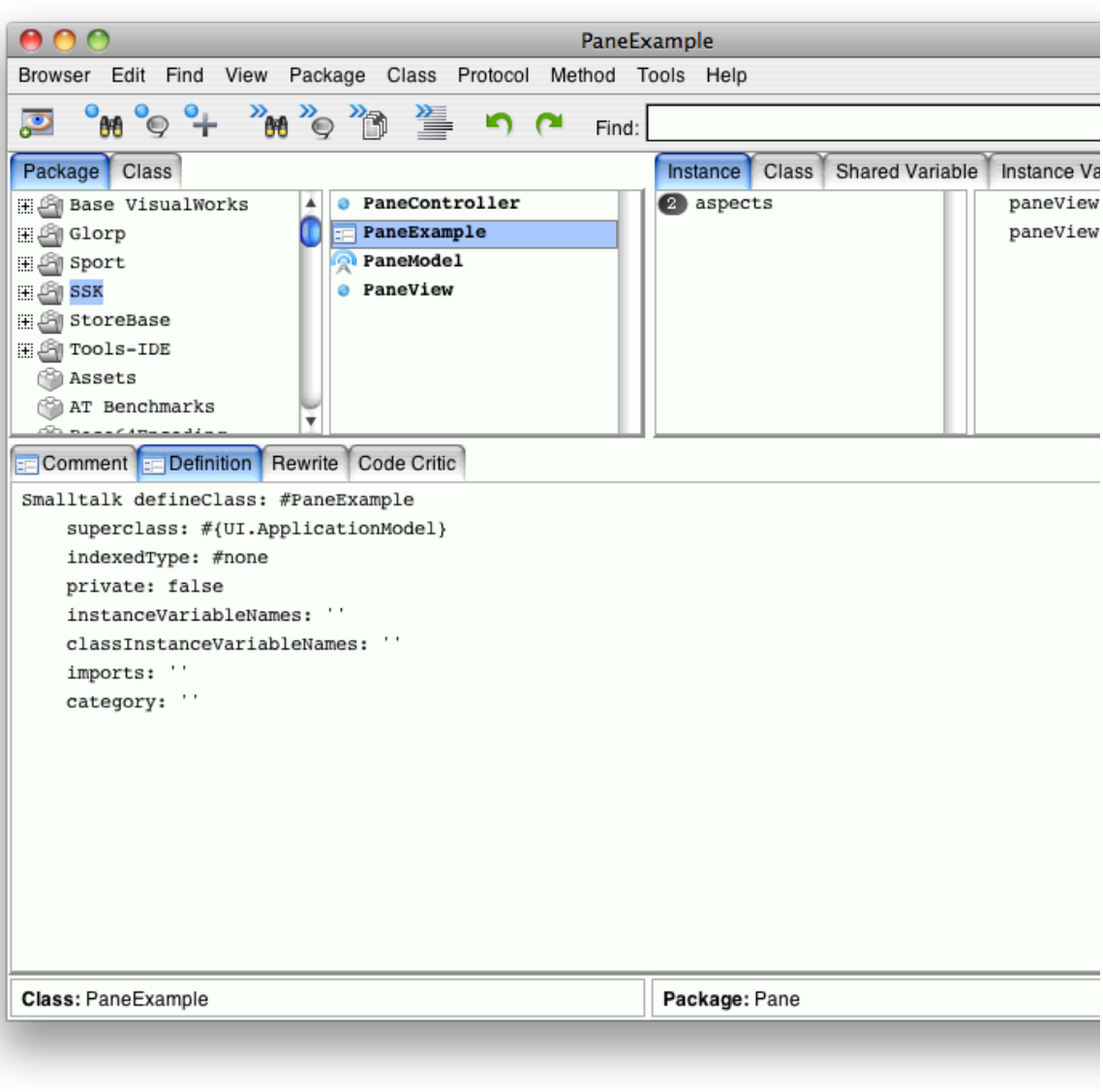
今回の USB メモリ内のファイルは前回と変わらず
pictures
.st が2つ

VisualWorks771ncWithJun790ForMac ディレクトリの構成は以下のように
pictures
Smalltalk.app
SSK_PaneMVC_20110511.st
VisualWorksWithJun/

Smalltalk.app を用いて起動
File Browser から File in する
System Browser から SSK が入っているかどうかを確認

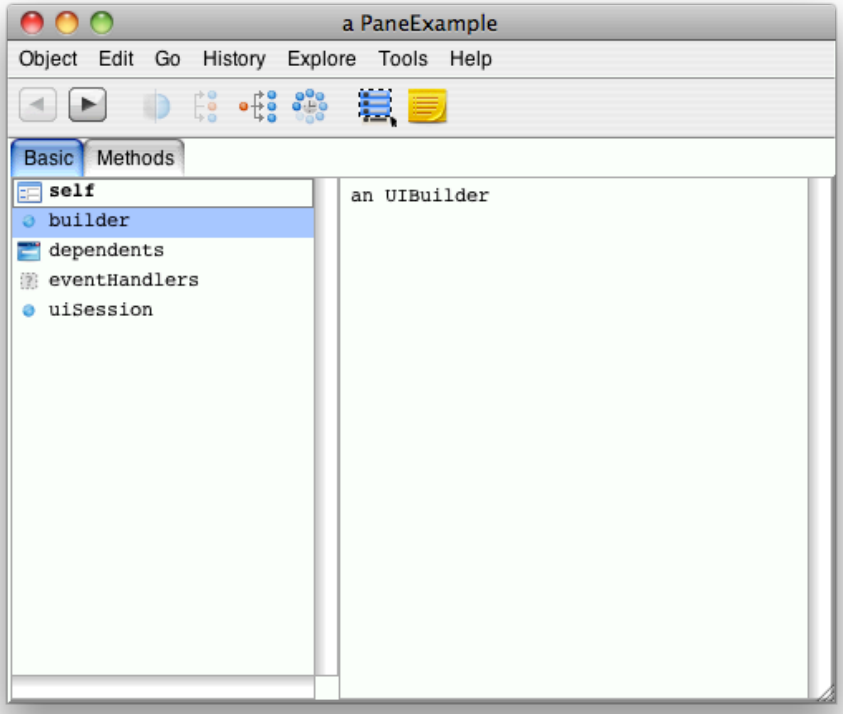
=====復習
PaneExample は Application Model がスーパークラス



