

USB メモリの中身

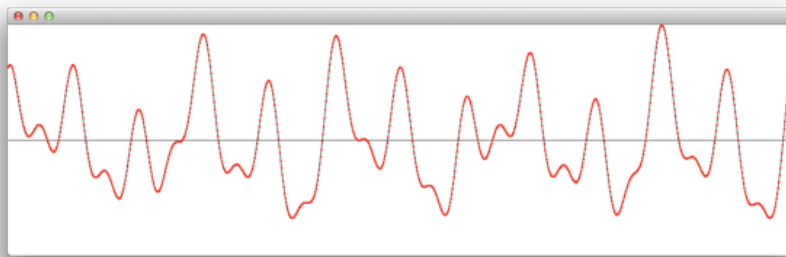
Fourier.tar.gz
generating.st
SSK_20111005_051900.st
SSK_20111005_204908.st
VisualWorks771ncWithJun790ForMac.zip
VisualWorks771ncWithJun790ForWin.zip

今回新しく出てきた物は、 generating.st のみ
今回付属の VisualWorks は既に CodeHighlight 済み

SSK_20111005_204908.st と generating.st の2つを VisualWorks771ncWithJun790ForMac の中に移動

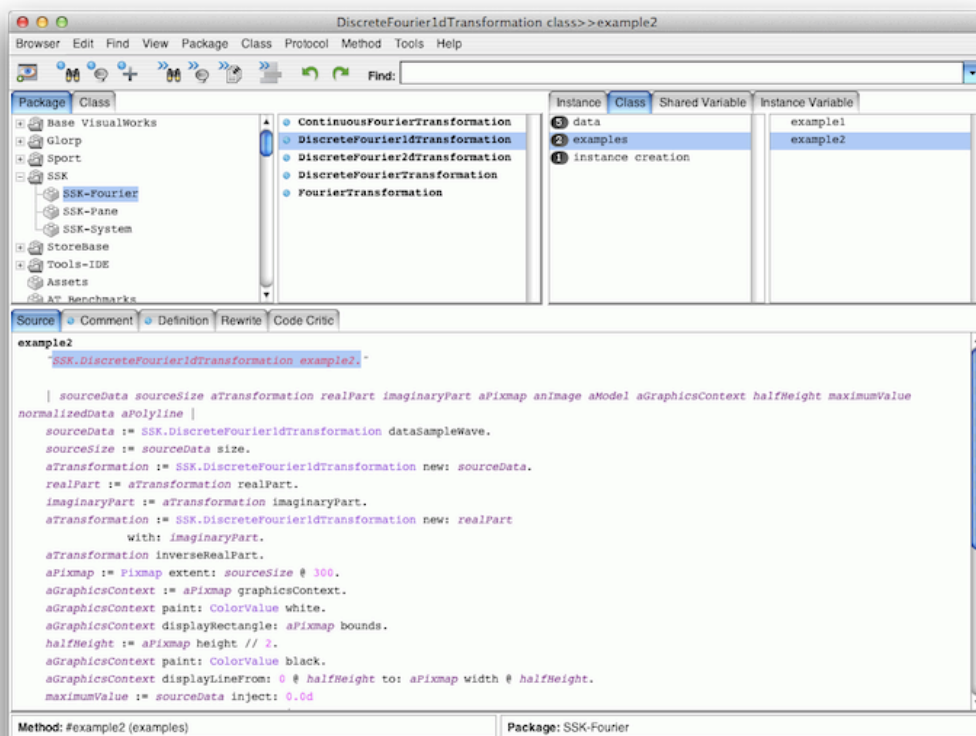
File Browser から SSK_20111005_204908.st を File in

先月は SSK バンドルの SSK-Fourier をいろいろやっていた
DiscreteFourier1dTransformation, Class, example2



