

技術情報

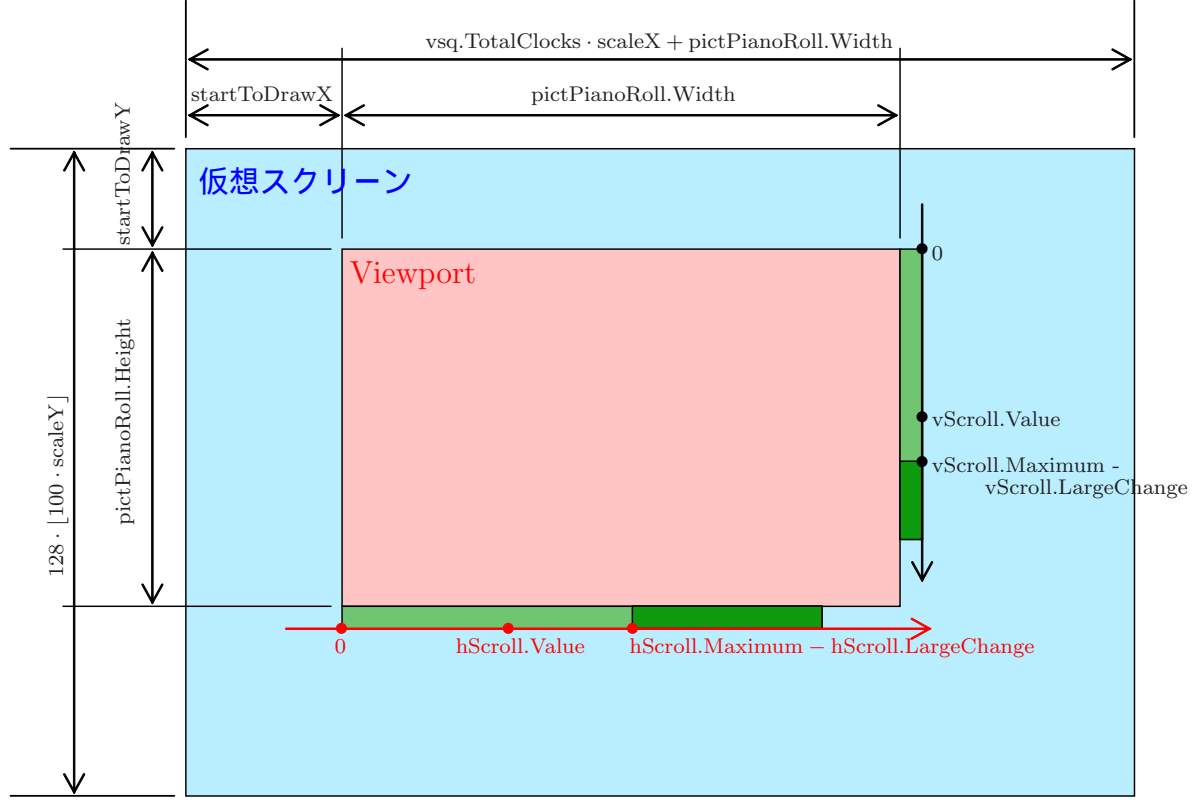
kbinani

目次

1	ピアノロールのスクロール動作の計算	3
1.1	水平スクロールバー	3
1.2	垂直スクロールバー	5
2	波形合成システム	7
2.1	概要	7
2.2	基底クラス: WaveUnit	7
2.3	インターフェース	8
2.3.1	WaveReceiver	8
2.3.2	WaveSender	8
2.3.3	WaveGenerator	8
2.4	WaveGenerator	8
2.4.1	VOCALOID	8
2.4.2	vConnect-STAND	8
2.4.3	UTAU	8
2.4.4	AquesTone	8
2.4.5	無音	8
2.5	WaveSender	8
2.5.1	WAVE ファイル読み込み	8
2.6	WaveReceiver	8
2.6.1	モニター	8
2.6.2	WAVE ファイル書き込み	8
2.7	WaveSender かつ WaveReceiver	8
2.7.1	増幅器	8
2.7.2	重ね合わせ	8
2.7.3	分割器	8
3	org.kbinani.vsq 名前空間にあるクラスの依存関係	9
4	動作確認用のチェックリスト	10