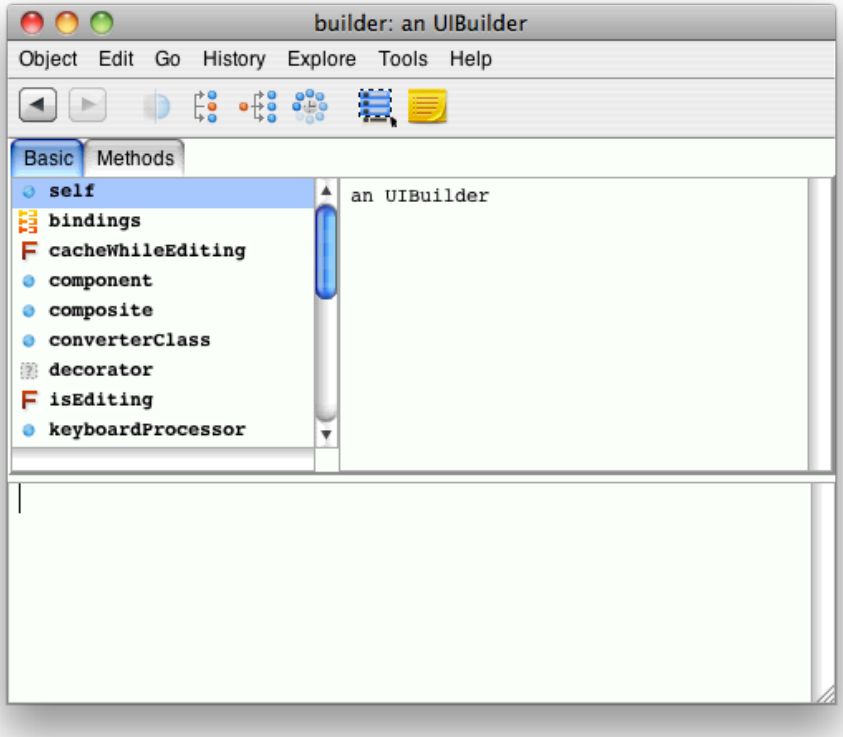
先月、最後にいじったのは example32 でしたね。

example32 で、最後の anExample が最後に return されるのは何処よ？
2つの窓はいったいどこから何処へ？
dependents に表示されているものをどんどんたどっていくと、行き着くところに行き着く
Builder に向かって、ウィンドウの名前 #○○ で、訪ねることができる

builder をダブルクリックで、 builder が self に



右上の黄色いので、 workspace を開いて



self componentAt: #paneView1 <-- print it すると
a SpecWrapper on: a BorderedWrapper on: a PaneView

(self componentAt: #paneView1) widget <-- print it すると
a PaneView

PaneExample から見ると
(self builder componentAt: #paneView1) widget <-- print it
a PaneView

builder の中の component には一つしかないのだが、これは直近の component しか表示してくれない
全ての component を見るには、 builder 中の namedComponents に全ての component を見る
が、この様に、 namedComponents を見るのではなくて、上でやったような方法で追いかけるようにしましょう。
IdentityDictionary (#paneView2 -> a SpecWrapper on: a BorderedWrapper on: a PaneView #paneView1 -> a SpecWrapper on: a BorderedWrapper on: a PaneView)

何回も追っていけば、 dependent からちゃんとたどれます。
いっぱいたどらないと行けなくて大変だけど
昔はこれしかなかったそうです。

=====復習終わり

MVC なので、どこから攻めますかね？
Model から攻めることに

PaneModel を覗くと、 Instance の accessing の中には label と picture はあるし、 initialize-release もとにかく作ってある(自動生成)けど、どうにもならないので修正

label と picture は自動生成の内容でかまわないが、 インデントがちょっとおかしいので、 Format をして、 Accept しておく

initialize-release の中の initialize を書き換える
initialize

```
super initialize.
label := nil.
picture := nil.
^self
```

label: を修正

label: anObject
```
label := anObject
```

と、 anObject はちょっと、おおざっぱすぎるので、修正

2種類あり
label: aString

```
(aString isKindOf: String) ifFalse: [^nil].
label := aString
```