そうそう、こんな感じだった

generating.st を File in



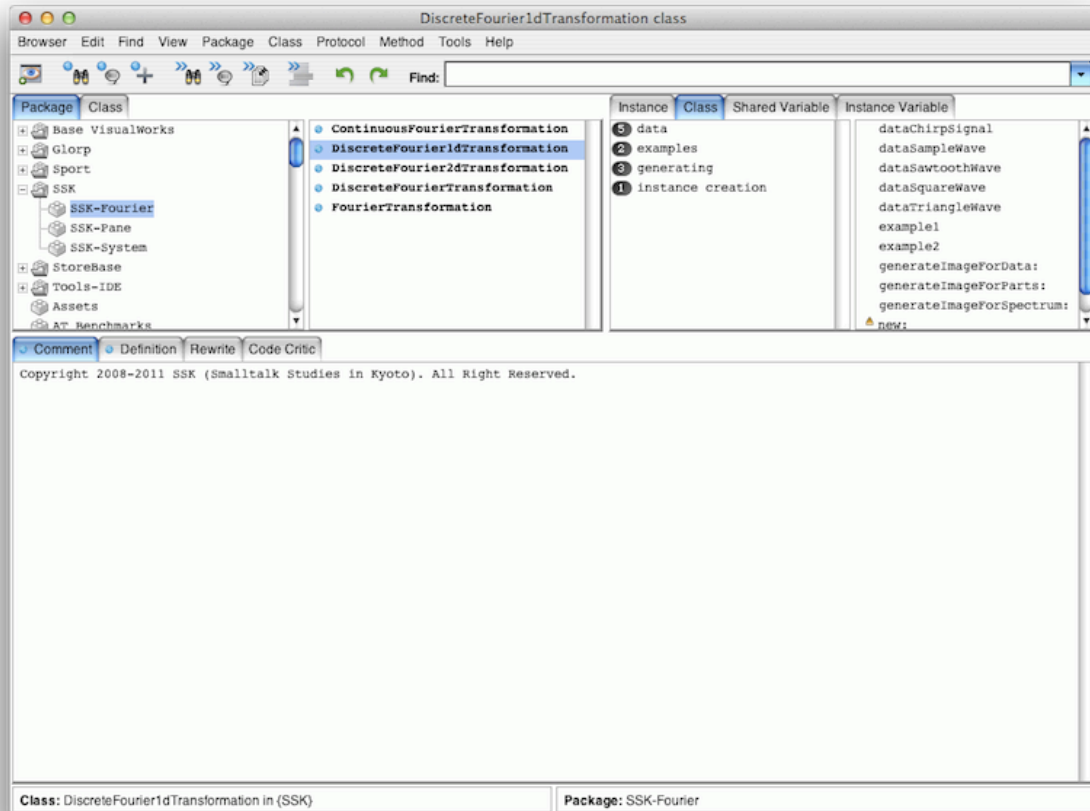
The screenshot shows the VisualWorks IDE interface. The title bar reads "DiscreteFourier1dTransformation class>>example2". The menu bar includes "Browser", "Edit", "Find", "View", "Package", "Class", "Protocol", "Method", "Tools", and "Help". The toolbar contains various icons for navigation and editing. The "Package" browser on the left shows a tree structure with "SSK-Fourier" selected. The "Class" browser in the center lists "ContinuousFourierTransformation", "DiscreteFourier1dTransformation", "DiscreteFourier2dTransformation", "DiscreteFourierTransformation", and "FourierTransformation". The "Instance" browser on the right shows "data", "examples", and "instance creation". The "Instance Variable" browser shows "example1" and "example2". The "Source" editor at the bottom displays the following code:

```
example2
  SSK.DiscreteFourier1dTransformation example2.

  | sourceData sourceSize aTransformation realPart imaginaryPart aPixmap anImage aModel aGraphicsContext halfHeight maximumValue
  normalizedData aPolyline |
  sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
  sourceSize := sourceData size.
  aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
  realPart := aTransformation realPart.
  imaginaryPart := aTransformation imaginaryPart.
  aTransformation := SSK.DiscreteFourier1dTransformation new: realPart
    with: imaginaryPart.
  aTransformation inverseRealPart.
  aPixmap := QPixmap extent: sourceSize @ 300.
  aGraphicsContext := aPixmap graphicsContext.
  aGraphicsContext paint: ColorValue white.
  aGraphicsContext displayRectangle: aPixmap bounds.
  halfHeight := aPixmap height // 2.
  aGraphicsContext paint: ColorValue black.
  aGraphicsContext displayLineFrom: 0 @ halfHeight to: aPixmap width @ halfHeight.
  maximumValue := sourceData inject: 0.0d
```

The status bar at the bottom indicates "Method: #example2 (examples)" and "Package: SSK-Fourier".

適当に別な場所を開いて

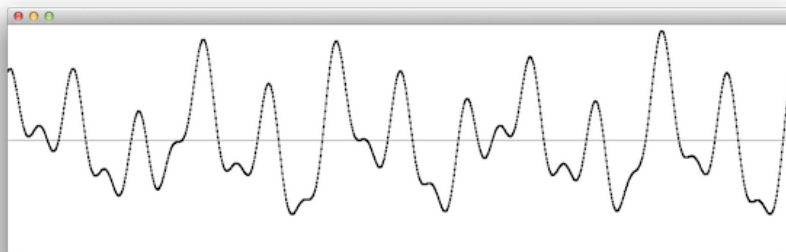


Class を開き直すと、generating が増える

example2 から example3 を追加
generating に実装されているものを使う

example3
"SSK.DiscreteFourier1dTransformation example3."

```
I sourceData aTransformation realPart imaginaryPart anImage aModel I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: sourceData.
aModel := SSK.PaneModel picture: anImage.
aModel open
```



example4 を追加

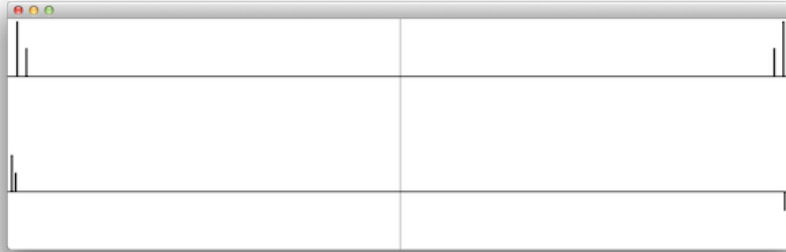
example4
"SSK.DiscreteFourier1dTransformation example4."

```
I sourceData aTransformation realPart imaginaryPart anImage aModel I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
```

```

imaginaryPart := aTransformation imaginaryPart.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation
    generateImageForParts: (Array with: realPart with: imaginaryPart).
aModel := SSK.PaneModel picture: anImage.
aModel open

```



こんな感じで出るんだけど、これでは要求仕様を満たしていない
内側に低周波、外側に高周波にしなければならないのだが、逆になっている

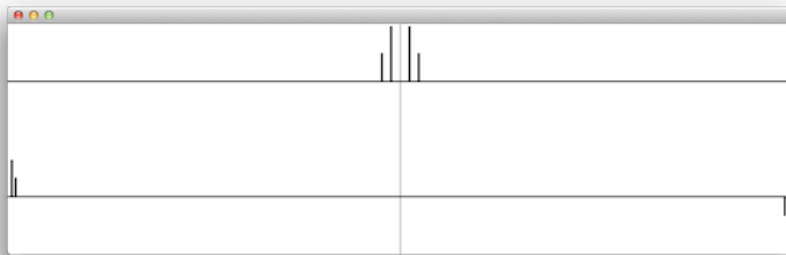
example4

```

"SSK.DiscreteFourier1dTransformation example4."

I sourceData aTransformation realPart imaginaryPart anImage aModel I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation
    generateImageForParts: (Array with: (aTransformation swap: realPart) with: imaginaryPart).
aModel := SSK.PaneModel picture: anImage.
aModel open

