

# The Squeak Environment

---

# Smalltalk Run-Time Architecture

---

## Virtual Machine + Image + Changes and Sources

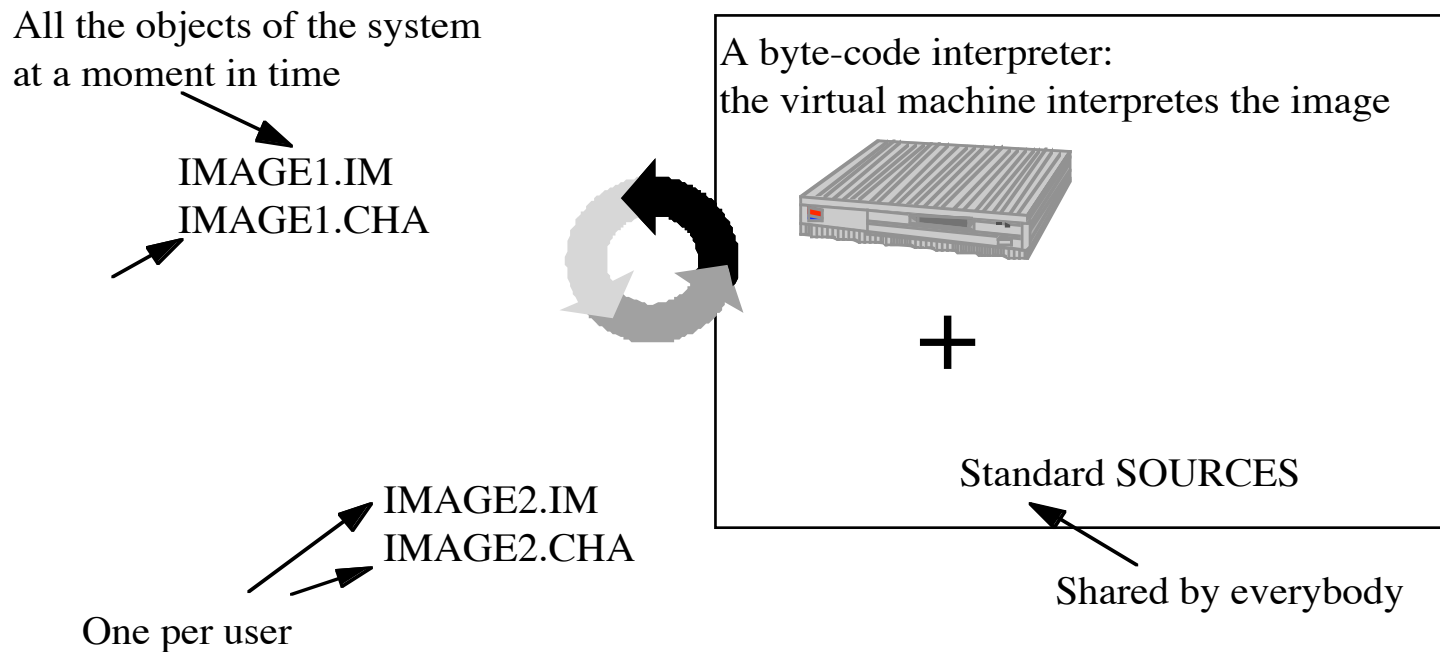


Image = bytecodes

Sources and changes = code (text)

