



## GS04: Tools and Environments

### Lab Session 1: Program Editors

The aim of this lab session is to explore how modern interactive development environments apply the principles and meet the requirements for program editors that we have discussed in the previous lecture. We will investigate the Eclipse Java Editor as a representative example.

#### Language-sensitive editing

Create a class in Java and while doing so explore how Eclipse parses your input on the fly, explore how it performs syntax-directed editing. Explore how Eclipse handles syntactic errors and how it supports pretty-printing.

#### Static-semantic correctness

Create a number of static semantic java errors (such as undefined variables, incompatible types, uncaught exceptions, unused declarations, uninitialised variables etc) in your Java class and explore how Eclipse detects and visualizes them. Note how static-semantic checking depends on syntactic correctness. Explore this by introducing a syntactic error and note that static semantic errors are no longer detected.

#### Auto-completion

Identify the different ways in which the Eclipse Java program editor avoids the introduction of static semantic errors by automatic completion of program text.

#### Browsing

Import the various Java sources from <http://pizza.cs.ucl.ac.uk:8082/gz04CW/> into a new Eclipse project and explore the browsing support that Eclipse provides to understand the relationship between the various classes.

#### Re-factoring

Explore how Eclipse supports you in making significant changes to Java code. Try to rename declarations, change their location in an inheritance hierarchy etc.

#### Documentation

Write some API documentation for the web service implementation in <http://pizza.cs.ucl.ac.uk:8082/gz04CW/> to explore the Eclipse support for API documentation generated with java-doc.