CS 535 Object-Oriented Programming & Design
Spring Semester, 2003
Doc 1 Introduction

Contents

Introduction .................................................................................................................. 2
Learning Smalltalk ....................................................................................................... 7
VisualWorks ................................................................................................................ 12
  Starting VisualWorks ............................................................................................... 13
Some VisualWorks Environment ............................................................................... 17
  Windows on Startup ................................................................................................. 18
Examples .................................................................................................................... 21
  Using the Transcript ................................................................................................. 23
Exiting from VisualWorks ......................................................................................... 24
Some Text Editing Short Cuts .................................................................................... 25

References


Smalltalk Best Practice Patterns, Kent Beck

Object-Oriented Design Heuristics, Arthur Riel

Software Productivity Research, Inc. (www.spr.com)

2003 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.
Introduction

Goal

Understand how to use classes & objects in code

Syntax of language is easy

How to create code that is:

• Understandable
• Modifiable
• Maintainable
• Reusable
Main Idea in Object-Oriented Programming

Related data and operations belong together

**Design Heuristic 3.3**

Beware of classes with many public accessor methods

Many accessors indicate that related data and behavior are not being kept in one place
Kent Beck’s Indicators of Good Style

Once and only once

Don’t repeat

• Logic
• Methods

Lots of little pieces

Rates of change

Don’t put two rates of change together

Don’t mix variables that change hourly with those that change monthly
Why Smalltalk

- Best language to learn object-oriented thinking
- Productivity Gains

**Software Productivity Research Study**

<table>
<thead>
<tr>
<th>Language</th>
<th>Lines of code/function point</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smalltalk</td>
<td>21</td>
</tr>
<tr>
<td>Ada 95</td>
<td>49</td>
</tr>
<tr>
<td>Java</td>
<td>53</td>
</tr>
<tr>
<td>C++</td>
<td>53</td>
</tr>
<tr>
<td>COBOL</td>
<td>107</td>
</tr>
<tr>
<td>C</td>
<td>128</td>
</tr>
</tbody>
</table>
Why Smalltalk

• Experience with dynamic typed language

• Reading Code

• Tool Support in software development
Learning Smalltalk

• Smalltalk language syntax

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

• Smalltalk Programming Environment

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

• Smalltalk Class Library

  Smalltalk 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
Some History

1967  Simula-67

Language developed in Norway for simulations

Use classes and objects

Late 1960’s Alan Kay – Father of Personal Computer

Kay Ph. D. thesis addresses the question:

How will we interact with notebook size computers?

Dynabook

1970-80 Xerox Parc

Alan Kay, Dan Ingalls, Ted Kaehler and others work on Smalltalk

Small – Originally for children

talk - Code is to communicate
Smalltalk Influences

Object-Oriented Programming

GUI

  Macintosh
  Windows

Refactoring

Extreme Programming
Versions of Smalltalk

VisualWorks

VisualAge for Smalltalk

Squeak

Dolphin
Smalltalk MT

Smalltalk X

Smallscript (.NET Smalltalk)

PocketSmalltalk
Smalltalk & Bytecode

Smalltalk is compiled to a bytecode for a virtual machine

Bytecode is same on all machines

VisualWorks has VM's for:

  Windows
  Macintosh
  Unix

VisualWork’s virtual machine (VM) uses a JIT to compile bytecodes

Just-in-time compilers (JIT)
  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
**VisualWorks**

*Parts of VisualWorks*

Executable Virtual Machine (visual, visual.exe)

This is the VM that interprets Smalltalk bytecode

visual.sou

Source code for most of class library

visual.cha

Source code for changes & new classes
Does not exist until after you first use VisualWorks

visual.im

Bytecode of sources that are executed

At first the image will appear to be an IDE for Smalltalk

parcels

Code bundles
Starting VisualWorks

See the class wiki for instructions on downloading VW 5i4

Before Starting VisualWorks

Before you start VisualWorks make a copy of visual.im

You will need it later

Starting VisualWorks on Windows

Method 1

Drag and drop the image file on the Visual application or visual.exe

Method 2

Double click on the image file

The first time you do this you may get a dialog asking for the application to run the image. Select visual. You will have to find it first. It is in the bin directory.
Starting VisualWorks on Macintosh