---

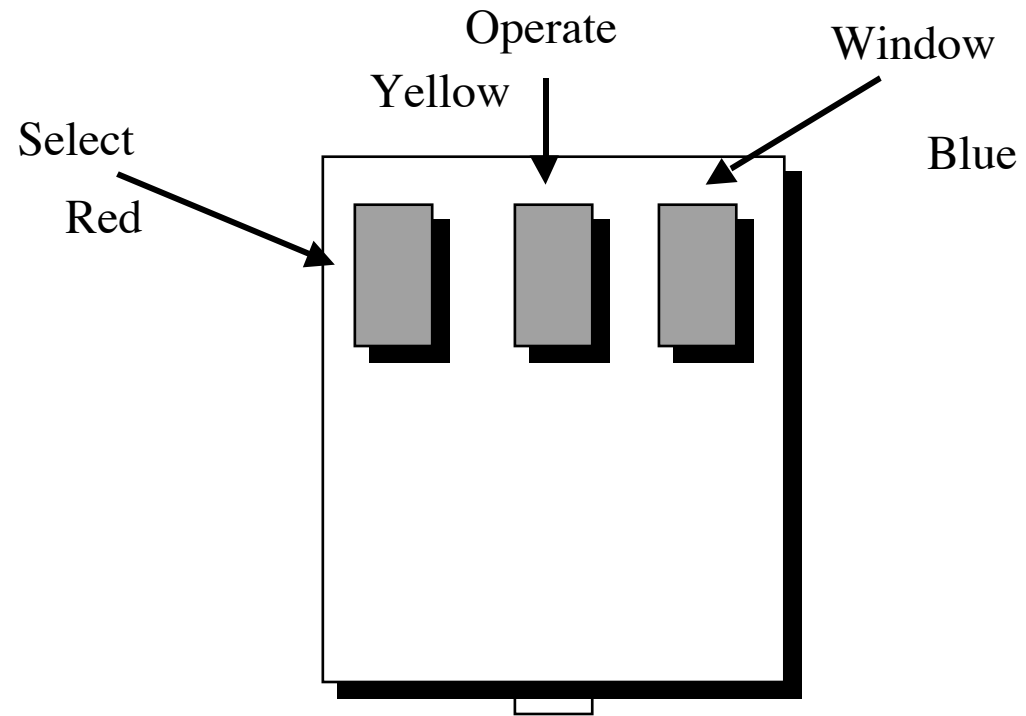
# Runtime Architecture

---

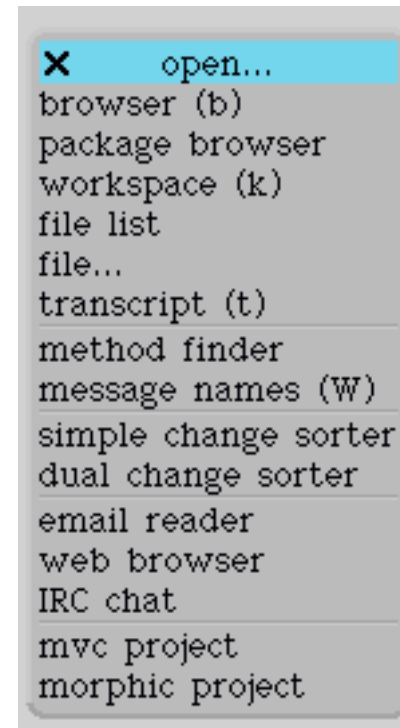
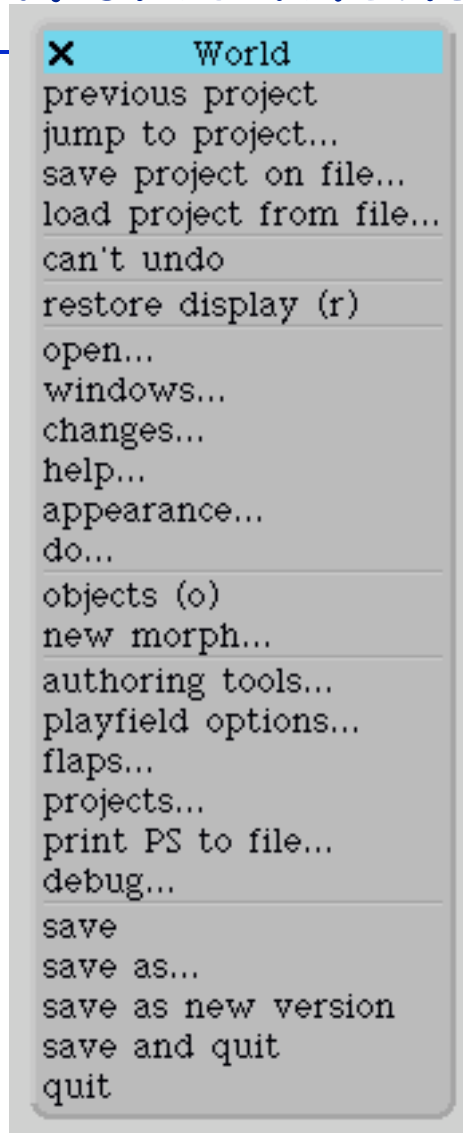
- The byte-code is in fact translated into native code by a just-in-time compiler.
- The source and the changes are not necessary for interpreting the byte-code, this is just for the development. Normally they are removed for deployment.
- An application can be delivered as some byte-code files that will be executed with a VM. The development image is stripped to remove the unnecessary development components.