```



実部だけはちゃんとひっくり返った

同様に虚部もひっくり返す

example4

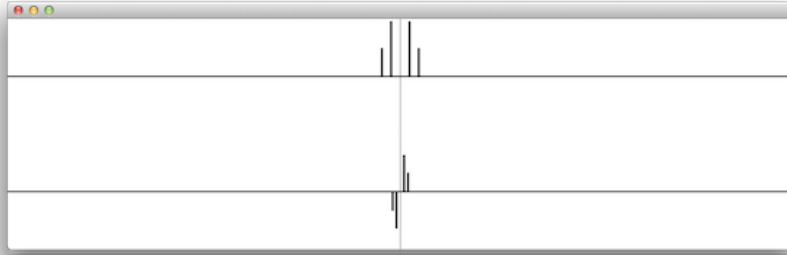
```

"SSK.DiscreteFourier1dTransformation example4."

I sourceData aTransformation realPart imaginaryPart anImage aModel I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation
    generateImageForParts: (Array with: (aTransformation swap: realPart)
        with: (aTransformation swap: imaginaryPart)).
aModel := SSK.PaneModel picture: anImage.
aModel open

```

generateImageForParts: ではなく generateImageForRealpart: imaginaryPart: と書く方がわかりやすいといえば、わかりやすい
が、とりあえず、このまま進めることにする



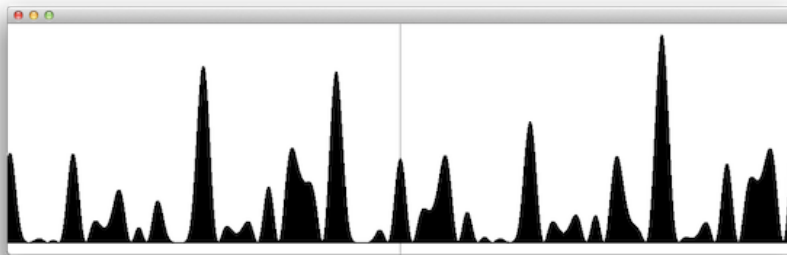
example5 を作る

example5

```
"SSK.DiscreteFourier1dTransformation example5."

I sourceData aTransformation realPart imaginaryPart anImage aModel I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation
    generateImageForSpectrum: powerSpectrum"←とりあえず、これを nil に".
aModel := SSK.PaneModel picture: anImage.
aModel open
```

powerSpectrum が居ないので、とりあえず、nil にして Accept してどこに居るのか探す
DiscreteFourier1dTransformation, Instance, spectrum, powerSpectrum に居る



これは、実は逆変換になってる (処理の順番がおかしいので)

example5

```
"SSK.DiscreteFourier1dTransformation example5."

I sourceData aTransformation realPart imaginaryPart anImage aModel powerSpectrum I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
powerSpectrum := aTransformation powerSpectrum.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation generateImageForSpectrum: powerSpectrum.
aModel := SSK.PaneModel picture: anImage.
aModel open
```

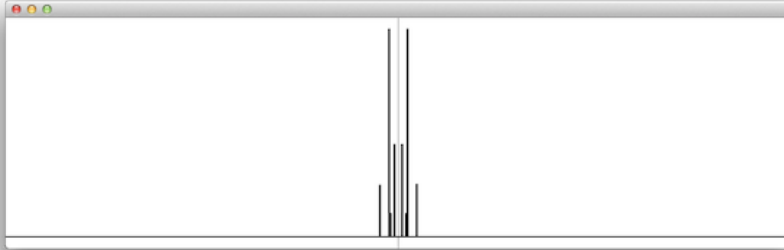


またもや、ひっくり返っているので、swap する

example5

```
"SSK.DiscreteFourier1dTransformation example5."
```

```
I sourceData aTransformation realPart imaginaryPart anImage aModel powerSpectrum I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
powerSpectrum := aTransformation powerSpectrum.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation
    generateImageForSpectrum: (aTransformation swap: powerSpectrum).
aModel := SSK.PaneModel picture: anImage.
aModel open
```



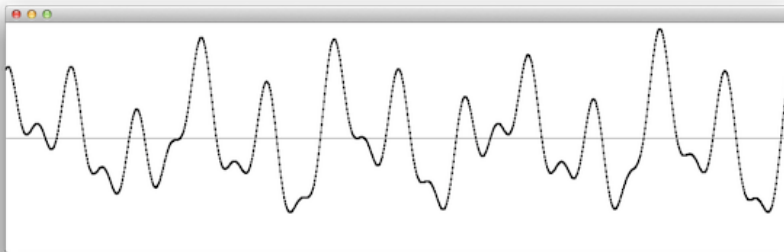
example6 を作る

example6

```
"SSK.DiscreteFourier1dTransformation example6."
```

```
I sourceData aTransformation realPart imaginaryPart anImage aModel powerSpectrum inverseData I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
powerSpectrum := aTransformation powerSpectrum.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
inverseData := aTransformation inverseRealPart.
aTransformation inverseRealPart.
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: inverseData.
aModel := SSK.PaneModel picture: anImage.
aModel open
```

(powerSpectrum (青部分)は proceed しておく)



example7 として、 example{3,4,5,6} をひとまとめにした物を作る

example7

```
"SSK.DiscreteFourier1dTransformation example7."
```

```
I sourceData aTransformation realPart imaginaryPart anImage aModel powerSpectrum inverseData aPoint I
sourceData := SSK.DiscreteFourier1dTransformation dataSampleWave.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
powerSpectrum := aTransformation powerSpectrum.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
inverseData := aTransformation inverseRealPart.
aPoint := 50 @ 50.
(Array
with: #generateImageForData: -> (Array with: sourceData with: 'Source Data')
with: #generateImageForParts: -> (Array
with: (Array with: (aTransformation swap: realPart) with: (aTransformation swap: imaginaryPart))
with: 'Real Part and Imaginary Part')
with: #generateImageForSpectrum:
-> (Array with: (aTransformation swap: powerSpectrum) with: 'Power Spectrum'))
```

```

with: #generateImageForData: -> (Array with: inverseData with: 'Inverse Data')) do:
  [:anAssociation |
  anImage := SSK.DiscreteFourier1dTransformation perform: anAssociation key
              with: anAssociation value first.
  aModel := SSK.PaneModel picture: anImage.
  aModel label: anAssociation value last.
  aModel openAt: aPoint.
  aPoint := aPoint translatedBy: 25 @ 25]