こちらは、 String でなかったら捨ててしまう。

こちらの方が Smalltalk っぽい
label: aString

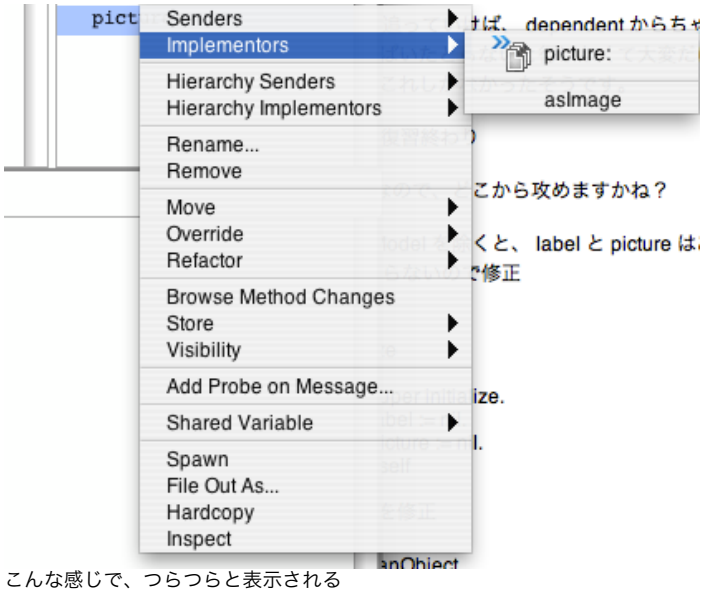
```
label := aString asString
```

何とかして、 String に変えるように頑張る。

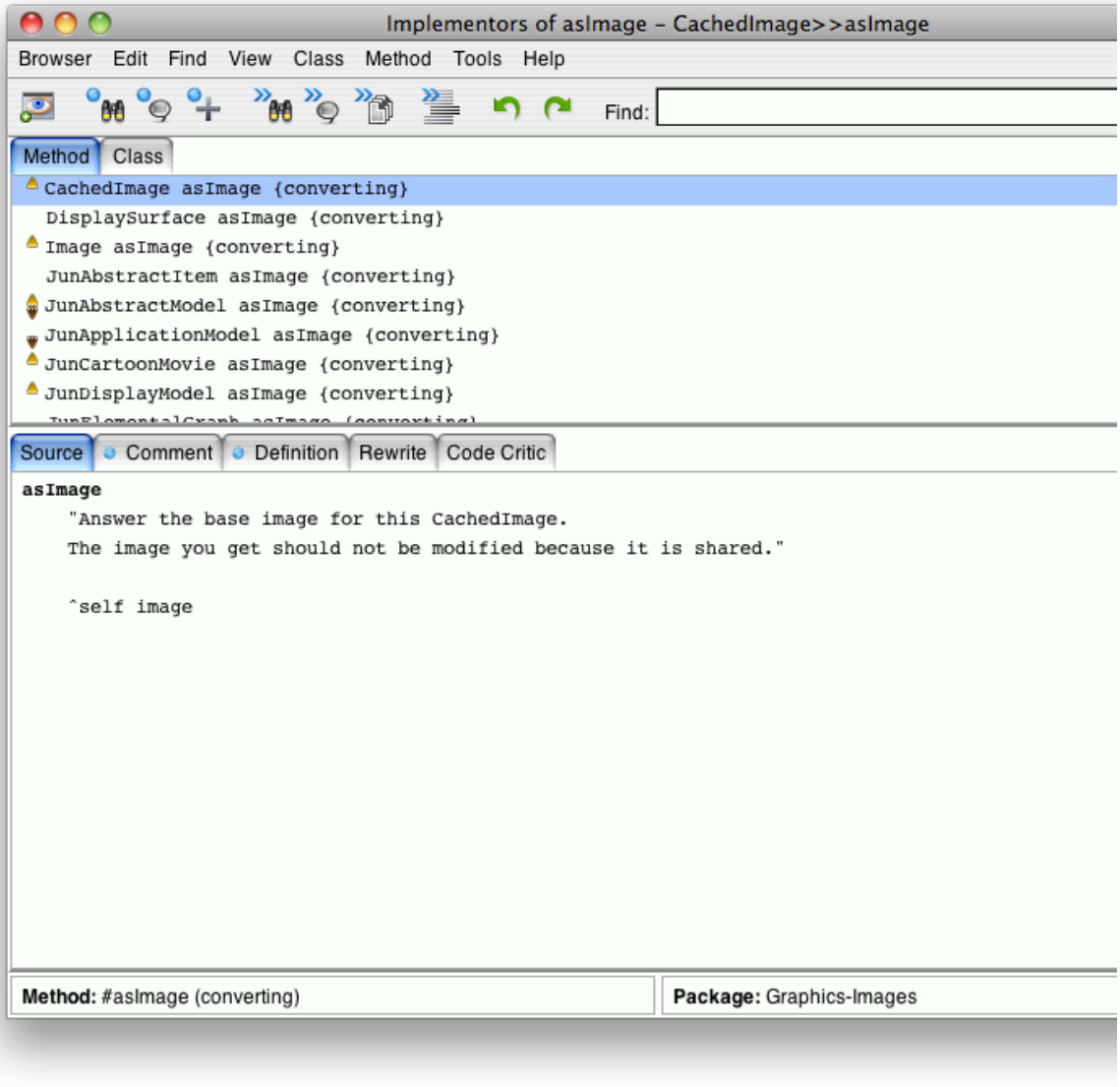
picture: も同様に修正
picture: anImage

```
picture := anImage asImage
```

asImage が出来ること
asImage を選択



こんな感じで、つらつらと表示される

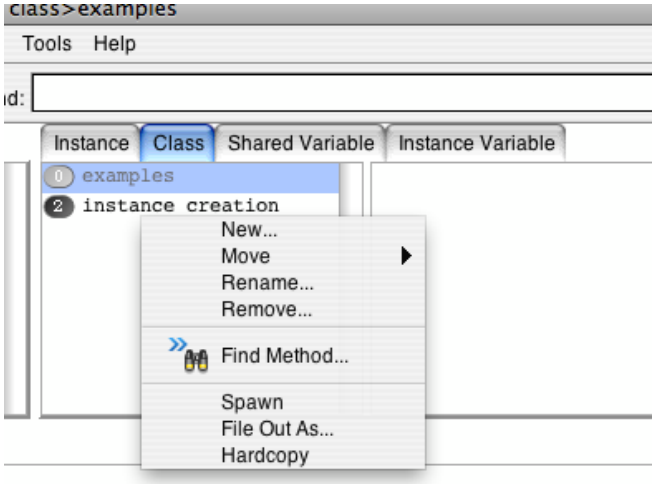


PaneModel の クラスメソッド instance creation の new を変更

new

```
	^(super new)
		initialize
```
よりも、
new
```
	^(super new)
		initialize;
		youself
```
の方が、ロバスト(頑強)

PaneModel に 新しいクラスメソッドを追加する



exapmle1 を追加

とにかくおきまりの
example1
```
	"PaneModel example1."

	| aModel |
	aModel := PaneModel new.
	^aModel
```

aModel に対して、いろいろしたくなりますよね

example1
```
	"PaneModel example1."

	| aModel |
	aModel := PaneModel new.
	aModel label: 'fumiya'.
	aModel picture: (JunImageUtility fromDisplay: (0 @ 0 extent: 256 @ 256)). "ディスプレイから一部分を切り取る"
	^aModel
```

Inspect it してみる
dependents には何もない
label には先ほど入力した物 # ここでは、 fumiya
picture には切り取った物 # picture をダブルクリックして、 self を見てみると、確かに、ディスプレイ左上を切り取った物がある

example シリーズは将来の自分のためにこのクラスの使い方をメモしておくのだ

example2 をいじることに
今度はディスプレイから切り取るのではなく、ファイルから読み込もう

example2
```
	"PaneModel example2."

	| aModel aDirectory |
	aModel := PaneModel new.
	aModel label: 'fumiya'.
	aDirectory := Filename defaultDirectory directory construct: 'pictures'. "<-- directory で一つ上のディレクトリに上がる"
	aDirectory inspect. "<-- Do it で、 Inspect it と同様の効果が得られる"
	aModel picture: (JunImageUtility fromDisplay: (0 @ 0 extent: 256 @ 256)).
	^aModel
```

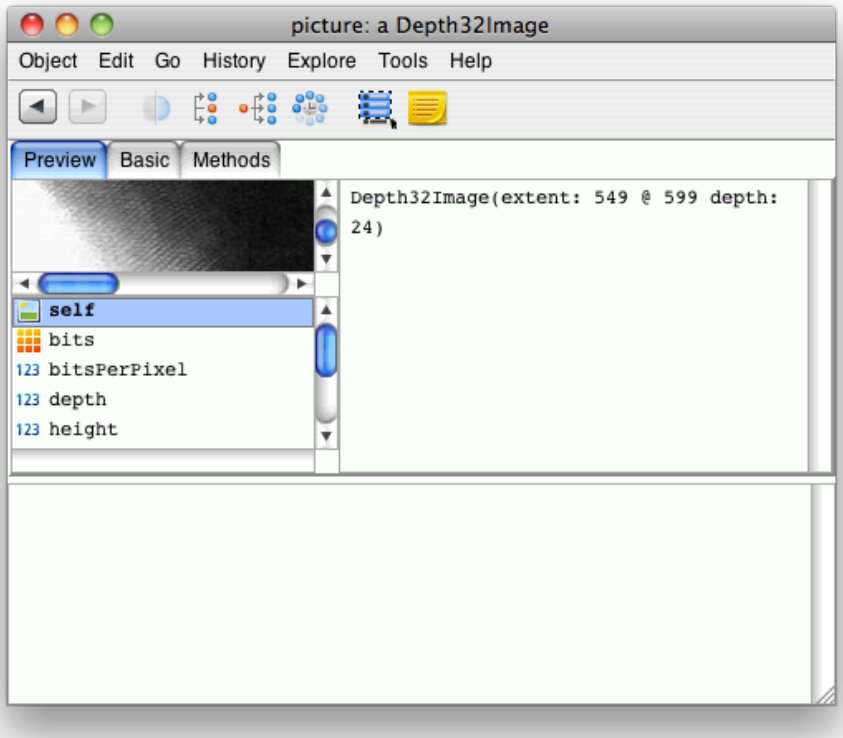
defaultDirectory とは current Directory のこと # 僕は VisualWorksWithJun
/ や :\ で、ディレクトリ操作(移動)しようとしないこと
他のプラットホームで動かなくなるので
inspect it して、本当に目的のディレクトリが取得できているのか確認
a MacOSXFilename('/Users/fumiya/Documents/src/Smalltalk/VisualWorks771ncWithJun790ForMac/pictures')

