

Unity5 サウンドクラスライブラリ

USnd デザイナー向けチュートリアル

最終更新日: 2019/05/27

■ 目次

- ・ 更新履歴
- ・ 概要
- ・ 用語説明
- ・ USnd の仕組み
- ・ サウンド組み込みについて
- ・ Unity 導入
- ・ Unity で決められているフォルダについて
- ・ チュートリアル – USnd 用サウンドテーブル生成
- ・ チュートリアル – AudioClip を読み込む
- ・ チュートリアル – 読み込むテーブルの設定
- ・ チュートリアル – Editor 上で実行する
- ・ AudioManager の使い方
- ・ AudioMasterSettings パラメータ
- ・ AudioCategorySettings パラメータ
- ・ AudioLabelSettings パラメータ
- ・ データ更新時の影響範囲
- ・ サウンドデバッグツール
- ・ XML を使ったパラメータ更新の注意点
- ・ パラメータエディタの使い方
- ・ 3D サウンドの設定方法
- ・ JSON 形式で 3D サウンドのパラメータ設定を行う方法
- ・ 3D サウンドパラメータを実行中に変更する方法
- ・ JSON 形式をインポートしてパラメータを変更する
- ・ Streaming 設定時の注意
- ・ Android で SoundPool を使用して再生する方法
- ・ XML エクセルに更新があったときの最新エクセルへの移行方法
- ・ AudioClip の設定を一括変更する方法
- ・ コール履歴を保存・出力する方法
- ・ トラブルシューティング

■更新履歴

- ・ 2015/06/04 SetAssetImporter の説明を追加
- ・ 2015/06/08 AssetBundle 生成方法について説明を追加
- ・ 2015/08/21 Plugins の配置と一部説明不足だった箇所を修正
- ・ 2015/09/08 Tips を追加、ヘッダーの生成方法を追加
- ・ 2015/09/17 パラメータの数値範囲の説明を追加、RestoreTime の説明文を修正
- ・ 2015/10/05 Cache パラメータ追加に伴い項目の説明を追加
- ・ 2015/10/08 USndViewer の XML ファイル名指定を複数設定できるように変更したので説明文も修正
- ・ 2015/11/11 サウンドデバッグツール USndTool の説明を追加
- ・ 2015/11/13 XML を使ったパラメータ更新の注意点を追加
- ・ 2015/12/09 USndViewer の GUI 更新を説明に反映
- ・ 2016/01/13 USnd ver2.0.0 の追加・修正点を説明に反映
- ・ 2016/01/22 UsndViewer の更新を説明に反映
- ・ 2016/01/29 USnd ver2.1.0 で 3D サウンド対応、3D サウンド設定について説明を追加
- ・ 2016/01/29 NameList 更新機能を追加、説明に反映
- ・ 2016/02/01 AudioMixerSettings と UnityEngine.Audio.AudioMixer の扱い変更に伴い修正
- ・ 2016/02/08 USndViewer 上での AudioMixer 設定反映方法と ExposedParameters に関する説明を追加、Streaming 設定時の注意点について説明を追加
- ・ 2016/08/30 XML パラメータについて記載順番・空欄時の説明を追加。
- ・ 2016/10/13 USnd ver2.6 で、Android で SoundPool を使用できるようになったので使用方法など追加
- ・ 2016/10/19 USndViewer NameList 更新時に自動で NameList を Reimport するように修正
- ・ 2016/12/06 Usnd ver2.7.0 再生間隔を設定するパラメータ Interval を追加、XML 編集エクセルのパラメータ追加時の移行方法説明を記載
- ・ 2017/09/14 USnd ver2.10.0 JSON 形式のファイルの取り扱い方の説明を追加
- ・ 2017/11/15 USnd ver2.11.1 USndTool のログについてトラブルシューティングに追記
- ・ 2018/05/14 AudioClip の設定を一括変更する方法を追記 (USndAudioAssetLoader.cs を追加)
- ・ 2018/11/12 コール履歴を保存・出力する方法を追記 (USndOutputCallLog.cs を追加)
- ・ 2019/05/27 AudioMixer の UpdateMode について記載を追加

■概要

USnd は Unity5 用サウンドクラスライブラリです。

Unity5 の純正 API を使用してオーディオの再生を行い、Unity に足りない機能を USnd 側で吸収し再生管理を行うクラスです。

一部機能はプラグインで実装しています。

下記のような機能が使用できます。

サポートしている機能

- ・ラベル単位で発音数制御を行う
- ・ラベルを任意のグループにわけ、発音制御を行う
- ・フェードイン、フェードアウト
- ・最後に停止した位置から再生を再開する
- ・ピッチをセント単位で扱う
- ・ピッチを指定した時間で変更する
- ・パンを指定した時間で変更する
- ・ランダムボリューム
- ・ランダムピッチ
- ・ランダムパン
- ・ランダムに指定した AudioSource を再生する
- ・グループ化した単位でのダッキング
- ・グループ化した単位での音量変更
- ・ミュート
- ・AudioClip のロード/アンロード(Preload Audio Data 無効時)
- ・AudioSnapshot の設定切り替え
- ・Android の SoundPool API を使用した再生

プラグインでサポートしている機能

- ・他のアプリケーションがオーディオ再生しているか取得する(iOS のみ)
- ・端末スピーカーから再生されているか取得する
- ・マナーモードか取得する

~~・Android のオーディオフィォーカス取得(有効時のみ、デフォルトは無効)~~

→Unity5.3.5 から Unity 側でオーディオフィォーカスを取得するようになりました

UnityEditor のみの機能

- ・コンパイル設定 (サウンド関連の define 設定切替)
- ・USndTool(ロード済み情報の参照、再生インスタンスの表示、ログ表示)
- ・パラメータエディタ (マスター、カテゴリ、ラベルの設定を実行中に変更する)
- ・テーブルログ (読み込んだテーブル情報のログを表示)

Unity でサポートされている機能

- ・ サステインループの読み込み(Windows 版 SoundForge の smpl チャンクを反映)
 - ・ AudioMixer によるボリューム管理
 - ・ AudioMixer によるダッキング
 - ・ AudioMixer によるエフェクト設定
- etc...

■用語説明

Unity の用語

AudioClip	オーディオファイルを Unity に読み込んだ直後の形式です。 ストリーミング、解凍してメモリに展開などロード方式の設定、圧縮形式の指定(スマートフォンは Vorbis を使用します)、圧縮レートの指定やロード時の挙動に関する設定が行えます。 AudioClip は AssetBundle 化可能です。
AudioSource	AudioSource は AudioClip を再生するためのクラスです。シーンに AudioClip を登録すると自動的に AudioSource も生成されます。 AudioSource はそのままでは AssetBundle 化できません。 USnd ではシステム側で AudioSource を生成するので、デザイナー側で AudioSource を用意することはありません。
GameObject	Unity でゲームを作るための部品です。
Prefab	GameObject を汎用的に扱うための機能です。
Scene	Unity のゲーム画面の土台になるものです。 シーンの中にゲームオブジェクトが含まれ、ゲームオブジェクトに AudioSource など色々な機能を提供する機能が含まれています。
AudioMixer	Unity5 で追加された、ボリュームやエフェクトをグループ化して操作するための機能です。現実のミキサーと似たようなものと思って差し支えありません。 特定のシーンに登録などする必要はなく、プロジェクト上に存在していれば機能します。 AssetBundle の指定ができますが、ver5.0.0f4 時点では AssetBunde で扱えないようです。AssetBundle から読み込むことはできますが、含まれる形式が UnityEditor.Audio.AudioMixerController なので実機で使えません。パラメータの変更はアプリ更新が必須になります。(設定したパラメータをプログラムから操作することは可能です。)
AssetBundle	リソースデータをバージョン管理しつつダウンロードする仕組みです。
Resources	Unity プロジェクトに含めるリソースを配置するフォルダ名です。ここに置いたデータはアプリ内蔵データになります。
meta	Unity が自動的に生成するファイル。UnityEditor の Inspector で設定したパラメータが書き込まれています。1 ファイルに対して 1 つ同名の meta が自動生成されます。meta ごとのプロジェクトにコピーすると、パラメータを引き継ぐことができます。

USnd の用語

Label	再生する 1 オーディオを表します。ラベルには AudioClip 名、ボリューム、再生制御方法などの情報が含まれています。ラベルにランダム再生が設定されている場合は、ひとつのラベルから複数のオーディオが再生される場合があります。ランダム再生で指定したラベルが再生されるとき、対象のラベルのランダム設定は無視されますが、それ以外のパラメータは各ラベルの設定に従います。 ラベルは文字列で管理します。ラベル名は重複できません。
Instance	ラベルの再生命令を実行し、実際に発音されたオーディオをインスタンスと呼びます。インスタンス ID は 0 以外の数値（マイナスを含む）で表現されます。
Category	複数のラベルをまとめてボリューム設定したり、同じカテゴリに所属するラベルの現在の発音数で発音制御を行ったりするための単位です。ダッキングもカテゴリ単位で行います。
Master	複数のカテゴリをまとめてボリューム設定するための単位です。 ゲームのコンフィグでボリューム設定を行うときなどに使用します。
XML	USnd でラベル、カテゴリ、マスターのパラメータ更新を、外部ファイルを用いて行う際に使用するファイル形式です。
XSD	XML が正しいかチェックするためのファイルです。XSD はなくても動きます。 Unity でそのまま読み込めないので拡張子を bytes にする必要があります。
JSON	USnd でラベル、カテゴリ、マスターのパラメータ更新を、外部ファイルを用いて行う際に使用するファイル形式です。 XML 同様、文字列形式です。出力は USndXML_Tools を使用します。 Audio3DSettings の JSON 形式出力は Unity 上で行います。
LoadID	まとめてロード/アンロードするための ID です。ラベルに設定できます。 ロード ID は UnityEditor で編集するか、XML 読み込み時にプログラムで指定します。
Ducking	一時的に指定したカテゴリの音量を変更する機能です。

ここまでは **USnd Documentation.pdf** と同じ内容です。

以下はデザイナー向けのチュートリアルです。使用できる API やプログラム関連の情報は **USnd Documentation.pdf** の内容をご確認ください。

■USnd の仕組み

USnd は Unity の機能を使って再生を行います。リソースの管理・エディットは Unity 上で行いますので、UnityEditor のライセンスが必要です。

フリー版もありますが会社では使用できません。Pro ライセンスを準備してください。iOS, Android ライセンスはビルドしなければ不要なので、ビルドする必要があるか相談の上購入申請してください。

通常、Unity でオーディオを再生するときは AudioClip を UnityEditor に登録、シーンに AudioClip を配置して AudioSource を作成。AudioSource の設定を変更してバランス調整などを行いますが、この方法では発音制御など細かい設定は行えません。

USnd では AudioClip とサウンドテーブルファイルを読み込み、AudioSource を USnd 内で生成・管理して再生を行います。ver2 からサウンドテーブルからの読み込みのみサポートするようになりました。

USnd の再生管理の構造は DNSound と同じく Master が一番上の階層にあり、Category が Master の下に存在します。Master は音量設定のみ、Category は発音制御と音量設定が可能です。そしてすべての Label がどこかの Category に所属します。Label には再生するオーディオファイルのほか、ボリューム、フェードイン・アウト、ランダム設定、ダッキング設定など再生に関わる多数の項目が設定可能です。

■サウンド組み込みについて

ver2からはサウンドテーブル(XMLまたはバイナリファイル)から必要情報を読み込み、USnd内でAudioSourceを生成・使いまわして使用方法のみになりました。

DNSoundと異なりUnityがコンバートを行なうので、用意するデータはリニアPCM形式のファイル(wav)です。サウンドテーブルはエクセルで編集します。サウンドテーブルの保存形式はXMLまたはエクセルVBAで出力するバイナリファイルです。

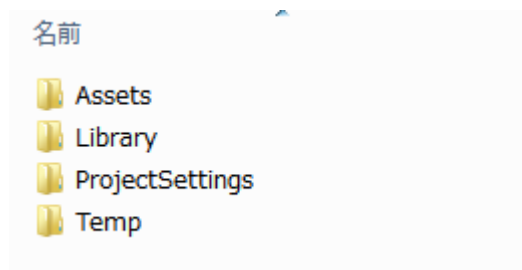
最終的にゲームに実装するのはバイナリファイルになると思われますが、開発中などテキストで扱ったほうが、都合が良い場合はXMLを使用してください。

作成したデータの確認はPC上ではUSnd一式に含まれるUSndViewerで行なえます。USndViewerを使って実機確認することは可能ですが、Android/iOS向けにビルドするにはAndroid/iOSのライセンスが必要です。音量調整などはゲーム実行中にUnityEditor上で行なうことが可能です。最終的なバランス調整は実機で行なってください。

■Unity 導入

・Unity プロジェクトのフォルダ構成

Unity はある程度フォルダの配置が決められています。プロジェクトの基本となる構成は下図のようなフォルダ構成です。既存のプロジェクトを開く場合は、Assets の置いてある階層を指定してプロジェクトを開きます。



・Unity のシーンファイル

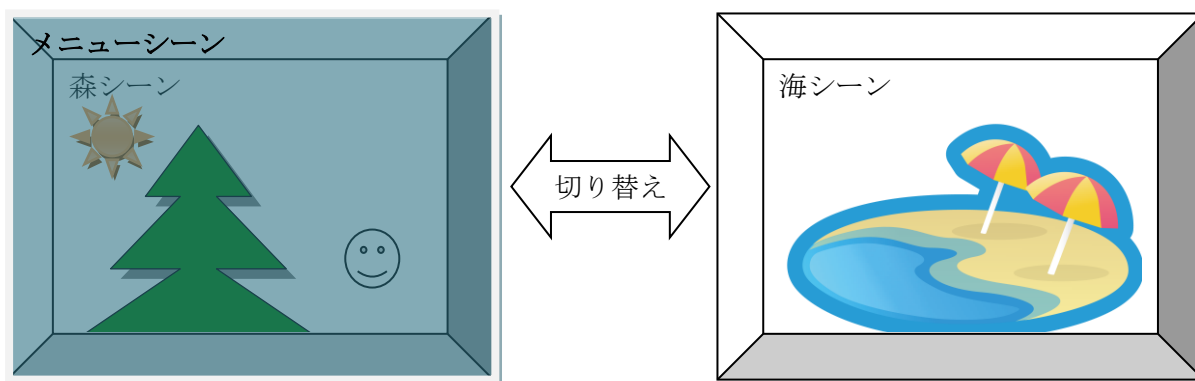
Unity は「シーン」という土台にゲームオブジェクトを配置していくことでゲームを構築します。シーンファイルは拡張子「.unity」という名前のファイルです。

