CS 683 Emerging Technologies: Embracing Change
Spring Semester, 2001
Doc 2 Starting Squeak
Contents

Learning Squeak 2
  Parts of Squeak 4
  Starting Squeak 5
Squeak Smalltalk Syntax 8
  Sample Program 10
  Assignment 11
  Variables 13
  Messages 14
    Unary Messages 16
    Binary Messages 17
    Keyword Messages 18
    Precedence 19
    Cascading Messages 22

References

Squeak Smalltalk: A Quick Reference, Andrew C. Greenberg, http://www.mucow.com/squeak-qref.html

Object-Oriented Design with Smalltalk — a Pure Object Language and its Environment, Ducasse, University of Bern, Lecture notes 2000/2001, http://www.iam.unibe.ch/~ducasse/WebPages/Smalltalk/ST00_01.pdf

Smalltalk Best Practice Patterns, Kent Beck, Prentice Hall, 1997

Reading

Squeak: Object-Oriented Design with Multimedia Applications, Guzdial, Chapter 1, and Chapter 2 sections 2.1- 2.3

Copyright ©, All rights reserved. 2001 SDSU & Roger Whitney, 5500 Campanile Drive, San Diego, CA 92182-7700 USA. OpenContent (http://www.opencontent.org/opl.shtml) license defines the copyright on this document.

Learning Squeak

To learn Squeak one must master:

- Smalltalk language syntax

  While Smalltalk syntax is simple it is not like C/C++/Java

- Squeak Programming Environment

  Requires more effort to learn at first, but worth the effort

- Squeak Class Library

  Squeak has a large library of useful code

  Don't code without it

- Object-oriented thinking

  This is the hardest part

- Smalltalk culture

  Smalltalker's have standard ways to code & solve problems

  See Smalltalk Best Practice Patterns by Kent Beck

Squeak & Bytecode

Squeak Smalltalk is compiled to a bytecode for a virtual machine language

Squeak's virtual machine (VM) interprets the bytecode

Squeak has VM's for:

- Windows 98/95, NT, ME, CE, DOS
- Macintosh, OS X
- Unix: DEC Alpha, HPUX, Linix, NetBSD, SGI, Sun
- Be OS, OS/2
- PDAs: Zaurus, Acorn RiscOS, DEC Itsy

Squeak can be used to develop applications for Palm Pilot

Squeak plugins for IE and Netscape allow Squeaklets run in web browsers

Squeak does have a just-in-time compiler. I do not know if it is in the main distribution yet.

Just-in-time compilers

- Compile bytecode to native machine code
- Cache the native machine code
- Run the native machine code
- Usually runs faster than interpreting bytecode

Smalltalk started using just-in-time compilers in early 1980s

Parts of Squeak

Executable Virtual Machine (Squeak, Squeak.exe)

- This is the VM that interprets Squeak bytecode

SqueakV2.sources

- Source code for most of Squeak class library

Squeak2.8.changes

- Source code for changes & new classes

Squeak2.8.image

- Bytecode of sources that are executed
- At first the image will appear to be an IDE for Squeak

As you use Squeak the image file will change. When you make changes to code they are saved in the change file. At some point you will want to start over with a clean copy of the image. Before you use Squeak make a backup copy of the image file and the changes file. When you want a fresh copy of the image, use copies of the backups.

Starting Squeak

You can download Squeak at: http://www.squeak.org/

Starting Squeak on Windows

Method 1

Drag and drop the Squeak image file on the Squeak application or Squeak.exe

Method 2

Double click on the Squeak image file

The first time you do this you may get a dialog asking for the application to run Squeak. Select the Squeak application or Squeak.exe

Starting Squeak on Macintosh

Method 1

Drag and drop the Squeak image file on the Squeak VM

Method 2

Double click on the Squeak image file

Starting Squeak on UNIX

Type:
squeak squeakImageFilename

where you need to replace squeakImageFilename with the actual name of the image file you wish to run

You path has to be set to include the program squeak

Running Squeak on Campus

Squeak is available in the Love Library computer lab and in BA 120. BA-120 is not yet open. I have not yet used Squeak on the Macs in Love Library lab.

Squeak is not available on Rohan. Squeak runs fine on UNIX. However, updating Squeak windows on the x-terms on campus is painfully slow. There is too much traffic to the X-term labs to usefully run Squeak there. (I also find KDE too slow to use in the lab - why do students still use it?)

Squeak in Love Library Computer lab - PC

Squeak is in two locations on each PC in the lab.

