

Tools shaped our mind...

<http://stephane.ducasse.free.fr>
<http://www.pharo.org>

Inria

 Université
de Lille

 cnrs

Tools

- Shape our mind....
- Get moldable tools so that **you CAN adapt them to you and your process** and not the inverse
- Build fast **your own** tools

Pharo has **amazing**
moldable tools

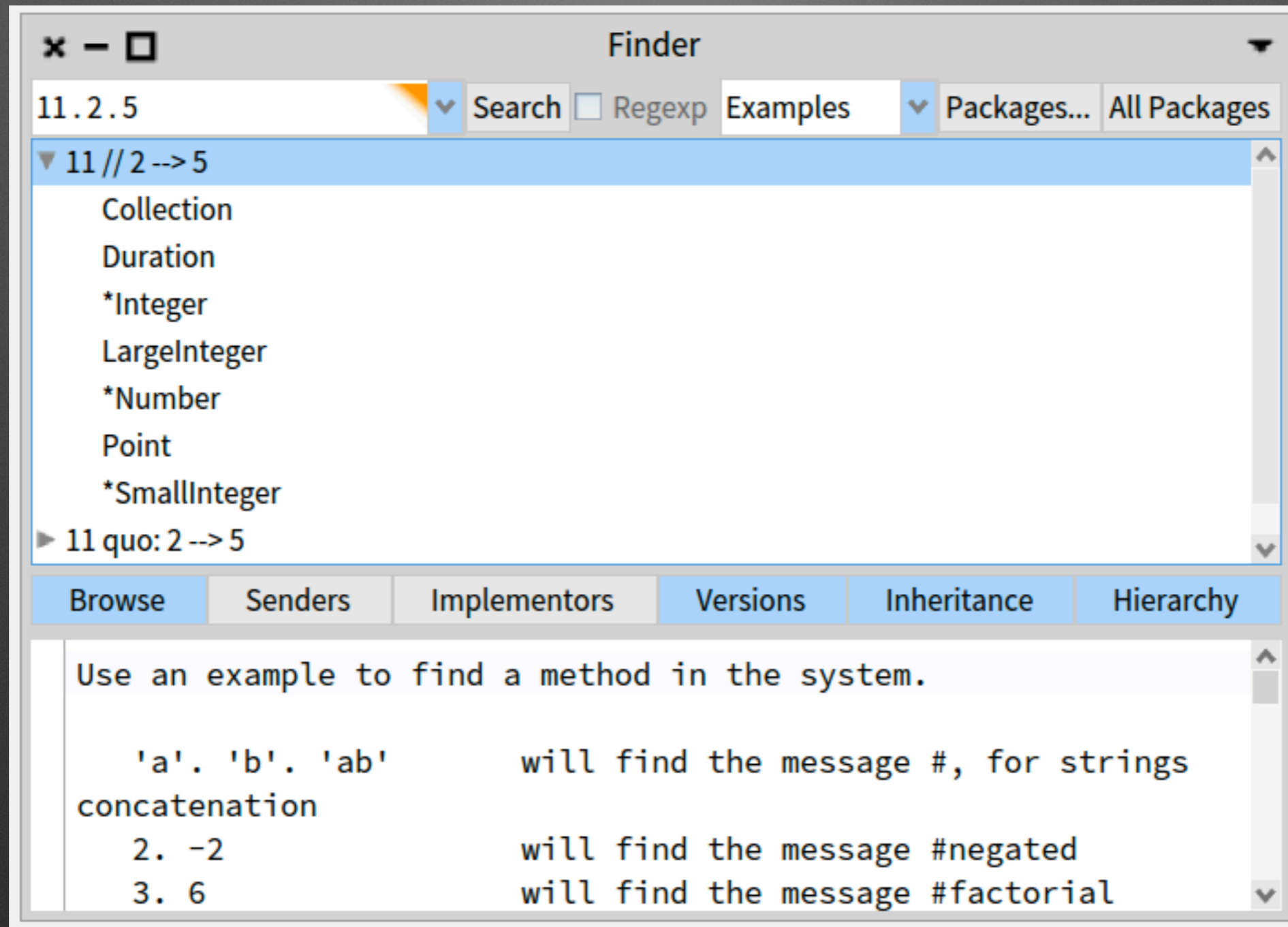
How to find information?

- Libraries are large
- You know what you want
- You do not know how to express it

Ask Finder example-based queries

- Provide objects and results
- Get the messages that match

11 ??? 2 should give 5



111 ??? 2 should give 5.5

Finder

11.2.5.5 Search Regexp Examples Packages... All Packages

- FileSystem class
- FloatArray
- *Fraction**
- IceNode
- Integer
- LargeInteger
- *Number
- Path
- Path class

Browse Senders Implementors Versions Inheritance Hierarchy

```
/ aNumber
  "Answer the result of dividing the receiver by aNumber."
  aNumber isFraction
    ifTrue: [^self * aNumber reciprocal].
  ^ aNumber adaptToFraction: self andSend: #/
```

What are the messages send to \$0 that return true

The screenshot shows the Ruby Finder tool interface. The search bar contains '\$0.isDigit'. The search results are displayed in a list view, showing various messages sent to \$0 that return true. The 'Character' class is highlighted in blue. Below the list, there are tabs for 'Browse', 'Senders', 'Implementors', 'Versions', 'Inheritance', and 'Hierarchy'. The 'isDigit' method is selected, and its implementation is shown in the bottom pane.

```
$0.isAlphaNumeric --> true
$0.isCharacter --> true
$0.isCompletionCharacter --> true
$0.isDecimalDigit --> true
$0.isDigit --> true
Character
$0.isLiteral --> true
$0.isOctetCharacter --> true
$0.isSafeForHTTP --> true
$0.shouldBePrintedAsLiteral --> true
$0.tokenish --> true
```

isDigit

```
"Return whether the receiver is a digit."
"$1.isDigit >>> true"
"$0.isDigit >>> true"

^ self.characterSet.isDigit: self
```


**Customized object
interaction/presentations**

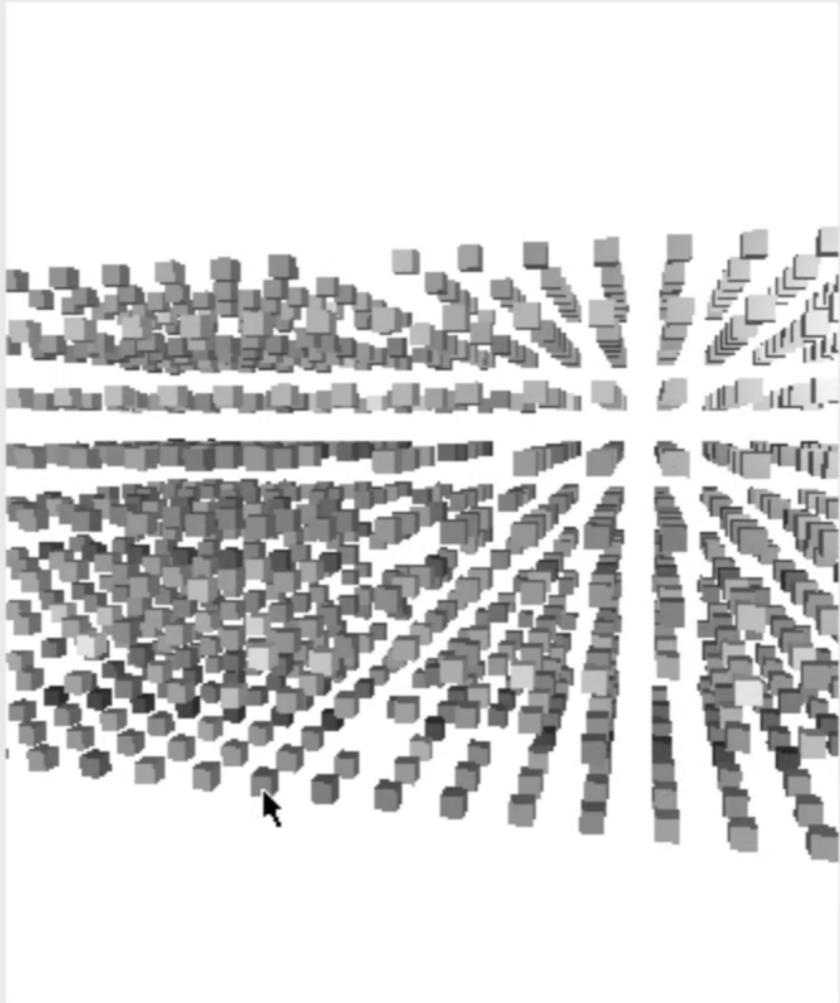
Inspecting Live a 3D object

Playground

```
data := TestData new data.  
  
cube := MatrixCube new initWithData: data.  
cube view
```