例えば、森ステージ、海ステージの2種類が存在するゲームを作る場合、シーンは2種類用意します。そしてそれぞれのステージにエディタ上で必要なオブジェクトを配置していきます。

共通でメニューを表示したいのですがそれぞれのステージに組み込むと効率が悪いので、メニューも別のシーンとして持つことにします。メニューシーンは常時読み込んでおき、メニューに重ねるように森、海シーンを切り替えて表示させる、といったような仕組みで Unity はゲームを構築しています。

上記の例では場面ごとにシーンを用意しましたが、ひとつのシーンでゲームを作る場合もあります。

オーディオも他のオブジェクトと同じようにシーンの上に配置します。



・UnityEditor 上での実行の仕方

Unity は UnityEditor 上でゲームの動作チェックを行うことができます。

まずは開始したいシーンを UnityEditor の Project ウィンドウでダブルクリックして開きます。プロジェクトによりますがプロジェクト名、Base、Boot など開始するシーンにはそれらしい名前が付けられていることが多いです。

その状態で UnityEditor 画面上の再生ボタンを押すことでゲームの実行を開始します。

データが更新されると自動でビルドを行い実行できる状態かチェックするので、ビルドボタンはありません。データが不正な状態だと実行ボタンをおしてもゲームは起動しません。

・データのリロード方法

こちらでもゲームのビルド同様、UnityEditor がアクティブになると自動でロードされます。

画面右下にロード中のインジケータが表示されるので、表示が消えたら読み込み完了です。

うまく読み込めていないときは、読み込みたいプロジェクトウィンドウのオブジェクトを選択し、右クリックメニューから Refresh(Ctrl+R)すると再読み込みされます。

・Unity を扱う上での注意点

Unity は日本語ファイルの取り扱いがあまりうまくないので、フォルダ名やファイル名に日本語や全角を入れな
いください。Unity プロジェクトを配置したフォルダのパスに日本語が含まれている場合も正しく動作しま
せん。

デスクトップはあぶないので、C ドライブや D ドライブの直下に半角英数のフォルダを作り、その中に配置する
運用にしておくトラブルが少ないです。

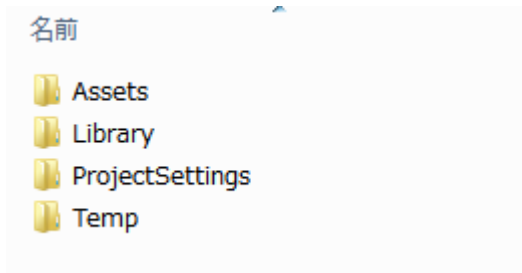
・Unity で編集した内容のセーブ方法について

Apply ボタンが付いているものは編集したら Apply を押してください。そのまま別の画面に移動すると適用する
か破棄するかダイアログボックスで確認が促されます。

Prefab など Apply がないときは UnityEditor 終了時に自動的にセーブされますが、Unity が落ちることがある
ので編集したら Ctrl+S で保存するようにしてください。

■Unity で決められているフォルダについて

Unity プロジェクトのルートフォルダは下図のように Assets フォルダのある階層です。



ゲームに組み込むリソースやスクリプトは Assets の中に配置します。命名ルールが Unity 側で設けられていて、それに従って必要なファイルを配置します。

下記のフォルダは Unity によって名前と用途が決められています。Unity 側で指定されたフォルダの中に用途にあったファイルを配置します。このフォルダの下階層に関しては、名前は指定されていないので、自由にフォルダ名を設定することができます。

フォルダ名	内容
Scripts	ゲームを構成するスクリプトファイルを配置します。 USnd フォルダはこの中に配置します。
Editor	エディタ拡張用のスクリプトファイルを配置します。ゲームのスクリプトはここには配置できません。
Plugins	Unity の機能を拡張するプラグインを配置します。スクリプトファイルと DLL を置きます。
Resources	ゲームに組み込むリソースを配置します。デザインデータやオーディオファイル、UnityEditor で作成したオブジェクトなどが含まれます。ここに置いたリソースはアプリ内蔵データとして組み込まれます。
StreamingAssets	ゲームで使用するリソースを配置します。デザインデータやオーディオファイル、UnityEditor で作成したオブジェクトなどが含まれます。ここに置いたリソースはアプリには Resources とは別の形で組み込まれます。 その他の利用法として、後で AssetBundle 化してアプリ生成時には取り除くデータなど、外部データとして使用する予定のファイルもここに置きます。

■チュートリアル – USnd 用サウンドテーブル生成

ここからは配布されているデータの準備方法と、USnd 一式に含まれる USndViewer で確認するまでの方法を説明します。

設定を記述するエクセルファイルの生成、対応付けの追加、ゲームで使用する設定ファイルの出力は **USndXML_Tools.xls** で行うことができますようになりました。

詳しくは「**USndXML_Tools 説明書**」をご確認ください。

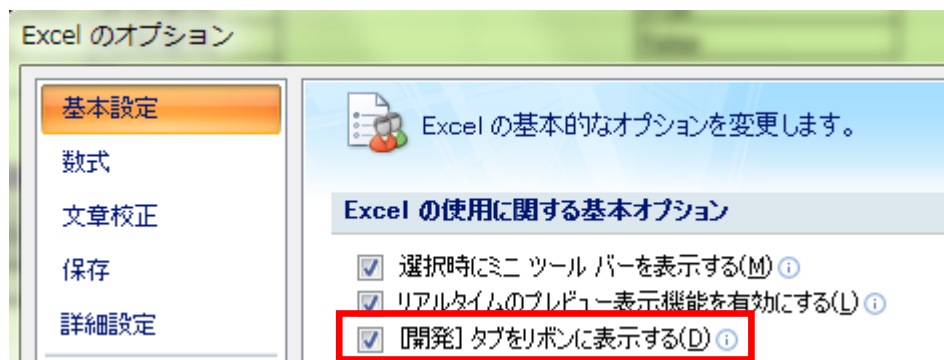
まずサウンドテーブルの準備ですが、XML フォルダ内にある **USndXML.xls** を使用します。XML は直接メモ帳などで編集することができますが、エクセルでも編集が可能です。今回はテストデータとして、**USndXML.xls** に元々記載されている記載例の内容をそのまま使います。

各項目の説明や数値範囲は後述の **AudioMasterSettings**、**AudioCategorySettings**、**AudioLabelSettings** のパラメータを参照してください。

1. XML テーブルを出力する

開発タブが表示されていない場合は左上の Office ロゴ->Excel のオプションを開いてください。

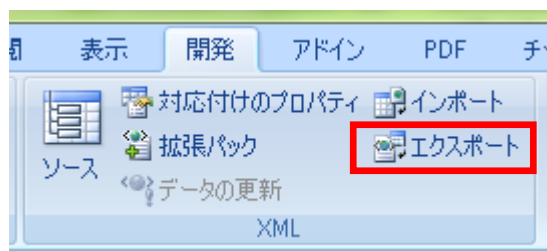
基本設定の[開発]タブをリボンに表示するにチェックをいれてください。



Master,Category シートを選択し、テーブルの適当なセルをクリックしてカーソルが置かれていた状態にします。下図のようにセルをクリックして黒い枠がテーブル上にある状態です。

MasterName	Volume
BGM	1
SE	1
VOICE	1

開発タブ->XML メニューの中からエクスポートを選択します。



保存場所とファイル名を指定すればエクスポート完了です。今回は **MasterSettings** という名前で保存します。

同シートのカテゴリテーブルと、別シートのラベルテーブルもそれぞれ **CategorySettings,LabelSettings** という名前で保存しておきます。

保存時に「対応付けを選択してください」というメッセージが表示された場合は書き出したいテーブル名が含まれている対応付けを選択します。**MasterSettings** なら **MasterSettings_**対応付けを選択します。

2. バイナリテーブルを出力する

XML かバイナリテーブルどちらか一形式を読み込めば再生が可能ですが、今回は両方作成しておきます。

「バイナリ作成」シートを開いてください。

Masterテーブル名	保存ファイル名	ID
テーブル1	MasterSettings	0

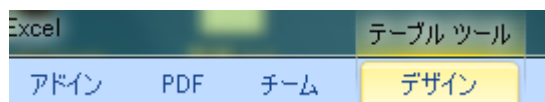
上図のようなテーブルが 3 つ並んでいます。

Master, Category, Label、3 種類のテーブル分、表が用意されています。

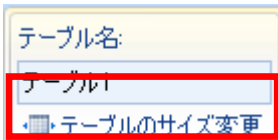
最初の列「***テーブル名」は各テーブルの、エクセル上で設定されているテーブル名を記入します。テーブル名を確認するには、Master テーブルの場合は「Master,Category」テーブルを開き、下図のようにセルをクリックして黒い枠がテーブル上にある状態にします。

MasterName	Volume
BGM	1
SE	1
VOICE	1

カーソルがテーブル上にある状態になっていると、上部メニューにテーブルツールの「デザイン」という項目が出るので選択します。



「デザイン」メニュー内、一番左にテーブル名が表示されます。テーブル名を変更することもできます。



この「テーブル 1」が現在 Master テーブルに設定されているテーブル名です。

「バイナリ作成」シートの表の Master テーブル名に「テーブル 1」を指定すると、エクセル上の「テーブル 1」が書き出し対象になります。Label など複数テーブルを用意する場合は、既にある行の下に追加して行ってください。テーブルの下への行追加は何行行なっても問題ありません。

Tips

B4 のセルから下方向にセルの内容を見て、記載があったら指定されているテーブル名の内容を調べるようになっています。そのため表の上に行を挿入したり、テーブル名の下にセルにコメントなどの文字を入力したりしないようにしてください。

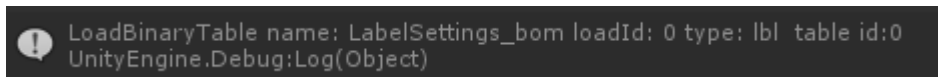
Master と Category、Category と Label の表の間も列の挿入は行なわないようにしてください。

次に保存ファイル名を指定します。現在設定されているファイル名は **MasterSettings** です。自動的に拡張子「.bytes」が付加されるので拡張子は必要ありません。

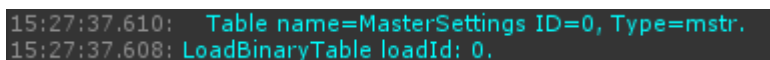
ID はテーブルに指定する任意の整数(4byte)です。プログラム中では特に使用しない値なので、使用しない場合は何も指定しなくても問題ありません。

ロード時に下記のようなログが出るので、読み込むテーブルバージョンが更新されたかどうかの目安にしてください。ID は XML には含まれないので、バイナリテーブル読み込み時のみです。

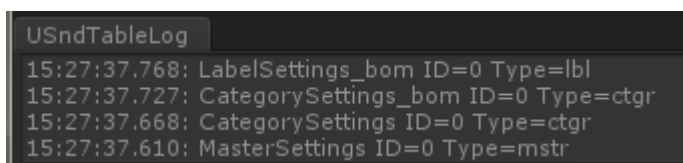
Unity の Console に表示されるログ



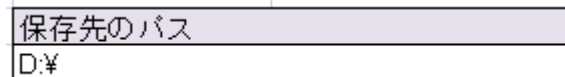
USndToolEditor に表示されるログ



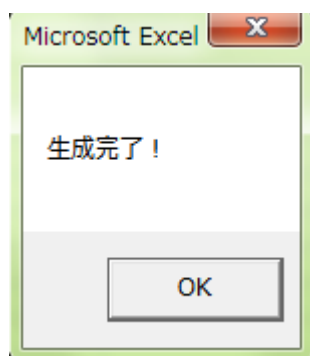
拡張メニュー->USnd->テーブルログに表示されるログ（バイナリ形式のときはファイル名、ID、ファイルタイプを表示）



以上の 3 項目をそれぞれ記入し、書き出し先を設定します。



デフォルトの指定は D ドライブの直下なので、適当なフォルダを指定してください。
設定が終わったら「バイナリファイルを生成」ボタンを押します。



上記のメッセージが表示されたら生成成功です。

作成した XML とバイナリファイルは、USndViewer/Assets の中にある Resources フォルダに配置します。XML は Resources/XML に、バイナリファイルは Resources/Binary に配置しておいてください。

以上でサウンドテーブルの生成は完了です。

■チュートリアル — AudioClip を読み込む

続いて AudioClip を読み込みます。

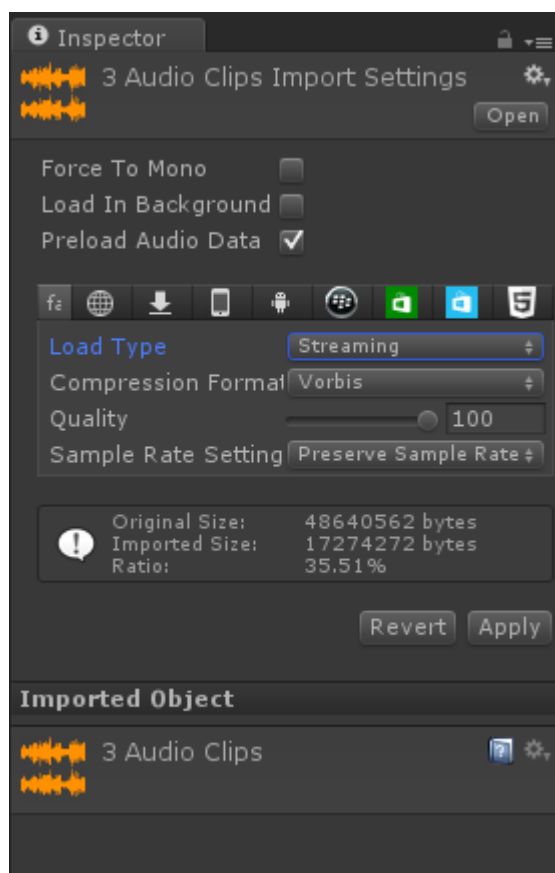
USndViewer/Assets/Resources/AudioClip にサンプルデータを配置してください。配置するのは Unity 起動中でも起動前でも大丈夫です。