Method 1

Drag and drop the image file on the visual application

Method 2

Double click on the image file
Starting VisualWorks on UNIX

Type:
  visual imageFilename &

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

You path has to be set to include the program visual
VisualWorks on Rohan

Image requires ~9MegBytes of disk space

Copy /opt/smalltalk/vw7nc/image/visualnc.im to a local directory

Change permissions so you have write permission on the local copy

Set the VISUALWORK environment variable

    setenv VISUALWORKS /opt/smalltalk/vw7nc (csh or tcsh)

    set VISUALWORKS='/opt/smalltalk/vw7nc'(sh)

Add the following to you path

    /opt/smalltalk/vw7nc/bin/solaris

To start the image:

    visual imageName
Some VisualWorks Environment

VisualWorks uses three logical buttons

- **Select** button
  Selects objects and text

- **Operate** button
  Opens a menu with context-sensitive commands

- **Window** button
  Opens a menu with window commands

### Mapping Logical to Physical Mouse Buttons

<table>
<thead>
<tr>
<th></th>
<th>3-Button</th>
<th>2-Button</th>
<th>1-Button</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select</td>
<td>Left button</td>
<td>Left Button</td>
<td>Button</td>
</tr>
<tr>
<td>Operate</td>
<td>Right button</td>
<td>Right button</td>
<td>Ctrl+Button</td>
</tr>
<tr>
<td>Window</td>
<td>Middle button</td>
<td>Ctrl+Left button</td>
<td>command+Button</td>
</tr>
</tbody>
</table>

You should perform the action described in the next few pages. One learns what one does.
Windows on Startup
Launcher

Ivan Tomek’s Introduction is a good way to start to learn about Smalltalk
Lesson Runner

- Introduction
- Fundamentals
- Class Library 1
- Class Library 2
- Writing SLessons
- Lesson Browser Tests

Lesson Description

Why Use Smalltalk anyway?

Exit Lesson Runner

Launch Lesson...
Do it (ctrl-d)
Compile and execute the selected code

Print it (ctrl-p)
Same as "do it" but also prints the result of running the code

Inspect it (ctrl-i)
Same as "do it" but also opens an inspector window on the result of running the code

Debug it (ctrl-d)
Opens the debugger to allow you to step through the selected code. We will cover the debugger later.
Examples

Do it
Print it
Using the Transcript

5 timesRepeat:
    Transcript
        show: 'Hello World';
        cr]
Exiting from VisualWorks

Selecting "Exit VisualWorks..." will prompt you to save the current image before exiting. The file name can be entered as desired.

The image file name is "lectureNotes".
Some Text Editing Short Cuts
Selection shortcuts (double click)

To select text, use the following double-click shortcuts.
Double-click at start or end of a window to select all text in the window.
Double-click at start of line to select the line. Does not work on the first line.
Double-click at end of line to select the line. Does on work on the last line.
Double-click just after an opening (or just before the closing) of ' or [ or ( or " selects
all text surrounded by the symbols.
Double-click inside a word or selector to select the word or selector.

Ctrl keys

Press at the same time the control key and the second key to:

<Ctrl> f inserts ifFalse: into the text.
<Ctrl> t inserts ifTrue: into the text.
<Ctrl> g inserts := into the text.
<Ctrl> d inserts today's date into the text.
<Ctrl> s (search or find) finds the next instance of the string in your copy buffer (last
copied or cut string)
<Ctrl> e (replace) opens a replace dialog
<Ctrl> a (find) opens a find dialog
<Ctrl> c is equivalent to Copy,
<Ctrl> z will Undo the most recent text change,
<Ctrl> v is equivalent to Paste

ESC keys

Press and release the ESC key then press the second key to:

ESC b changes the selected text to bold.
ESC i changes the selected text to italic.
ESC u underlines the selected text.
When the letter is uppercase (B, I, U), the effect is reversed: ESC U removes
underline, etc.
ESC + increases the font size of the selected text
ESC - decreases font size of the selected text
ESC followed by < or ' or [ or ( adds surrounding < ' " [ ( to the selected text.
ESC followed by <tab> selects the text just typed in
ESC x removes style changes to current selection and returns to default font.