a RWView

Raw Live Meta



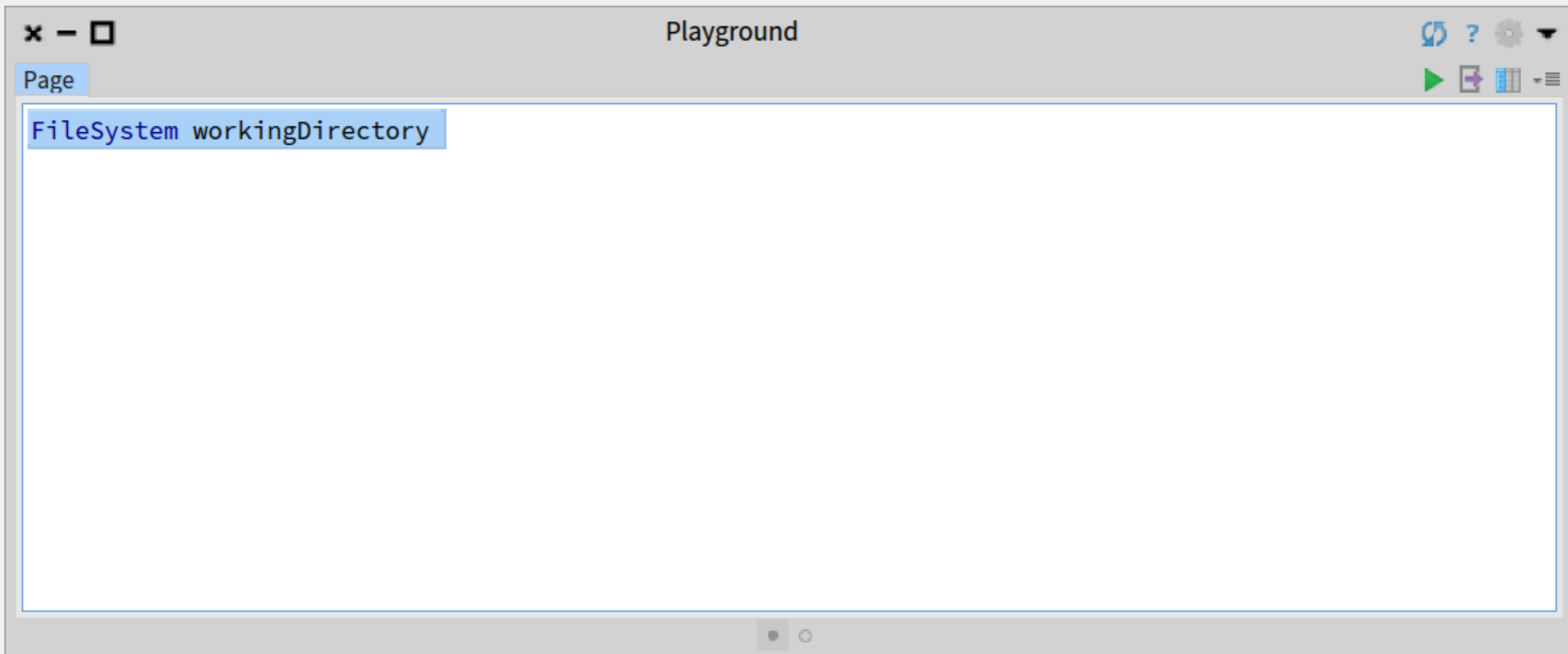
an Array [4 items] ('2000-01-01' 'Malawi' 'France'... x

Index	Item
1	'2000-01-01'
2	'Malawi'
3	'France'
4	'400.0'

enter search query (example: "each = 5")

The views of a file reference

Looking at a file reference



Oh! a file browser in my inspector!

The image shows a web browser window titled "Playground" with a developer console and a file inspector. The console displays the text "FileSystem workingDirectory". The file inspector shows a directory listing for a file reference located at "/Users/ducasse/Documents/Pharo/images/F...". The listing includes the following items:

Name	Size	
..	0 B	2
pharo-local	0 B	2
logo.png	25.82 kB	2
pharo.version	3 B	2
ReadMe.txt	63 B	2
meta-inf.ston	1.17 kB	2
P8-MasterClass.image	70.22 MB	2
P8-MasterClass.changes	1.16 kB	2
Archive.zip	27.24 kB	2
Pharo8.0-32bit-0932da8.sources	37.94 MB	2

But I have a file reference: a dull object

The screenshot shows a Playground window with a code editor on the left and an object inspector on the right. The code editor contains the text `FileSystem workingDirectory`. The object inspector shows a `FileReference` object with the following properties:

Variable	Value
self	/Users/ducasse/Documents/Pharo/images/I
filesystem	a FileSystem
path	Path / 'Users' / 'ducasse' / 'Documents' / 'Ph

Below the table, the object's class and parent are shown:

```
"File @  
/Users/ducasse/Documents/Pharo/images/P8-MasterCl  
ass"  
self parent File @ /Users/ducasse/Documents/Pharo/images
```

Quite boring object

The screenshot shows a Squeak Playground window with the following components:

- Code Area:** Contains the text `FileSystem workingDirectory`.
- Inspector:** Shows the inspection of a `FileReference` object. The title bar indicates the object's path: `a FileReference (/Users/ducasse/Documents/Pharo/images/F...)`.
- Inspector Table:** A table with two columns: `Variable` and `Value`.

Variable	Value
<code>self</code>	<code>/Users/ducasse/Documents/Pharo/images/I</code>
<code>filesystem</code>	<code>a FileSystem</code>
<code>path</code>	<code>Path / 'Users' / 'ducasse' / 'Documents' / 'Ph</code>

Below the table, the object's class and instance variables are displayed:

```
"File @  
/Users/ducasse/Documents/Pharo/images/P8-MasterCl  
ass"  
self fullName '/Users/ducasse/Documents/Pharo/images/P8-MasterClass'
```

We can see the png ;)


Playground

a FileReference (/Users/ducasse/Documents/Pharo/images/P8-M...)

Items Raw Meta

Name	Size	
..	0 B	2
pharo-local	0 B	2
logo.png	25.82 kB	2
pharo.version	3 B	2
ReadMe.txt	63 B	2
meta-inf.ston	1.17 kB	2
P8-MasterClass.image	70.22 MB	2
P8-MasterClass.changes	1.16 kB	2
Archive.zip	27.24 kB	2
Pharo8.0-32bit-0932da8.sources	37.94 MB	2

Picture Contents Raw Meta

The image shows the Pharo logo, which consists of the word "Pharo" in a blue, italicized sans-serif font. The letter "o" is replaced by a circular emblem containing a stylized lighthouse with a red top and a white body with blue stripes.

Looking inside that PNG file

The image shows a software interface with two main panes. The left pane is a file explorer titled 'Playground' showing a directory listing. The right pane is a hex editor showing the raw bytes of a selected file, with a 'Contents' tab active.

File Explorer (Left Pane):

Name	Size
..	0 B
pharo-local	0 B
logo.png	25.82 kB
pharo.version	3 B
ReadMe.txt	63 B
meta-inf.ston	1.17 kB
P8-MasterClass.image	70.22 MB
P8-MasterClass.changes	1.16 kB
Archive.zip	27.24 kB
Pharo8.0-32bit-0932da8.sources	37.94 MB

Hex Editor (Right Pane):

Offset	Hex	ASCII
1	00000000 89 50 4E 47 0D 0A 1A 0A 00 00 00	
2	0D 49 48 44 52 .PNG.....IHDR	
3	00000010 00 00 01 77 00 00 00 90 08 06 00	
4	00 00 F3 F6 2B ...w.....+	
5	00000020 70 00 00 0A D1 69 43 43 50 49 43	
6	43 20 50 72 6F p....iCCPICC Pro	
7	00000030 66 69 6C 65 00 00 48 89 95 97 07	
8	54 53 69 16 C7 file..H....TSi..	
9	00000040 BF F7 D2 43 42 4B 08 45 4A E8 4D	
10	90 5E A5 84 1E ...CBK.EJ.M.^...	
11	00000050 40 41 3A D8 08 49 48 42 09 21 05	
12	15 3B 32 38 02 @A:...IHB.!...;28.	
13	00000060 23 8A 8A 08 96 01 1D 8A 28 38 16	

But still a file reference!