起動中に配置した場合は、UnityEditor にフォーカスが戻ると自動的にロードされます。ロードが開始されない場合はメニュー->Assets->Refresh を行なってください。

UnityEditor が読み込んだ WAV ファイルがそのまま AudioClip になります。

今回のテストデータは BGM なので、ストリーミング再生にします。Ctrl を押しながら選択すると、複数選択ができます。複数選択した状態にすると、まとめて編集が可能です。

Inspector の LoadType を Streaming に変更し、Apply ボタンを押して完了です。



LoadType の種類

Decompress on load … ロード時に音声を解凍します。

Compressed in memory … メモリ内では圧縮したままにし、再生時に解凍します。

Streaming … ストリーミング再生を行います。Unity5.3.1 時点では AudioClip として読み込んだ時点でファイルが開いた状態になるため、ボイスデータなど大量のファイルを Streaming で開こうとすると OS の上限を超えたときにアプリがクラッシュします (iOS は 260 ファイルほどが上限)。デザインデータなど他のデータ読み込み時にもファイルは開くので、大量に Streaming に設定するのは避け、遅延が気にならないものは

Compressed in memory を使用するようにしてください。

その他、**Load In Background** にチェックを入れるとバックグラウンドでロードを行います。**Preload Audio Data** と一緒に使うと効果的です。

Preload Audio Data は事前にロードするタイミングをコントロールします。チェックをはずすとスクリプトからロード/アンロードのタイミングをコントロールできます。

Preload Audio Data のチェックを入れておかないと **AssetBundle** から読み込むときは **AssetBundle** を開きっぱなしにしないと鳴らないなど、制約もあるのでチームのプログラマと相談してオプションは決めてください。

以上で **AudioClip** の読み込みは完了です。

■チュートリアル — 読み込むテーブルの設定

USndViewer には必要なスクリプト、ゲームオブジェクトが配置済みです。Resources にサウンドテーブルを読み込むのに必要なデータを配置すれば動作が確認できます。

Resources の中のフォルダ名は固定になっています。

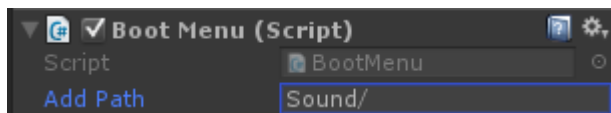
AudioClip …… 読み込むオーディオデータを配置、配置するデータは WAV 形式。

XXXBinary …… バイナリ形式のサウンドテーブルを配置、Master, Category, Label それぞれ分けておく。

XXXXXML …… XML 形式のサウンドテーブルを配置、Master, Category, Label それぞれ分けておく。

NameList …… 読み込むサウンドテーブルと、AudioClip 内のフォルダ構造を記載したリストを配置。

これらのフォルダは固定の名前です。Resources 内で一度フォルダを分ける場合、例えば Resources/Sound というフォルダの中にデータをまとめて置きたい場合は、Boot シーンの Canvas に設定されている BootMenu の AddPath に「Sound/」と入力します。



Resources 直下に AudioClip 等フォルダを配置する場合、AddPath は空欄のままです。

AudioClip 内のフォルダ構成の指定は特にないので、自由にフォルダを作成してください。

それ以外は各フォルダの直下にファイルを配置してください。

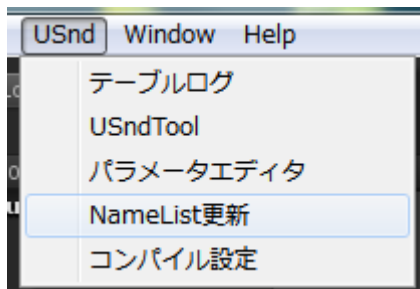
XXXBinary には Master, Category, Label のそれぞれのバイナリ形式(.bytes)のサウンドテーブルを配置します。

XXXXXML も同様に XML 形式のサウンドテーブルを配置します。一方の形式のデータのみ使用する場合は、バイナリ形式、XML 形式どちらかにサウンドテーブルが配置されていれば起動できます。

NameList には読み込むファイル名およびディレクトリ構成を記載します。

拡張メニューの USnd→NameList 更新で自動的に下記のテキストファイルを生成します。

ログに Complete と表示されれば成功です。



AddPath を追加している場合は、Assets/Editor/CreateNameList.cs に Assets/Resources 以下の必要なパスが記載されているので、正しいパスに書き直してください。

上記のように Resources 以下に Sound ディレクトリがある場合は

"Assets/Resources/Sound/NameList/MasterName.txt";

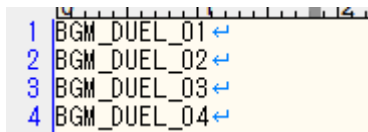
と書き直します。

生成されるテキストの内容は以下のとおりです。直接編集しても大丈夫です。

MasterName.txt … マスターが記載されているテーブル一覧を記載
CategoryName.txt … カテゴリが記載されているテーブル一覧を記載
LabelName.txt … ラベルが記載されているテーブル一覧を記載
Directory.txt … AudioClip 内のフォルダ構造を記載

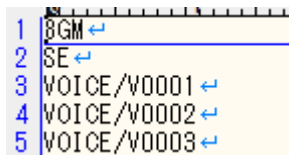
XXXName.txt にはサウンドテーブルのファイル名一覧を記載します。

SJIS(改行コード CRLF)で区切り文字なしで 1 行ずつファイル名を記載します。Windows でテキストを普通に保存すれば SJIS(改行コード CRLF)になります。



```
1 BGM_DUEL_01
2 BGM_DUEL_02
3 BGM_DUEL_03
4 BGM_DUEL_04
```

Directory.txt は AudioClip 内のフォルダ構造を、XXXName.txt と同様に 1 行ずつ記載します。



```
1 BGM
2 SE
3 VOICE/V0001
4 VOICE/V0002
5 VOICE/V0003
```

VOICE/V0001 のように、VOICE フォルダの中にさらにフォルダがある場合は「/」で区切ってオーディオデータが置いてある階層を記載します。フォルダ名の最後には「/」は入力しません。

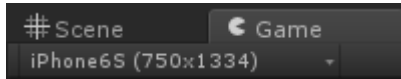
以上で読み込むテーブルの設定は完了です。

■チュートリアル — Editor 上で実行する

一通りの準備が終わったら Editor 上で動作を確認します。

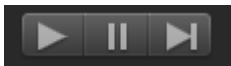
最初のシーンは「Boot」なので、Boot シーンがロードされた状態にしてください。Project ウィンドウのツリー上の Assets を選択すると、Assets の直下に Boot シーンが配置されているのが確認できます。Boot のアイコンをダブルクリックすると Boot シーンがアクティブになります。

次に実行時の画面サイズを指定しておきます。画面サイズの指定は Game タブのすぐ下にある▼を押すと選択できます。



デフォルトで上記の iPhone6S は存在しないので、▼を押して表示されたリストの一番下の「+」をクリックします。追加する画面情報を入力する UI が表示されるので、Label に表示名（上記の場合は iPhone6S）、Type は Fixed Resolution のまま変更なし、Width&Height に 750、1334 を入力して OK ボタンを押します。追加された iPhone6S を選択しておいてください。

UnityEditor の上側中央のボタンで実行・一時停止を行ないます。



一番左が再生、中央が一時停止です。一番左のボタンを押してプロジェクトを実行します。Unity は更新があるたびに自動でビルドを行なうので、ボタンを押すだけで即実行できます。

起動すると下図のような画面が表示されます。

Start

- ☒ BGM_DUEL_01
- ☒ BGM_DUEL_02
- ☒ BGM_DUEL_03
- ☒ BGM_DUEL_04
- ☒ BGM_DUEL_05
- ☒ BGM_DUEL_06
- ☒ BGM_DUEL_07
- ☒ BGM_DUEL_08
- ☒ BGM_DUEL_09
- ☒ BGM_DUEL_10
- ☒ SE_3D_SMN_01_04041
- ☒ SE_3D_SMN_02_04007
- ☒ SE_3D_SMN_03_04088
- ☐ SE_3D_SMN_04_04000

Select ON/OFF

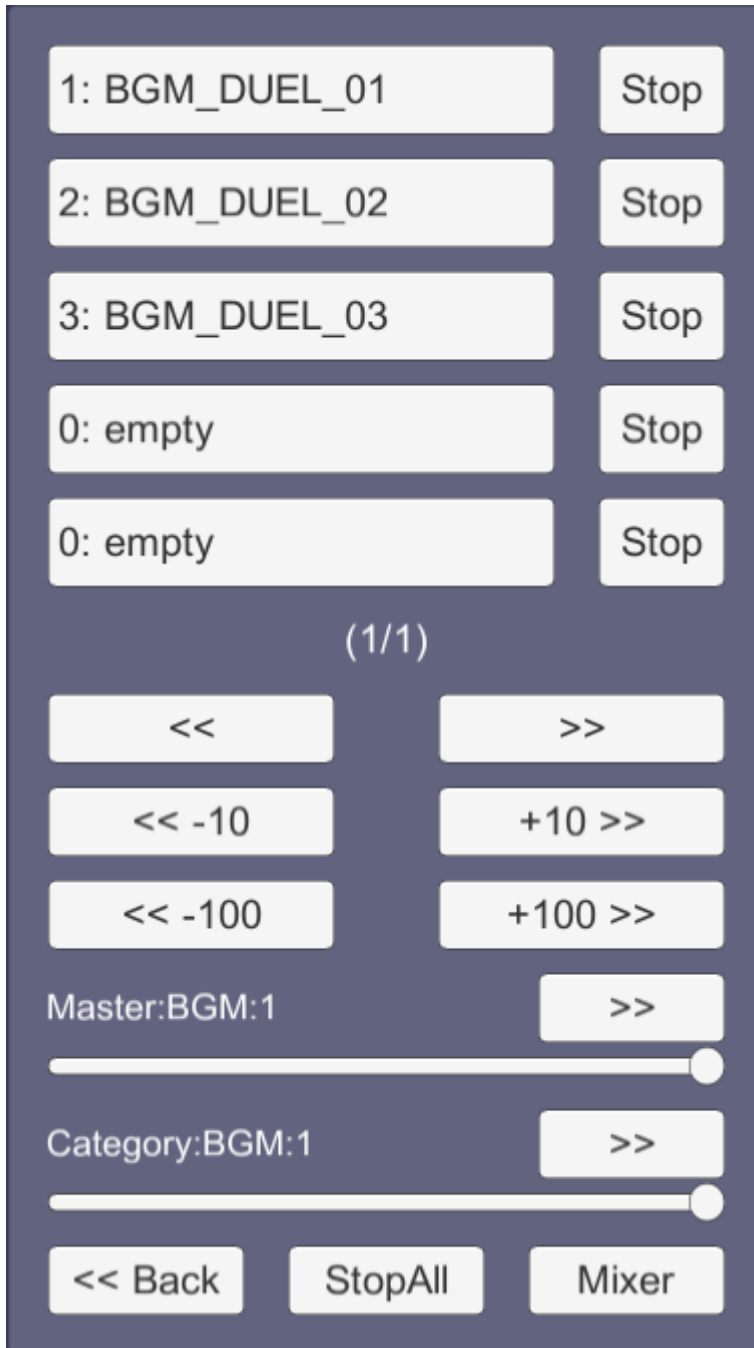
☐ LoadBinary

LabelName に指定したラベルサウンドテーブルのうち、どのファイルを読み込むか候補が表示されます。読み込みたいもののみチェックをつけてください。

上記例では、すべてにチェックが入ったまま起動した場合、上から順に BGM_DUEL_01、BGM_DUEL_02・・・と読み込まれます。既に存在するラベル名が後から読み込むテーブルに含まれていた場合、後から読み込んだテーブルで情報を上書きします。

読み込みたいテーブルを選択したら「Start」ボタンを押します。

Start を押して起動が完了すると下図のような画面が表示されます。



ラベル名の記載されているボタンを押すと再生を開始します。右側の **STOP Label** で、ラベル単位で再生を停止します。

その下の「<<」「>>」ボタンはページ送りボタンです。その下のスライダーは **Master**、**Category** のボリュームコントロールです。

StopAll ですべての再生を停止、**Back** で読み込むラベルの選択画面に戻ります。

Mixer は次の項目を参照してください。

以上で **USnd** 用のデータの準備と **USndViewer** の基本的な使い方のチュートリアルは終了です。

■AudioMixer の使い方

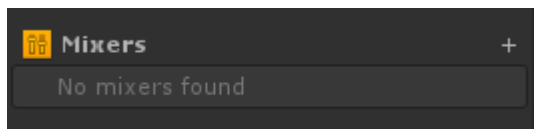
Unity5 から AuidoMixer という機能が入りました。名前のとおり **Mixer** です。

なお、AudioMixer は設定しなくても音は鳴ります。

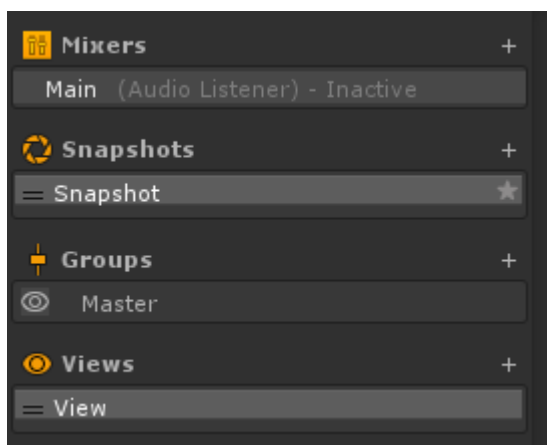
今回は Snapshot という機能をつかって、通常の状態とフィルタをかけた状態、2つの状態に切り替えるような設定の仕方を説明します。