1 ピアノロールのスクロール動作の計算

水平および垂直スクロールバー・コントロールの設定と、画面の描画状態との関係を決めるための計算方法を述べる。



上図のように、ビューポートのオフセットは $startToDrawX$ と $startToDrawY$ で定義される。また、水平スクロール・コントロール $hScroll$ の $Value$ プロパティの単位は $clock$ 、垂直スクロール・コントロール $vScroll$ の $Value$ プロパティの単位は $cent$ である。ピアノロールの画面上の横方向の表示倍率を $scaleX[pixel/clock]$ 、縦方向の表示倍率を $scaleY[pixel/cent]$ とする。

$startToDrawX$ 、 $startToDrawY$ は次で定義される：

$$\begin{cases} startToDrawX = hScroll.Value \cdot scaleX, \\ startToDrawY = vScroll.Value \cdot scaleY. \end{cases} \quad (1)$$

1.1 水平スクロールバー

$startToDrawX$ の最大値および最小値は次のように決める。

$$\max(startToDrawX) = vsq.TotalClocks \cdot scaleX, \quad (2)$$

$$\min(startToDrawX) = 0. \quad (3)$$

水平スクロールバーが一番右にスクロールされたときに、 $hScroll.Value$ が最大値をとらねばならないから、 $startToDrawX$ の最大値を $scaleX$ で割ったものが、 $hScroll.Value$ の

最大値となるようにする．また，水平スクロールバーの動作の仕様から `hScroll.Value` の最大値は `hScroll.Maximum - hScroll.LargeChange` であるので，次の式が成り立つ．

$$\frac{\max(\text{startToDrawX})}{\text{scaleX}} = \text{hScroll.Maximum} - \text{hScroll.LargeChange} \quad (4)$$

さらに，外観上スクロールバーの可動範囲とスクロールボックスの幅の比が，仮想スクリーンとビューポートの幅の比と同一であると美観が良いので，次の式が成り立つようにする．

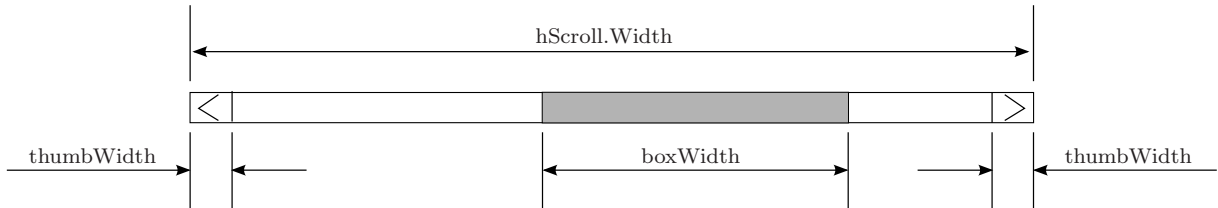
$$\frac{\text{pictPianoRoll.Width}}{\text{vsq.TotalClocks} \cdot \text{scaleX} + \text{pictPianoRoll.Width}} = \frac{\text{hScroll.LargeChange}}{\text{hScroll.Maximum}} \quad (5)$$

式 (4) と式 (5) を連立させて解くと，`hScroll.Maximum` と `hScroll.LargeChange` は次で定まる．

$$\text{hScroll.Maximum} = \text{vsq.TotalClocks} + \frac{\text{pictPianoRoll.Width}}{\text{scaleX}}, \quad (6)$$

$$\text{hScroll.LargeChange} = \frac{\text{pictPianoRoll.Width}}{\text{scaleX}}. \quad (7)$$

ここで，スクロールボックスの実際の表示幅がいくらになるかを調べる．水平スクロールバーの構造はだいたい下図のようになっている．



スクロールバーの動作の仕様から，次が成り立つ．

$$\frac{\text{boxWidth}}{\text{hScroll.Width} - 2 \cdot \text{thumbWidth}} = \frac{\text{hScroll.LargeChange}}{\text{hScroll.Maximum}}. \quad (8)$$