The screenshot shows a 'Playground' window with two panes. The left pane displays a file browser view with a table of files. The right pane shows the 'Raw' view of a selected file, displaying its internal structure and metadata.

File Browser View (Left Pane):

Name	Size	Count
..	0 B	2
pharo-local	0 B	2
logo.png	25.82 kB	2
pharo.version	3 B	2
ReadMe.txt	63 B	2
meta-inf.ston	1.17 kB	2
P8-MasterClass.image	70.22 MB	2
P8-MasterClass.changes	1.16 kB	2
Archive.zip	27.24 kB	2
Pharo8.0-32bit-0932da8.sources	37.94 MB	2

Raw View (Right Pane):

Variable | Value

- self | /Users/ducasse/Documents/Pharo/images/...
- filesystem | a FileSystem
- path | Path / 'Users' / 'ducasse' / 'Documents' / 'Ph...

"File @
/Users/ducasse/Documents/Pharo/images/P8-MasterCl
ass/logo.png"

self fullName | '/Users/ducasse/Documents/Pharo/images/P8-MasterClass/logo.png'

See! an archive '.zip'

The image shows a screenshot of a file explorer window titled "Playground" and a hex editor window. The file explorer window displays a directory listing with columns for Name, Size, and a third column (likely permissions or type). The files listed are:

Name	Size	
..	0 B	2
pharo-local	0 B	2
logo.png	25.82 kB	2
pharo.version	3 B	2
ReadMe.txt	63 B	2
meta-inf.ston	1.17 kB	2
P8-MasterClass.image	70.22 MB	2
P8-MasterClass.changes	1.16 kB	2
Archive.zip	27.24 kB	2
Pharo8.0-32bit-0932da8.sources	37.94 MB	2

The hex editor window shows the raw contents of the selected file, "Archive.zip". The hex data is displayed in a table with columns for line number, hex values, and ASCII characters. The first few lines of the hex data are:

Line	Hex	ASCII
1	00000000 50 4B 03 04 14 00 08 00 08 00 E1	
2	AA 16 51 00 00 PK.....Q..	PK.....Q..
3	00000010 00 00 00 00 00 00 00 00 00 00 08	
4	00 10 00 6C 6F lolo
5	00000020 67 6F 2E 70 6E 67 55 58 0C 00 A4	
6	70 41 5F 96 70 go.pngUX...pA_.p	go.pngUX...pA_.p
7	00000030 41 5F F7 01 14 00 6C B7 63 90 68	
8	41 B3 25 DA B6 A_....l.c.hA.%..	A_....l.c.hA.%..
9	00000040 6D DB B6 6D BB FB B4 6D DB B6 79	
10	DA B6 6D DB B6 m..m...m..y..m..	m..m...m..y..m..
11	00000050 6D 5B EF 7C F7 DE 99 79 F1 DE D4	
12	8F 8A 55 99 2B m[. ...y.....U.+	m[. ...y.....U.+
13	00000060 57 46 66 C4 DE 55 19 A1 20 27 0E	

Kind of clear...

The image shows a 'Playground' application window with two side-by-side file reference panes. The left pane shows a directory listing with 'Archive.zip' selected. The right pane shows a similar listing with 'pharo.version' selected.

Left Pane: FileReference (/Users/ducasse/Documents/Pharo/images/P8-M...)

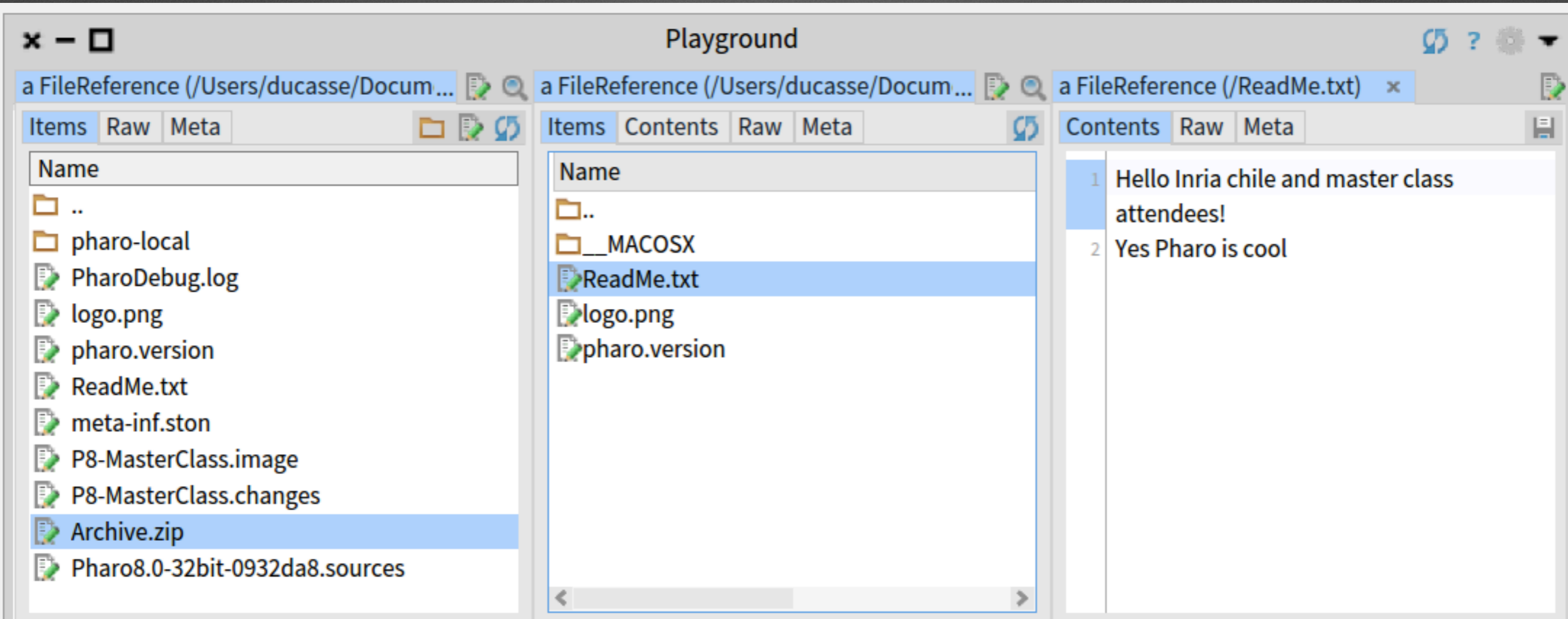
Name	Size	...
..	0 B	2
pharo-local	0 B	2
logo.png	25.82 kB	2
pharo.version	3 B	2
ReadMe.txt	63 B	2
meta-inf.ston	1.17 kB	2
P8-MasterClass.image	70.22 MB	2
P8-MasterClass.changes	1.16 kB	2
Archive.zip	27.24 kB	2
Pharo8.0-32bit-0932da8.sources	37.94 MB	2

Right Pane: FileReference (/Users/ducasse/Documents/Pharo/images/F...)

Name	Size	...
..	0 B	2
__MACOSX	0 B	2
ReadMe.txt	0 B	2
logo.png	0 B	2
pharo.version	0 B	2

An object can expose multiple interactive views!

- You can use the best view for your task!
- You can add views to your domain objects



**It is supra cool but it is not
magic**

Implementing a pane!

The screenshot shows the Smalltalk IDE interface for the class `AbstractFileReference` with the `gtInspectorPngIn` instance side selected. The left pane shows a package browser with `FileSystem-Disk` selected. The middle pane shows the class hierarchy, and the right pane shows the instance side methods, with `gtInspectorPngIn` highlighted. The bottom pane shows the implementation code for `gtInspectorPngIn`.

```
gtInspectorPngIn: composite
  <gtInspectorPresentationOrder: 0>
  composite morph
    title: 'Picture';
    display: [ self binaryReadStreamDo: [ :stream | PNGReadWriter formFromStream: stream ] ];
    when: [ self isFile and:
      [ self mimeTypes notNil and:
        [ self mimeTypes first matches: ZnMimeType imagePng ] ] ]
```

8/8 [9] GT-InspectorExtensions-Core extension F +

Ok files are boring...
What about *inside*
the system?



