Why Study Smalltalk

- Purest OO language, encourage OO programming
- Can inspect and change objects and the runtime system itself at run time
- Pioneered in many things
  - Graphical user interface (Window, Menu, Mouse)
  - Personal workstation
  - Push OO into success
    - I invented the term Object-Oriented, and I can tell you I did not have C++ in mind. -- Alan Kay
- Has an active community
- Could have deserved more popularity

Key Features

- Very simple syntax
- Everything is object
  - True, nil
- Class is object
- No control: if, loop are objects
- Dynamically typed:
  - Variable has no type.
  - The class hierarchy is the type system.
- The language is together with its interactive runtime system
  - Runtime written in the language itself
  - Can change the system on-the-fly
  - Debug system state (image), object, class hierarchy
- All data is private (protected), all methods are public.
- No manual memory management and pointers

History

- 1967: Inspired by Simula67, the first OO language
- 1971: Started by Dynabook project (Alan Kay)
  - Hardware: GUI, pointing device, external storage, etc.
  - Later led to Alto (“laptop”)
  - Software: for Children
- 1973: Xerox Alto Computer
  - First Smalltalk environment
  - The Alto Computer 1974 - Video
- 1972: Smalltalk-72
- 1976: Smalltalk-76
- 1980: Smalltalk-80
- Major contributors of Smalltalk:
  - Alan Kay (2003 Turing Award), Adele Goldberg, Daniel Ingalls, et. Al.

Smalltalk Today

- Squeak
  - Seaside: a dynamic web application development framework
  - Croquet: an open source platform for collaborative 3D multi-user online applications.
  - Sophie: a digital media assembly tool to create multimedia documents.
  - Scratch: a toolkit for children to write games, animated stories.
  - Cincom VisualWorks, IBM VisualAge, GNU Smalltalk, Dolphin, ...
Resources

Squeak:
- Download at http://www.squeak.org/
- Quick Tutorial:
  A Development Example for Squeak 3.9
- A Free Book:
  Squeak by Example
  http://www.iam.unibe.ch/~scg/SBE/

Required Reading

Readings:
- Smalltalk by Example, Chapter 1-6
  (Chapter 7-9 are also recommended, especially sections related to Array)
- Section "10.6 Smalltalk" in the textbook.

Smalltalk, Squeak video

- http://www.youtube.com/watch?v=y_3l08tI5wQ

The language is together with its interactive runtime system

Smalltalk Runtime

- Runtime written in the language itself
- Can change the system on-the-fly
- Debug system state (image), object, class hierarchy

Squeak Demo

- The files (.exe, .changes, .img, .sources)
- Menu and Mouse
  (red-button (usually left button), yellow-button (usually right button),
  blue-button(Alt+red, "morphic halo") )
- Save changes into .img and .sources
- Transcript: system console, log
- Workspace: Run code snippet, text documents, ...
  do it, print it, inspect it, explore it
- System Browser  (Class, Object, Instance variable, Method)
- Hierarchy Browser
- Method Finder
Demo: Race Car

http://www.youtube.com/watch?v=y_3l08tI5wQ

Demo: Quinto Game

Squeak by Example, Chapter 2

Lecture 13 – OO Programming, Spring 2008
CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2008

1. Create new class category: SBE-Quinto

2. Define new class: SBECell

   Object subclass: SBECell
   instanceVariableNames: ‘’
   poolDictionaries: ‘’
   category: ‘SBE-Quinto’

   SimpleSwitchMorph subclass: #SBECell
   instanceVariableNames: ‘mouseAction’
   poolDictionaries: ‘’
   category: ‘SBE-Quinto’

3. Add methods: initialize

   initialize
   super initialize.
   self borderWidth: 2.
   bounds := 0@0 corner: 16@16.
   offColor := Color paleYellow.
   onColor := Color paleBlue darker.
   self useSquareCorners.
   self turnOff

4. Inspect an object

   SBECell new (inspect it)
   self openInWorld (do it)

5. Add new class: SBEGame

   BorderedMorph subclass: SBEGame
   instanceVariableNames: ‘cells’
   poolDictionaries: ‘’
   category: ‘SBE-Quinto’

6. Add methods: SBEGame>>initialize

   initialize
   super initialize.
   n := self cellsPerSide.
   sampleCell := SBECell new.
   width := sampleCell width.
   height := sampleCell height.
   self bounds: (5@5 extent: ((width*n) @(height*n)) + (2 * self borderWidth)).
   cells := Matrix new: n tabulate: [
   :i :j | self
   newCellAt: i at: j ].

7. Put methods into category (protocol)
Demo: Quinto Game

8. Two more methods for SBECeLL
   mouseAction: aBlock
   mouseAction := aBlock
   mouseUp: anEvent
   mouseAction value

9. more methods for SBEGame
   cellsPerSide
   "The number of cells along each side of the game"
   ^10
   toggleNeighboursOfCellAt: i at: j
   (i > 1) ifTrue: [ (cells at: i - 1 at: j) toggleState].
   (i < self cellsPerSide) ifTrue: [ (cells at: i + 1 at: j) toggleState].
   (j > 1) ifTrue: [ (cells at: i at: j - 1) toggleState].
   (j < self cellsPerSide) ifTrue: [ (cells at: i at: j + 1) toggleState].

Demo: Quinto Game

9. More SBEGame methods
   initialize
   | sampleCell width height n |
   super initialize.
   n := self cellsPerSide.
   sampleCell := SBECell new.
   width := sampleCell width.
   height := sampleCell height.
   self bounds: (5 @ 5 extent: width * n @ (height * n) + (2 * self borderWidth)).
   cells := Matrix new: n tabulate: [:i :j | self newCellAt: i at: j].

Demo: Quinto Game

• File Out and File In

Demo: MineSweeper