Improving Usability of Refactoring Tools

Emerson Murphy-Hill
emerson@cs.pdx.edu
Maseeh College of Engineering & Computer Science

Problems and Solutions

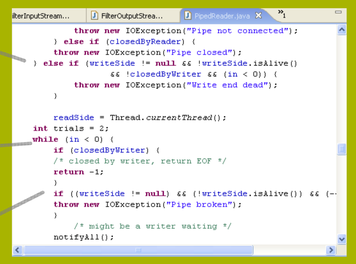
I observed 11 programmers over about ½ hour each. Each attempted to perform Extract Method refactorings on several large code bases using Eclipse. I observed two main difficulties.

Observation 1: Statements are Hard to Select

inconsistent indentation

multi-line statements

long lines

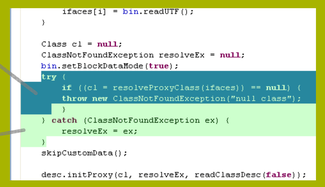


My Solutions

SelectionAssist

mouse selection

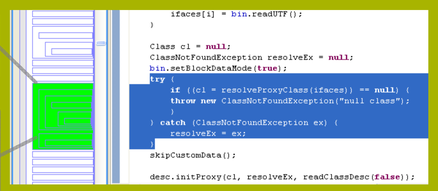
statement extent visual cue



BoxView

box = statement

click box to select a statement

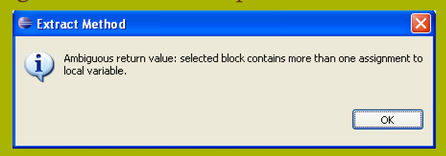


Observation 2: Error Messages are Bad

Scary to programmers

Insufficiently descriptive

Only one error detected



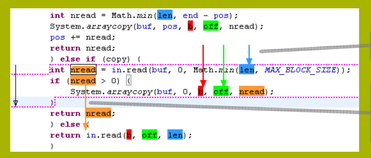
Problem point not identified

All errors look the same

My Solution

Refactoring Annotations

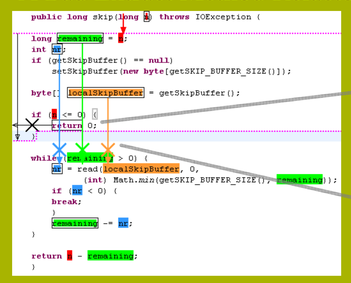
A Good Extract Method Refactoring



arguments

returned variable

A Bad Extract Method Refactoring



conditional return

multiple return values

Usability Principles

Selection tools should:

- Be lightweight and unobtrusive
- Abstract away code formatting
- Help select meaningful program constructs (task dependent)
- Allow the programmer to use either the keyboard or mouse

Expressions of violations of refactoring preconditions should:

- Indicate the location of every error
- Indicate the amount of work required to recover
- Be easily distinguishable from one another
- Indicate errors as relations between program elements

Experimental Validation

To simulate the two problematic stages of the Extract Method refactoring (at left), I asked each of 16 object-oriented programming students to perform two tasks.

Task 1: Select All if Statements

Programmers tried to select about 50 if statements using the keyboard/mouse, Selection Assist, and Box View. I measured the number of mis-selected statements and the time to perform the selection.

Task 2: Find Violated Preconditions

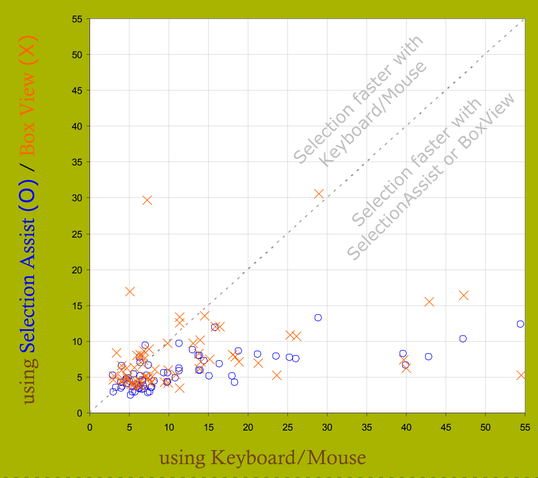
Programmers tried to identify all Extract Method precondition violations in 8 segments of code using the Eclipse Extract Method Wizard and Refactoring Annotations. I measured the number of mis-identifications and time to identify all violations.

Summary of Results

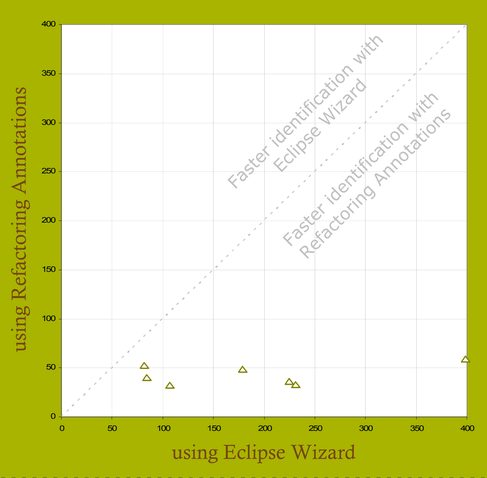
	Mis-Selected If Statements	Correctly Selected If Statements	Mean Selection Time
Mouse/Keyboard	37	303	10.4 seconds
SelectionAssist	6	355	5.5 seconds
BoxView	2	357	7.8 seconds

	Failed to Identify Error	Identified Non-Error	Mean Elapsed Time
Eclipse Wizard	11	28	164 seconds
Refactoring Annotations	1	6	46 seconds

Mean Time (in Seconds) to Correctly Select Each if Statement



Mean Time (in Seconds) to Correctly Identify Precondition Violations for Each Method



Would You Be Likely to Use This Tool Again?

Tools Available at http://multiview.cs.pdx.edu/refactoring

Research Supported by the National Science Foundation grant number CCF-0520346