容

インゲームでサウンドのプロファイルを確認

ゲーム中に壮大なローレベルダンジョン襲撃で突然サウンドがプレイヤーを包囲して、同時に画面でビジュアルエフェクトの大渦が噴出する場面で、フレームレートの低下が発生したとします。サウンドがフレームレートの低下に関与したかを判断するために、ゲームプロファイラ機能とゲームオブジェクトプロファイラ機能を使いパフォーマンスを分析できます。

ゲームプロファイラを使い、以下を分析します。

- モンスター、ビジュアルエフェクト、物理的オブジェクトなどに関連する多数のサウンドが、プラットフォームのストリーミング能力をどう使用したか
- 位置を示すアンビエントサウンドなどのバックグラウンド音が、いつ、どのようにバーチャルボイスに移行したか
- モンスターの様々な音声にどのエフェクトプラグインが適用されているか、またCPU負荷にどう影響したか

ゲームオブジェクトプロファイラを使い、以下を分析します。

- 各モンスターのサウンドの減衰半径が、他のモンスターの減衰半径と、どう作用したか

- ゲームオブジェクト、例えば発射型のマジックのビジュアルエフェクトなどが、他のエフェクトやモンスターとの関係に応じて、どう動いたか
- サイドチェイン機能を動かすRTPCによって、戦闘に関連するサウンド再生がどう影響されたか

このようにしてゲームプロファイラやゲームオブジェクトプロファイラを使い、ゲーム稼働時のサウンドスケープを細部まで確認することができます。

ゲームプロファイラの詳細情報：

Wwise Help > Finishing Your Project > **Profiling**

Video Tutorial - Wwise Profiler Overview

インスタンスリミット、プライオリティ、バーチャルボイス

プロジェクトが完成に近づいた時点で初めて、インスタンスリミット、プライオリティ付け、バーチャルボイス移行などのエフェクトの結果が、ゲームプレイ中に再生されるサウンドとして実際に聞けます。これらの懸念事項をプロジェクトのスタートから予測して開発中に対策を進めることで、大掛かりな変更を実装するためのリソースが確保できなくなったプロジェクトの終盤において、役立つことがあります。

再生制限

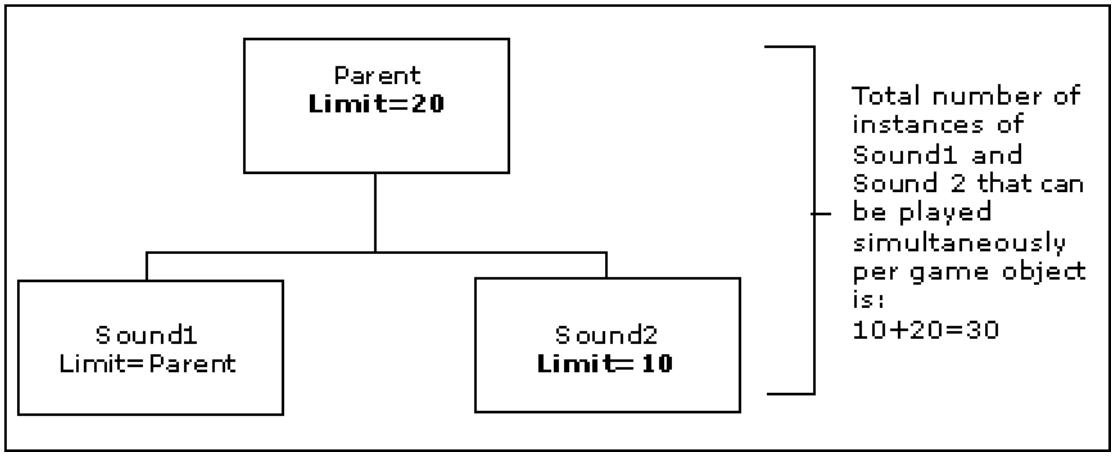
ゲームの限られたリソースやデザイン制約に対応するために、ゲーム中は常にサウンンド、ミュージック、モーションの各オブジェクトを最適化する必要があります。ゲーム中の最適化には、以下の2つの手法があります。

- 1つのゲームオブジェクトで再生できるサウンド、ミュージック、モーションのインスタンス数を制限
- バスを通るサウンド、ミュージック、モーションのインスタンスの合計数を制限