AudioMixer ウィンドウが開いていない場合は、メニューの **Window** から **AudioMixer** を選択してウィンドウを開いてください。

Mixers と書いてある項目の横の「+」ボタンを押してください。



一番上の階層なので「Main」という名前にしておきます。

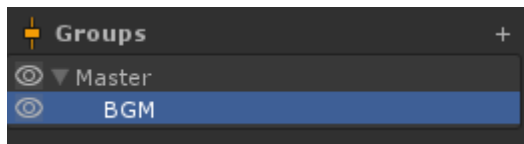


Tips

Mixers に指定した名前がそのままファイル名になります。上記のように指定した場合は **Main.mixer** が Assets 直下に作られます。

Groups の「+」を押してください。

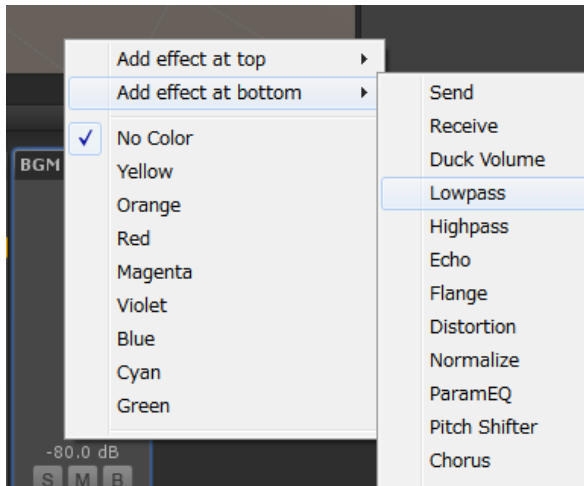
Master の下に新しいミキサーが追加されるので、名前を「BGM」にします。



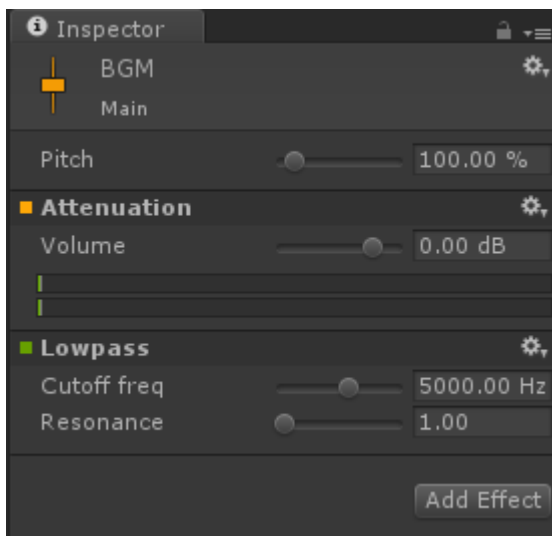
このグループ名はサウンドテーブルの **UnityMixerName** に使用します。チュートリアルで作成したデータには既にグループ名：BGM が指定されています。

次に Snapshot を試すために BGM のミキサーに Lowpass フィルタを追加します。

BGM ミキサーの上で右クリック-> Add effect at bottom->Lowpass を選択してください。



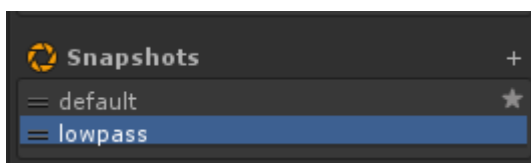
Inspector を見ると下図のように Lowpass の項目が追加されたことが確認できます。



デフォルトだと Cutoff freq が 5000Hz になっているので最大値 22000Hz に変更して掛かっていない状態にしておきます。

追加したエフェクトは Snapshot を切り替えても存在します。Snapshot はあくまでパラメータを覚えておける機能です。Snapshot を切り替えてもエフェクトの追加・削除はできません。

Mixer の項目 Snapshots に最初から追加されている「Snapshot」の名前を変更して「default」にします。Snapshots の横の「+」ボタンを押して新しい Snapshot を追加します。名前は「lowpass」にしました。

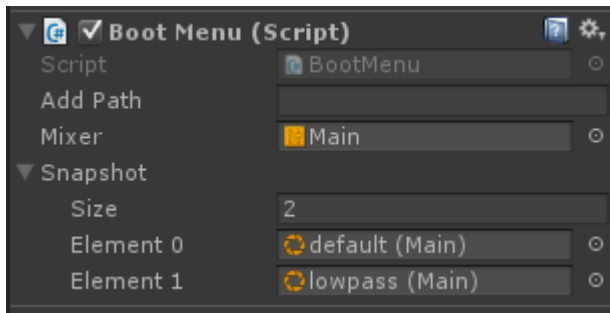


lowpass が選択されている状態で BGM ミキサーを選択し、Lowpass の Cutoff freq を分かりやすい値にします。選択中の lowpass Snapshot のパラメータが編集されます。default を選択してみると、default Snapshot の Cutoff freq は変更されていないことが確認できます。

以上でミキサーの設定は完了です。

USndViewer 上で動作を確認するには AudioManager と Snapshot を Inspector 上で設定する必要があります。ゲーム本体へ実装する際はプログラマ側で完結するので特殊な操作は必要ありません。

Boot シーンを開き、Hierarchy から Canvas を選択します。

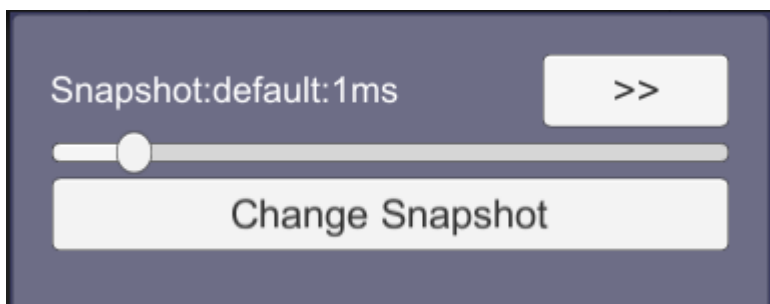


AddPath と同じく、BootMenu スクリプトに設定するようになっていました。

Mixer の UI 右側にある◎をクリックし、割り当てる AudioMixer を設定します。Snapshot の Size に切り替える Snapshot の数を入力、Element にそれぞれ Snapshot を設定します。

この状態で起動すると、指定した AudioMixer を経由して音が再生されるようになります。

Snapshot の切替を試すには、起動後のメイン画面右下の「Mixer」ボタンを押してください。



現在の Snapshot から指定した Snapshot へ、指定秒数で変化させることができます。

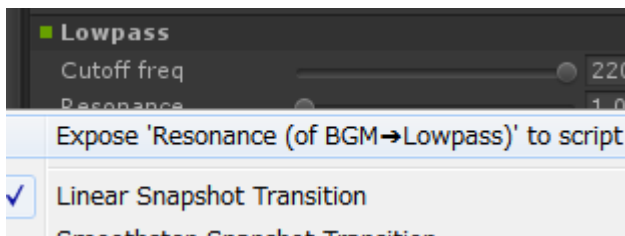
UI の使い方は Master ボリューム、Category ボリュームと同じで、「Change Snapshot」を押すと切替を実行します。

以上で実機での確認方法の説明は終了です。

最後に AudioMixer のパラメータをスクリプトから編集する方法を説明します。

AudioMixer のパラメータはスクリプトから変更することができますが、あらかじめミキサーに設定をしておく必要があります。

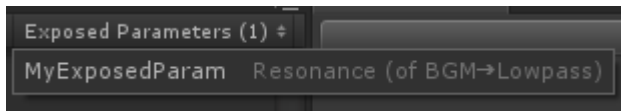
変更したいエフェクトが登録されているミキサーグループを選択し、Inspector 上でスクリプトで制御したパラメータの上で右クリックします。



上記例では Lowpass の Resonance の上で右クリックしたので「Expose 'Resonance(of BGM→Lowpass)' to script」というメニューになっています。「Expose 'Resonance(of BGM→Lowpass)' to script」を選択すると、AudioMixer タブの右側にある Exposed Parameters(0)が(1)になります。



この部分をクリックすると以下のように表示されます。



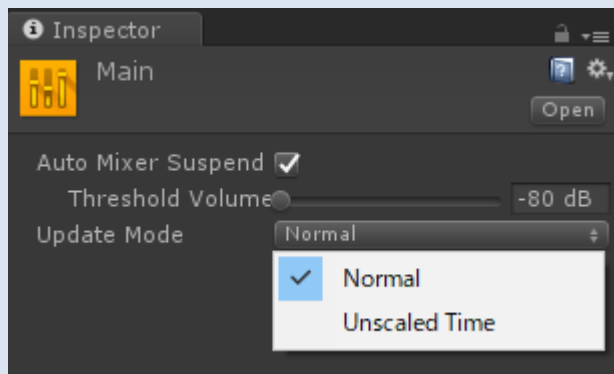
「MyExposedParam」というのがスクリプトから **Resonance** にアクセスするときの名前です。ダブルクリックすると編集モードになるので、分かりやすい名前に変更してください。例えば「**Resonance**」という名前にした場合、プログラマに「**Resonance**」という名前にしたことと、UnityEditor 上のパラメータの数値範囲 (**Resonance** は 1～10、絶対に使用しない数値範囲がある場合は 1～3 など有効にしたい範囲のみでも可) を伝えてください。

パラメータの関連付けを削除したい場合は、登録したときと同様 **Inspector** 上で右クリックしてメニューを表示します。先ほど登録メニューだった位置が「**Unexpose**」になっているので、**Unexpose** を選択すると登録が解除されます。

以上の作業でスクリプトから **AudioMixer** 上のパラメータが操作できるようになります。

Tips

ゲーム中で **Snapshot** の切り替えの速度が指定した秒数通りに動いていない場合は **Mixer** の設定を変更します。**Mixers** の項目（上記例だと **Main**）を選択し、Inspector の **Update Mode** を **UnscaledTime** に変更すると、実時間で更新されます。



■AudioMasterSettings パラメータ

項目名	説明
MasterName	[必須]マスター名を半角英数で入力します。空欄にはできません。
Volume	ボリュームを 0.0～1.0 で入力します。記載なしのときは 1 になります。

必ず最初の列が MasterName、次に Volume の順番で記載します。MasterName が重複していたときは、後から出現した MasterName の設定で上書きされます。

■AudioCategorySettings パラメータ

項目名	説明
CategoryName	[必須]カテゴリ名を半角英数で入力します。
MaxNum	カテゴリの最大発音数、0 で無制限です。空欄のときは 0 です。
Volume	ボリュームを 0.0～1.0 で入力します。空欄のときは 1 です。
MasterName	[必須]カテゴリが所属するマスター名を入力します。

必ず最初の列が **CategoryName**、他のパラメータは順番が入れ替わっていても読み込み可能です。
CategoryName が重複していた場合は、後から出現した **CategoryName** の内容で上書きされます。

■AudioLabelSettings パラメータ

項目が多いのでいくつかに分けて説明します。

XML で空欄の扱いは、数値を入れる項目は 0、文字列は未指定、true/false は false として扱います。一部例外として、IsStealOldest、AutoRestore は true、RestoreTime は-1 です。

基本情報

項目名	説明
LabelName	[必須]ラベル名です。半角英数で入力してください。
FileName	[必須]オーディオファイル名を入力します。AudioClip 名として参照するので、パス・拡張子は含まないそのままのファイル名を指定してください。
Loop	ループするときは True を、しないときは空欄か False を指定します。参照するループ情報は Windows 版 SoundForge で入力するサステインループです。リージョンは認識しません。空欄のときは False と認識されます。
Volume	ラベルボリュームです。範囲は 0.0～1.0 です。空欄のときは 1 になります。 小数点の有効範囲は指定していないので 0.00001 など有効です。 最終的な音量 = MasterVolume × CategoryVolume × LabelVolume UnityMixer を設定している場合は最終的な音量が UnityMixer に行きます。

必ず最初の列が LabelName、他のパラメータは順番が入れ替わっていても読み込み可能です。LabelName が重複している場合は、後から出現した LabelName の設定で上書きされます。

発音制御

項目名	説明
CategoryBehavior	カテゴリの最大発音数を越えたときの挙動を設定します。 STEAL_OLDEST・・・後発優先 JUST_FAIL・・・先発優先 QUEUE・・・後発優先、フェードを待ってから再生 空欄のときは STEAL_OLDEST になります。
Priority	USnd 発音制御時のプライオリティです。数値が小さいほうが優先度が高くなります。 範囲は 0～127 で、0 が最高値、127 が最低値です。 空欄のときは 64 になります。
CategoryName	[必須]Category テーブルに設定してあるカテゴリ名を半角

	英数で入力します。
SingleGroup	任意の文字列を指定します。 同じ名前が指定された異なるラベル同士、同時に発音できるのは1ラベルのみになります。このときの制御は先発優先でプライオリティは判定されません。同一ラベル内の複数発音は許可されます。
MaxNum	同ラベル内の最大発音数です。0で無制限です。(カテゴリにMAXがある場合はカテゴリに従う) Unity エディタ上の最大値は30です。 空欄のときは0になります。
Is Steal Oldest	ラベルの発音数を越えたときに後発優先にするときはTrue、先発優先のときはFalse。デフォルト(空欄)はTrueです。

再生時の Unity 設定

項目名	説明
UnityMixerName	Unity のミキサーを使用する場合、ミキサーの名前を指定します。
SpatialGroup	3D サウンドのパラメータグループ名を指定します。
Delay	再生開始までのディレイを秒数で指定します。空欄のときは0になります。
Interval	再生を行ってから次に再生可能になるまでの間隔を秒数で指定します。