A class is an object we can inspect!

The screenshot shows a Python Playground window with a code editor on the left and an object inspector on the right. The code editor contains the word "Point". The object inspector shows the details of a "Point" class object, including its superclass (Object), method dictionary, format string, layout, organization, subclasses, name, and class pool.

Playground

Page

Point

a Point class (Point)

Raw Defir... Meth... All R... All Ref Com... Inst\... Insta... Meta

Variable	Value
self	Point
superclass	Object
methodDict	a MethodDictionary [103 items] (size 103)
format	65538
layout	a FixedLayout
organization	a ClassOrganization
subclasses	nil
name	#Point
classPool	a Dictionary [0 items] /

"Point"
self

“A class has a method dictionary”
they said... let us verify

The screenshot shows a Ruby Playground window with two panes. The left pane displays the internal structure of a `Point` class, and the right pane displays the contents of its `MethodDictionary`.

Left Pane: a Point class (Point)

Variable	Value
self	Point
superclass	Object
methodDict	a MethodDictionary [103 items] (size 103)
format	65538
layout	a FixedLayout
organization	a ClassOrganization
subclasses	nil
name	#Point

Below the table, the source code for the `Point` class is visible:

```
"Point"  
self
```

Right Pane: a MethodDictionary [103 items] (size 103)

Key	Value
#reflectedAbout:	Point>>#reflectedAbout:
#rotateBy:centerAt:	Point>>#rotateBy:centerAt:
#adaptToNumber:andSend:	Point>>#adaptToNumber:andSend:
#squaredDistanceTo:	Point>>#squaredDistanceTo:
#adaptToCollection:andSend:	Point>>#adaptToCollection:andSend:
#theta	Point>>#theta
#transposed	Point>>#transposed
#-	Point>>#-
#fourDirections	Point>>#fourDirections
#crossProduct:	Point>>#crossProduct:
#scaleFrom:to:	Point>>#scaleFrom:to:
#veryDeepCopyWith:	Point>>#veryDeepCopyWith:

Dissecting one method object

The screenshot shows the Ruby Inspector tool window titled "Inspector on a CompiledMethod (Point>>#degrees)". The main content area displays the internal structure of the method object, organized into a table with "Variable" and "Value" columns. The variables listed are self, literal1 through literal8, and bc 89. The values are Point>>#degrees, 90.0, 270.0, #asFloat, #arcTan, #radiansToDegrees, 360.0, 180.0, #ifTrue:ifFalse:, and 0. Below the table, the source code for the method is visible, starting with "Point>>#degrees" and "self".

Variable	Value
{ } self	Point>>#degrees
▶ Σ literal1	90.0
▶ Σ literal2	270.0
▶ ¶ literal3	#asFloat
▶ ¶ literal4	#arcTan
▶ ¶ literal5	#radiansToDegrees
▶ Σ literal6	360.0
▶ Σ literal7	180.0
▶ ¶ literal8	#ifTrue:ifFalse:
▶ Σ bc 89	0

```
"Point>>#degrees"  
self
```

I do not want to be a compiler!

Inspector on a CompiledMethod (Point>>#degrees)

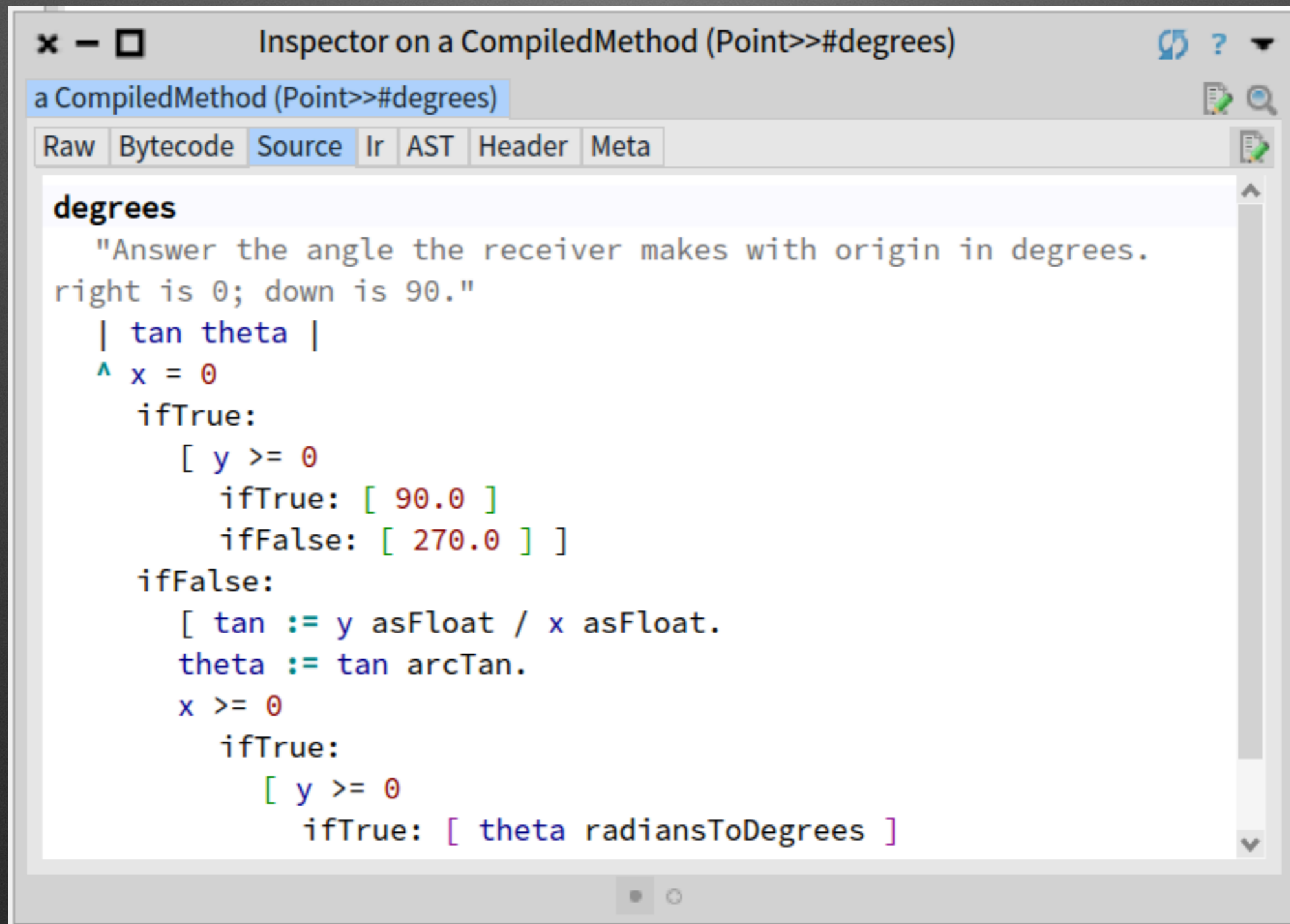
a CompiledMethod (Point>>#degrees)

Raw Bytecode Source Ir AST Header Meta

Variable	Value
▶ Σ bc 89	0
▶ Σ bc 90	117
▶ Σ bc 91	182
▶ Σ bc 92	172
▶ Σ bc 93	9
▶ Σ bc 94	1
▶ Σ bc 95	117
▶ Σ bc 96	181
▶ Σ bc 97	153
▶ Σ bc 98	32