ゲームオブジェクトごとに再生制限を設定

どちらかの制限数に到達した時点で、Wwiseはオブジェクトのプライオリティ設定に基づいて、停止するオブジェクトと再生するオブジェクトを判断します。同等プライオリティのオブジェクトが複数ある場合は、最新または最古のインスタンスが停止されるように事前に設定できます。

再生数の制限をアクターミキサーやインタラクティブミュージックに対して設定すると、同一構成の中で1つのゲームオブジェクトに対して再生できるインスタンス数を制御できます。なお、子オブジェクトが親の再生制限をオーバーライドする場合、その構成内で定義された全ての制限数の合計が、再生できるインスタンスの総数となります。つまり、親の制限数が20で、子の制限数が10であれば、可能なインスタンスの最大数は30です。



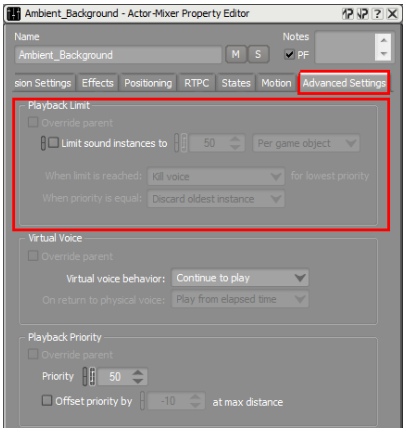
オーディオバスの再生制限を設定

マスターミキサーに対して再生制限を設定すると、バスに同時に入れられるサウンド、ミュージック、モーションのインスタンス数が制限されます。各オブジェクトのプライオリティは既にアクターミキサーやインタラクティブミュージックのレベルで指定してあるので、バスにおける再生プライオリティは設定しません。

グローバルな再生制限

新しいサウンドオブジェクト、ミュージックオブジェクト、またはモーションオブジェクトがアクターミキサーやインタラクティブミュージックのレベルで、キルされず（停止されず）バーチャルボイスに切り替わらなければ、次にマスターミキサーのレベルに送られます。この段階で、バスを同時に通れるボイスの最大数が、グローバル再生リミットによって制限されます。

再生制限は、サウンドオブジェクト、アクターミキサー、またはオーディオバスのProperty Editor画面を開き、Advanced Settingsタブで行います。



Advanced SettingsタブでPlayback Limit（再生制限）の数や動作を設定

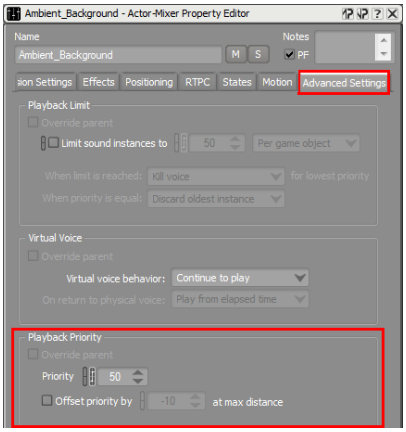
再生プライオリティの設定

プロパティ設定によって、1つのアクターミキサー構成にあるサウンドオブジェクトやモーションオブジェクトの優先順位が決まります。早い段階から、アクターミキサーの最上レベルの親コンテナで、一般的なプライオリティをサウンドタイプに基づき設定する努力をします。また、サウンドの内容を良く理解した上でクリティカルなサウンドを洗い出し、それに高いプライオリティを付与して必ず再生されるように手配すれば、適切に整理されてプロジェクトの最終段階の作業が楽になります。

Designer Note

再生プライオリティは、サウンドオブジェクトやモーションオブジェクトからリスナーまでの距離で変更することもできます。WwiseではAttenuation Editor（減衰エディター）画面で定義したMax distance（最大距離）値を使って、プライオリティをオフセットします。オフセットの大きさは、オブジェクトからリスナーまでの相対位置によって決まります。ソースポイントではオフセットが適用されず、減衰の最大距離ではオフセット設定値がそのまま適用され、その間はWwiseが直線で結びます。

プライオリティ設定は、サウンドオブジェクト、アクターミキサー、またはオーディオバスのProperty Editor画面を開き、Advanced Settingsタブで行います。