C:\Squeak
G:\Pc\cfg\squeak\Application

The G drive is a network drive. The Squeak files on this drive are read only. This will cause problems when you save the image or make any code changes. So do not use this copy of Squeak. Unfortunately this copy of Squeak is referenced in the start menu. Do not use that.

Use the copy of Squeak on the C drive. The copy on the C drive is read/write accessible. This means that if you use Squeak and make changes, the next user will see your changes. This may confuse them and allow them to copy your work. Use clean copies of the image and change files (you can copy them from the G drive). When you are done remove your working copy of the image and change files. We will see later how to save your work.

Squeak Smalltalk Syntax

The Xerox team spent 10 years developing Smalltalk

They thought carefully about the syntax of the language

Smalltalk syntax is

- Different from other languages
- Simple and compact

Basic Rules

Everything in Smalltalk is an object

All actions are done by sending a message to an object

Every object is an instance of a class

All classes have a parent class

Object is the root class

Entire Language Example
From Ralph Johnson

exampleWithNumber: x

"This is a small method that illustrates every part of Smalltalk method syntax except primitives, which aren't very standard. It has unary, binary, and key word messages, declares arguments and temporaries (but not block temporaries), accesses a global variable (but not and instance variable), uses literals (array, character, symbol, string, integer, float), uses the pseudo variable true false, nil, self, and super, and has sequence, assignment, return and cascade. It has both zero argument and one argument blocks. It doesn't do anything useful, though"

```
| y |
true & false not & (nil isNil) ifFalse: [self halt].
y := self size + super size.
#($ a #a 'a' 1 1.0)
    do: [: each | Transcript
            show: (each class name);
            show: (each printString);
            show: ' '].
↑ x < y
```

Sample Program

```
"A Sample comment"

| a b |
a ← 'this is a string'.                    "← (left arrow)is assignment"
a ←  'this is " a string that contains
      a single quote and a newline'.

a ← 'concat' , 'inate'.
a := 5.                                    ":= is assignment"
a := 1 + "comments ignored"  1.

b := 2 raisedTo: 5.
↑a + b                                     "↑ (up arrow) means return"
```

Type the '^' character to get ↑ (up arrow)

Type the '_' underbar character to get ← (left arrow)

Assignment

Squeak has two ways to indicate an assignment

· :=
· ←

← is the '_' underbar character (shift-hyphen)

The arrow character ← rather expressive. It is not on the standard keyboard, so Squeak displays _ character as the left arrow character. As a result you can not use the _ character in variable names in Squeak. Other versions of Smalltalk no longer use this symbol. Select one way to indicate assignment and use it consistently. Do not mix them as in the above program. In these notes I will use := just because it is easier to display in Squeak and html.

Multiple Assignments

Assignment statements return values!

```
| a b |
a := b := 3 + 4.
```

a and b now contain 7

Statement Separator

```
| cat dog |
cat := 5.
dog := cat + 2
```

A period is used as a statement separator

A period is optional after the last statement

Identifiers

An identifier (any name) in Squeak is of the form:

letter (letter | digit)*

Variables

```
| cat dog |
cat := 5.
dog := cat + 2.
```

Vertical bars at the top of a program declare variables

Variables must be declared

All variables are references to objects

Variables are initialized to nil

As we will see numbers in Smalltalk are objects. Internally references to objects require pointers. Always using a pointer to refer to a number would slow arithemitic operations. Most Smalltalk virtual machines will store numbers directly in a variable. At the programming level one does not see any difference in how numbers and other objects are handled.

Messages

In most languages >, if else are hard coded in the grammar

In Smalltalk operations are defined as methods in a class

+ is a method in the Integer class

In 3 + 4, + is a message sent to the integer 3

Using messages rather than hard coded grammar makes

- Parsing code simpler
- Language extensible

Three type of Messages

- Binary

  1 + 2
  12 / 6

- Unary

  12.3 printString
  '123' asNumber

- Keyword

  'Hi mom' copyFrom: 1 to: 3

All messages contain:

- Receiver
- Selector
- Zero or more arguments

Messages always return a value

Unary Messages

Format:
aReceiver aSelector

12 factorial

12 is the receiver
factorial is the selector
returns 479001600

'this is a string' reversed

'this is a string' is the receiver
reversed is the selector
returns 'gnirts a si siht'

'12345' isAllDigits

returns true

Each class in Smalltalk defines its own unary methods

Binary Messages

Format:

aReceiver aSelector anArgument

2 + 4

2 is the receiver
+ is the selector
4 is the argument
returns 6

Binary selectors are

- Arithmetic, comparison and logical operations
- One or two characters taken from:

  + - / \ * ~ < > = @ % | & ! ? ,

  Second character is never -

Using the above rules you can create your own binary messages in Smalltalk. You can make @? a binary method in a class.