"Point>>#degrees"
self

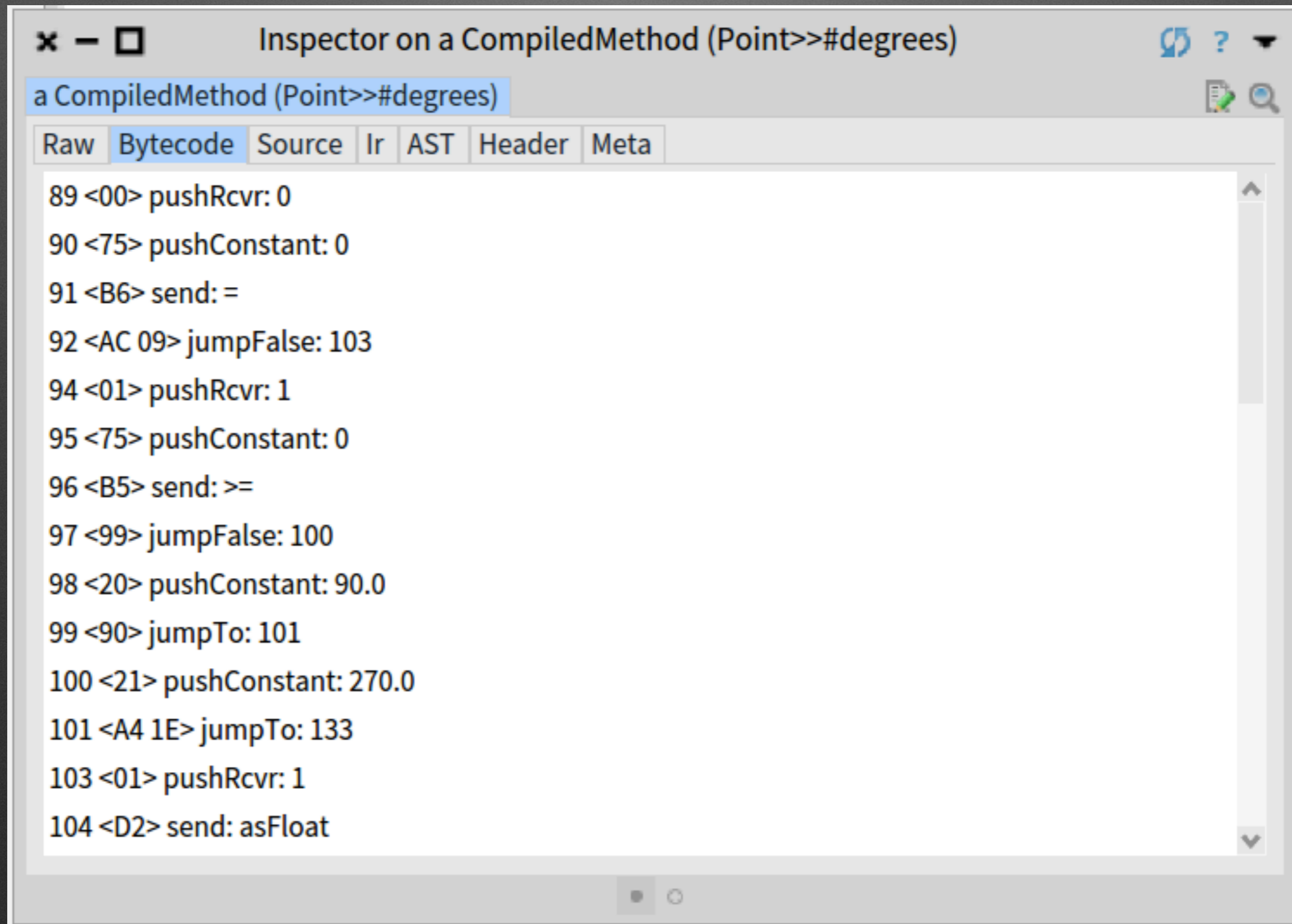
It looks like a method



The screenshot shows an IDE window titled "Inspector on a CompiledMethod (Point>>#degrees)". The window contains a tabbed interface with "Source" selected. The source code for the "degrees" method is displayed, showing a docstring and several conditional branches based on the sign of the x and y coordinates.

```
degrees
  "Answer the angle the receiver makes with origin in degrees.
  right is 0; down is 90."
  | tan theta |
  ^ x = 0
  ifTrue:
    [ y >= 0
      ifTrue: [ 90.0 ]
      ifFalse: [ 270.0 ] ]
  ifFalse:
    [ tan := y asFloat / x asFloat.
      theta := tan arcTan.
      x >= 0
        ifTrue:
          [ y >= 0
            ifTrue: [ theta radiansToDegrees ]
```

Numbers are not that obscure



The screenshot shows a debugger window titled "Inspector on a CompiledMethod (Point>>#degrees)". The window contains a list of bytecode instructions for a method named "a CompiledMethod (Point>>#degrees)". The instructions are displayed in a table-like format with columns for line number, bytecode, and instruction name.

Line	Bytecode	Instruction
89	<00>	pushRcvr: 0
90	<75>	pushConstant: 0
91	<B6>	send: =
92	<AC 09>	jumpFalse: 103
94	<01>	pushRcvr: 1
95	<75>	pushConstant: 0
96	<B5>	send: >=
97	<99>	jumpFalse: 100
98	<20>	pushConstant: 90.0
99	<90>	jumpTo: 101
100	<21>	pushConstant: 270.0
101	<A4 1E>	jumpTo: 133
103	<01>	pushRcvr: 1
104	<D2>	send: asFloat

And mapping them to the good abstraction helps

The image shows a screenshot of an IDE's Inspector window, split into two panes. The left pane shows the AST of a compiled method, and the right pane shows the source code of the same method.

Inspector on a CompiledMethod (Point>>#degrees)

a CompiledMethod (Point>>#degrees)

- Raw
- Bytecode
- Source
- Ir
- AST
- Header
- Meta

▼ RBMethodNode(degrees "Answer the angle the receiver makes with")

- ▼ RBSequenceNode(| tan theta | ^ x = 0 ifTrue: [y >= 0 ifTrue: [90.0]])
- RBTemporaryNode(tan)
- RBTemporaryNode(theta)
- ▼ RBReturnNode(^ x = 0 ifTrue: [y >= 0 ifTrue: [90.0]])
- ▼ RBMessageNode(x = 0 ifTrue: [y >= 0 ifTrue: [90.0]])
- ▼ RBMessageNode(x = 0)
- RBInstanceVariableNode(x)
- RBLiteralValueNode(0)
- ▼ RBBlockNode([y >= 0 ifTrue: [90.0] ifFalse: [270.0]])
- ▼ RBSequenceNode(y >= 0 ifTrue: [90.0] ifFalse: [270.0])
- ▼ RBMessageNode(y >= 0 ifTrue: [90.0] ifFalse: [270.0])
- ▶ RBMessageNode(y >= 0)
- ▶ RBBlockNode([90.0])
- ▶ RBBlockNode([270.0])

a RBMessageNode (RBMessageNode(y >= 0))

- Raw
- Tree
- Scopes
- Source cc...
- AST Dump
- Meta

degrees

"Answer the angle the receiver makes with origin in degrees. right is 0; down is 90."

```
| tan theta |
^ x = 0
ifTrue:
  [ y >= 0
    ifTrue: [ 90.0 ]
    ifFalse: [ 270.0 ] ]
ifFalse:
  [ tan := y asFloat / x asFloat.
    theta := tan arcTan.
    x >= 0
      ifTrue:
        [ y >= 0
          ifTrue: [ theta radiansToDegrees ]
          ifFalse: [ 360.0 + theta
            radiansToDegrees ] ]
```

Yes pushRcvr: 1 means the second field!

The image shows a screenshot of the Inspector on a CompiledMethod (Point>>#degrees) in a development environment. The window is split into two panes. The left pane shows the bytecode for the method, and the right pane shows the symbolic bytecode for the selected instruction.