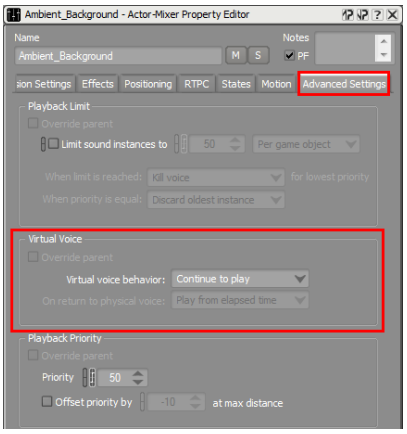
Advanced SettingsタブでPlayback Priority（再生プライオリティ）を設定

バーチャルボイスについて

多数のサウンドが同時再生されてもパフォーマンスレベルが最適に保たれるように、一定ボリューム以下のサウンドが貴重な処理能力やメモリーを消費しないようにします。耳に聞こえないサウンドは、サウンドエンジンで再生せずにバーチャルボイスリストに入れることができます。Wwiseはリストに入れたサウンドの管理と監視を継続しますが、リストに入れられるとサウンドエンジンが処理しないので、ハードウェアのアクティブボイスを使わずにすみます。

バーチャルボイス機能を使うと、閾値以下のサウンドや再生制限を超えたサウンドは、ボリュームレベルによってフィジカル（物理的）ボイスとバーチャルボイスの間を行き来します。WwiseのProject Settings画面でユーザーが設定した閾値にボリュームが到達すると、そのサウンドはバーチャルボイスリストに追加され、サウンド処理が停止します。その後、サウンドが減衰の最大距離を示す半径の中に移動するなど、ボリュームレベルが上がれば、バーチャルボイスリストから外されてフィジカルボイスとなり、再びサウンドエンジンに処理され始めます。

バーチャルボイスの動作設定は、サウンドオブジェクト、アクターミキサー、またはオーディオバスのProperty Editor画面を開き、Advanced Settingsタブで行います。



Advanced SettingsタブでVirtual Voiceの動作を設定

フィジカルボイスからバーチャルボイスに移行するときのサウンドの動作を、以下の3つのオプションから選びます。

- **Continue to play（再生を継続）** - 再生しても耳には聞こえないが、オブジェクトをフィジカルボイスとして再生し続ける。
- **Kill voice（ボイスをキルする）** - オブジェクトの再生を停止する。フェードアウトは適用されない。
- **Send to virtual voice（バーチャルボイスへ送る）** - オブジェクトをバーチャルボイスリストに送る。

サウンドエンジンは、バーチャルボイスリストに送られたサウンドオブジェクトやモーションオブジェクトの特定のパラメータを監視しますが、オーディオ処理やモーション処理は実行されません。

Send to Virtual Voiceを選択すると、サウンドオブジェクトやモーションオブジェクトがバーチャルボイスリストからフィジカルボイスに戻る時の動作を、以下の3つのオプションから選びます。

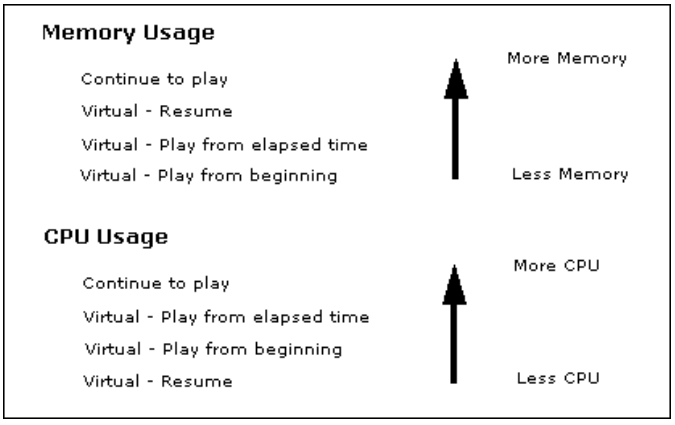
- **Play from beginning（最初から再生）** - オブジェクトを最初から再生する。これを選択すると、オブジェクトのループカウントがリセットされる。
- **Play from elapsed time（時間経過後から再生）** - オブジェクトの再生が中断されなかったかのように、再生を続ける。

なお、このオプションはサンプルアキュレートではないので、フィジカルボイスに戻ったサウンドが再生中の他のサウンドと揃わないことがあります。

- **Resume（再開）** - フィジカルボイスからバーチャルボイスリストに移行する時にサウンドを一時停止して、再びフィジカルボイスになった時に再生を再開する。