```

青色部分で処理したい物を準備して
 緑色部分でぐるぐる回す

example8 として、ずらっと開いてくれる物を作る

example8

```
"SSK.DiscreteFourier1dTransformation example8."
```

```

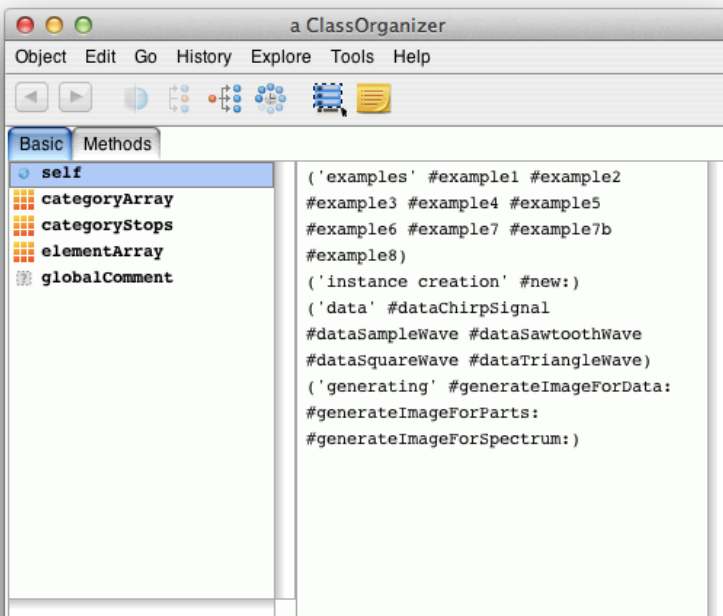
I aPoint |
aPoint := 50 @ 50.
(SSK.DiscreteFourier1dTransformation class organization listAtCategoryNamed: #data) do:
  [:aSelector |
  I sourceData aTransformation realPart imaginaryPart anImage aModel powerSpectrum inverseData |
  sourceData := SSK.DiscreteFourier1dTransformation perform: aSelector.
  aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
  realPart := aTransformation realPart.
  imaginaryPart := aTransformation imaginaryPart.
  powerSpectrum := aTransformation powerSpectrum.
  aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
  inverseData := aTransformation inverseRealPart.
  (Array
   with: #generateImageForData: -> (Array with: sourceData with: 'Source Data')
   with: #generateImageForParts: -> (Array
    with: (Array with: (aTransformation swap: realPart) with: (aTransformation swap: imaginaryPart))
    with: 'Real Part and Imaginary Part')
   with: #generateImageForSpectrum:
    -> (Array with: (aTransformation swap: powerSpectrum) with: 'Power Spectrum')
   with: #generateImageForData: -> (Array with: inverseData with: 'Inverse Data')) do:
    [:anAssociation |
    anImage := SSK.DiscreteFourier1dTransformation perform: anAssociation key
                with: anAssociation value first.
    aModel := SSK.PaneModel picture: anImage.
    aModel label: anAssociation value last.
    aModel openAt: aPoint.
    aPoint := aPoint translatedBy: 25 @ 25]]

```

Do it すると 20枚の Window が開いて大変…
 (データ5種類に波形4種類)

上記、青色部分の説明

SSK.DiscreteFourier1dTransformation class organization を inspect it すると、



こんな感じで、メソッドのリストとかがどさっと返ってくる

SSK.DiscreteFourier1dTransformation class organization listAtCategoryNamed: #'data'
こうすることで、data の物だけが返ってくるようになる

豆知識

Smalltalk には Programming by example. という言葉がある
すぐに実装せずに、example を作れとのこと

DiscreteFourier1dTransformation の Definition を見ると

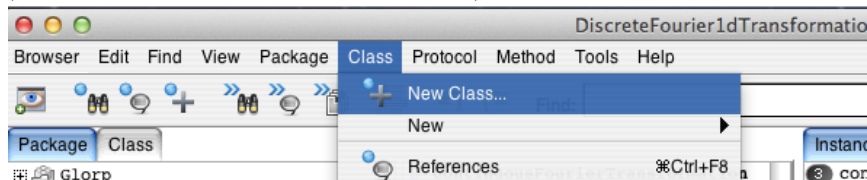
```
Smalltalk.SSK defineClass: #DiscreteFourier1dTransformation
  superclass: #(SSK.DiscreteFourierTransformation)
  indexedType: #none
  private: false
  instanceVariableNames: ''
  classInstanceVariableNames: ''
  imports: ''
  category: 'SSK-Fourier'
```

ApplicationModel をスーパークラスにしたものを使いたい(GUI のウィンドウビルダーが使えるので！)

名前を変えて accept するといつも通り別に増えるので、気にせず変更する

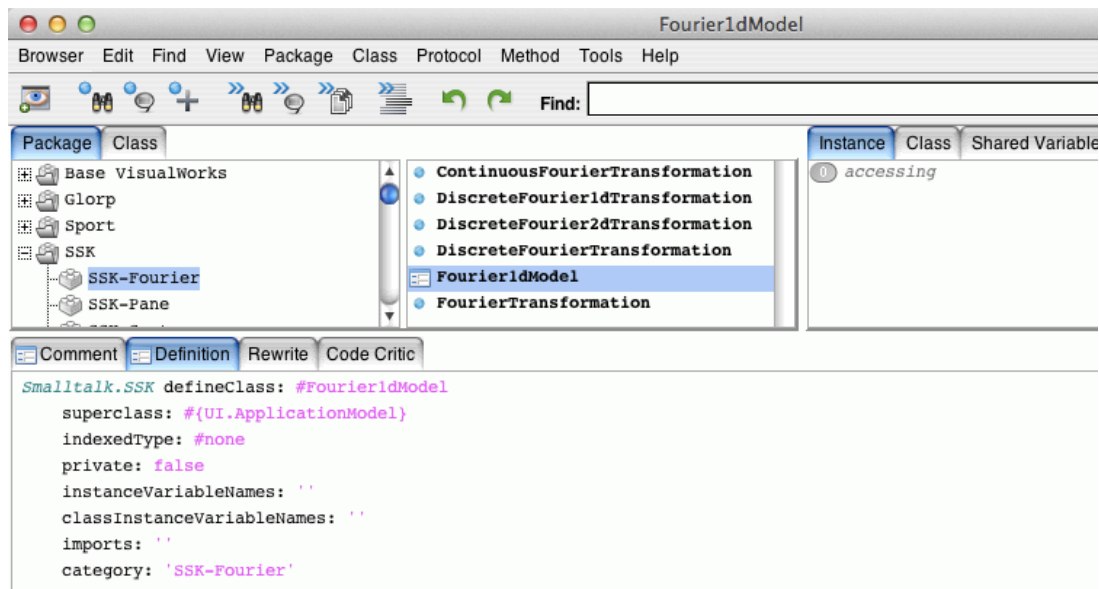
```
Smalltalk.SSK defineClass: #Fourier1dModel
  superclass: #(UI.ApplicationModel)
  indexedType: #none
  private: false
  instanceVariableNames: 'sourceData realPart imaginaryPart powerSpectrum inverseData '
  classInstanceVariableNames: ''
  imports: ''
  category: 'SSK-Fourier'
```