# Mouse Semantics

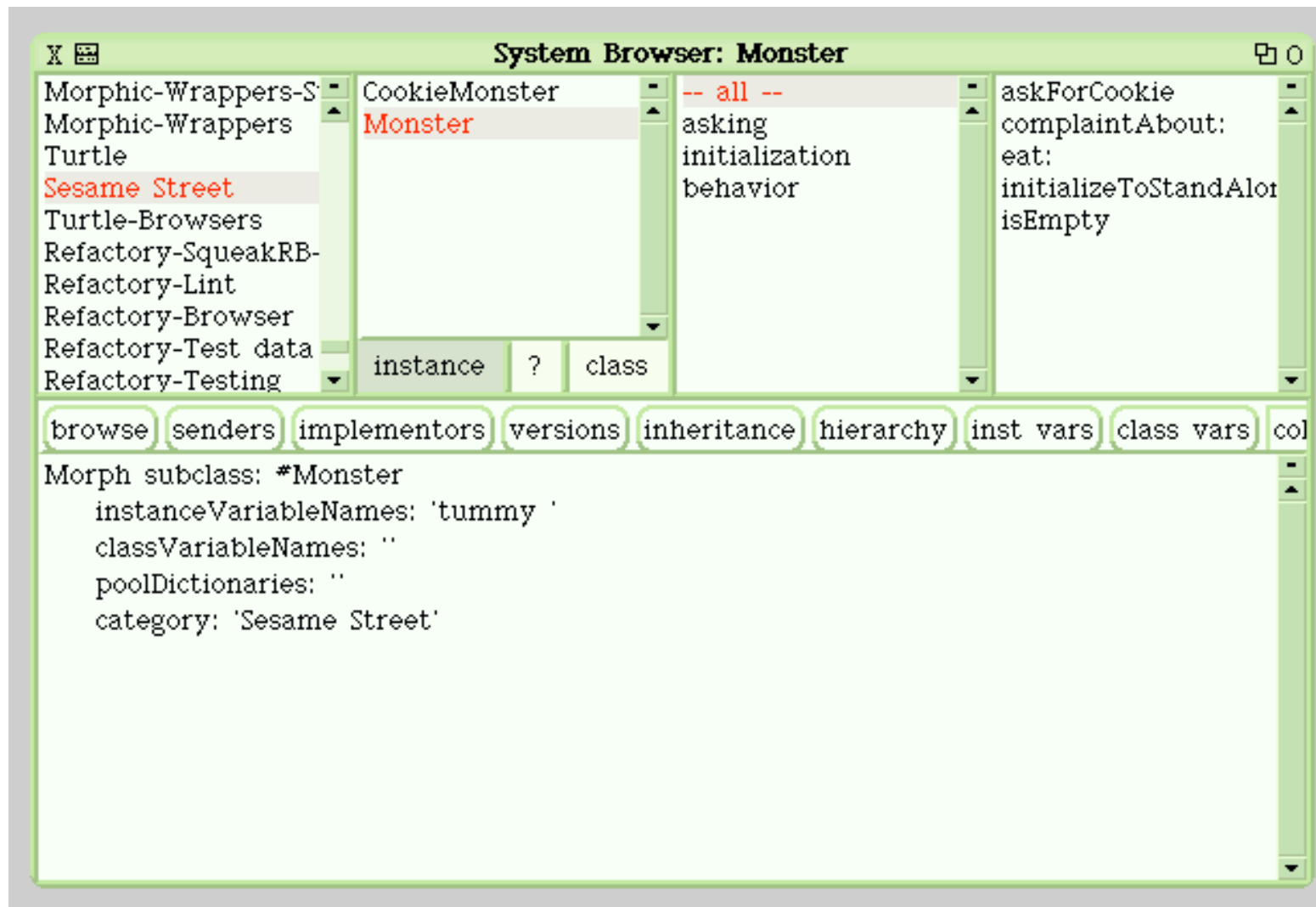
---



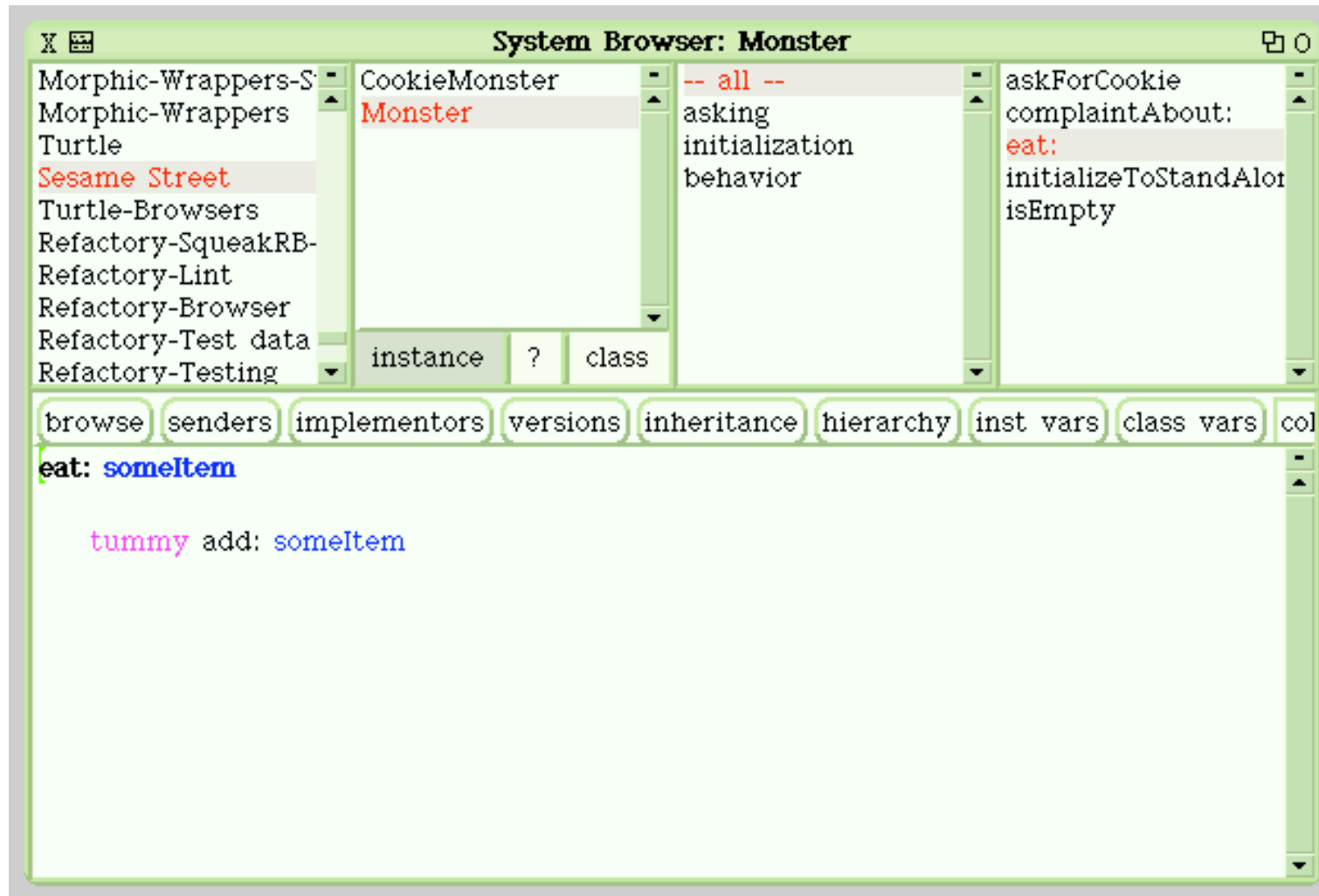
# World Menu and Open Menu



# Browsing a class



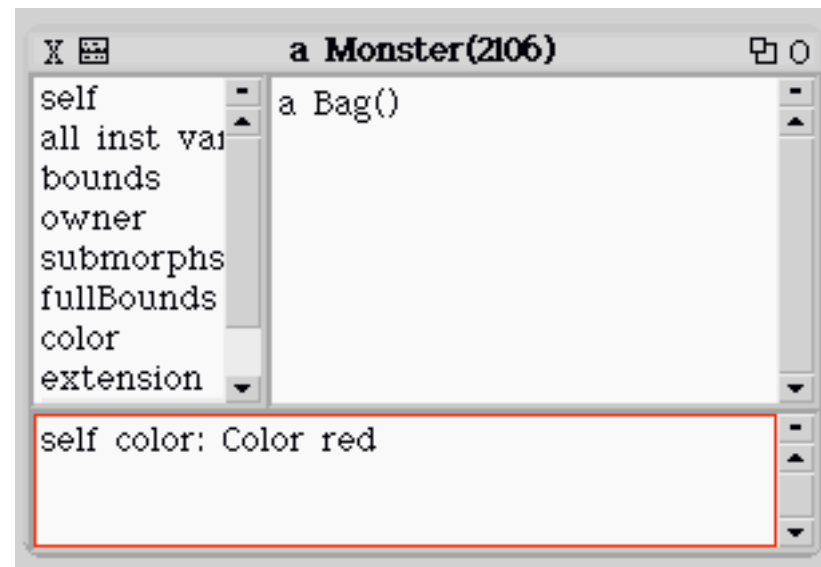