従って，

$$\text{boxWidth} = \frac{\text{hScroll.LargeChange}}{\text{hScroll.Maximum}} (\text{hScroll.Width} - 2 \cdot \text{thumbWidth}). \quad (9)$$

この `boxWidth` は，式 (6) と式 (7) により決まるが，この大きさがある値以下になると非常に扱いづらくなると考えられる．そこで，この `boxWidth` の最小値を `min(boxWidth)` とし，式 (6) と式 (7) から求めた値がこの値より小さくなる場合は，以下の要領で計算することにする．

`boxWidth` を値が `min(boxWidth)` の定数として扱う．すると，式 (5) は成り立たなくなるが，これは仕方がない．見た目が悪いよりも，スクロールボックスが操作しにくいことの方が駄目だと思うからである．従って最初式 (4) と式 (5) を連立させて解いた代わりに，式 (4) と式 (9) を連立させて解く．解は，

$$\text{hScroll.Maximum} = \frac{\text{vsq.TotalClocks}(\text{hScroll.Width} - 2 \cdot \text{thumbWidth})}{(\text{hScroll.Width} - 2 \cdot \text{thumbWidth}) - \text{boxWidth}}, \quad (10)$$

$$\text{hScroll.LargeChange} = \frac{\text{vsq.TotalClocks} \cdot \text{boxWidth}}{(\text{hScroll.Width} - 2 \cdot \text{thumbWidth}) - \text{boxWidth}}, \quad (11)$$

となる．

1.2 垂直スクロールバー

`startToDrawY` の最大値および最小値は次のように決める．

$$\max(\text{startToDrawY}) = 128 \cdot \lfloor 100 \cdot \text{scaleY} \rfloor - \text{pictPianoRoll.Height}, \quad (12)$$

$$\min(\text{startToDrawY}) = 0. \quad (13)$$

垂直スクロールバーが一番下にスクロールされたときに，`vScroll.Value` が最大値をとらねばならないから，`startToDrawY` の最大値を `scaleY` で割ったものが，`vScroll.Value` の最大値となるようにする．また，垂直スクロールバーの動作の仕様から `vScroll.Value` の最大値は `vScroll.Maximum - vScroll.LargeChange` であるので，次の式が成り立つ．

$$\frac{\max(\text{startToDrawY})}{\text{scaleY}} = \text{vScroll.Maximum} - \text{vScroll.LargeChange}. \quad (14)$$

さらに，外観上スクロールバーの稼動範囲とスクロールボックスの高さの比が，仮想スクリーンとビューポートの高さの比と同一であると美観が良いので，次の式が成り立つようにする．

$$\frac{\text{pictPianoRoll.Height}}{128 \cdot \lfloor 100 \cdot \text{scaleY} \rfloor} = \frac{\text{vScroll.LargeChange}}{\text{vScroll.Maximum}}. \quad (15)$$

式 (14) と式 (15) を連立させて解くと，`vScroll.Maximum` と `vScroll.LargeChange` は次で定まる．

$$\text{vScroll.Maximum} = \frac{128 \cdot \lfloor 100 \cdot \text{scaleY} \rfloor}{\text{scaleY}}, \quad (16)$$

$$\text{vScroll.LargeChange} = \frac{\text{pictPianoRoll.Height}}{\text{scaleY}}. \quad (17)$$

ここで，スクロールボックスの実際の表示高さがいくらになるかを調べる．水平スクロールバーの場合と大体同じであるので，スクロールボックスの高さ `boxHeight` について，次が成り立つ．

$$\text{boxHeight} = \frac{\text{vScroll.LargeChange}}{\text{vScroll.Maximum}}(\text{vScroll.Height} - 2 \cdot \text{thumbHeight}). \quad (18)$$