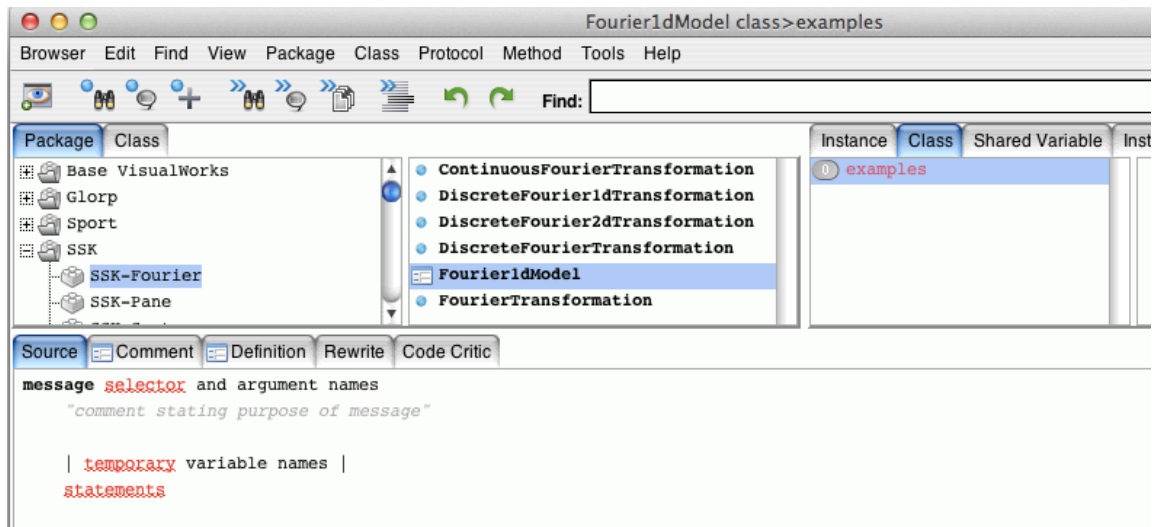
(この New Class という物を使って新しく作ることも出来る)



Comment に 「!」 がつくので適当に付け替える

(例えば、 Copyright 2008-2011 SSK (Smalltalk Studies in Kyoto). All Right Reserved. とか)





example から作る
下準備

Fourier1dModel, Class, examples, example1
example1

```
"SSK.Fourier1dModel example1."

| aModel |
aModel := SSK.Fourier1dModel new.
aModel open.
^aModel
```

こんなのを作るんだぞー！と最低限必要な物を取りあえず書いておく

で、改めてやりたいことに即して書き直し

example1

```
"SSK.Fourier1dModel example1."

| aModel |
aModel := SSK.Fourier1dModel new: SSK.DiscreteFourier1dTransformation dataSampleWave.
aModel open.
^aModel
```

インスタンスを作れるように

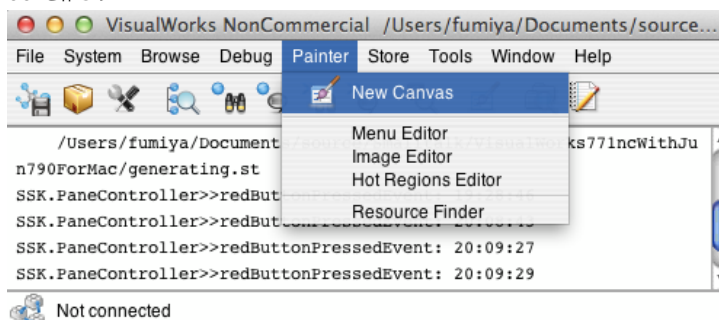
Fourier1dModel, Class, instance creation, new:
new: sourceData

```
^(self new)
  setSourceData: sourceData;
  yourself
```