再生・停止時の挙動

項目名	説明
Pan	パンを数値範囲-1(左)~0(中央)~1(右)で指定します。空欄のときは0になります。
Pitch	ピッチをセント単位で設定します。数値範囲は-1200~1200です。 空欄のときは0になります。
IsLastSamples	True のとき最後に停止した位置を保持し、次に再生開始するときに前回の位置から再生します。False のときは Pause からの復帰以外は必ず頭から再生を行います。 空欄のときは False になります。
FadeInTime	頭から再生開始するときのフェードイン時間を秒単位で指定します。 空欄のときは0になります。
FadeOutTime	停止するときのフェードアウト時間を秒単位で指定します。 空欄のときは0になります。

FadeInOldSamples	Is Last Samples が有効なとき、途中から再生開始するときのフェードイン時間を秒単位で指定します。 空欄のときは False になります。
FadeOutOnPause	ポーズ設定時のフェードアウト時間を秒単位で指定します。 空欄のときは 0 になります。
FadeInOffPause	ポーズ解除時のフェードイン時間を秒単位で指定します。 空欄のときは 0 になります。

ボリュームとピッチのランダム設定

項目名	説明
IsVolRnd	ボリュームのランダム再生を行うときは True 。 空欄のときは False になります。
IncVol	前回のランダム結果とかぶらないようにしたい場合は True 、完全ランダムの場合は False 。 空欄のときは False になります。
VolRndMin	ランダムボリュームの取る最小値を指定します。 空欄のときは 0 になります。
VolRndMax	ランダムボリュームの取る最大値を指定します。 空欄のときは 0 になります。
VolRndUnit	ランダム値の粒度を指定します。 0.2 を指定した場合、 0.2 の倍数がランダム値になります。 空欄のときは 0 になります。
IsPitchRnd	ピッチのランダム再生を行うときは True 。 空欄のときは False になります。
IncPitch	前回のランダム結果とかぶらないようにしたい場合は True 、完全ランダムの場合は False 。 空欄のときは False になります。
PitchRndMin	ランダムピッチの取る最小値を指定します。 空欄のときは 0 になります。
PitchRndMax	ランダムピッチの取る最大値を指定します。 空欄のときは 0 になります。
PitchRndUnit	ランダム値の粒度を指定します。 空欄のときは 0 になります。

パンとオーディオソースのランダム設定

項目名	説明
IsPanRnd	パンのランダム再生を行うとき True 。 空欄のときは False になります。
IncPan	前回のランダム結果とかぶらないようにしたい場合は True 、

	完全ランダムの場合は False 。 空欄のときは False になります。
PanRndMin	ランダムパンの取る最小値を指定します。数値範囲は-1(左)～0(中央)～1(右)です。 空欄のときは 0 になります。
PanRndMax	ランダムパンの取る最大値を指定します。数値範囲は-1(左)～0(中央)～1(右)です。 空欄のときは 0 になります。
PanRndUnit	ランダム値の粒度を指定します。 空欄のときは 0 になります。
IsRndSrc	ランダム再生を行うとき True 。 空欄のときは False になります。
IncSrc	前回のランダム結果とかぶらないようにしたい場合は True 、 完全ランダムの場合は False 。 空欄のときは False になります。
RndSrc	ランダム再生する他のラベル名を指定。 セル内で改行してランダム再生するラベル名を入力。カンマ など区切りは不要です。 ランダム候補にはラベル自体を含むので、このセルに2つの ラベル名を指定した場合、このラベルを含めた3つの候補か ら再生されます。

再生時にピッチ、パンを動かす設定

項目名	説明
IsMovePitch	再生開始時に指定した秒数で開始位置から到達位置までピ ッチを移動します。 空欄のときは False になります。
PitchStart	開始ピッチ位置 空欄のときは 0 になります。
PitchEnd	到達ピッチ位置 空欄のときは 0 になります。
PitchMoveTime	開始位置から到達位置まで数値が変化するのにかかる時間 を秒数で指定。 空欄のときは 0 になります。
IsMovePan	再生開始時に指定した秒数で開始位置から到達位置までパ ンを移動します。 空欄のときは False になります。
PanStart	開始パン位置、数値範囲は-1(左)～0(中央)～1(右)です。 空欄のときは 0 になります。
PanEnd	到達パン位置、数値範囲は-1(左)～0(中央)～1(右)です。

	空欄のときは 0 になります。
PanMoveTime	開始位置から到達位置まで数値が変化するのにかかる時間を秒数で指定。 空欄のときは 0 になります。

ダッキング関連の設定

項目	説明
DuckingCategory	ラベルの再生開始時にダッキングするカテゴリ名を指定します。複数カテゴリがあるときはカンマなどで区切らず改行して記入してください。
DuckStart	ダッキング開始時のフェードアウト時間を秒数で指定。 空欄のときは 0 になります。
DuckEnd	ダッキング終了時のフェードイン時間を秒数で指定。 空欄のときは 0 になります。
Duck Vol	一番ボリュームがさがっているときのカテゴリ音量係数を指定。数値範囲は 0～1 です。 空欄のときは 0 になります。 ダッキング対象の最終ボリューム = $\text{MasterVolume} \times \text{CategoryVolume} \times \text{LabelVolume} \times \text{DuckVol}$
AutoRestore	ラベルが停止したとき自動的に復帰させる場合は True 。手動で復帰させる場合は False 。デフォルト(空欄)は True です。
RestoreTime	Auto Restore Ducking が有効なとき、指定した秒数が経過したらダッキングを解除します。デフォルト(空欄)は -1 で、-1 を指定すると停止したときに復帰します。
IsAndroidNative	Android のとき SoundPool で再生を行いたいときは true 、デフォルト (空欄) は false です。 SoundPool を使用するときは短いワンショットの SE のみ再生するようにしてください。 この機能は基本使用しません。どうしても遅延をなくさないと困るときに使用してください。

■データ更新時の影響範囲

AudioMixer のパラメータを更新

MIXER ファイル(.mixer)のみ更新します。

シーン上のオブジェクトを追加・更新

シーンファイル(.unity)が更新されます。

Prefab を編集

Prefab ファイル(.prefab)のみ更新されます。

AudioClip のパラメータを変更

変更した AudioClip のメタファイル(.meta)のみ更新されます。

複数の AudioClip を一つの AssetBundle にしている状態で、1 つの AudioClip を更新したとき

変更したデータがひとつでも、その AssetBundle に含まれているデータはすべて再 AssetBundle 化されます。

10 個のデータを含む AssetBundle のうち 1 個を更新したので差分の 1 個のみ配信、ではなく、10 個分すべてのサイズが配信対象になります。

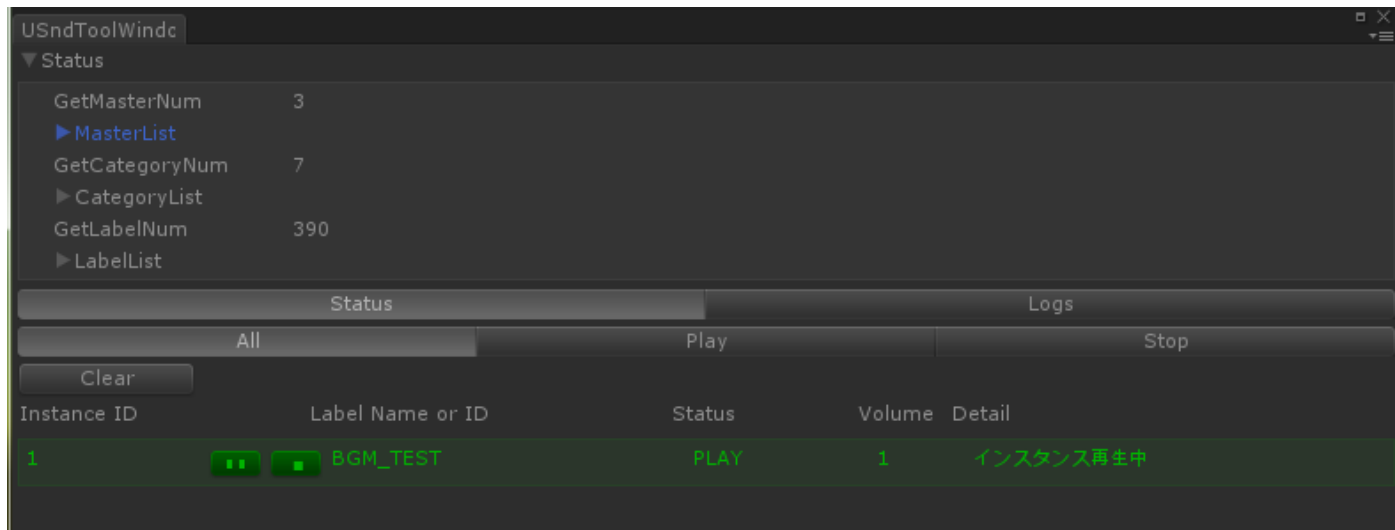
■サウンドデバッグツール

実行中に拡張メニューの USnd->USndTool を選択すると、下図のようなウィンドウが開きます。

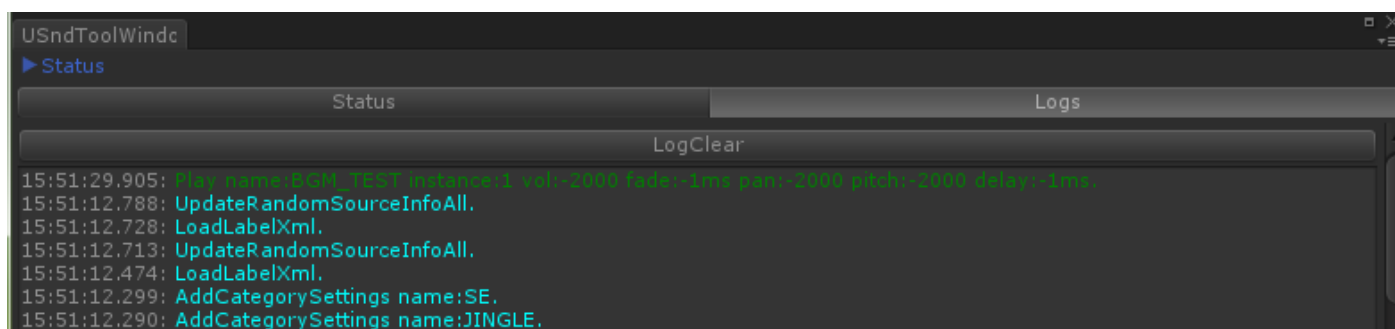
現在読み込んでいる Master、Category、Label の情報と、現在再生中のインスタンス情報が表示されます。

MasterList、CategoryList、LabelList に表示されるのは現在ロード済みの情報です。

LabelList 上のボタンで再生確認することもできます。



Logs タブに切り替えると、USnd のシステムログが表示されます。



再生中のインスタンス情報とログは新しいものが上に追加されていきます。

ログは一部の API のみ出力を行っているので、ほしい情報がログに出ていない場合はご相談ください。ログは Unity 上で表示されるのは 15000 文字前後までです。Copy ボタンを押すと AudioDefine.cs に定義されている LOG_MAX 行数までは保存されているので、目当てのログが消えていたら Copy→テキストエディタに貼り付けして確認してください。

表示されないときは USnd->コンパイル設定を開き、USND_EDIT_MODE にチェックが入っているか確認してください。USND_EDIT_MODE にチェックが入っていると USndTool の処理を行ないます。また、USND_DEBUG_LOG にチェックが入っていると Console に USnd のログが表示されます。USndTool のログは USND_DEBUG_LOG が無効でも表示されます。

■XML を使ったパラメータ更新の注意点

配信後にアプリ更新なしでデータのみアップデートするといったことが必要になってきます。

XML を読み込んだときに同じ名前のラベルが存在する場合、新しい XML から読み込んだパラメータで既存のラベル内容を更新することができます。

この機能を使うと後からパラメータを変更することができます。このとき、指定のないパラメータは元の状態から変更されません。

例えば、元々のラベルのピッチが 100 だった場合、新しい XML でピッチの指定を空欄にしているときはパラメータの更新が行われないのでピッチは 100 のままになります。通常、ピッチのデフォルト値である 0 にするときは空欄でも問題ありませんが、指定済みのパラメータを上書きして 0 に戻したい場合は、明示的に 0 を指定する必要があることに注意してください。

■パラメータエディタの使い方

拡張メニューの USnd->パラメータエディタを選択すると、パラメータエディタが開きます。

Master、Category、Label の現在の設定の確認と、実行中にパラメータを編集して試すことができます。

USndViewer を再生画面まで立ち上げると下図のような表示になります。



左側のビューで表示したい項目を選択すると右側にパラメータが表示されます。

パラメータを変更すると、変更した内容で再生を試すことが出来ます。

「表示方法を変更する」ボタンを押すと、左側のビューが画面上部に折りたためる形で配置され、その下に詳細画面が表示されます。使いやすいレイアウトを選択してください。

変更したパラメータは「選択中の項目をクリップボードにコピー」を押すと、選択中の項目だけテキスト形式でコピーされます。上図の場合は **BG_MENU** がコピーされます。

タブ区切りで項目を並べたテキストがコピーされるので、エクセルを開き変更した項目名の上にカーソルを置いて **Ctrl+C** でコピーすると、そのまま貼り付けができます。

「すべての項目をクリップボードにコピー」を行なった場合、全項目分の貼り付けが可能です。

実行中に変更した内容は、Unity の実行を停止すると消えてしまうので注意してください。

■3D サウンドの設定方法

JSON 形式で 3D サウンドパラメータが読み込み可能になったので、Unity の AudioSource 上で編集したパラメータを JSON 形式で出力、読み込みが可能になりました。

下記記載の従来の方法と平行して使用可能です。詳しくは次の項目「JSON 形式で 3D サウンドのパラメータ設定を行う方法」を確認してください。

3D サウンドは Unity 標準のパラメータ設定は使用しません。

ラベルの SpatialGroup に指定した名前の Audio3DSettings の設定に従い、3D サウンドパラメータが設定されます。1 ラベルごとに 3D サウンドパラメータを設定するのではなく、SpatialGroup の単位でパラメータを設定するようになっています。