Inspector on a CompiledMethod (Point>>#degrees)

a CompiledMethod (Point>>#degrees)

Raw	Bytecode	Source	Ir	AST	Header	Meta
89	<00>	pushRcvr: 0				
90	<75>	pushConstant: 0				
91	<B6>	send: =				
92	<AC 09>	jumpFalse: 103				
94	<01>	pushRcvr: 1				
95	<75>	pushConstant: 0				
96	<B5>	send: >=				
97	<99>	jumpFalse: 100				
98	<20>	pushConstant: 90.0				
99	<90>	jumpTo: 101				
100	<21>	pushConstant: 270.0				
101	<A4 1E>	jumpTo: 133				
103	<01>	pushRcvr: 1				
104	<D2>	send: asFloat				
105	<00>	pushRcvr: 0				
106	<D2>	send: asFloat				

a SymbolicBytecode (94 <01> pushRcvr: 1)

```
origin in degrees. right is 0; down is 90."
| tan theta |
^ x = 0
ifTrue:
  [ y >= 0
    ifTrue: [ 90.0 ]
    ifFalse: [ 270.0 ] ]
ifFalse:
  [ tan := y asFloat / x asFloat.
    theta := tan arcTan.
    x >= 0
      ifTrue:
        [ y >= 0
          ifTrue: [ theta radiansToDegrees ]
          ifFalse: [ 360.0 + theta
radiansToDegrees ] ]
      ifFalse: [ 180.0 + theta
radiansToDegrees ] ]
```


Pharo Pro devs

- Get **productivity boost**
- Xtreme TDD
 - write test,
 - test fails and
 - **code in debugger**

Hot update on the fly
customizable debugger

Halt

Bytecode

Stack

Proceed Restart Into Over Through

PDFCellElement	getSubElementsWith:styleSheet:
PDFCellElement(PDFComposite)	generateCodeSegmentsCollectionWi
PDFCellElement(PDFComposite)	generateCodeSegmentWith:styleShe
PDFDataTableElement(PDFComposite)	generateCodeSegmentsCollectionWi [:aSubElement aSubElement generateCodeSe
Array(SequenceableCollection)	collect:

Source

Where is? Browse

```
generateCodeSegmentsCollectionWith: aPDFGenerator styleSheet: compositeStyleSheet format: aFormat
^ (self getSubElementsWith: aPDFGenerator styleSheet: compositeStyleSheet)
  collect: [ :aSubElement |
    aSubElement
      generateCodeSegmentWith: aPDFGenerator
      styleSheet: (aSubElement buildCompositeStyleSheetFrom: compositeStyleSheet)
      format: aFormat ]
```

Variables

Type	Variable	Value
implicit	self	a PDFCellElement
parameter	aFormat	a PDFA4Format
parameter	aPDFGenerator	a PDFGenerator
parameter	compositeStyleSheet	a StyleSheet

dimension: 80 mm @ 20 mm;

Halt in OCDBox>>name:

Class	Method	Context
OCDBox	name:	UndefinedObject>>Dolt
UndefinedObject	Dolt	CompiledMethod>>valueWithReceiver
CompiledMethod	valueWithReceiver:arguments:	[aCompiledMethod
SmalltalkEditor	debug:receiver:in:	[self value.

Bytecode Breakpoints Sindarin

Type	Target	Method
<input type="checkbox"/> Breakpoint	self	ReflectiveExamples>>example
<input checked="" type="checkbox"/> Breakpoint	self	OCDBox>>initialize
<input checked="" type="checkbox"/> Breakpoint	self	OCDBox>>name:
<input checked="" type="checkbox"/> Breakpoint	self	OCDBox>>name:
<input checked="" type="checkbox"/> Halt	OCDBox	OCDBox>>name:
<input type="checkbox"/> Breakpoint	OCDBox	OCDBox>>name:
<input type="checkbox"/> Breakpoint	self	OCDBox>>removeElement:
<input type="checkbox"/> Breakpoint	self	OrderedCollection>>removeElement:
<input type="checkbox"/> Halt	StHaltCacheTest	StHaltCacheTest>>testInitial
<input checked="" type="checkbox"/> Halt	StHaltCacheTest	StHaltCacheTest>>testInitial

```

1 name: anObject
2   self halt.
3   name := anObject
  
```

Specific view

Bytecode Breakpoints Sindarin

```

33 <4C> self
34 <80> send: halt
35 <D8> pop
36 <40> pushTemp: 0
37 <C9> popIntoRcvr: 1
38 <58> returnSelf
  
```

Variable	Value
stackTop	an OCDBox
0 [anObject]	'i'
rcvr: 0 [elements]	an OrderedCollection [0 items]
rcvr: 1 [name]	''

Receiver in: a StDebuggerContext (OCDBox>>name:)

an OCDBox

Variable	Value
[arg] anObject	'i'
self	an OCDBox
elements	an OrderedCollection [0 items]
name	''
Temps	a Dictionary [1 item] (#anObject->'i')
anObject	'i'
stackTop	an OCDBox

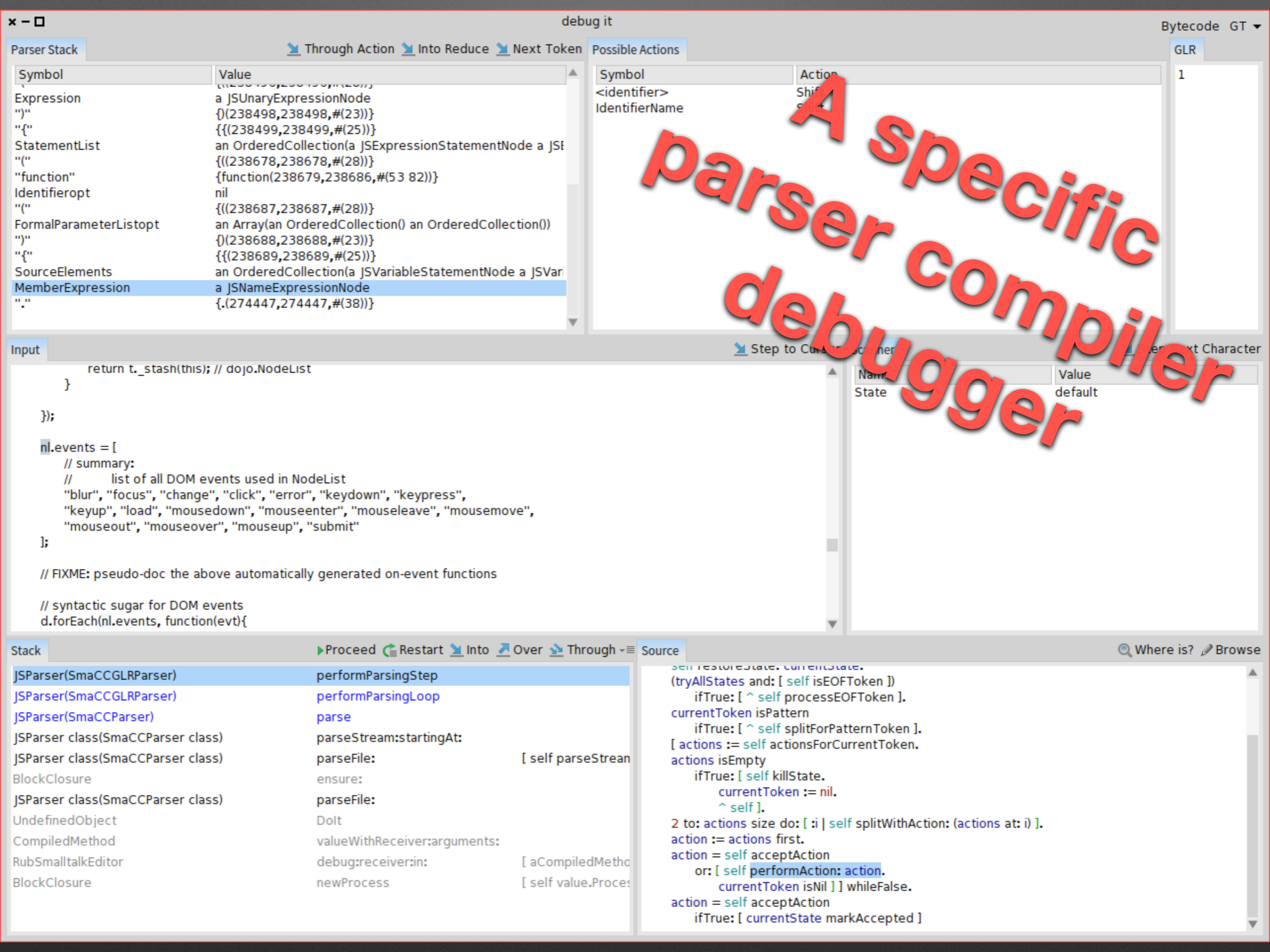
Raw	Breakpoints	Meta
	<input checked="" type="checkbox"/> Halt	OCDBox
		OCDBox>>name:
1		
1		"an OCDBox"
2		self

Receiver in: a StDebuggerContext (OCDBox>>name:)

an OCDBox

Variable	Value
[arg] anObject	'i'
self	an OCDBox
elements	an OrderedCollection [0 items]
name	''
Temps	a Dictionary [1 item] (#anObject->'i')
anObject	'i'
stackTop	an OCDBox