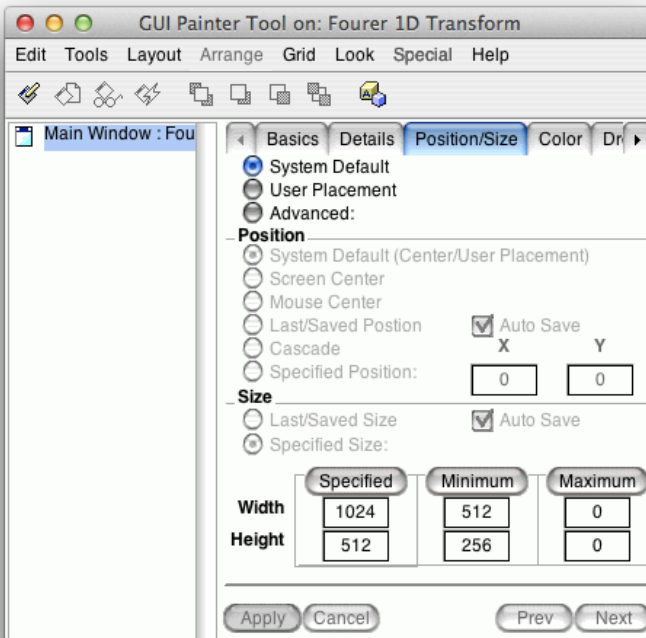
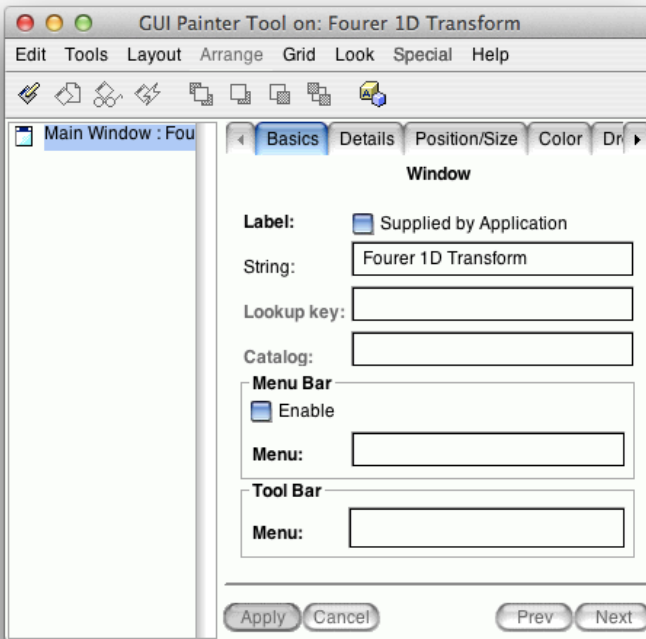
Fourier1dModel, Instance, private, setSourceData:
setSourceData: aCollection

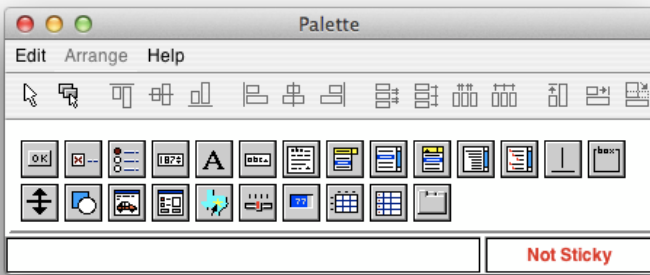
```
| aTransformation |
sourceData := aCollection.
aTransformation := SSK.DiscreteFourier1dTransformation new: sourceData.
realPart := aTransformation realPart.
imaginaryPart := aTransformation imaginaryPart.
powerSpectrum := aTransformation powerSpectrum.
aTransformation := SSK.DiscreteFourier1dTransformation new: realPart with: imaginaryPart.
inverseData := aTransformation inverseRealPart
```

