・ファンクションリファレンス

@Engine

```
bool EngineInitialize(unsigned short sourceCapacity=32,unsigned short actorCapacity=256);
void EngineUpdate(float deltaTime);
```

@Listener

```
void ListenerPostion(float x, float y, float z);
void ListenerDirection(float x, float y, float z);
void ListenerUpVector(float x,float y,float z);
void ListenerVelocity(float x, float y, float z);
void ListenerVolume(float volume);
```

@World

```
void WorldSpeedOfSound(float speedObSound);
void WorldDopplerFactor(float dopplerFactor);
void WorldDopplerVelocity(float velocity);
```

@class Player

```
bool Init(const char*);
void Play(float timeToGrow=0.0f);
void Stop(float timeToStop=0.0f);
void Pouse()const;
void SetLooping(bool io);
void SetVolume(float vol);
void SetPitch(float pitch);
void Reset();
```

@class Player3D

```
void SetPosition(float x, float y, float z);
void SetVelocity(float x, float y, float z);
void SetReferenceDistance(float referenceDistance);
void SetMaxDistance(float maxDistance);
void SetMinVolume(float minVolume);
```

```
void SetMaxVolume(float maxVolume);
```

@class Player3DCustom

```
SetDirection(float x, float y, float z);
SetConeInnerAngle(float coneInnerAngle);
SetConeOuterAngle(float coneOuterAngle);
SetConeOuterVolume(float coneOuterVolume);
SetRollOffFactor(float rolloff);
```

はじめに<namespace Sound>

ライブラリ関数,またクラスは全て namespace <Sound> でくくられています

<サウンドエンジン>

サウンドを使うには不可欠な関数軍です,単体の関数ですが
Engine???()という関数名がついていることでしょう
.

<func> bool EngineInitialize(unsigned short sourceCapacity=32,unsigned short actorCapacity=256);

エンジンを初期化します,
第一引数は,同時に再生する音の数です.
第二引数は Player 変数を同時に初期化できる数を表します.
デフォルト引数で sourc が 32,actor が 256 となってます,使用していて足りなくなったら値を変更するとよいでしょう.
過度に作れないように,sourceCapacity の最大値は 64 まで,
actorCapacity は 512 までと、私が設定しています,
※ 必ず最初に使用してください(初期化していなければ,全ての関数が無効になります)

<func> void EngineUpdate(float deltaTime);

裏でいろいろしています.毎フレームに一回呼ぶようにしてください
引数はフレーム間の更新にかかった経過時間です<デルタタイム>
一応,使用しなくても音はなってしまいますが,
3D の音が上手く反映できないのでなにかおかしいと思ったら忘れてないか確かめてみてください

<リスナー>

リスナーはその名の通り,聴き手の位置や向きなどの情報を設定します.

後述の 3D サウンドを使わない(Player クラスのみ使用)場合は必要ありませんが,

3D サウンドを使う際は必須の設定項目になるでしょう.

関数名は Listener???()となっているでしょう

```
<func> void ListenerPostion(float x, float y, float z);
        * リスナーの位置を変更する.
        * @param x,y,z ：リスナーの空間座標(位置ベクトル)
        *                  デフォルト値{0.0f,0.0f,0.0f}
        *
        * @note 3Dオブジェクトを利用する場合,リスナーの位置,向き,
        *       3Dオブジェクトの座標などで,聞こえる方向や大きさが計算される
        */
<func>void ListenerDirection(float x, float y, float z);
        * リスナーの速度を設定する.
        *
        * @param x,y,z ：リスナーの速度ベクトル
        *                                  デフォルト値{0.0f,0.0f,0.0f}
        *
        * @note 実際に動くわけではなくドップラー効果の計算に使わる値(紛らわしい)

<func>void ListenerUpVector(float x,float y,float z);
        * リスナーの向きベクトルを設定する
        *
        * @param x,y,z ：方向ベクトル(単位ベクトル)
        *                                  デフォルト値{0.0f,0.0f-1.0f}
        *
        * @note 3Dオブジェクトを利用する場合,リスナーの位置,向き,
        *                  3Dオブジェクトの座標などで,聞こえる方向や大きさが計算される

<func>void ListenerVelocity(float x, float y, float z);
        * リスナーの上方向ベクトルを設定する
        *
```

```
        * @param x,y,z :方向ベクトル(単位ベクトル)
        *                                   デフォルト値{0.0f,1.0f,0.0f}
        *
        * @note 3Dオブジェクトを利用する場合,(略)...上参照

<func>void ListenerVolume(float volume);
        * マスター音量と思ってよい,
        *
        * @param volume   :音量[0.0 ~ 1.0f]
        *                                    デフォルト値{1.0f}
        *
        * @note Playerクラスで個々の音量を調節することも可能
```