directory を抜いた状態で実行してみると
a MacOSXFilename('/Users/fumiya/Documents/src/Smalltalk/VisualWorks771ncWithJun790ForMac/VisualWorksWithJun/pictures')
おおー。予想通り

inspect 先で
self asURI asString <-- print it
'file:///Users/fumiya/Documents/src/Smalltalk/VisualWorks771ncWithJun790ForMac/pictures'
URI や URL では、 / とか使っちゃっても問題ない

example2
```
"PaneModel example2."

| aModel aDirectory aFilename anImage |
aModel := PaneModel new.
aModel label: 'fumiya'.
aDirectory := Filename defaultDirectory directory construct: 'pictures'.
aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
anImage := JunImageUtility fromFile: aFilename.
aModel picture: anImage.
^aModel
```

Inspect it して、 picture を確認してみると、確かにファイルを取得できている



次はラベルがちゃんと、人物の名前になるほうがいいよねぇ。

Find Class で、 Filename クラスをひいてみる
そうすると、先ほど使った asURI はちゃんとある
asURI

```
^FileURL fromFilename: self
```

ずらずら見てみると…
tail というものが見つかる

リスト要素の一番最後の物をくれる

```
tail
	"Answer the filename suffix as a String."

	| index nm |
	nm := self asString.
	(index := self lastSeparatorIndex) notNil
		ifTrue:	[^nm copyFrom: index+1 to: nm size]
		ifFalse:	[^nm copy]


example2
	"PaneModel example2."

	| aModel aDirectory aFilename anImage |
	aModel := PaneModel new.
	aDirectory := Filename defaultDirectory directory construct: 'pictures'.
	aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
	anImage := JunImageUtility fromFile: aFilename.
	aModel picture: anImage.
	aModel label: aFilename tail. "<-- ここは、 label: の側で、 asString で、文字列にするようになっているので、なんとか文字列にしてくれるはずである"
	^aModel
```

Inspect it すると、

```
'BernhardRiemann.jpg'
```

と、ちゃんとファイル名が帰ってくる

さてさて、 label にするには .jpg が邪魔だな

.jpg を引っぺがすようなメソッドもちゃんと存在するらしい

Filename の中に、 extension というメソッドがある

実装を見てみると、

```
extension
	"Answer the receiver's extension if any.  This is the characters from the
	 last occurrence of a period to the end, inclusive. E.g. the extension of
	 'visual.sou' is '.sou'. Answer nil if none.  Note that e.g. .login has no
	 extension."

	| string periodIndex |
	string := self tail.
	periodIndex := string lastIndexOf: $..
	^periodIndex > 1 ifTrue: [string copyFrom: periodIndex to: string size]
```

periodIndex って、いいのかそれ

Instance の Utilities の中に色々あったり

Instance の parsing というものに splitExtension なるものもあるぞ

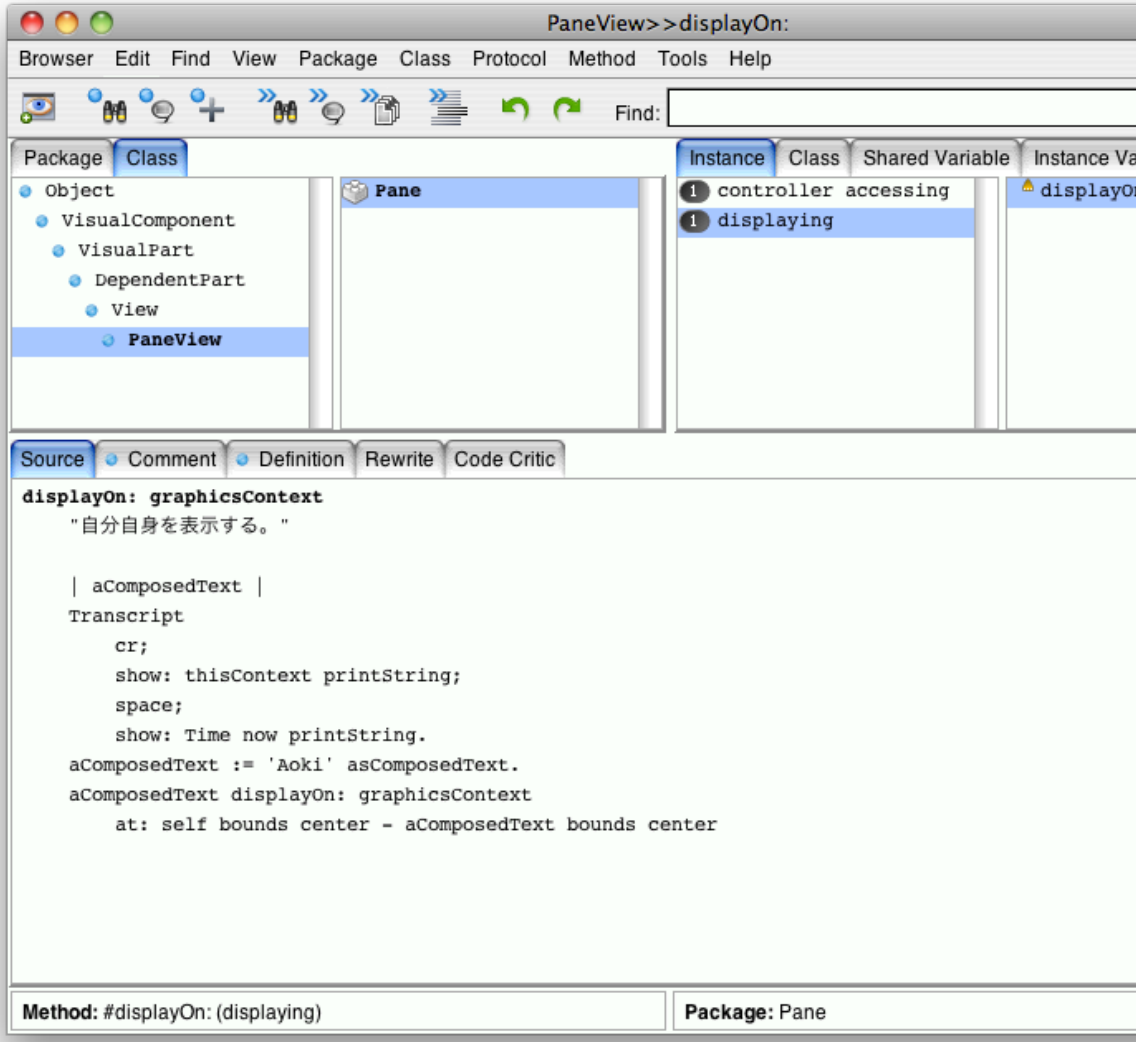
```
example2
	"PaneModel example2."

	| aModel aDirectory aFilename anImage |
	aModel := PaneModel new.
	aDirectory := Filename defaultDirectory directory construct: 'pictures'.
	aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
	anImage := JunImageUtility fromFile: aFilename.
	aModel picture: anImage.
	aModel label: (Filename splitExtension: aFilename tail) first. "<-- ただし、ちょっとお下品"
	^aModel
```

```
'BernhardRiemann'
```