GUI を作ろう



4つの物を入れる外枠作り
こんなのが開く

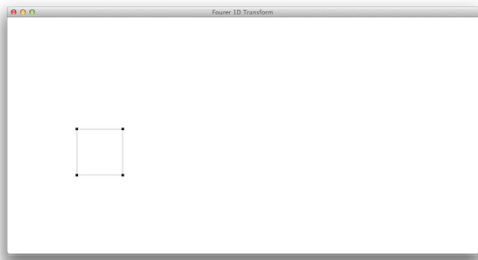




の

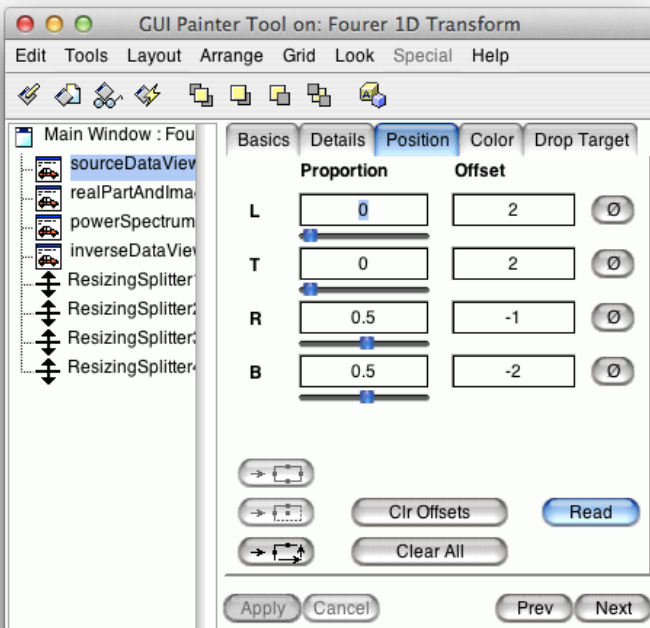


これ View Holder を使う

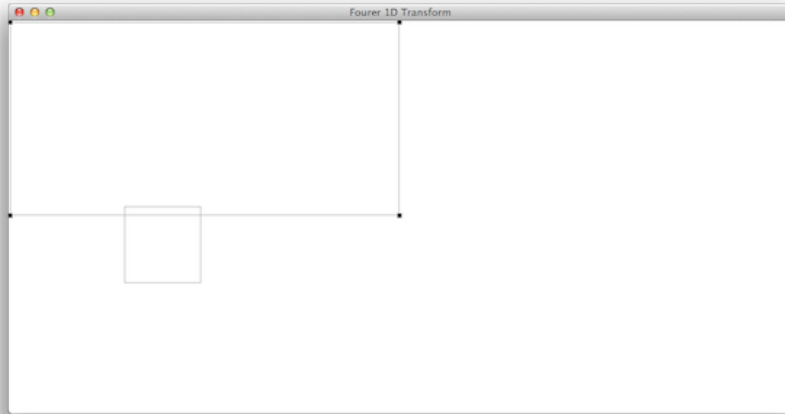


適当に設置

変更したい View を選択して、



設定

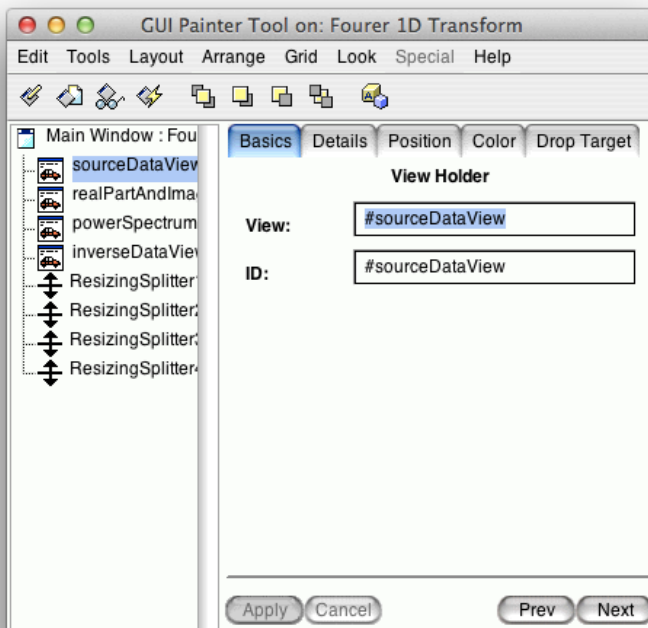


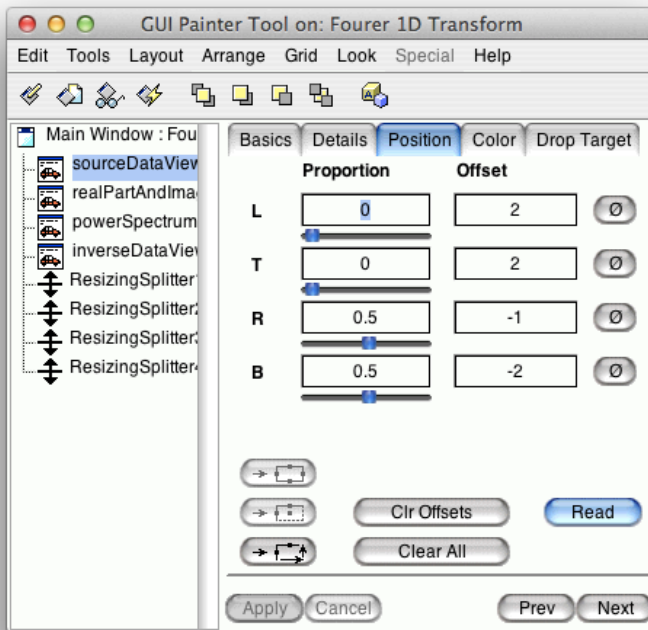
こんな感じで調整



一番上の Copy Widget を使って複製を作れる(Paste Widget で貼り付け)

設定は





ほか3つは文字で

View, ID 共に

	L(Proportion, Offset)	T	R	B
#sourceDataView	0,2	0,2	0.5,-1	0.5,-2
#realPartAndImaginaryPartView	0.5,1	0,2	1,-2	0.5,-2
#powerSpectrum	0,2	0.5,1	0.5,-1	1,-2
#inverseDataView	0.5,1	0.5,1	1,-2	1,-2