<ワールド>

エンジンと違って必須項目でない,またリスナーにも属さない全体設定項目です
openALは与えた距離や速度によって音の減衰やドップラー効果などを計算して音を変化させます,こちらではその計算に音速等の設定ができます.

```
<func>void WorldSpeedOfSound(float speedObSound);
        * @param speedOfSound : 音の速さ[0.0001~]
        *                          デフォルト値{ 343.3}
        * @note ドップラー効果に影響がある値
<func>void WorldDopplerFactor(float dopplerFactor);
        * 多分全てのオブジェクトのvelocityのスケールだと思われ(予想)
        * @param dopperFactor :  [0.0f ~]
        *                          デフォルト値{1.0f}
        * @note よくわかってない.勉強不足
        * --以下公式から和訳して引用--
        * ソースとリスナーの速度を調整します。
<func>void WorldDopplerVelocity(float velocity);
        * 多分全てのオブジェクトのvelocityの乗数だと思われ(予想)
        * @param velocity:リファレンスにもRangeが乗ってない...
        *                  デフォルト値{不明}
        * @note よくわかってない.勉強不足
```

＊ --以下公式から和訳して引用--

＊ 音速に適用される乗数。

<プレイヤーオブジェクト>

音を鳴らすために必要なクラス.プレイヤーの変数を作成してパラメータを設定することで音を鳴らすことができる.

<func>bool Init(const char*);

＊ Init関数,バッファーにアクターオブジェクトを作成する.

＊ @param filename: waveファイル(.wav)

＊ @note waveファイル(ステレオorモノラル)のみ受け付ける

＊ Initしないと他のメンバ関数が使用できない.

<func>void Play(float timeToGrow=0.0f);

＊ 音を再生する

＊ @param timeToGrow ： 音が徐々に大きくなる時間(秒)

＊ @note ： 引数を設定すると徐々に大きくなって時間が経つと本来の音量になる

<func>void Stop(float timeToStop=0.0f);

＊ 再生中であれば停止する.

＊ @param timeToStop ：音が止まるまでの時間(秒)

＊ @note 引数を設定すると音が止まるまで徐々に減衰していく

<func>void Pouse()const;

＊ 音を一時停止する.次回再生の際は途中から再生される

＊@note 乱用するとソースが足りなくなるので 途中で止める必要のないものはStopで止めよう,

<func>void SetLooping(bool io);

＊ ループを設定する

＊ @param io ： true く ループオン ： false く ループオフ

＊ デフォルト値 false

<func>void SetVolume(float vol);

＊ 音量を設定する

＊ @param volume:設定する音量(0.0f~0.1f)

```
        *                                   デフォルト値(0.8f)
        * @note 本来のoenal側のデフォルトは1.0fなのだが,爆音である

<func>void SetPitch(float pitch);
        * ピッチを設定する
        * @param pitch :設定するピッチ(0.0f~)
        *                   デフォルト値(1.0f);
        * @note 半分にすれば1オクターブ下がる,二倍にすれば1オクターブ上がる,以上

<func>void Reset();
        * アクターの破棄を手動で行います
        * ※行わない場合は変数が破棄されるときにデストラクタで自動で行われる.
```