# Browsing methods



# Inspector

---

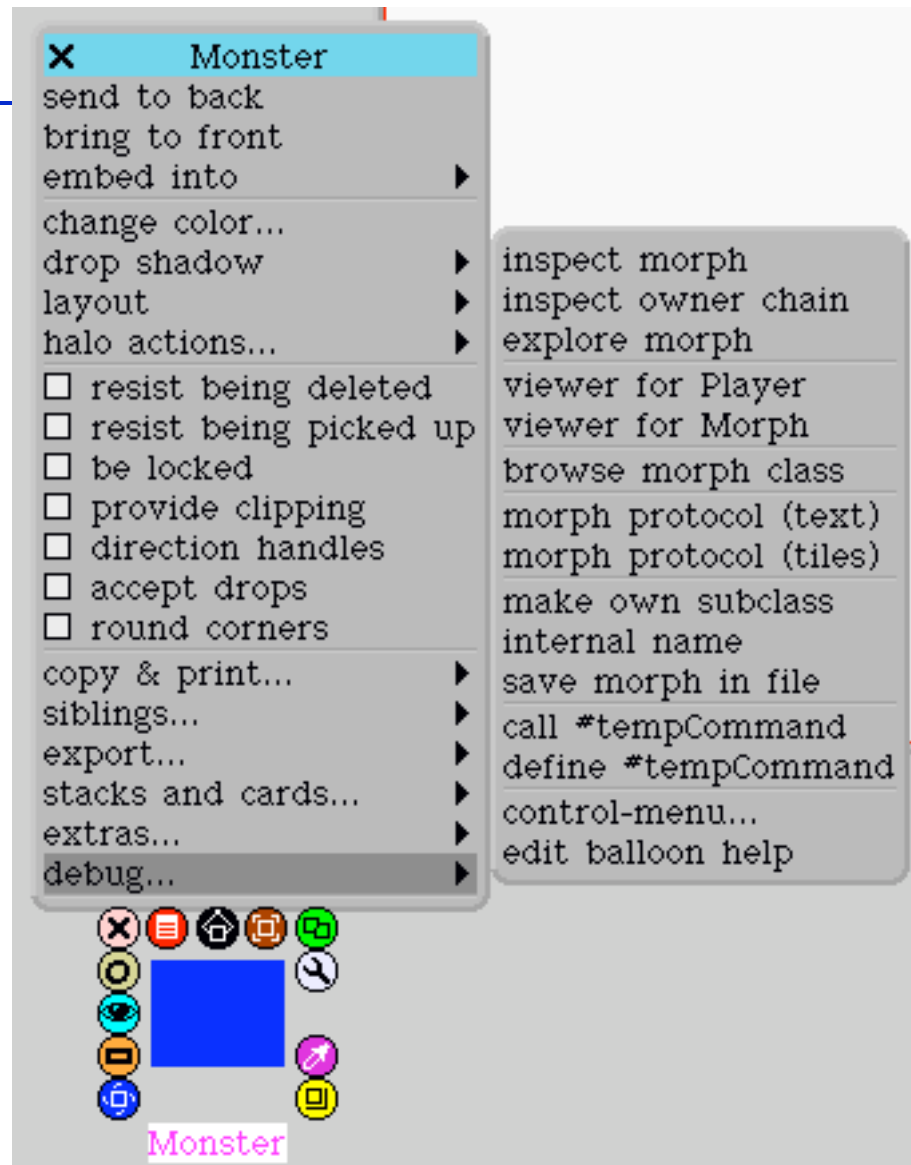
- To look inside objects
- Violates encapsulation!!!
- `Monster new inspect`