と、 .jpg がとれました

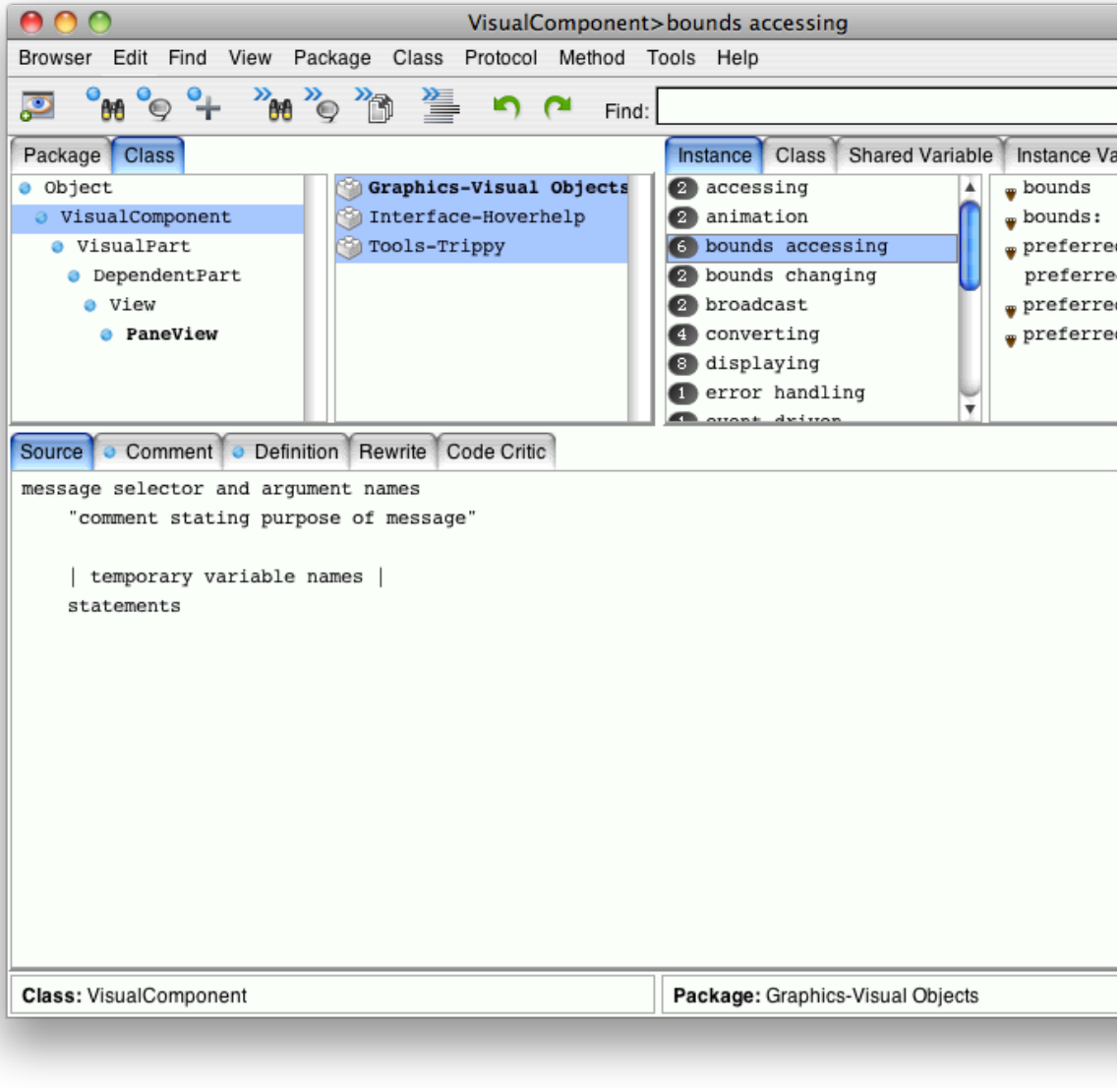
次は、 View をいじるぞ

PaneView の Class 構造を見てみる

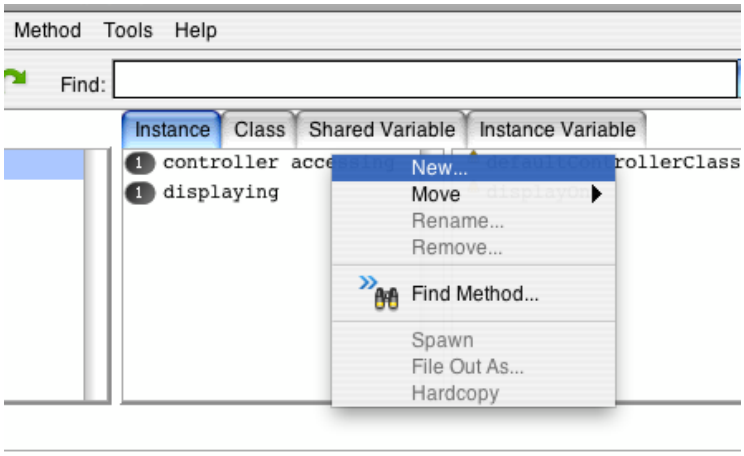


ずいぶん深い
VisualComponent には
bounds accessing の下に bounds というものがある
中身はコメントだらけ
Smalltalk でこういう類の物は hook に違いないと思うわけですよ

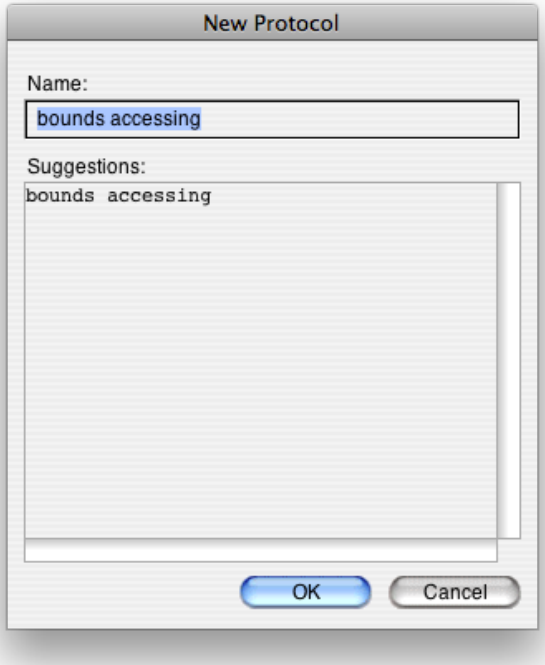
ということで、選択して、
bounds: newBounds だけまるまるコピーしておく
bounds accessing を選択しておいて、