Keyword Messages

Format:
receiver keyword1: argument1 keyword2: argument2 …

12 min: 6

12 is the receiver
min: is a selector with only one keyword
6 is the argument
returns 6

'this is a string' copyFrom: 1 to: 7

'this is a string' is the receiver
copyFrom:to: is the selector with two keywords
1 and 7 are the arguments
returns 'this is'

'this is a string'
findString: 'string'
startingAt: 4
caseSensitive: true

Precedence

First unary messages are parsed left to right
Binary messages are parsed left to right after unary messages
Keyword messages are parsed after binary messages

Parenthesis change the order of evaluation

Expression	Result
3 + 4 * 2	14
3 + (4 * 2)	11
5 + 3 factorial	11
(5 + 3) factorial	40320
'12' asNumber + 2	14

Arithmetical operations do not use normal mathematical precedence rules

Parenthesis must be used to separate multiple keyword messages in one statement

```
'this is a string' reversed
    findString: ('the cat is white' copyFrom: 9 to: 10)
    startingAt: 1 + 2
     caseSensitive: 2 + 2 = 4
```

Transcript

Special output window
Similar in purpose to Java's System.out and C++'s out

Useful Transcript messages:

open
  open the window

clear
  clear the window

show: aString
  display aString in the window

nextPutAll: aString
  add aString to the display buffer

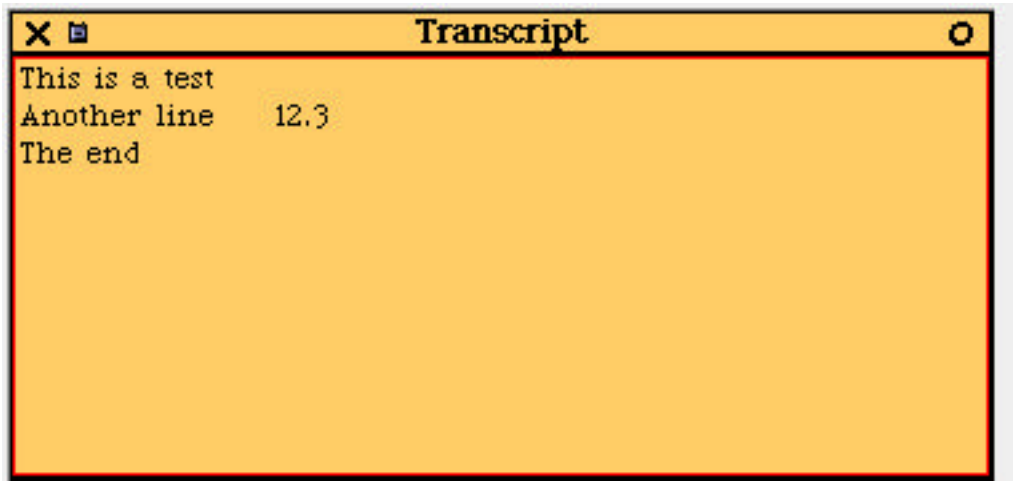
endEntry
  put contents of display buffer in window
  empty the buffer

tab cr space crtab crtab: anInteger
  put given character in the display buffer

Sample Program

```
Transcript open.
Transcript show: 'This is a test'.
Transcript cr.
Transcript show: 'Another line'.
Transcript tab.
Transcript show: 12.3.
Transcript cr.
Transcript show: 'The end'.
```

Result of Running Program



Cascading Messages

Format:

receiver selector1 [arg] ; selector2 [arg] ; ...

A cascade sends multiple messages to the same receiver

Messages are sent from left to right to the same receiver

Transcript

```
open;
show: 'This is a test';
cr;
show: 'Another line';
tab;
show: 12.3;
cr;
show: 'The end'.
```

Cascade Versus Compound Messages

Expression	Result
'hi mom' reversed asUppercase	'MOM IH'
'hi mom' reversed; asUppercase	'HI MOM'

Compound

In a compound message each message is sent to the result of the previous message

'hi mom' reversed asUppercase

First send reversed to 'hi mom'

The result is 'mom ih'

Now send asUppercase to 'mom ih'

Cascade

In a cascade message each message is sent to the same receiver

'hi mom' reversed; asUppercase

First send reversed to 'hi mom'

The result is not used

Now send asUppercase to 'hi mom'