# Direct Manipulation

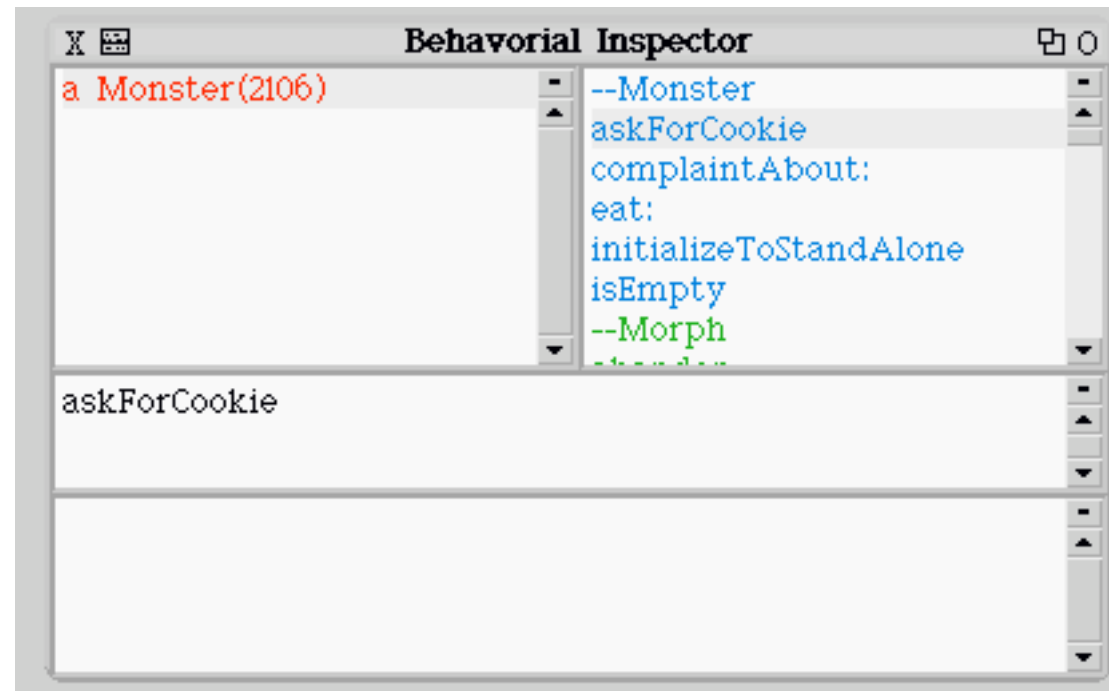
- Bring the halo
- Experiment



# Behavioral Inspector

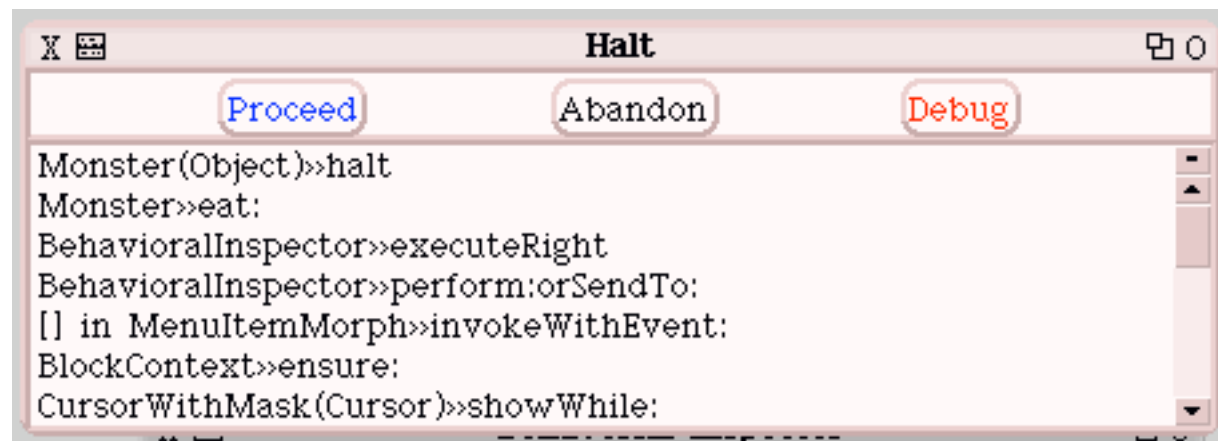
---

- Do not break encapsulation
- myObject behavioralInspect



# Debugger?

---



# Debugger !!!

The screenshot shows a debugger window titled "Halt". The stack trace is as follows:

```
Monster(Object)>>halt
Monster>eat:
BehavioralInspector>>executeRight
BehavioralInspector>>perform:orSendTo:
[] in MenuItemMorph>>invokeWithEvent:
BlockContext>>ensure:
CursorWithMask(Cursor)>>showWhile:
```

Below the stack trace is a toolbar with buttons: Proceed, Restart, Send, Step, Through, Full Stack, Where, and Browse.

The main display area shows the current execution point:

```
eat: C
    tummy add: C
```

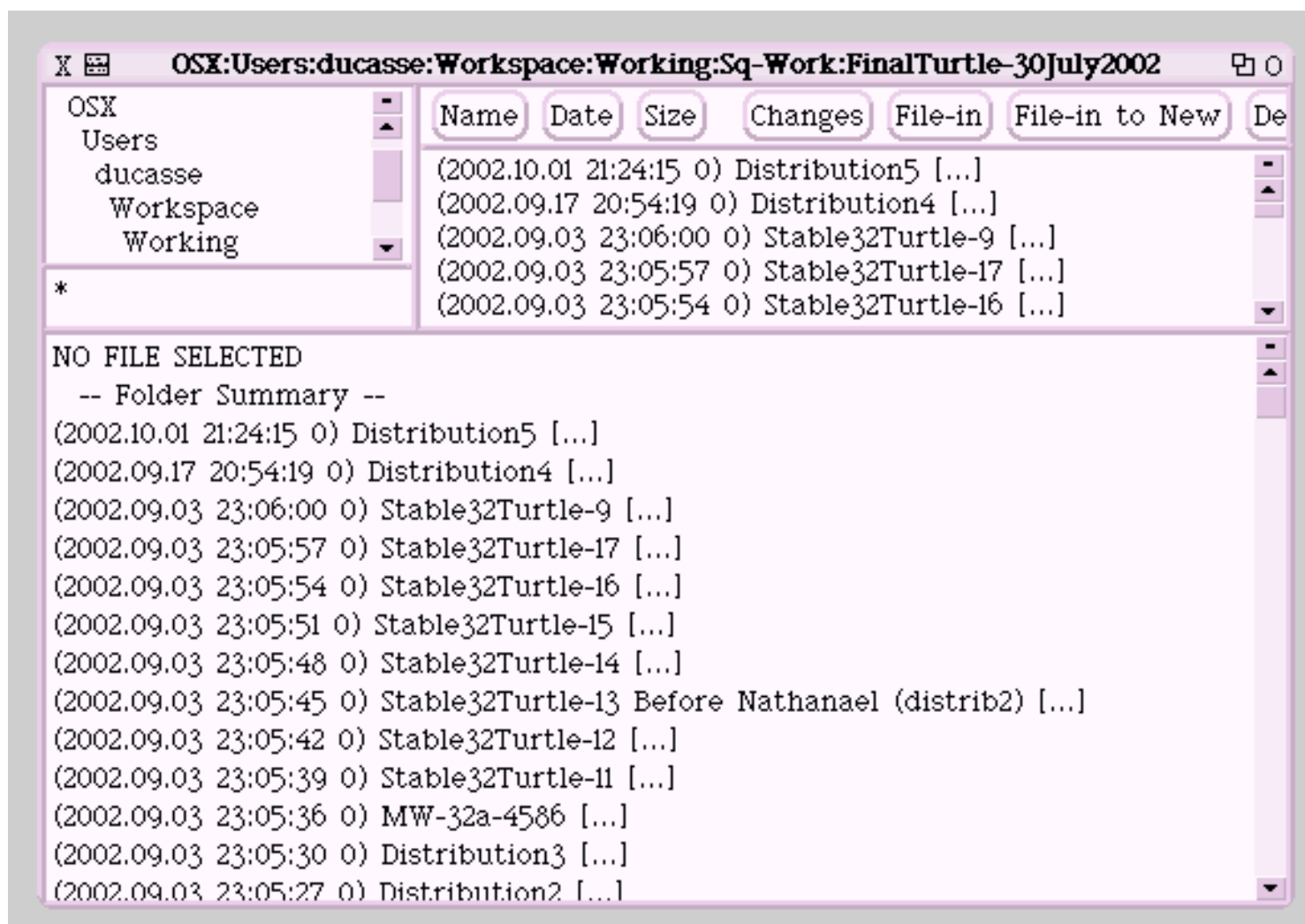
At the bottom, there are two variable inspectors. The left one shows the following variables:

bounds	a Bag()
owner	
submorphs	
fullBounds	
color	
extension	
tummy	

The right variable inspector shows:

thisContext	1
all temp vars	
C	

# FileList



# ChangeSorter: to sort your changes

The screenshot displays the ChangeSorter interface. On the left, a vertical list of change sets is shown, with 'Sesame' selected. On the right, a detailed view of the 'Sesame' change set is shown, titled 'Changes go to "Sesame"'. This view is organized into a grid of four panes. The top-left pane shows a list of files, with 'CookieMonster' and 'Monster' highlighted. The top-right pane shows a list of classes, with 'AniTurtle' and 'AniTurtle class' highlighted. The bottom-left pane shows a list of methods, with 'askForCookie' highlighted. The bottom-right pane shows the code for the 'askForCookie' method, which includes a comment: '↑ FillInTheBlank request: 'Give me cookie!!! (please)''.