下表の通り、各オプションのパフォーマンス特性が異なります。

Behavior	CPU cost	Memory cost
Play from beginning	Medium: Voice stops being serviced when virtual. Some extra operations are done when switching between virtual and physical.	Low: All internal processing buffers are flushed when virtual.
Play from elapsed time	High: Voice needs to be serviced at each buffer when virtual. Some extra operations are done when switching between virtual and physical.	Low: All internal processing buffers are flushed when virtual.
Resume	Low: Voice stops being serviced when virtual. No operations occur when switching.	High: All internal processing buffers are retained when virtual.



Designer Note

ストリーミングサウンドは、バーチャルになった時点でI/O帯域幅を使わなくなります。Play from beginningやPlay from elapsed timeの動作が選択されていると、I/Oバッファーがフラッシュされます。その結果、ボイスがバーチャルからフィジカルに切り替わる際にサウンドが聞こえるまで遅延があります。

つまり、インスタンスやプライオリティを自由に設定することで、ゲームのサウンドが実際に変わってきます。アクターミキサー階層やマスターミキサー階層のどのレベルでプロパティを定義するにしろ、細心の注意が必要です。限定的に設定を変更すると重要なサウンドが適切に聞こえなくなることがあり、逆に設定を無視すると好ましくない動作を引き起こすことがあります。

インスタンスリミット、プライオリティ、バーチャルボイスについての詳細情報：

Video Tutorial - Voice Management

Wwise Knowledge Base - Tips to reduce memory usage

Wwise Knowledge Base - Playback instance limits (including global limits)

Wwise Knowledge Base - Playback Limit and Priority: Use Case Scenarios

Wwise Knowledge Base - Working with object priority and virtual voices

Wwise Knowledge Base - How does playback limit overriding work?

Wwise Knowledge Base - Virtual voices: What’s calculated and what’s not

Wwise Help > Using Sounds and Motion to Enhance Gameplay > Managing the Priority of Sounds and Motion > **Understanding How Wwise Prioritizes Sounds and Motion Objects**

ゲームエンジンへのインテグレーション

ゲームエンジンとオーディオエンジンの2つのオーサリングアプリケーションの機能をつなげられることは、最も過小評価されているワークフロー改善点の1つです。共通する情報セットでリンクされる2つのツールの間のやり取りを使って、サウンドデザイナーの日々のタスクの一部である反復作業を前進させることができます。SoundFrameは、Wwiseと他のオーサリングアプリケーションの間のコミュニケーションを、洗練されたモジュラー方式を使って、簡単に無駄なく処理するためのソリューション群を提示します。

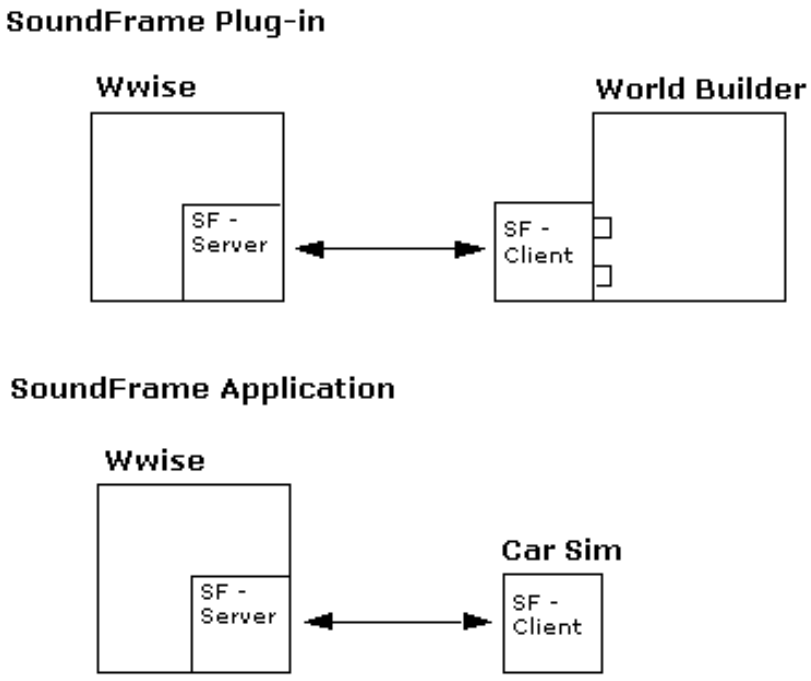
Designer Note

SoundFrameで大部分のSound Engine APIにアクセスできます。ですから、アプリケーション内でイベント再生や、ステート、スイッチ、RTPC、トリガー、環境などの修正が可能となります。このAPIを使って、ゲームエンジンを使わず、サウンドバンクも生成せずに、実際のゲームシナリオをWwiseで直接シミュレーションできます。