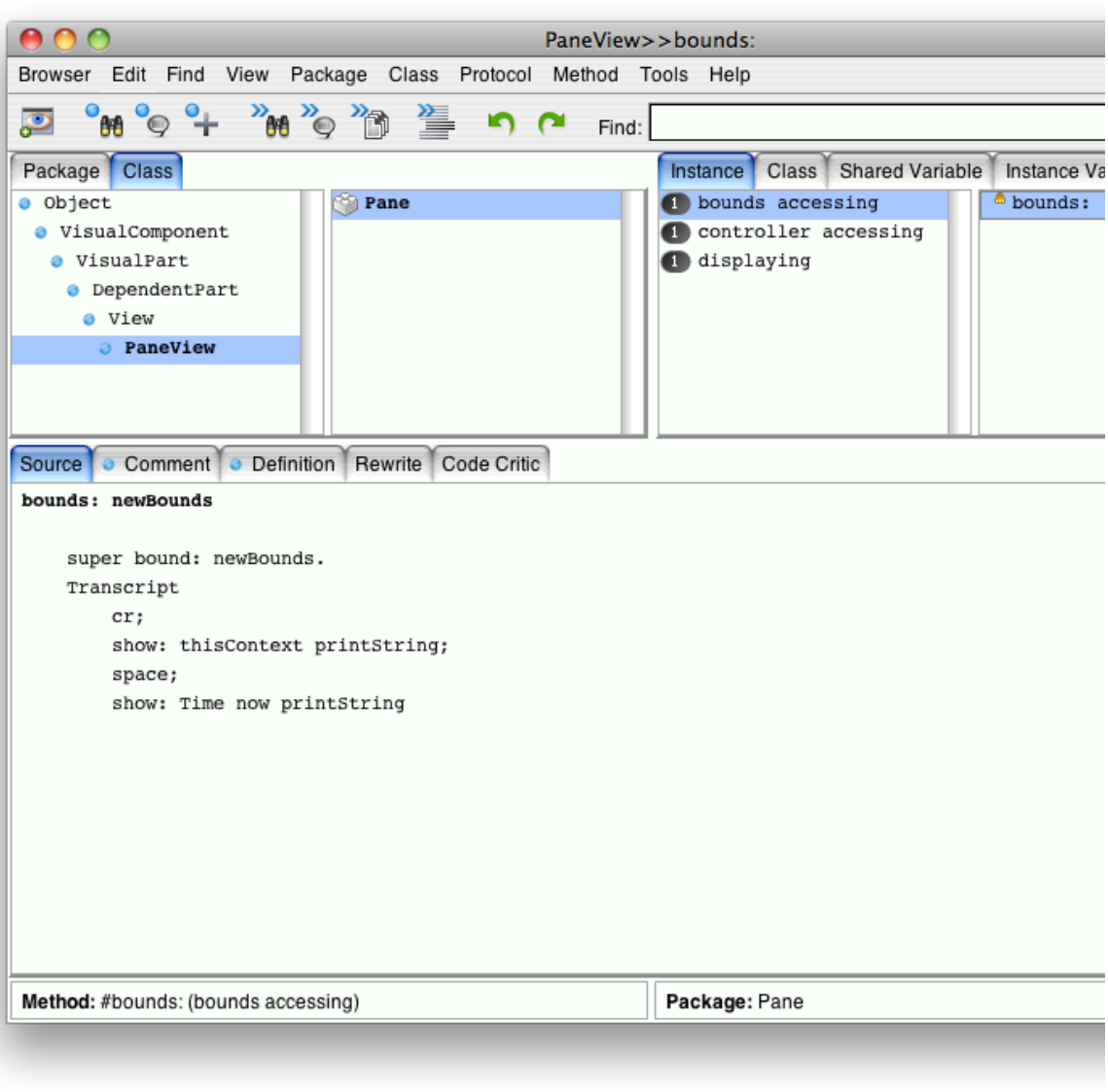
PaneView の Instance で New を選んで、



勝手に先ほどの物を選んでくれると



bounds accessing に先ほどコピーしておいて bounds: newBounds を追加



上から持ってきたメソッドは何もしていないとコメントに書いてあっても、実は何かしているかも知れない(クラス構造が深いのでなおさら)ので、ちゃんと、 super は呼び出しておくこと

bounds: newBounds

```
super bounds: newBounds.
Transcript
    cr;
    show: newBounds printString;
    space;
    show: Time now printString
```

example32 を実行してみると、確かに、これが呼び出されていることが分かる

0 @ 0 corner: 333 @ 274 20:12:02 <-- この
0 @ 0 corner: 333 @ 274 20:12:02 <-- 2つ
PaneView>>displayOn: 20:12:02
PaneView>>displayOn: 20:12:02

PaneExample の paneView{1,2} を変更
とりあえず、窓の数だけ、あるからなぁ…

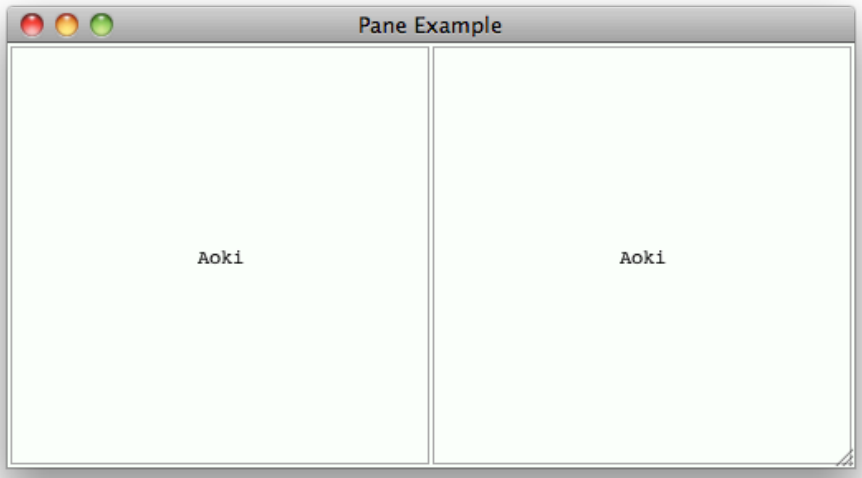
paneView1

```
| aModel aView aDirectory aFilename anImage |
aModel := PaneModel new.
aDirectory := Filename defaultDirectory directory construct: 'pictures'.
aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
anImage := JunImageUtility fromFile: aFilename.
aModel picture: anImage.
```

```
	aModel label: (Filename splitExtension: aFilename tail) first.
	aView := PaneView model: aModel.
	^aView
```