<プレイヤー3D>

プレイヤーだけだと立体音響で音を鳴らすことはできません.3Dで音楽を鳴らすには
こちらのクラスで音を鳴らしてください.
また.立体音響で音を鳴らす場合はListenerの設定は必要不可欠となるので,そちらの設定も
しっかり行いましょう

```
<func>void SetPosition(float x, float y, float z);
        * @param x,y,z :空間上の位置
        *                   デフォルト値{0.0f,0.0f,0.0f}
        * @note Listenerの座標と向きとオブジェクトの位置から,
        *                   音が聞こえる方向や、音の減衰などが決まる.

<func>void SetVelocity(float x, float y, float z);
        * 速度を設定する,位置が動くわけではなくドップラー効果の計算に使用する.
        * @param x,y,z :速度ベクトル
        *                   デフォルト値{0.0f,0.0f,0.0f}
        * @note

<func>void SetReferenceDistance(float referenceDistance);
        * @param referenceDistance :音が半分減衰する距離
        *                               デフォルト値{1.0f}
        * @note 距離による音量の減衰に影響する値:長いほど減衰しにくい,短いほど減衰しやす
```

*い.

<func>void SetMaxDistance(float maxDistance);

* 距離による減衰がなくなる境界線を設定する
* @param maxDistance :減衰がなくなるまでの距離(リスナーからの相対距離)
* デフォルト値{100.0f}※openALではdistanceModelの設定によるため勝手に決めてる
* @note referenceDistance(上記)とうまく組み合わせて使えって公式が言ってた

<func>void SetMinVolume(float minVolume);

* 音量の最低値の設定(計算適用語)
* @param minVolume :min音量(0.0f~)
* デフォルト値{0.0f}
* @note 減衰によって音が消えてほしくない場合は設定するとよい

<func>void SetMaxVolume(float maxVolume);

* 音量の最大値の設定(計算適用後)
* @param maxVolume :max音量(0.0f~1.0f)
* デフォルト値{1.0f}
* @note 上手く使ってください(てきとー)

<プレイヤー3D カスタム>

あなたがもしプレイヤー3Dクラスで物足りなくなった時がこのクラスの使い時かと思います,<カスタム>というだけあって,元のプレイヤー3Dの設定項目に加えて,こちらでしかせっていできない項目があります,ちなみに設定項目は全てopenALの機能です,
個人的に頻繁に使いそうなものは3D,そうでないものは3Dカスタムに振り分けています,
設定変数はバッファで全て保持するので3Dより,3Dカスタムの方がメモリの消費量は上がるので,カスタムの方のパラメータを設定しない場合はプレイヤーカスタムの方を使うのが
好ましいでしょう.

<func>SetDirection(float x, float y, float z);

* 向きを変更する
* @param x,y,z ：方向ベクトル(単位ベクトル)
* デフォルト値{0.0f,0.0f,0.0f}

```
        * @note 後述の音の方向による減衰の設定と深く関係する

<func>SetConeInnerAngle(float coneInnerAngle);
        * コーンのインナー角の設定
        * @param coneInnerAngle    :円錐の頂点の角度(0.0f~360.0f)
        *                          デフォルト値{360.0f}
        * @note ConeOutorVolumeの減衰が適用されない角度.向きの設定はDistanceで,

<func>SetConeOuterAngle(float coneOuterAngle);
        * コーンのアウター角の設定
        * @param ConeOuterAngle :円錐の頂点の角度(0.0f~360.0f)
        *                                                  デフォルト値{360.0f}
        * @note インナー角とアウター角の間で徐々に減衰されていくイメージ

<func>SetConeOuterVolume(float coneOuterVolume);
        * アウター角の外の音量を設定する
        * @param coneOuterVolume :max音量(0.0f~1.0f)
        *                          デフォルト値{1.0f}
        * @note 音源の向きと内側円錐,外側円錐,外型円錐の外の音量...四人はセットなんです.

<func>SetRollOffFactor(float rolloff);
        * @param rolloff :ロールオフの値 [0.0f - ]
        *                  デフォルト値{1.0f}
        *
        * @note 美味しいよね(本人も使い方がわからない)
        * --以下公式から和訳して引用--
        * 距離減衰を誇張または減少させるための乗数
        * 0.0では、距離の減衰は発生しません。
```