この座標系は左下基点になっている

Proportion は長さ Offset で調節

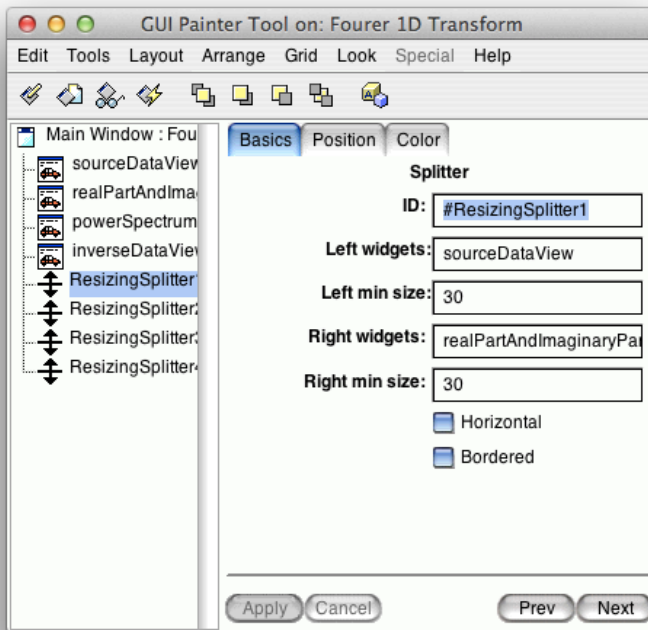
resize に対応するために今度は



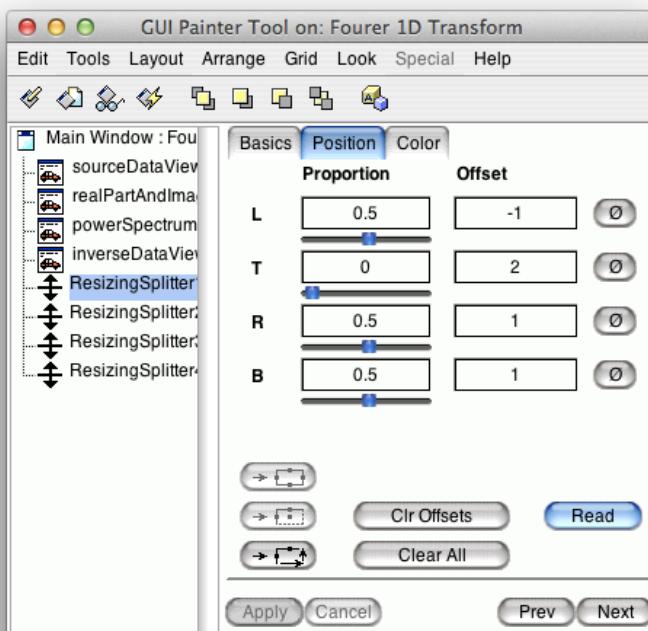
を使う

また、適当におく

同様にコピーして4つ用意

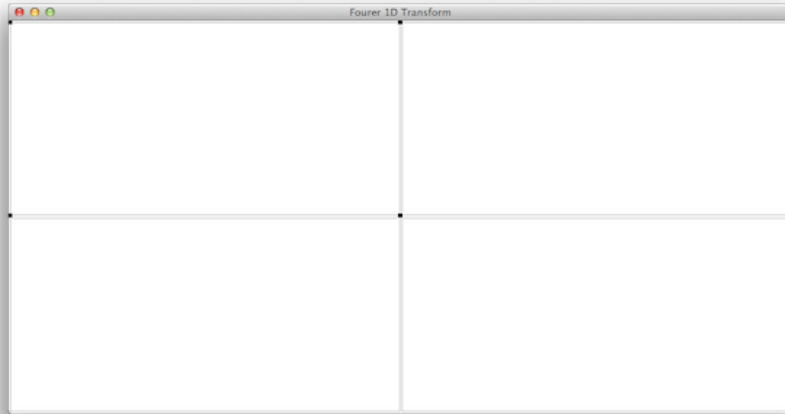


Horizontal で縦横切り替え(オフで縦向きのパール出来る(横向きに resize できる))
 Bordered で resize 用の線を見せるか見せないかを設定

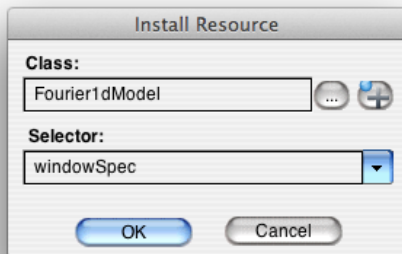


ID(ここは何でも良い)	Horizontal	Left(Top) widget	Right(Bottom) widget	L	T	R	B
#ResizingSplitter1	Off	sourceDataView	realPartAndImaginaryPartView	0.5,-1	0,2	0.5,1	0.5,1
#ResizingSplitter2	On	realPartAndImaginaryPartView	inverseDataView	0.5,1	0.5,-1	1,-2	0.5,1
#ResizingSplitter3	Off	powerSpectrumView	inverseDataView	0.5,-1	0.5,-1	0.5,1	1,-2
#ResizingSplitter4	On	sourceDataView	powerSpectrumView	0,2	0.5,-1	0.5,-1	0.5,1

こんな感じになったら



で Install (もしくは、Edit から)
Install 先はこれ



先ほど、指定した View を実装する
Fourier1dModel, Instance, aspects, sourceDataView
sourceDataView

```
I aView anImage aModel I  
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: sourceData.  
aModel := SSK.PaneModel picture: anImage.  
aModel label: 'Source Data'.  
aView := PaneView model: aModel.  
aView alignmentSymbol: #center.  
^aView
```

ほか3つはとりあえず何も描画しない状態で実装...していたのだが、間に合わなかったのとおりあえず同じような感じで書いておく

Fourier1dModel, Instance, aspects, inverseDataView
inverseDataView

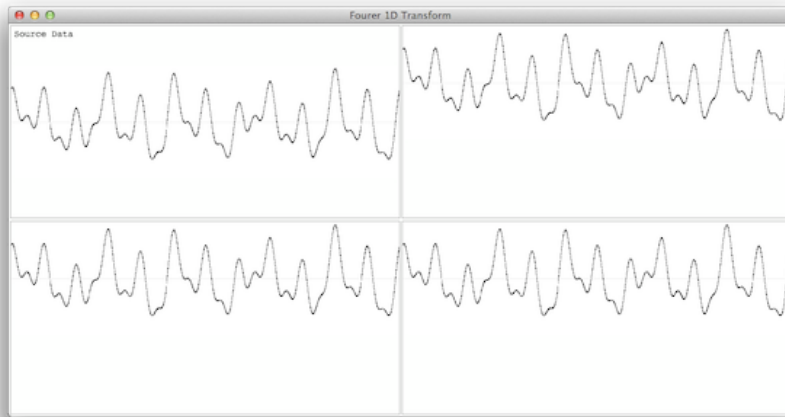
```
I aModel aView anImage I  
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: sourceData.  
aModel := SSK.PaneModel picture: anImage.  
aView := PaneView model: aModel.  
^aView
```

Fourier1dModel, Instance, aspects, powerSpectrumView
powerSpectrumView

```
I aModel aView anImage I  
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: sourceData.  
aModel := SSK.PaneModel picture: anImage.  
aView := PaneView model: aModel.  
^aView
```

Fourier1dModel, Instance, aspects, realPartAndImaginaryPartView
realPartAndImaginaryPartView

```
I aModel aView anImage I  
anImage := SSK.DiscreteFourier1dTransformation generateImageForData: sourceData.  
aModel := SSK.PaneModel picture: anImage.  
aView := PaneView model: aModel.  
^aView
```



こうなる
 (ただし、左上以外は本来表示したい物と異なる)

今回はここまで！

コレを使って保存

```

programmanager class>>save
Browser Edit Find View Package Class Protocol Method Tools Help
Find:
Package Class
├─ Glorp
├─ Sport
├─ SSK
│  ├─ SSK-Fourier
│  ├─ SSK-Pane
│  └─ SSK-System
Instance Class Shared Variable Instance Va
6 accessing
2 comments
1 saving
Source Comment Definition Rewrite Code Critic
save
"SSK.ProgramManager save."

| aBundle dateString timeString aDirectory aFilename aFileManager |
aBundle := self bundle.
dateString := JunCalendarModel stringFromDate select: [:each | each isDigit].
timeString := JunCalendarModel stringFromTime select: [:each | each isDigit].
aDirectory := Filename defaultDirectory directory.
aFilename := aDirectory construct: 'SSK_' , dateString , '_' , timeString , '.st'.
Cursor write showWhile:
  [aFileManager := SourceCodeStream write: aFilename.
  [aBundle fileOutOn: aFileManager] ensure: [aFileManager close]]
  
```