Raw	Breakpoints	Meta
	<input checked="" type="checkbox"/> Halt	OCDBox
		OCDBox>>name:
1		
1		"an OCDBox"
2		self



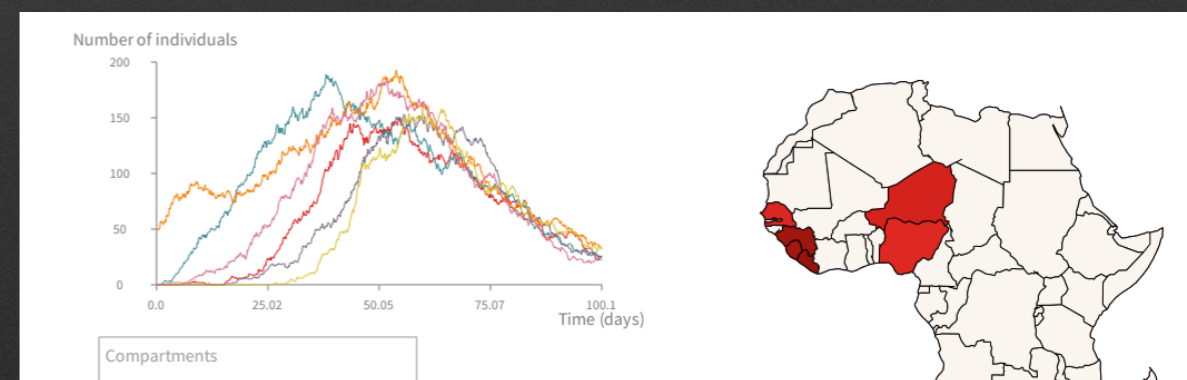
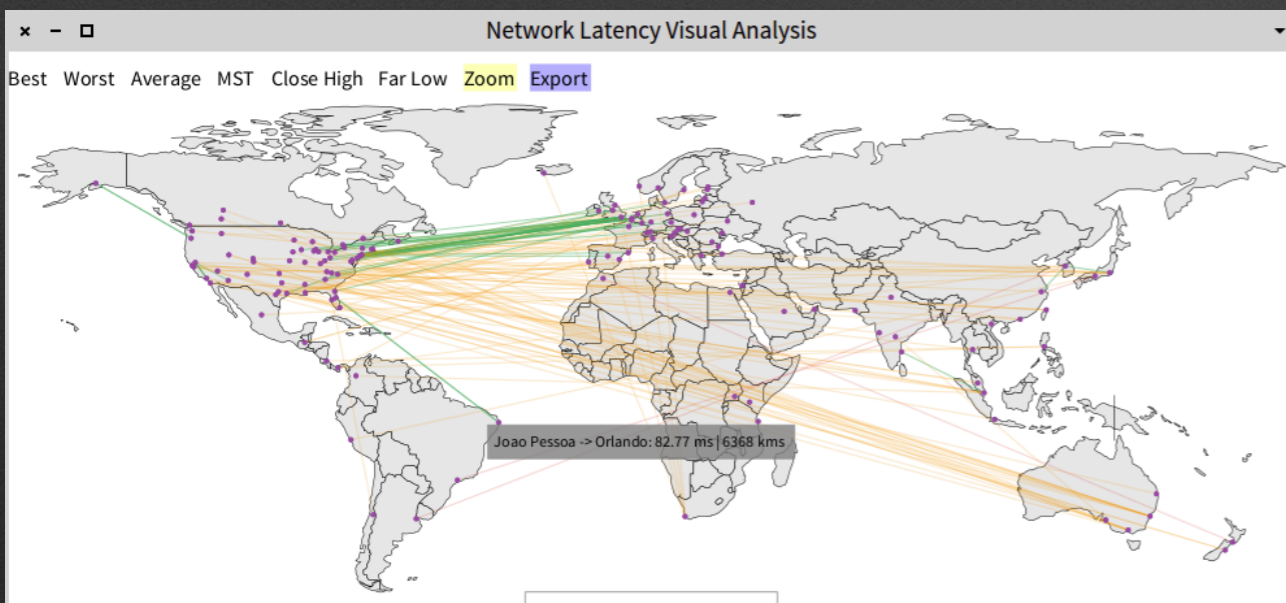
Live visualisation scripting

- The next level
- Roassal 30 by Prof. A. Bergel/Object Profile University of Chile at Santiago
- Simply gorgeous
- Check <http://agilevisualization.com>

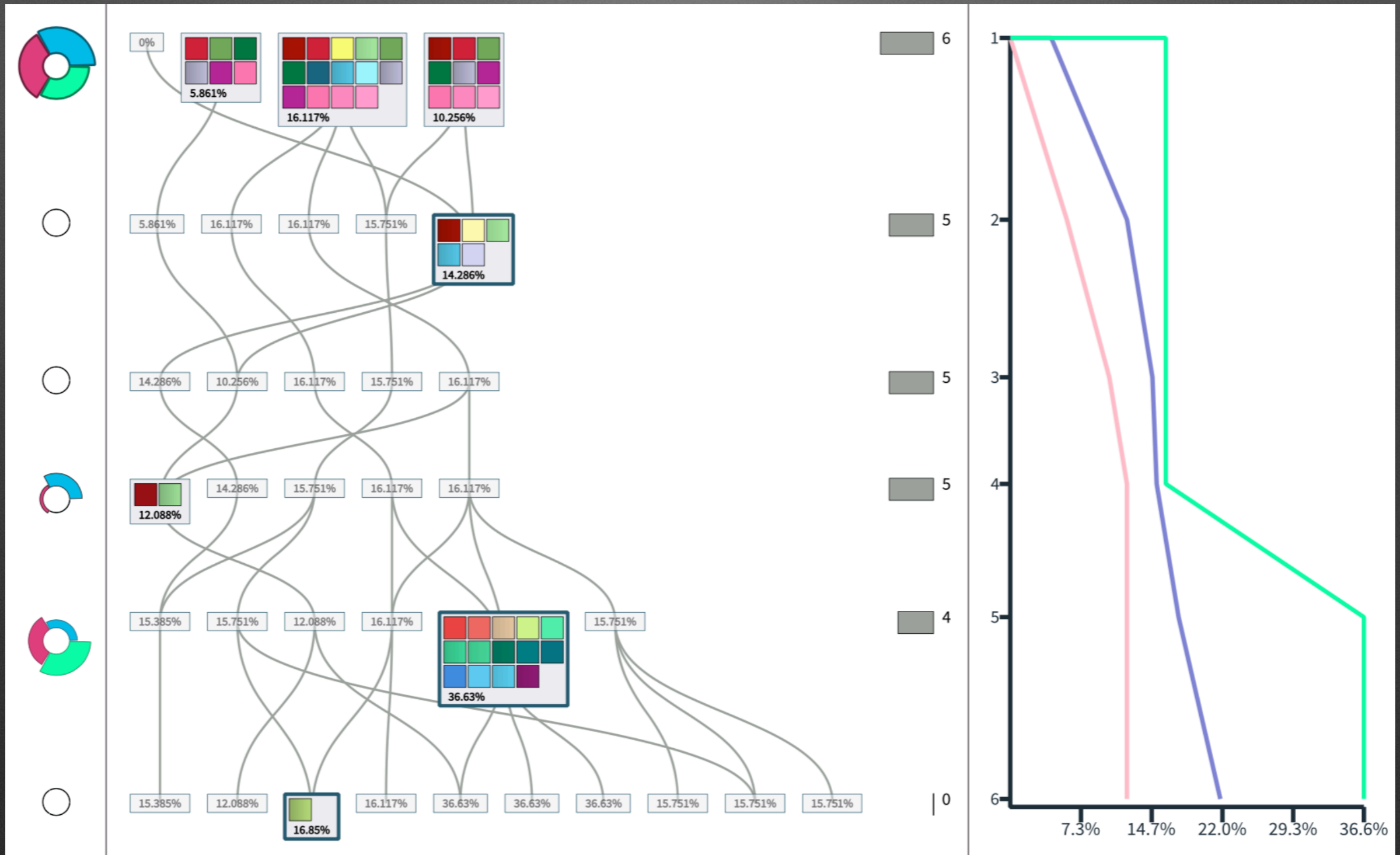
Scripting visualisations

```
b := RTMondrian new.  
  b shape rectangle  
  withBorder;  
  width: [ :cls | cls numberOfVariabl  
  height: [ :cls | cls numberOfMethod
```

```
b nodes: Collection withAllSubclasses.  
b edges connectToAll: [ :cls | cls sub  
b layout tree.  
b normalizer  
  normalizeColorAsGray: [ :cls |  
cls numberOfLinesOfCode ].  
b
```