ラベルへの設定は SpatialGroup に任意の名前を設定することで、3D サウンドパラメータが有効になります。サウンドテーブル側の 3D サウンド指定は以上です。

次に Audio3DSettings パラメータを設定する方法を説明します。USnd が組み込まれているプロジェクト上でメニューの Assets→Create→USnd→Create Audio3DSettings Instance を実行し、Audio3DSettings のインスタンスを生成します。Project ウィンドウで選択しているフォルダの直下に生成されるので、適当なフォルダを選択してから実行してください。

作成すると下図のようなアイコンのアセットが生成されます。



作成されたアイコンを選択すると Inspector にパラメータが表示されます。

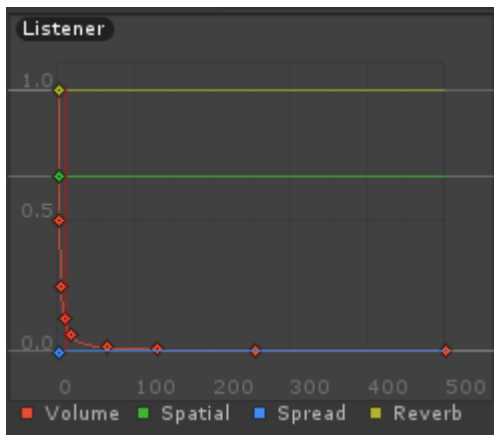


これが 3D サウンドで指定できるパラメータです。AudioSource の 3D サウンドパラメータと同じものです。

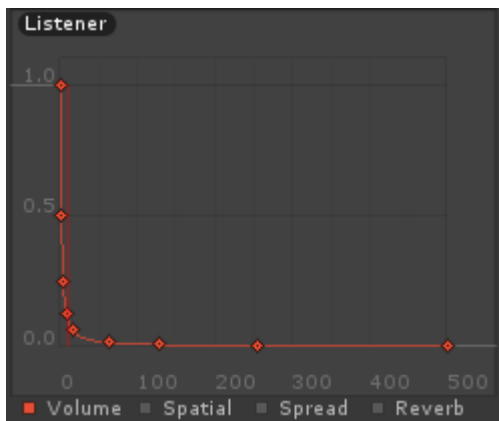
参考：<http://docs.unity3d.com/ja/current/Manual/class-AudioSource.html>

項目名	説明
Name	SpatialGroup に指定した文字列を入力します。
Spatial Blend	2D サウンドと 3D サウンドのミックスバランスです。1 で完全な 3D サウンドになり、0 に近づくほど 2D サウンドと混ざった状態で再生されます。0 で完全に 2D サウンドになります。
ReverbZoneMix	リバーブゾーンに送られるボリュームを指定します。
DopplerLevel	ドップラー効果の掛かる度合いです。0 を指定するとエフェクトが掛かりません。
Spread	スピーカー空間で 3D サウンドに対する広がり角度を指定します。
RolloffMode	減衰カーブの設定で、Logarithmic と Linear を設定すると Unity が指定したグラフで減衰します。Custom を指定すると後述の Custom Rolloff Curve に従います。
Min Distance	減衰カーブが適用される最短距離で、例えば 1m を指定すると 1m 以内にある音源には減衰は適用されません。1m 以上はなれると、グラフの設定に従って減衰を始めます。
Max Distance	減衰カーブが適用される最大距離で、例えば 1m を指定すると 1m 以上はなれた音源には適用されず音量が一定になります。1m 以下に近付くと、グラフの設定に従って音量が大きくなります。
Custom Rolloff Curve	Rolloff Mode で Custom を設定した場合に有効になります。音量の距離減衰カーブを自分で設定するためのパラメータです。
Spatial Blend Curve	SpatialBlend パラメータを距離に応じて変化させたいときに設定します。グラフのポイントが 2 つ以上のとき Spatial Blend Curve の設定が有効になります。
Reverb Zone Mix Curve	ReverbZoneMix パラメータを距離に応じて変更させたいときに設定します。グラフのポイントが 2 つ以上のとき Reverb ZoneMix Curve の設定が有効になります。
Spread Curve	Spread パラメータを距離に応じて変更させたいときに設定します。グラフのポイントが 2 つ以上のとき Spread Curve の設定が有効になります。

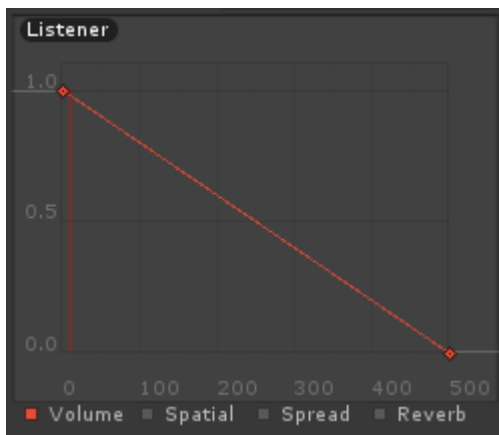
CustomRolloffCurve(赤)、SpatialBlendCurve(緑)、ReverbZoneMixCurve(黄)、SpreadCurve(青)の AudioSource 上のグラフは下図のように表示されます。USnd ではこのグラフ自体は編集できず、個別のグラフを編集しますが最終的には再生中の AudioSource を選択するとこのようなグラフで表示されます。



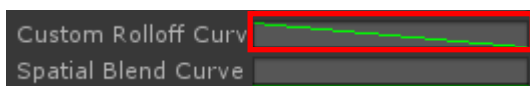
RolloffMode で Logarithmic を選択すると下図のようなカーブになります。



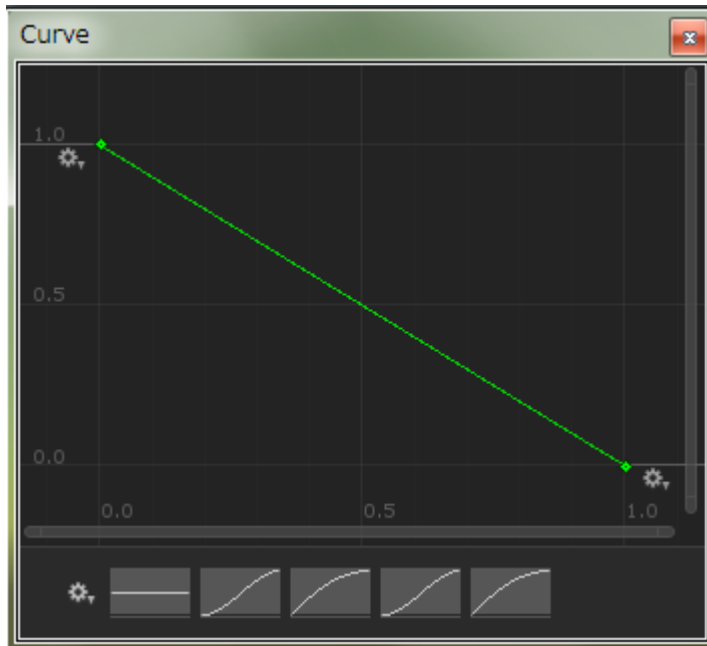
RolloffMode を Linear にすると下図のようなカーブです。



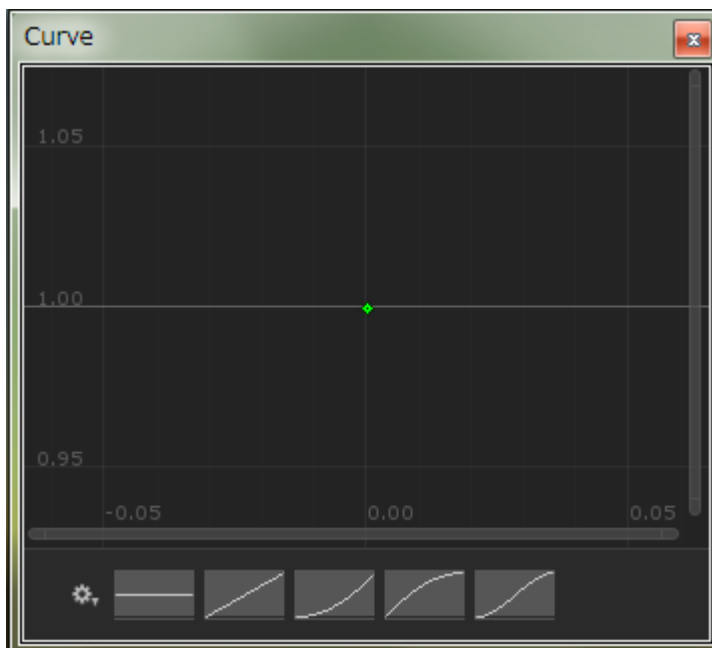
自分でカーブを作る場合は Audio3DSettings を選択し、Inspector 上の緑色の線が描画されている部分（赤枠の部分）をクリックします。



クリックすると、下図のようなウィンドウが開きます。



このグラフの X 軸 0.0～1.0 が MinDistance～MaxDistance の距離に相当します。Y 軸は 0.0 で無音～1.0 で Max 音量です。緑の戦場をダブルクリックするか右クリックで **AddKey** を選ぶとキー（緑の点）が追加できます。**CustomRolloffCurve** の場合は最初から 2 つのキーが指定されていますが、他の項目はそれぞれの項目のデフォルト値（Spatial と ReverbZone が 1、Spread は 0）に 1 つだけキーが設定されています。



グラフ下のデフォルトのグラフを選ぶか、適当な場所をダブルクリックまたは緑の点から左右に伸びる灰色のライン上を右クリックして **AddKey** を押すと 2 つ目のキーが追加できます。デフォルトの描画では最初に置いてあるポイントが中央になってしまうため、グラフの数値範囲が中途半端になっていますが X 軸 0～1.0、Y 軸 0～1.0(ReverbZoneMix は 1.1)の範囲にグラフを設定してください。マウスホイールでグラフの拡大縮小が行なえます。多少範囲外になっても数値範囲以上・以下にはならないのでぴったり合わせる必要はありません。キーの上で右クリック→**EditKey** で数値を直接入力することもできます。

CustomRolloffCurve ではなく **Logarithmic** または **Linear** グラフが使いたい場合は **RolloffMode** を変更してください。

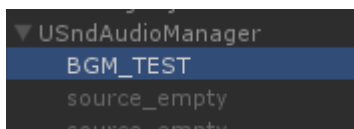
その他のグラフを無効にしたい場合は、グラフ上のキーを削除して一つだけキーが残っている状態に戻してください。

なお、リスナーの位置は Unity のデフォルトでメインカメラになっています。リスナーの位置を変更したい場合はチームのプログラマに相談してください。

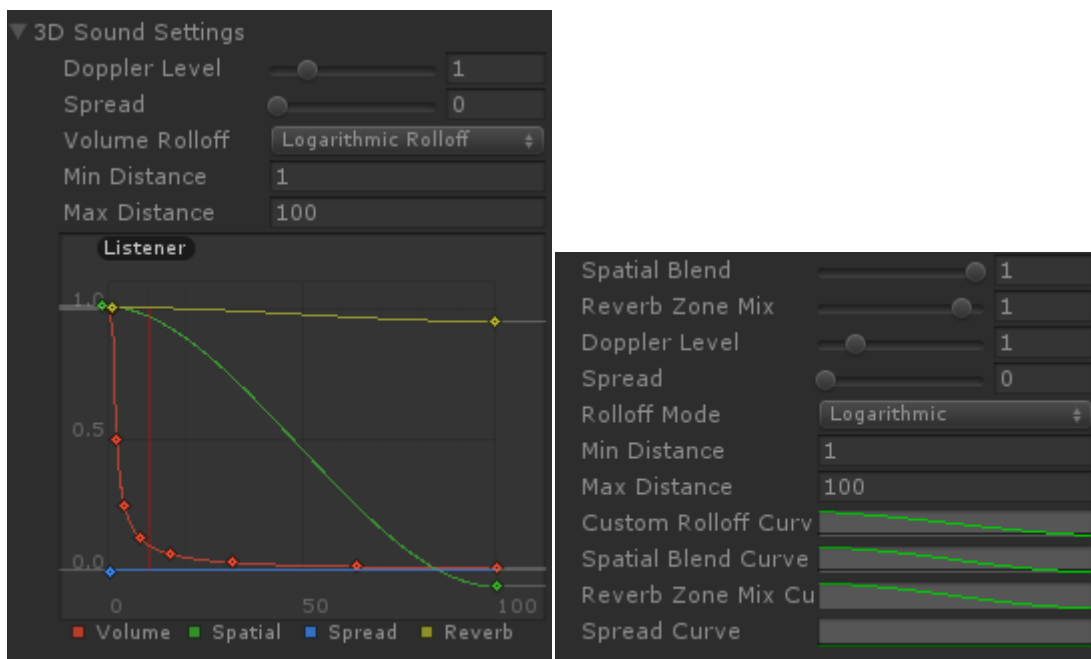
3D サウンドのパラメータ設定は以上です。

次にパラメータの調整の仕方について説明です。環境音などループする音源の場合、再生中の **AudioSource** パラメータを変更すると鳴らしながら調整ができます。

ゲーム実行中、**Hierarchy** の中に **USndAudioManager** というオブジェクトがあるので畳んである内容を開きます。再生ポートとして用意された **AudioSource** オブジェクトが子になっています。再生を開始すると割り当てられた **AudioSource** にラベル名が設定されるので該当 **AudioSource** を選択します。



Inspector を開き、3D Sound Settings に指定した **SpatialGroup** のパラメータが反映された状態が表示されていることを確認してください。



データがループしている場合は **Inspector** 上で 3D Sound Settings を変更すると、変更した状態でどんな鳴り方になるか確認ができます。ただし、**Audio3DSettings** には反映されないので注意してください。