SoundFrame SDKを使えば、ワールドをビルドするアプリケーション（Unity、Unreal Editor/UnrealEd、Maya®、3ds Max®、社内の独自ツールなど）に直接組み込めるプラグインを作れます。このようなプラグインはコミュニケーションフレームワークの上に築かれるので、イベントの再生、ゲームシンク変更のトリガー、ポジション情報の修正など多数のWwise機能を、ワールドビルドのアプリケーションから直接実行できます。また、アニメーション内の特定ポイントへのイベントの組み込み、ゲームテキスチャーへのスイッチのマッピング、減衰半径のビジュアル化、ゾーンの環境リバーブの割り当てなど、数多くの設定が可能となります。

SoundFrame機能とは

SoundFrame APIで作成したアプリケーションやプラグインは、WwiseのSoundcaster機能と似た仕組みをとります。Soundcaster機能と同様に、イベント、サウンド、ゲームシンクなどをトリガーして多様なゲームシナリオを再現できます。最大の違いは、SoundFrameを使ったアプリケーションやプラグインがWwiseの外部にあることです。2つのアプリケーションが互いにやり取りするには、同一マシンにインストールして稼働させる必要があります。SoundFrameはクライアント/サーバー型の関係を使い、Wwiseと双方向リンクを成立させます。このようなコミュニケーション様式を採用することで、高度なゲームシナリオを素早く効率的に検証することができます。



SoundFrameのアプリケーションやプラグインによってトリガーされたサウンドは、Wwiseで再生されます。SoundFrameがWwiseをサウンドエンジンとして使うので、サウンドエンジン関連の制約がありません。つまり、更新内容をライブでテストし検証でき、開発サイクルのどの段階にあっても大幅な時間の節約となります。

Soundcasterのシミュレーション能力をもう一歩先に押し進めれば、SoundFrameを使って達成できるオーディオシミュレーションの数や種類はアイディア次第で拡張できます。

ゲームエンジンとのインテグレーションの、他のテクニック

SoundFrameが提供するフレームワークを使い、Wwiseとのツールセットインテグレーションを迅速に開発することができますが、ゲームエンジンとのインテグレーションを独自に実践したいと思う方もいるでしょう。開発環境で必要なこともある深いインテグレーションのためのロードマップとして、WwiseのXML SchemaをSDKと共に提供しています。

Programmer Note

ワークユニット用のXML SchemaはWwise実装のフォルダー[Schemas]にあります（ObjectDataSchema.N.xsd - 最大数のものを使うこと）。このスキーマでワークユニットのファイルのフォーマットを理解することができ、生成されたXMLファイルの検証にも利用できます。また簡単なプロジェクトを使ってWwiseに保存される拡張子.WWUのファイルを見れば、情報の整理方式が分かります。

最適化のまとめ

Wwiseで開発するプロジェクトの詳細情報を初期段階にできるだけ多く入手することが、建設的なワークフロー方式の確立につながります。途中で頻繁に小さな改善を加えれば、プロジェクトの最後でよく発生する落とし穴を避けられます。理想的には、最適化が要求される開発マイルストーンに到達する頃には、既に制限あるリソースの最良の管理方法について検討してあるべきです。研究と熟考された実装の積み重ねで、あらゆるゲームで、素晴らしいサウンドを実現できるツールを活用できます。

本章では、以下を説明しました。

- プラットフォームごとの採用と除外
- プロパティのリンクとアンリンク
- エフェクトのレンダリング
- Wwiseのプロファイリング機能の役割
- ゲームへの接続
- プロファイラを使ったデータキャプチャ
- インゲームでサウンドのプロファイルを確認
- インスタンスリミット、プライオリティ、バーチャルボイス
- 再生制限
- ゲームオブジェクトごとに再生制限を設定
- グローバルな再生制限
- 再生プライオリティの設定
- バーチャルボイスについて
- ゲームエンジンへのインテグレーション
- SoundFrame機能とは
- ゲームエンジンとのインテグレーションのその他のテクニック