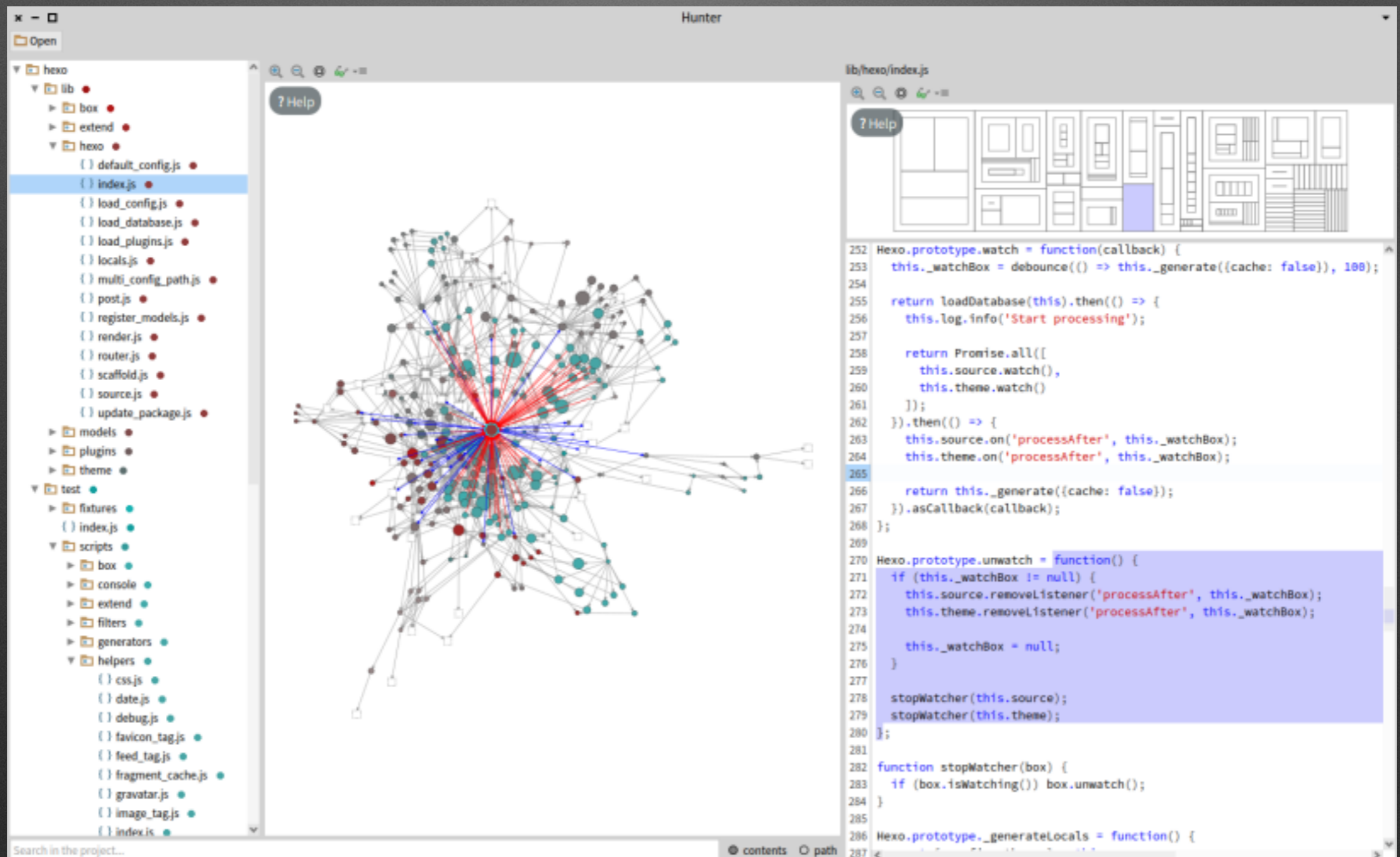


Execution of IA generating tests



Building your own tool

- Javascript analysis



The screenshot displays the Hunter tool interface, which is used for JavaScript analysis. The interface is divided into three main sections:

- File Tree (Left):** A hierarchical view of the project files. The 'hexo' directory is expanded, showing sub-directories like 'lib', 'test', and 'helpers', along with numerous JavaScript files such as 'index.js', 'load_config.js', and 'update_package.js'.
- Network Graph (Center):** A complex network graph representing the relationships between the analyzed JavaScript code. Nodes are represented by colored circles (red, blue, green, grey) and are interconnected by lines, forming a dense, star-like structure with many connections radiating from a central point.
- Code Editor (Right):** A view of the source code for 'lib/hexo/index.js'. The code is highlighted in blue, showing functions like 'Hexo.prototype.watch' and 'Hexo.prototype.unwatch'. The code includes logic for watching and unwatching resources, using promises and debouncing.

The Hunter tool provides a comprehensive view of the code's structure and dependencies, allowing for detailed analysis and visualization of the JavaScript code.

HTTP traffic analysis

- <http://youtu.be/rIBbeMdFCys>

The screenshot displays the Pharo IDE interface. At the top left is the Pharo logo. A Monticello Browser window is open, showing a repository at `http://smalltalkhub.com/mc/SvenVanCaekenberghe/f`. The browser lists several packages, including `HP35-Calculator-SvenVanCaekenberghe.17.mcz` through `.7.mcz`. Below the browser is a Playground window. The playground shows a log of events under the 'Announcements' tab. The selected event is: `2014-10-15T15:27:46.873225+ 2014-10-15 15:27:46 028 Response Read a ZnResponse(200 OK text/`. The right-hand side of the playground displays the state of the selected event, showing variables like `self`, `clientid`, `duration`, `id`, `response`, and `timestamp`.

Variable	Value
self	2014-10-15 15:27:46 028 Response Read a ZnResponse(200 OK text/
clientid	nil
duration	0
id	28
response	a ZnResponse(200 OK text/plain 11978)
timestamp	2014-10-15T15:27:46.873225+02:00

Below the variable table, a message box shows: `"2014-10-15 15:27:46 028 Response Read a ZnResponse(200 OK text/plain 11978) 0ms"` with `self` as the receiver.

Probabilistic Data Structure

- <https://github.com/osoco/PharoPDS>
- Defined new data structures
- And the analysis tools

The screenshot displays the 'PDS Algorithms Viewer' interface. The main window is titled 'PDS Algorithms Viewer' and features a menu bar with options: '+ New Bloom Filter', 'Analysis', 'Profiling', 'Benchmarking', and 'Reset'. The left sidebar shows 'PDS Algorithms' with 'Bloom Filter' selected. The main content area is divided into several sections:

- Add elements:** A text input field contains 'e.g. Madrid', followed by an 'Add' button. Below it, a message states 'london added to filter!'.
- Membership test:** A text input field contains 'E.g. London', followed by a 'Test' button.
- Bloom Filter BitSet:** A grid visualization showing the state of the filter. The grid is mostly green, with some blue squares indicating set bits. The grid is labeled 'In Out Center'.
- False-Positive Probability Curve:** A graph showing the False Positive Probability (FPP) as a function of Elements Added. The x-axis is 'Elements Added' (0.0 to 150.0) and the y-axis is 'False Positive Probability (FPP)' (0.0 to 0.15). A blue curve represents the 'FPP curve', and a red dot represents the 'Current FPP' at approximately (0, 0).

On the right side, a panel titled 'a PDSBloomFilter' displays a table of parameters:

Name	Value
'Target Elements (n)'	100
'Target FPP'	0.03
'Number of hashes (k)'	6
'Current Elements'	0
'Current FPP'	0.0

Empowering is the right word

- Moldable tools are powerful
- Productivity boost
- Tried to give you my feeling
- But “The idea of experience does not replace experience.” Alain

There is a meta question

**How to invent new things with
the same tools than any body
else?**



IT'S SAD HOW SOME PEOPLE
CAN'T HANDLE A LITTLE
VARIETY.

