

「フーリエ変換を使わない、エフェクトの  
かけかた。」

mlemon:hi-farm.net

「フーリエ変換を使わない、エフェクトの  
かけかた。」

mlemon:hi-farm.net

# おことわり

- 僕自身、まだしっかりと理解しきれていないところがあり、もっときれいにできるはず、などのご指摘があると思いますが、暖かくご指摘いただければ幸いです。ご理解のほどよろしく願いします。ありがとうございます。

Dynamic Sound Generationとは？

# Dynamic Sound Generationとは？

- コールバックを設定することにより、動的にサウンドを生成する機能。もちろんオーディオファイルのデータを再生することも可能。

# Dynamic Sound Generationとは？

- コールバックを設定することにより、動的にサウンドを生成する機能。もちろんオーディオファイルのデータを再生することも可能。

と、いうことは、

# Dynamic Sound Generationとは？

- コールバックを設定することにより、動的にサウンドを生成する機能。もちろんオーディオファイルのデータを再生することも可能。

と、いうことは、

**動的にサウンドを生成するときに、値を書き換えるとエフェクトをかけられるはず！**

エフェクトってというのは、例えば、

エフェクトってというのは、例えば、

- デイレイ (遅れて聴こえるやつ)
- ローパスフィルター (低い音が残る)
- ハイパスフィルター (高い音が残る)
- 再生速度の変更
- 音程の変更

エフェクトってというのは、例えば、

- デイレイ（遅れて聴こえるやつ）
- ローパスフィルター（低い音が残る）
- ハイパスフィルター（高い音が残る）
- 再生速度の変更
- 音程の変更

などがあります。

- ただ、普通にエフェクトをつけようとする、フーリエ変換なるものを避けて通る事ができません。

- 今回は、フーリエ変換抜きでできるエフェクトなので、これは考えないようにします。

まずは、普通に再生  
してみましよう。

ポイント

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成
- コールバック関数の中でByteArrayとしてデータを詰めこむ

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音になる

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音になる
- コールバック関数の中でByteArrayとしてデータを詰めこむ

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音がる
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音がる

# ポイント

- オーディオファイルから読み込むためのSound オブジェクトを作成  
ファイルの読み込みが完了。
- コールバックで再生するためのSoundオブジェクトを作成
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音がる
- コールバック関数の中でByteArrayとしてデータを詰めこむ
- 音がる
- データがなくなったら、再生が終わる

では、コードを見て  
みましよう。

- このコードの中で、`samplesCallback` イベントハンドラ内で、データの解析 + 再生をしているので、ここで色々するとどうやら音に変化をつける事ができそう。

では、やってみよう

# ボリリューム変更

- エフェクトとは言えないと思いますがけど、まずは動的に音を変える、ということ。

# ディレイ：方針

- ディレイは、少し前の音が響いているはずなので、音データをオブジェクトに蓄えて、少し前の音も重ねて鳴らすようにすればいいはず！！

# ディレイ：コード例

- サンプルを見てみましょう。

# ローパスフィルター

- ローパスフィルターとは、低音部分を残し、他の周波数帯をカットするフィルターです。
- これは、一つ前のデータと平均することによって実現可能です。

# ローパスフィルター

- では、コード例を見てみましょう。

# ハイパスフィルター

- これも本来ならしっかりと計算しないと実現できないのですが、簡易版なら、先ほどのローパスフィルターの原理を利用すると簡単に実現可能です。

# ハイパスフィルター

- では、コード例を見てみましょう。

他にも作れれば。

- すみません。今回はこの程度しかできなかつたです。

- 今後、フィルターの精度と種類を増やす事ができたら、あらためてsparkにコミットさせて下さい。

- みなさま、ご清聴ありがとうございました。

- ちなみに、ここでブログやってまして、後日このドキュメントをアップします。後は、奮闘の日々を書くようにします。
- <http://blog.hi-farm.net/>