paneView2

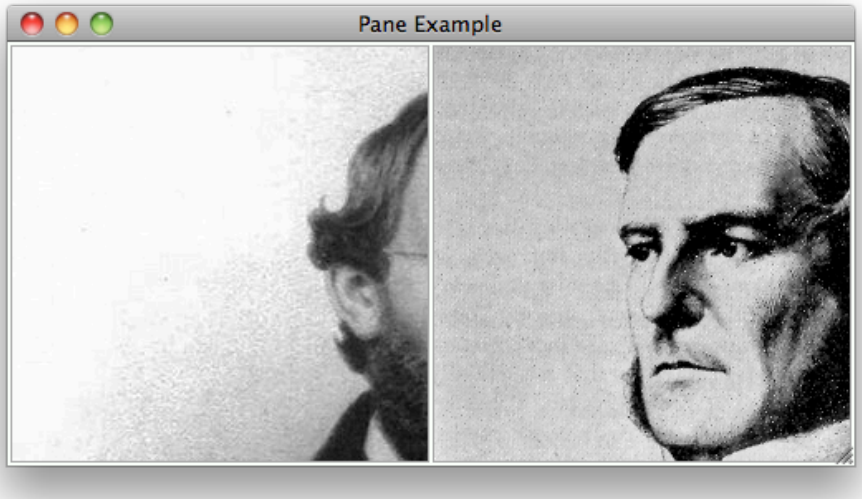
```
	| aModel aView aDirectory aFilename anImage |
	aModel := PaneModel new.
	aDirectory := Filename defaultDirectory directory construct: 'pictures'.
	aFilename := aDirectory construct: 'GeorgeBoole.jpg'.
	anImage := JunImageUtility fromFile: aFilename.
	aModel picture: anImage.
	aModel label: (Filename splitExtension: aFilename tail) first.
	aView := PaneView model: aModel.
	^aView
```



現時点では、何も変わっていない
ただし、画像自体は来ているはずなのである

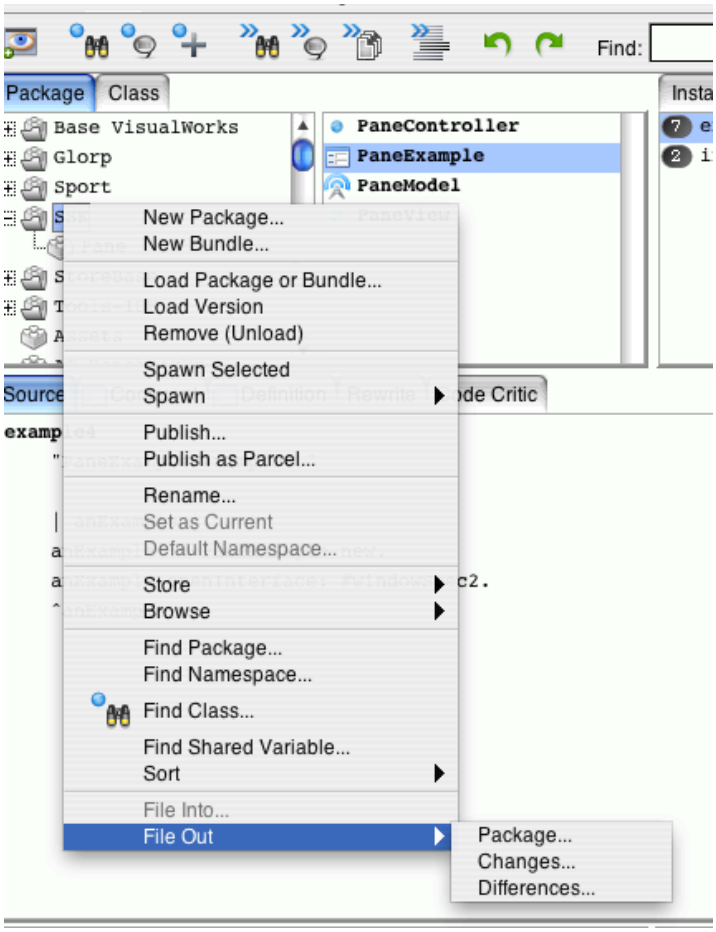
PaneView の displaying の displayOn: を変更する

```
displayOn: graphicsContext
	"自分自身を表示する。"

	self model picture ifNotNil: [:anImage | anImage displayOn: graphicsContext]
```



縮小してないし、アスペクト比とかも特に気にしてないから見切れてますな

結構進んだので、ちゃんと保存しましょう
Package を選んで



paneView の赤色部分は、誰がやるのが正しいのでしょうか？
paneView1

```
| aModel aView aDirectory aFilename anImage |
aModel := PaneModel new.
aDirectory := Filename defaultDirectory directory construct: 'pictures'.
aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
anImage := JunImageUtility fromFile: aFilename.
aModel picture: anImage.
aModel label: (Filename splitExtension: aFilename tail) first.
aView := PaneView model: aModel.
^aView
```

ファイルを持ってくるのは PaneExample の仕事なのか、 PaneModel の仕事なのか

Example の 都合を Model に突っ込むのはちょっと、気が引けるので、これは、 Example に残したままにしましょう
方針としては、ファイル名を与えると、ちゃんと画像を表示してくれるような感じが良いかな

PaneExample の Instance に defaults を追加
こんなメソッドを追加
defaultDirectoryOfPictures

```
^Filename defaultDirectory directory construct: 'pictures'
```

目的のディレクトリを返すメソッド

この様に書き換え
paneView1

```
| aModel aView aFilename anImage |
aModel := PaneModel new.
aFilename := self defaultDirectoryOfPictures construct: 'BernhardRiemann.jpg'.
anImage := JunImageUtility fromFile: aFilename.
aModel picture: anImage.
aModel label: (Filename splitExtension: aFilename tail) first.
```

```
	aView := PaneView model: aModel.
	^aView
```

赤色部分が変わったところ
ディレクトリを教えてくれるようになった

defaults は Instance にも Class にも必要なので、 Class にまるっとコピー
Class 側
defaultDirectoryOfPictures

```
	^Filename defaultDirectory directory construct: 'pictures'
```

Instance 側
defaultDirectoryOfPictures

```
	^self class defaultDirectoryOfPictures
```

Instance 側は Class 側にデリゲートすればよいのだ
ただし、ここで、 Instance 側を消してしまうと、Class 側を呼び出さなければならなくなり、呼び出しが煩雑になる

PaneExample の Inspect に、 private として、 PaneView にいちいちごちゃごちゃと書かなくても、ファイル名(拡張子付き)を渡すと自動生成してくれる物を作る

createPaneModel: filenameString

```
	| aModel aFilename anImage |
	aModel := PaneModel new.
	aFilename := self defaultDirectoryOfPictures construct: filenameString.
	anImage := JunImageUtility fromFile: aFilename.
	aModel picture: anImage.
	aModel label: (Filename splitExtension: aFilename tail) first.
	^aModel
```