ワンショットや、ループ音を鳴りなおしさせて確認する場合は、**Audio3DSettings** を変更します。

AssetBundle に含まれている場合は確認できませんが、**Resources** 以下にある場合は **Project** のツリーから該当の **Audio3DSettings Asset** を探して選択し、**Inspector** 上で編集してください。

次に再生されるときに新しいパラメータが反映されます。変更したパラメータは実行を停止しても保持されるので、試しに変更して前の状態に戻したいときはバックアップを取るなどしてから変更してください。

以上で調整方法の説明は終了です。

■JSON 形式で 3D サウンドのパラメータ設定を行う方法

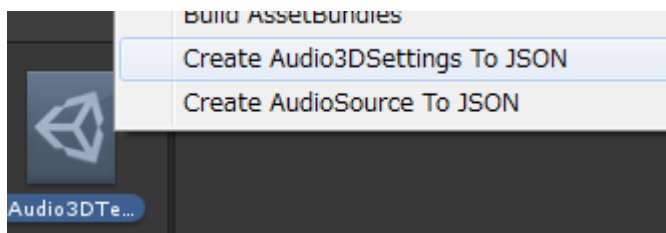
3D サウンドパラメータを JSON 形式で出力する方法は、前項の `Audio3DSettings` のインスタンスから作成する方法と、`AudioSource` から作成する方法があります。

Audio3DSettings のインスタンスから JSON を生成する

上記項目で作成したパラメータを右クリックして「Create Audio3DSettings To JSON」を選びます。

保存先のフォルダを選択するウィンドウが表示されるので、適当な場所を選択してください。ファイル名は選択したインスタンスと同じファイル名.json になります。

メニューに項目が表示されない場合は、Editor フォルダに `CreateAudio3DSettingsJson.cs` が存在するか確認してください。ファイルがない場合は Assets 以下に Editor フォルダを作成し、`CreateAudio3DSettingsJson.cs` をコピーしてください。

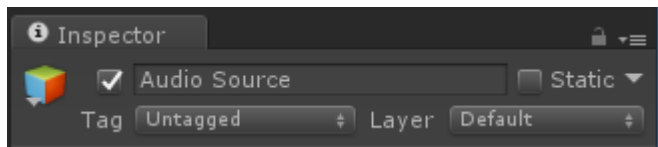


AudioSource から JSON を生成する

`AudioSource` から JSON を生成する場合は、`AudioSource` 自体はゲームプロジェクトには必要ないファイルなので、適当な Unity プロジェクトを作成してください。

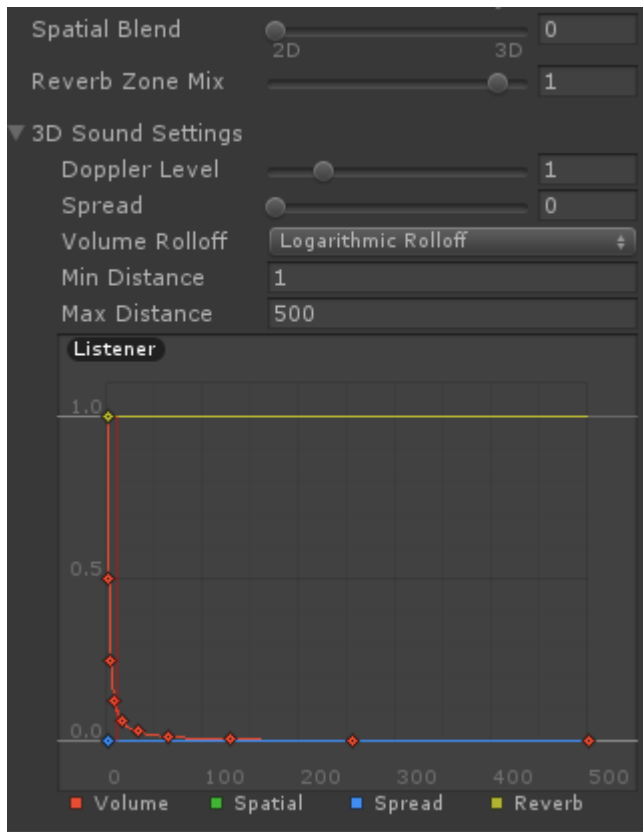
プロジェクトを作成したら、メニューの `GameObject`→`Audio`→`AudioSource` を選択して新しい `AudioSource` オブジェクトを作成します。

Hierarchy 上に新しく「Audio Source」が追加されるので、選択して Inspector を編集します。

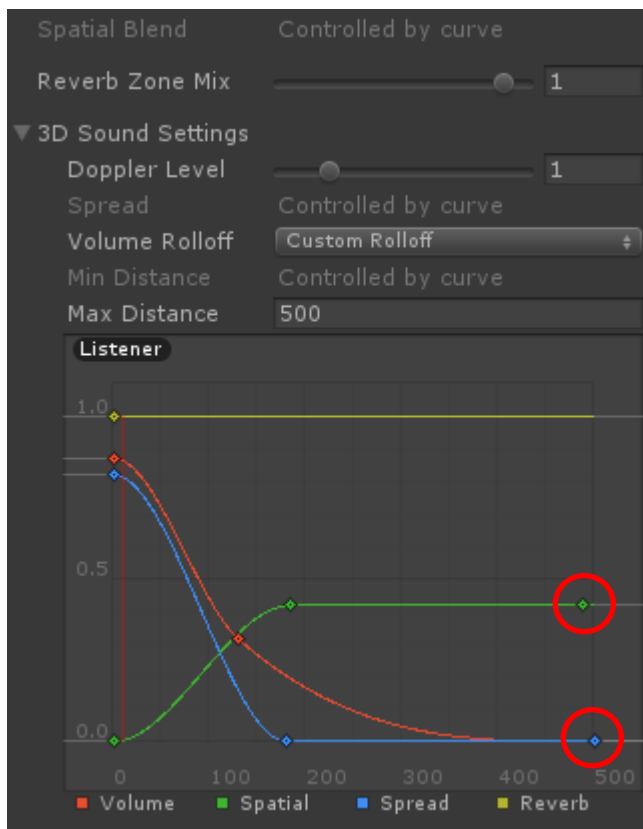


「Audio Source」と文字が入力されている箇所が `Audio3DSettings` の Name になります。3D パラメータの名前を入力してください。Unity 上では半角スペースは使用可能ですが、`Audio3DSettgins` の名前は半角スペースは使用できません。

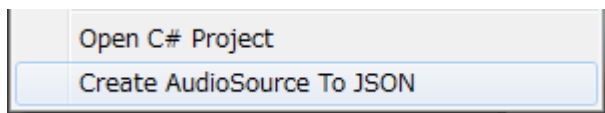
パラメータが複数ありますが、使用するパラメータは `Spatial Blend` 以下の部分です。



グラフの線上をダブルクリックするとポイントが追加できます。
 グラフを編集する際は、終端にもポイントを置くようにしてください。下図の赤丸で囲ったポイントがないと、直前のポイントの値が正しく反映されない場合があります。



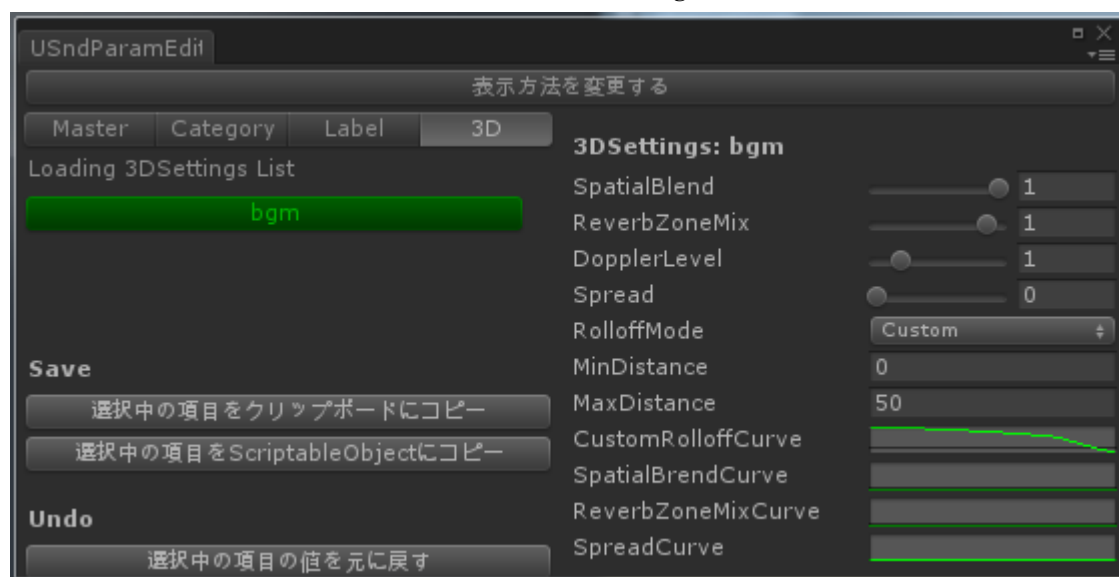
編集が終わったら、Hierarchy 上で出力する AudioSource が選択されていることを確認し、上部メニューの Assets→Create Audio Source To JSON を選択します。



Audio3DSettings インスタンスのときと同様、フォルダを選択するダイアログが開くので、保存先のフォルダを選択します。フォルダを選択すると AudioSource オブジェクトに設定された名前.json ファイルが生成されます。

■3D サウンドパラメータを実行中に変更する方法

ver2.12.0 からパラメータエディタで Audio3DSettings の値を実行中に変更できるように機能を追加しました。

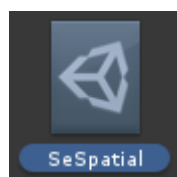


パラメータエディタの左上タブの「3D」を選択すると登録されている Audio3DSettings 一覧が表示されます。ボタンに SpatialName に設定した値が表示されます。編集したい SpatialName を選択すると、右側に現在のパラメータが表示されます。

UI 上のパラメータを変更すると、実行中の 3D サウンド設定も変更されます。再生中でもリアルタイムに変更することができます。

変更した値は実行中のパラメータを上書きしているだけなので、そのまま実行を停止すると変更は消えてしまいます。

上記「3D サウンドの設定方法」で紹介した Audio3DSettings のオブジェクトを Resources などから読み込んでいる場合（以下のようなアイコンが Resources などにある場合）は、



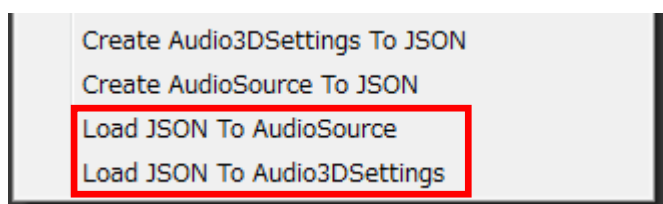
左側 Save メニューの「選択中の項目を ScriptableObject にコピー」ボタンを押すと、値がオブジェクトに格納された状態になります。一度ボタンを押すと Undo は効かなくなるので注意してください。

JSON から読み込んでいる場合は「選択中の項目をクリップボードにコピー」を押します。ボタンを押すと JSON 形式でパラメータがクリップボードにコピーされるので、JSON ファイルに貼り付けて上書きしてください。

パラメータの変更を元に戻すときは Undo メニューの「選択中の項目の値を元に戻す」をクリックします。

■JSON 形式をインポートしてパラメータを変更する

Editor フォルダに LoadJsonToAudioSource.cs を置いておくと、JSON 形式を読み込んで AudioSource または Audio3DSettings オブジェクトにパラメータを読み込むメニューが、メニューの Assets と右クリックメニューに追加されます。



インポート先の AudioSource を選択した状態で Load JSON To AudioSource を選択すると JSON ファイルを指定して、選択した AudioSource に JSON の内容を読み込みます。

Audio3DSettings に読み込みたいときは、Load JSON To Audio3DSettings を選択します。

メニューを選ぶと JSON ファイルを選択するウィンドウが開きます。JSON ファイルに記載されている spatialName と選択したオブジェクトの spatialName (AudioSource の場合はオブジェクト名) が一致する場合、パラメータを上書きします。

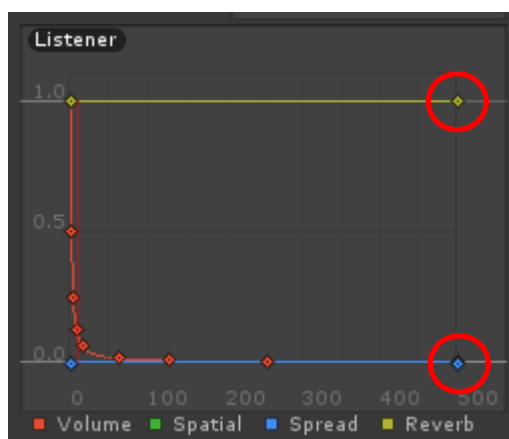
実行中にパラメータエディタで値の変更をしたときなど、生成元のオブジェクトに変更後の値を反映させるときに使用してください。

使用時の注意点としては、AudioSource へパラメータを読み込むとき、カーブのパラメータにポイントが 2 つセットされていないとパラメータの変更ができなくなるので注意してください。

Audio3DSettings から JSON を生成するなど、今までの USnd バージョンでは CustomRolloffCurve はデフォルトがリニアグラフになっていますが、それ以外の 3 種類のカーブは点になっていました。AudioSource のデフォルト値とあわせてある状態ですが JSON→AudioSource に読み込みを行うと値が正しく反映されなくなるため、ver2.12.0 から Audio3DSettings オブジェクトを生成するときは自動的にポイントを 2 つ入れて直線のグラフが入るよう変更しました。

Ver2.12.0 以降の Audio3DSettings のデフォルトをそのまま AudioSource に反映させると以下のようにポイントが追加されます。

AudioSource を元に JSON を作成している場合、JSON→AudioSource の反映を行うときに図のようにポイントが入っていないと値が AudioSource 上で変更できなくなるので、JSON から AudioSource へインポートを行いたい場合はグラフにキーを二つ置くようにしてください。