**Change Set**  
make changes go to me (m)  
new change set... (n)  
find...(f)  
show category... (s)  
select change set...  
rename change set (r)  
file out (o)  
mail to list  
browse methods (b)  
browse change set (B)  
copy all to other side (c)  
submerge into other side  
subtract other side  
add preamble (p)  
add postscript...  
category functions...  
destroy change set (x)  
more...

**Changes go to "Sesame"**

MW-base	CookieMonster	AniTurtle	AniTurtle
4917dupNavBar-sw	Monster	Monitor	AniTurtle class
Sesame		ExtendedTurtleBeha	
More About Sound		TurtleforMW	
Squeak in 3D		TurtleBrowsers	
Squeak and the Inte		TurtleEnvironment	
askForCookie			
complaintAbout:			
eat:			
initializeToStandAlone			
isEmpty			
askForCookie			
↑ FillInTheBlank request: 'Give me cookie!!! (please)'			

# Message Names Finder

The screenshot shows a window titled "Message names containing 'match:'". At the top, there is a search bar with the text "match:". Below the search bar, a list of message names is displayed, including "encodeMatch:distance:", "findMatch:lastLength:lastMatch:chainLength:", "howManyMatch:", "match:", "match:fields:", "match:inContext:", "reorderParametersToMatch:", "scaleToMatch:", "startingAt:match:startingAt:", and "sunitMatch:". The "match:" entry is highlighted. To the right of the list, a pane shows the details for the selected message, displaying "Parser match:", "PositionableStream match:", and "String match:". Below the list and details pane, there is a row of buttons: "browse", "senders", "implementors", "versions", "inheritance", "hierarchy", "inst vars", "class vars", and "col". At the bottom, a text area displays the following output:

```
'foo*baz' match: 'foo23bazo' false
'foo' match: 'Foo' true
'foo*baz*zort' match: 'foobazort' false
'foo*baz*zort' match: 'foobazzort' false
'*foo*zort' match: 'afoo3zortthenfoo3zort' true
'*foo*zort' match: 'afoodezortorfoo3zort' true
"
```

The image shows a screenshot of the Smalltalk IDE. A 'Selector Browser' window is open, displaying a list of selectors. The selector `'*b' match: 'ab' --> true` is highlighted. Below the browser, the definition of the `match: text` method is visible. The method's documentation states: "Answer whether text matches the pattern in this string. Matching ignores upper/lower case differences. Where this string contains \*, text may contain any character. Where this string contains \*, text may contain any sequence of characters." The method signature is `self startingAt: 1 match: text startingAt: 1`. The browser also shows other selectors like `'ab' caseSensitiveLessOr` and `'ab' includesAnyOf: '*b'`.



# Methods in ChangeSets + Versions

The screenshot shows an IDE window titled "Methods in Change Set Sesame". The main content area lists several methods for the class "Monster":

- Monster askForCookie {asking}
- Monster complaintAbout: {asking}
- Monster eat: {behavior}
- Monster initializeToStandAlone {initialization}
- Monster isEmpty {behavior}

Below the list are several buttons: "browse", "senders", "implementors", "versions", "inheritance", "hierarchy", "inst vars", "class vars", and "col". The "versions" button is selected, and the content below it shows "eat: someItem".

A secondary window titled "Recent versions of eat:" is open over the main window. It displays a list of recent versions:

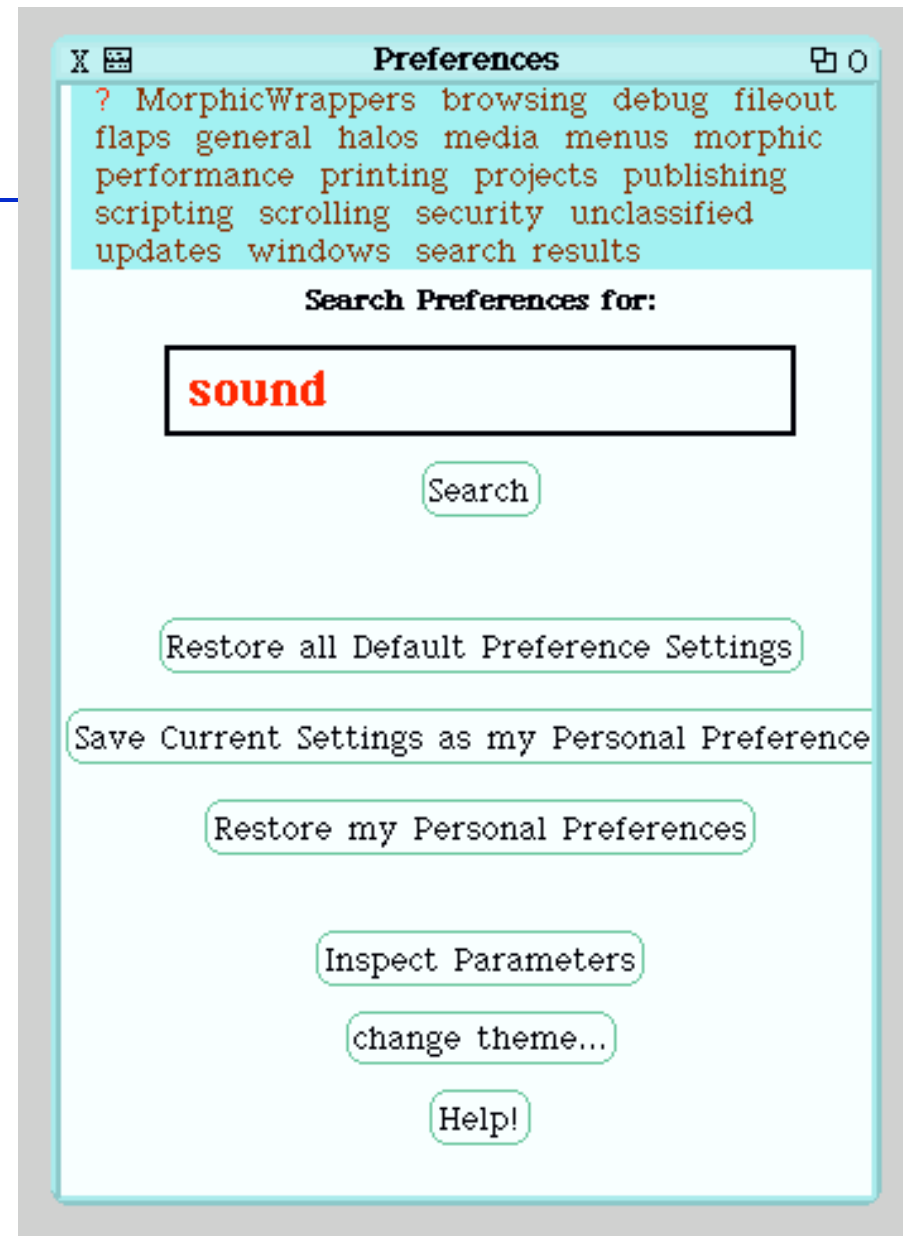
- sd 10/11/2002 21:55 Monster eat:
- sd 10/11/2002 21:31 Monster eat:

Below the list are buttons: "compare to current", "revert", "remove from changes", "help", "diffs", and "F". The "diffs" button is checked. The diff view shows the following changes:

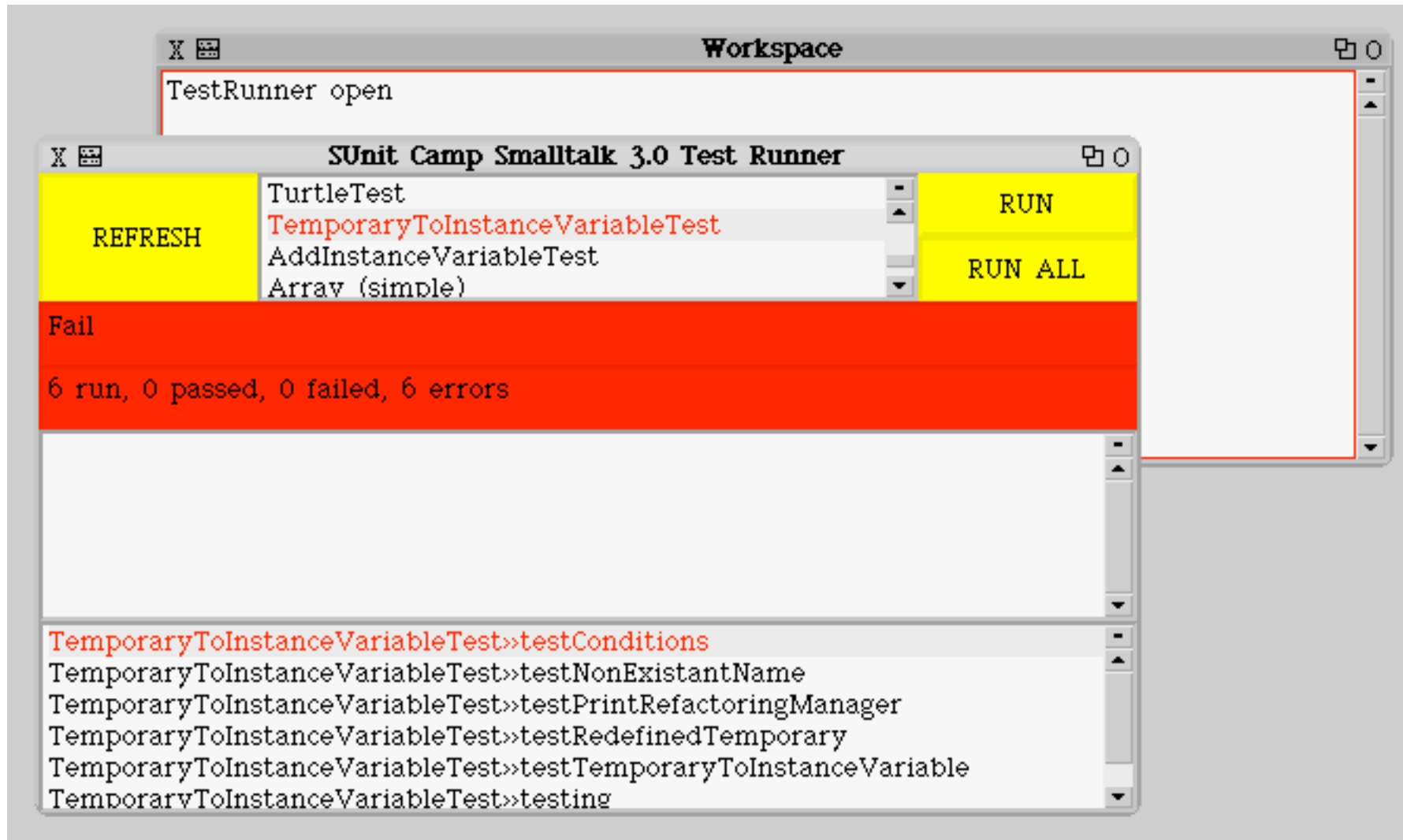
```
eat: someItem  
eat: someItem  
-  
tummy add: someItem
```