参考ドキュメントとチュートリアル

Wwise Help > Finishing Your Project > **Profiling**

Video Tutorial - Wwise Profiler Overview

Video Tutorial - Voice Management

Wwise Knowledge Base - Tips to reduce memory usage

Wwise Knowledge Base - Playback instance limits (including global limits)

Wwise Knowledge Base - Playback Limit and Priority: Use Case Scenarios

Wwise Knowledge Base - Working with object priority and virtual voices

Wwise Knowledge Base - How does playback limit overriding work?

Wwise Knowledge Base - Virtual voices: What’s calculated and what’s not

Wwise Help > Using Sounds and Motion to Enhance Gameplay > Managing the Priority of Sounds and Motion > Understanding How Wwise Prioritizes Sounds and Motion Objects

第13章 最後に

本物のアドベンチャーの開始

長くて困難なゲーム開発の過程で、試作品の失敗、やり直し作業、そして行き詰まりを、何度も経験することでしょう。戦いに向かって試行錯誤あふれる森を進む時は、様々な試練から身を守る頼もしい武器が必要です。ゲームオーディオの現場で、サウンドデザインと実装技術というマジックを手に、戦いを切り抜けて質の悪いサウンドに打ち勝って下さい。難題あふれる開発という任務において、持ち合わせておきたい技術を提供するのがWwiseです。効率性と分かりやすさ、そして広がる可能性が約束されています。いざ冒険へ！

Thank You

Special Thanks

Audiokinetic社：

本ドキュメントはAudiokineticチームの豊富な知識と支援、特にSimon AshbyとEtienne Caronには、専門用語や整合性に関する細かい確認をお願いして完成しました。また最終的なドキュメントのプレゼンテーションは、Bernard Rodrigueの高度な技術手腕によって完成しました。

編集：

Judy Lapalmeは句読点の誤りを要領よく訂正し、無駄のない文章への編成を実現させてくれました。

プロジェクトコンテンツ：

付属のWwise Project Adventureで使われるコンテンツは全て、Bay Area SoundのJulian Kwasneski氏（サウンドデザイン）とJared Emerson-Johnson氏（コンポーザー）によって、本プロジェクト専用に作成されました。

Continuity エキスパート（ベータテスター）：

Hrishikesh Dani、Luca Fusi、Jack Menhorn、Roel Sanchez、Michael Taylor、Rob Bridgett

（敬称略）

筆者について

Damian Kastbauer

Damian KastbauerはフリーランスのTechnical Sound Designerとして、「ノイズメーカー」とゲーム開発者の間の溝を埋めようと努力しています。ゲームオーディオ専用のオーサリングアプリケーションが提供するインタラクティブな技術を活用しながら、ダイナミックなサウンドを作成して「良いサウンドコンテンツ」を「素晴らしいサウンド」にすることが目標です。

ミネソタ州ミネアポリス市在住の筆者は、聡明で美しい妻、輝かしい2人娘、大きな毛むくじゃらの犬、そして気まぐれな猫と暮らしています。インタラクティブオーディオの魅力をいかに人々に伝えるかに没頭していない時は、家族と過ごしたり、ものづくりに励んだり、エフェクトペダルで変な音を作って楽しんでいます。

連絡先：damian@lostchocolatelab.com.

Q: 「Lost Chocolate Lab」とはチョコレートのLab（研究所）をなくしたという意味か、それともチョコレート色のLab（ラブラドール）がいなくなったということですか。
A: 見方によります。ラブラドール犬が迷子になったこともないし、自分のラボがあった記憶もありません．．．。

Q: ミネアポリスに住んで、どう仕事をしているのですか。
A: ホームスタジオでソースコントロールを利用した自宅勤務のほか、時々、開発者と直接オンサイトで仕事をします。

Q: ゲームオーディオの将来はどうなるでしょう。
A: ユニークで魅力的なエクスペリエンスを実現するために、インタラクティブオーディオの使用を増やそうと、みんなが協力するのでは。