■Streaming 設定時の注意点

Unity5.3.1 時点の仕様では AudioClip の LoadType に Streaming を設定しているときの挙動は以下のようになります。

AudioClip の Preload Audio Data 設定が有効なときの注意点

- ・ 1 ファイルあたりストリーミング用のメモリとして非再生時でも 158KB を消費する
- ・ AudioClip 化した時点でファイルを開くため、大量に AudioClip を読み込むと OS の上限をオーバーすることがある

AudioClip の Preload Audio Data 設定が無効なときの注意点

- ・ AssetBundle から Load を行なうとき、AudioClip を取得した後 AssetBundle.Unload(false)を実行すると Preload 時にファイルが開けなくなり再生失敗する

実装時は以下のことを注意してください。

- ・ Preload を指定した場合、AudioClip 化するファイル数が多くなりすぎないようにロードしておくデータを制限する
- ・ 長さの短いファイルは Compress In Memory を使用し、Streaming にしないようにする
- ・ Compress In Memory では厳しい場合は、Streaming に指定した上で Preload の指定を外す。その際 AssetBundle から読み込むときは Unload(false)を行なわないようにする

通常 AssetBundle からデータを読み込んだ後は、AssetBundle.Unload(false)を呼ぶ実装なので、Preload を外した Streaming 設定の AudioClip の実装だけ扱いが異なります。

AudioClip 用に処理を追加してもらう必要があるので、Preload を使うかどうかはプログラマと相談して設定するようにしてください。

Preload Audio Data を無効にするときは、以下のことに注意してください。

- ・ Preload を外した Streaming 設定のデータをランダム再生するときは、ランダム指定したオーディオが複数ラベルからコールされないようにしてください。

→Unity5.3.5p6 から Streaming は強制的に PreloadAudioData が無効な状態になりました。

■Android で SoundPool を使用して再生する方法

Unity は他のプラットフォームに比べると Android 上では発音遅延が大きいです。音楽ゲームのタップ音など遅延が気になる使い方をされる SE は UnityAPI で再生を行うと違和感があります。

そういった場合、JavaAPI の SoundPool を使用して再生することで、遅延を軽減することができます。

USnd ではサウンドテーブルの IsAndroidNative フラグを有効にすることで SoundPool を使用して再生を行うことができます。

ただし、SoundPool を使用したときは再生中の音量、パン、ピッチの変更、再生中か判定、停止、一時停止/再開ができません。短いワンショットの SE のみ SoundPool で再生するようにしてください。

デフォルトの SoundPool の同時発音数は 6 になっています。変更は可能ですが、この程度で間に合う SE を再生するための機能だと思ってください。

SoundPool を使って再生を行う場合、使用できるオーディオフォーマットは OggVorbis のみです。通常 UnityAPI で再生を行う場合、Resources フォルダや AssetBundle から読み込むので保護された状態でデータを扱えますが、SoundPool で再生するには一度端末ローカルにファイルを書き出す必要があります。不正にデータを抜かれると困るような場合は SoundPool では再生できません。

SoundPool で再生を行う場合はローディングの実装が通常と異なるので、SoundPool を使用する旨と、ラベルとそのラベルが使用するファイル名を実装を行なうプログラマへ伝えてください。データを渡す際は通常の Wave ファイルの他に OggVorbis に変換したファイルも必要です。Windows・iOS では通常通りの再生、Android では OggVorbis を使用して SoundPool で再生される状態になるので、通常データの準備も必要です。

■XML エクセルに更新があったときの最新エクセルへの移行方法

※以下は手動で行う場合です。現在は USndXML_Tools で自動で行えます。

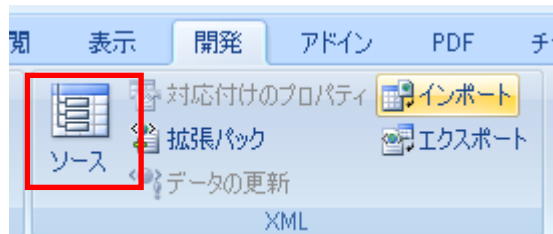
パラメータ追加などで USndXML.xls に変更があったとき、既存のテーブルから最新版へ移行する方法です。
まず XML を通常通りエクスポートし、最新版の USnd に含まれる LabelSettings.xsd を用意します。
どちらの方法もバイナリ形式を使用している場合は、作業後にテーブル名を確認してください。

・最新版の USndXML.xls に移す場合

サンプルで記載している内容を削除します。（記入されている行を選択して Delete キーを押す）
左上のマスにカーソルを置いて開発→インポートを選択し、書き出した XML を選択します。

・タイトル側の USndXML.xls に移す場合

既存のテーブルごと一旦削除します。
開発→ソースを選択し XML ソースウィンドウを表示します。



右側に下記のウィンドウが表示されます。

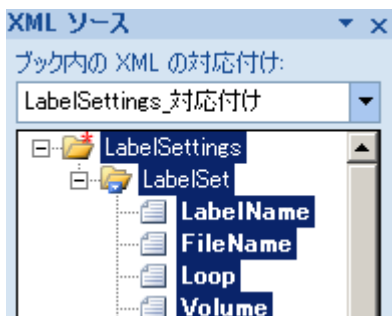


XML の対応付けボタンをクリックして、対応付け一覧を表示します。



削除したテーブルの対応付けを削除します。該当の対応付けを選択し、「削除」ボタンを押します。
 削除できたら「追加」を押して、新しい LabelSettings.xsd を選択します。
 追加できたら「OK」を押してウィンドウを閉じます。

XML の対応付けから追加した対応付け名を選択し、ツリーをクリックします。



この状態からテーブルを配置したいマスに向かってドラッグ&ドロップします。
 テーブルが自動的に作成されるので、テーブルにカーソルを置いて開発→インポートを選択し書き出した XML
 を選択してインポートを行なってください。
 複数テーブルがある場合はこれを繰り返してください。

以上。

■AudioClip の設定を一括変更する方法

USnd ver2.15.1 以降から USndAudioAssetLoader.cs を追加しました。
他のデバッグツール同様、UnityEditor 上で動作するツールです。AudioClip 設定を CSV 形式で出力と、CSV 形式を読み込んで AudioClip の設定を更新することができます。
USndAudioAssetLoader.cs がない場合、Assets 以下に Editor フォルダを作成し、USndAudioAssetLoader.cs を配置してください。USnd のほかのエディタが使用可能な場合は、他のスクリプトと同じフォルダにおいてください。USndAudioAssetLoader.cs 自体はプロジェクトに含める必要はないので、サウンド担当者のローカル環境のみに置くのでも大丈夫です。AudioClip の設定は meta ファイルに保存されるので、meta を共有すれば他の環境でも同じ設定になります。

画面上部のメニューから USnd→AudioClip 設定変更を選択します。



「CSV 作成」ボタンを押すと、Resources 以下にある AudioClip を検索し、現在 AudioClip に設定しているパラメータのリストを生成します。CSV の保存先は Assets フォルダ以下でないと指定できませんが、CSV 自体はプロジェクトに含める必要はないです。

「AudioClip 設定変更」を押すと、CSV を選択するウィンドウが開きます。読み込む CSV を選択すると、CSV に指定されている AudioClip の設定を CSV 上のパラメータに変更して再読み込みを行います。変更がある AudioClip のみ再読み込みを行います。AudioClip を選択した状態で実行した場合、Inspector の表示は自動的に更新されないなので F5 を押すか、別のオブジェクトを選択して Inspector の表示を更新してください。

CSV 項目の内容は以下の通りです。

項目名	内容
AssetPath	Assets フォルダからの AudioClip へのパスです。CSV 作成から出力した状態から変更する必要はありません。
LoadInBackground	ロードと同時に裏でデータを読み込みしておきたいとき TRUE にします。デフォルトは FALSE です。LoadType が Streaming のときは強制的に FALSE になります。
LoadType	データの取り扱い方を指定します。 DecompressOnLoad・・・AudioClip のロード時に解凍されます。 CompressedInMemory・・・オーディオデータは圧縮形式でメモリに保管されま

	<p>す。</p> <p>Streaming・・・ストリーミング再生を行います。</p>
CompressionFormat	<p>圧縮形式を指定します。デフォルトは Vorbis です。</p> <p>PCM、OggVorbis、ADPCM が選択できますが、基本 Vorbis から変更しません。</p>
Quality	<p>圧縮率を 0.00～1.00 で指定します。値は圧縮結果とソースファイルとのサイズの比にほぼ対応しています。</p>
SampleRateSettings	<p>サンプルレートを Unity 上で変更するか設定します。PreserveSampleRate がデフォルトです。</p> <p>PreserveSampleRate・・・インポート時にサンプルレートを変更しません。AudioClip のサンプルレートを使用します。</p> <p>OptimizeSampleRate・・・Unity がインポートした AudioClip に最適なサンプルレートを推測します。オーディオファイルはオーディオの品質を維持しながら最小のサンプルレートを選択します。</p> <p>OverrideSampleRate・・・指定した値でインポートしたオーディオファイルのサンプルレートを上書きします。</p>
SampleRateOverride	<p>SampleRateSettings が OverrideSampleRate に設定されているときに設定されるサンプルレートを指定します。</p>

実行時にエラーがあった場合は **Unity** の **Console** ウィンドウにメッセージが表示されます。

■ コール履歴を保存・出力する方法

USnd ver2.15.1 以降から USndCompileSettings.cs を追加しました。

他のデバッグツール同様、UnityEditor 上で動作するツールです。ゲームを実行した際、コールされたラベルを保存しゲーム起動中に任意のタイミングで、テキスト形式でコール履歴を出力します。

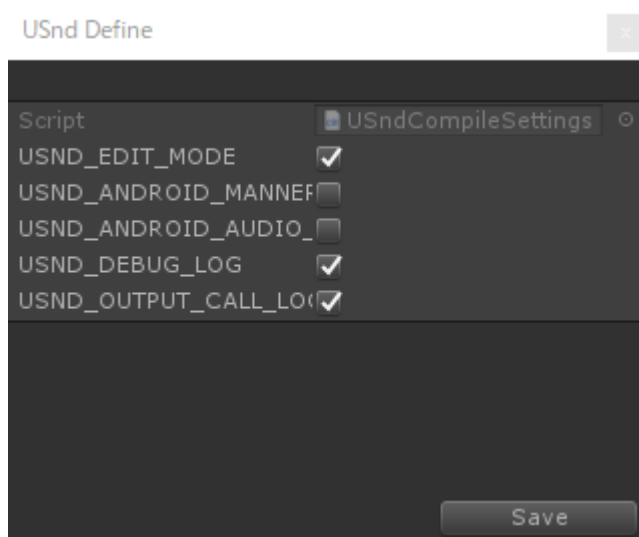
コール履歴は重複なし、アルファベット順にソートされます。

履歴の出力が行えるのはゲーム起動中のみです。

履歴の出力を行うには、まずコンパイル設定を有効にしてください。

メニューの USnd->コンパイル設定を選択し、USND_OUTPUT_CALL_LOG にチェックを入れ Save ボタンを押し、右下のインジケーターが消えるのを待ちます。

設定が有効になっていない場合、「コール履歴を出力」で表示されるウィンドウに「コンパイル設定で USND_OUTPUT_CALL_LOG を有効にしてください。」というメッセージが表示されます。



設定が終わったら、ゲームを起動します。

USnd->コール履歴を出力を選択し、ウィンドウを開きます。



ゲーム起動開始から履歴の記録を開始します。ゲーム中の任意のタイミングから記録を開始したい場合は、開始するタイミングで「履歴をクリア」ボタンを押してください。

出力したいタイミングで「履歴を出力」ボタンを押します。フォルダ選択ダイアログが開くので、保存する場所を指定します。指定したフォルダに CallLog.txt が作成されます。

履歴の出力はゲーム実行中でないと行えません。

■トラブルシューティング

Q. 再生ディレイとダッキングを設定したラベルを再生すると、再生開始していない時点からダッキングしてしまう。

A. 仕様です。再生開始時からダッキングしたい場合は、再生ディレイを設定せずプログラム制御で再生を遅延させるようにしてください。

Q. タップ音が遅延する。

A. Unity は Android だと遅延が大きいです。テーブルで `IsAndroidNative` フラグを `true` にし、`SoundPool` を使用して再生するようにしてください。ただし `SoundPool` は短いワンショットの SE しか再生できず、再生中にインスタンスを制御できません。また、Unity の API で再生するとエフェクトをかけていなくても何か効果がかかっているように聞こえるようなので、`SoundPool` のほうが音量が大きく聞こえる傾向にあるので注意してください。

その他の対策・他のプラットフォーム向けの対策としては、UnityEditor の Edit->ProjectSettings->Audio から DSP Buffer Size を変更して遅延が軽減するか試してください。

Q. 3D サウンドのグラフを編集したが値が無視される

A. うまく値が反映さないことがあるようなので、一定範囲以上で値の変化がない場合でもグラフの先端と終端にポイントを追加して試してみてください。（「JSON 形式で 3D サウンドのパラメータ設定を行う方法」項を参照）

Q. USndTool のログの行数を増やしたい

A. Unity 上で表示できるのが 15000 文字までなので Unity 上での表示は増やせません。ただし Copy ボタンを押してクリップボードに保存されるログは `AudioDefine.cs` で定義されている `LOG_MAX` 行数分保存されています。見たいログが消えているときは Copy ボタンを押し、テキストエディタ等で貼り付けを行い、ログを確認してください。また USndTool の Status の表示数は、同じく `AudioDefine.cs` の `SOUNDINFO_MAX` で変更可能です。

Q.Snapshot の切り替えが指定した秒数で実行されない。

A.TimeScale が変わっていると指定した時間で実行されません。Mixer の UpdateMode を Unscaled Time に変更してください。

Q.AudioMixer がバイナリ形式で出力されるようになった

A.Edit->Project Settings の Editor の中にある Asset Serialization を ForceText に設定して新しく作り直してください。Mixed か ForceBinary にするとバイナリ形式になります。