水平スクロールバーのときと同様に，`boxHeight` を，値が `min(boxHeight)` の定数として扱う．式 (14) と式 (18) を連立させて解く．解は，

$$\text{vScroll.Maximum} = \frac{128 \cdot \lfloor 100 \cdot \text{scaleY} \rfloor - \text{pictPianoRoll.Height}}{\text{scaleY}} \times \frac{\text{vScroll.Height} - 2 \cdot \text{thumbHeight}}{\text{vScroll.Height} - 2 \cdot \text{thumbHeight} - \text{boxHeight}}, \quad (19)$$

$$\text{vScroll.LargeChange} = \frac{128 \cdot \lfloor 100 \cdot \text{scaleY} \rfloor - \text{pictPianoRoll.Height}}{\text{scaleY}} \times \frac{\text{boxHeight}}{\text{vScroll.Height} - 2 \cdot \text{thumbHeight} - \text{boxHeight}}, \quad (20)$$

となる .

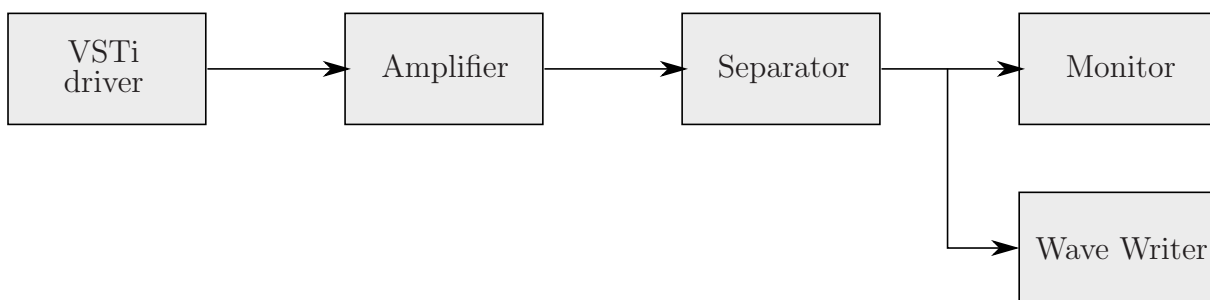
2 波形合成システム

2.1 概要

波形合成システムは、波形を処理する幾つかのクラスのインスタンスを、ブロック線図のようにつなぎ合わせて実現される。ここでいう「波形を処理」とは、

1. 生成
2. 増幅
3. 重ね合わせ
4. 分割

等のことである。例えば、VOCALOID VSTi を駆動してその音声波形をスピーカーからモニターすると同時に、Wave ファイルに記録する場合は、次のようなクラス間の接続が為されている。



上図では、波形を生成する生成器から、増幅器、分割器、音声モニター、および Wave 出力器が接続されている。この節では、これらを実装するための基底クラスや、それぞれの機能ごとの実際のクラス簡易なリファレンスと、使用例を述べる。

2.2 基底クラス: WaveUnit

このクラスは、波形を処理するクラスの継承元となる基底クラスである。クラスの外観は以下のようになっている。

```
class WaveUnit{
    public abstract int getVersion();
    public abstract void setConfig( string parameter );
    public virtual void setGlobalConfig(
        org.kbinani.cadencii.EditorConfig config ){};
    public virtual void paintTo(
        System.Drawing.Graphics graphics,
        int x,
        int y,
        int width,
        int height ){};
}
```

2.3 インターフェース

波形処理の種類には、波形を (active に) 生成する処理、波形を (passive に) 生成する処理、および波形を受け取る処理の3種類に大別できる。ここでいう active, passive な処理とは、それぞれ、自律して波形を生成し他の処理装置に波形を送り出す処理、および他からの波形の要求に応じて必要な量の波形を生成する処理のことをいう。