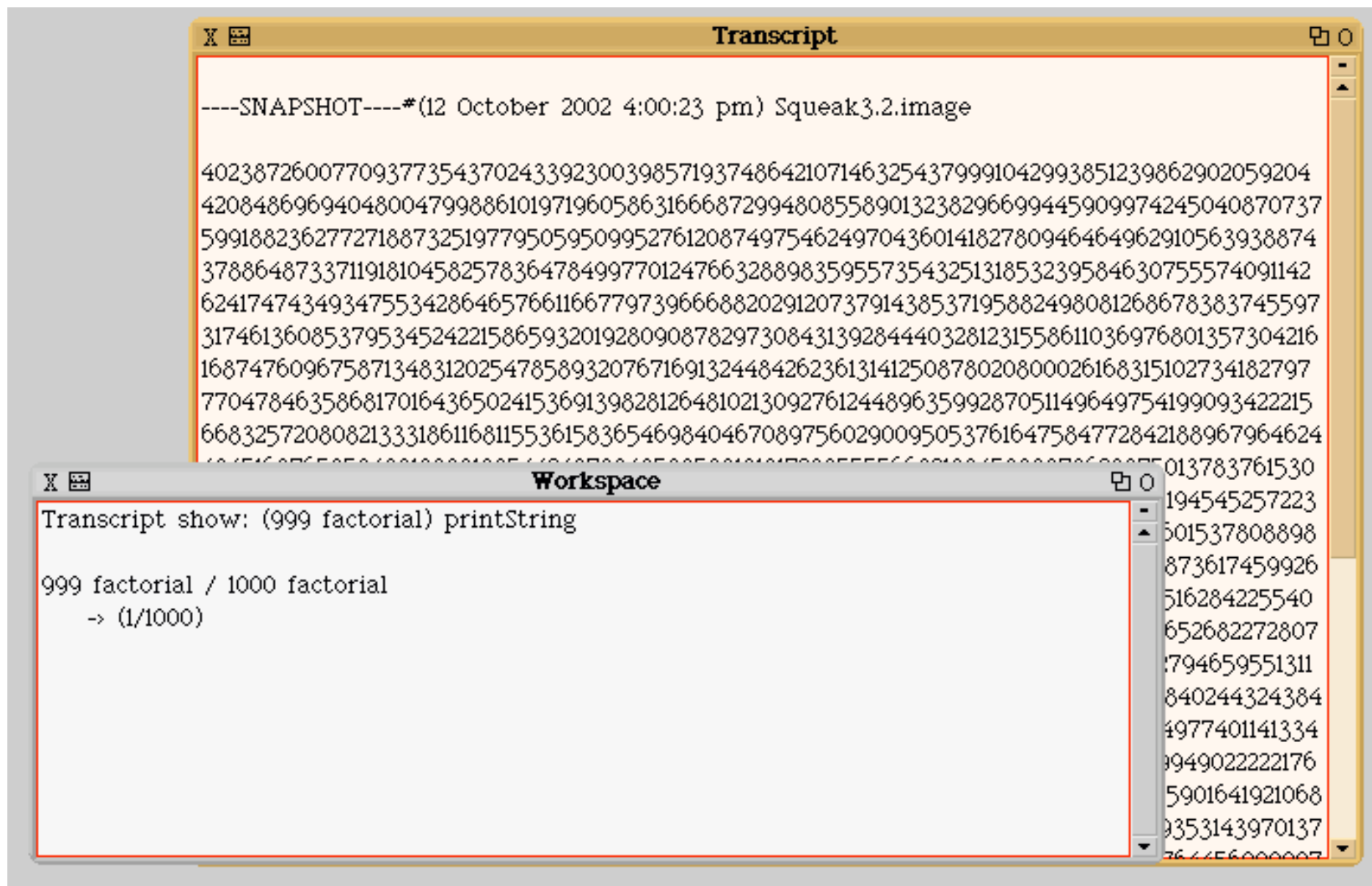
# Preferences

---



# SUnit

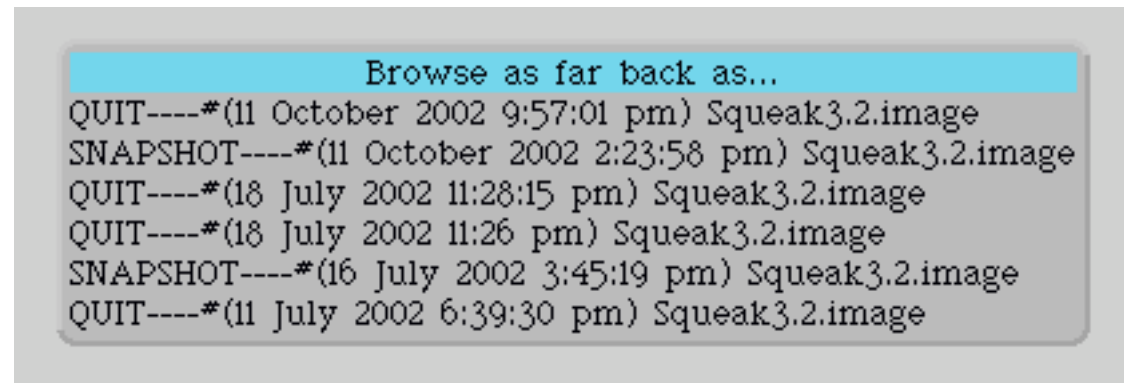
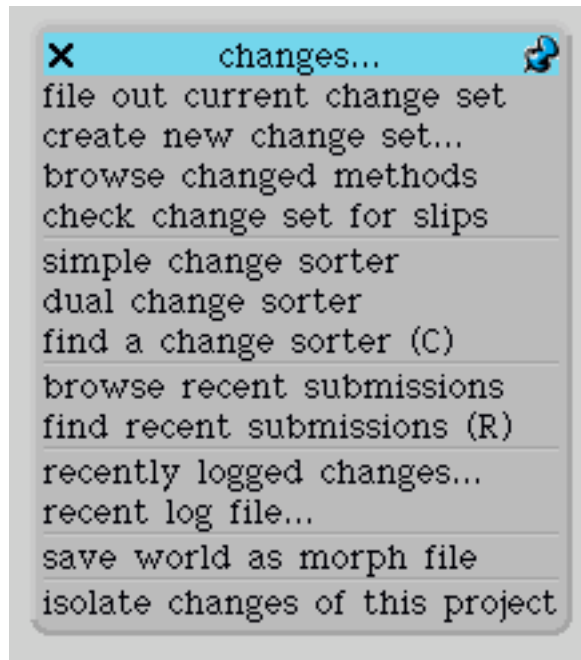




# Changes Menu...Recently logged files

---

- Everything you do is recorded



# Change your Mind

---

- Everything you do is recorded !!
- So try and learn how to recover your code
  
- You are smart so
  - Experiment,
  - learn for you, browse,
  - be aggressive, **\*\*\*all\*\*\*** the code is there