PaneView{1,2} を書き直す
paneView1

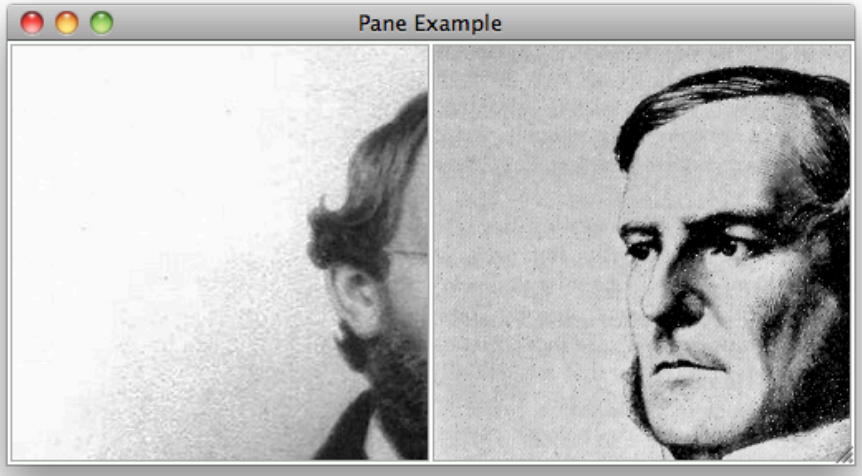
```
	| aModel aView |
	aModel := self createPaneModel: 'BernhardRiemann.jpg'.
	aView := PaneView model: aModel.
	^aView
```

paneView2

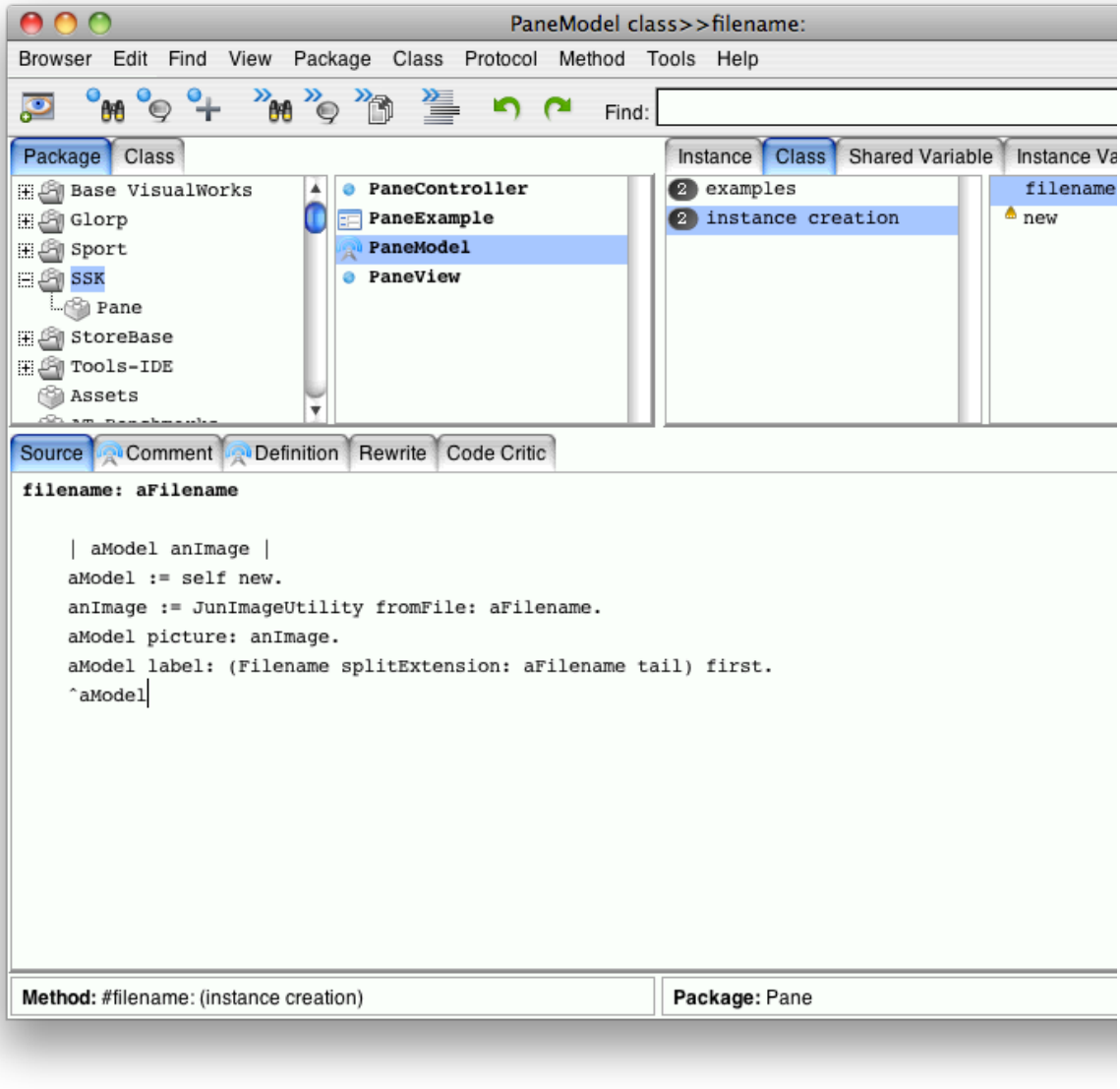
```
	| aModel aView |
	aModel := self createPaneModel: 'GeorgeBoole.jpg'.
	aView := PaneView model: aModel.
	^aView
```

ずいぶんすっきりと
さすがに、これらは書いておかないとどうしようもない

先ほどと変わらず



PaneModel の instance creation に filename からインスタンスを作れるようにするメソッドを追加する



filename: aFilename

```
	| aModel anImage |
	aModel := self new.
	anImage := JunImageUtility fromFile: aFilename.
	aModel picture: anImage.
	aModel label: (Filename splitExtension: aFilename tail) first.
	^aModel
```

PaneExample の private の createPaneModel: を書き直して

```
createPaneModel: localnameString

	| aFilename aModel |
	aFilename := self defaultDirectoryOfPictures construct: localnameString.
	aModel := PaneModel filename: aFilename.
	^aModel
```

PaneModel の example2 を変更

```
example2
	"PaneModel example2."

	| aModel aDirectory aFilename |
	aDirectory := Filename defaultDirectory directory construct: 'pictures'.
	aFilename := aDirectory construct: 'BernhardRiemann.jpg'.
	aModel := PaneModel filename: aFilename.
	^aModel
```

こんな感じに書き直すことも出来ますよ

最後は忘れずに保存して