2.3.1 WaveReceiver

2.3.2 WaveSender

2.3.3 WaveGenerator

2.4 WaveGenerator

2.4.1 VOCALOID

2.4.2 vConnect-STAND

2.4.3 UTAU

2.4.4 AquesTone

2.4.5 無音

2.5 WaveSender

2.5.1 WAVE ファイル読み込み

2.6 WaveReceiver

2.6.1 モニター

2.6.2 WAVE ファイル書き込み

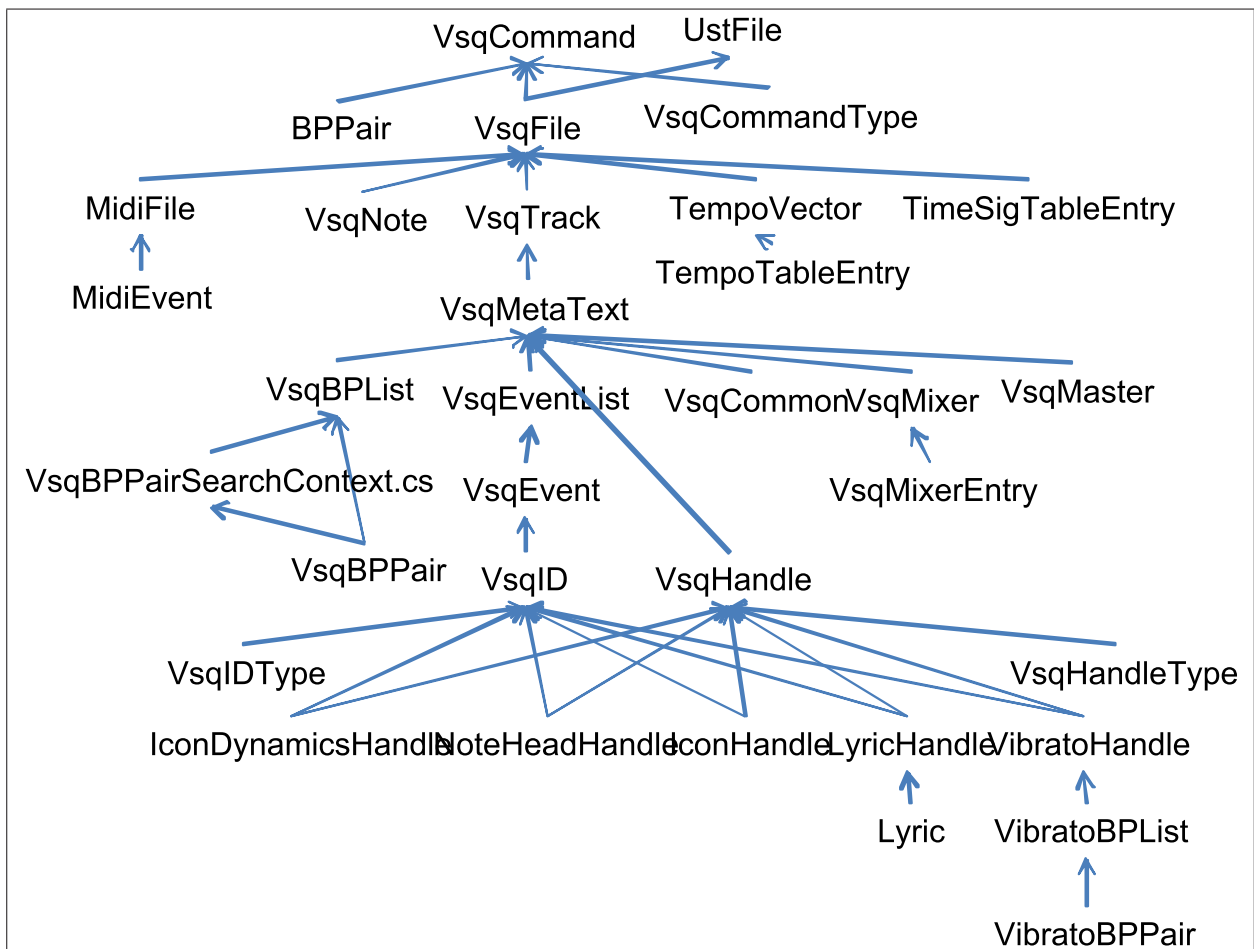
2.7 WaveSender かつ WaveReceiver

2.7.1 増幅器

2.7.2 重ね合わせ

2.7.3 分割器

3 org.kbinani.vsq 名前空間にあるクラスの依存関係



4 動作確認用のチェックリスト