Eclipse as a Teaching Platform for Kenya

Student: Thomas Timbul (MEng 4)
Supervised by: Robert Chatley

Publication

- Accepted for publication in the proceedings of ESEC/FSE 2005 (research demos)
- IBM Eclipse innovation award scheme

Presentation Contents

- Introduction to Kenya
- Motivation for KenyaEclipse
- Tool Demonstration
- Approach Criticism
- Conclusions
- Future Work
- Questions

Introduction to Kenya

- Robert Chatley 2001
- 'mini' Java
- Hides issues complicated for beginners
  - No packages
  - No qualified method call (`instance.call()`)
  - No access modifiers (`public`, `private`, …)
- Has its own 'IDE'
- Direct translation into full Java

Motivation for KenyaEclipse

- Promote the use of professional tools
  - Many students code using plain text editor
  - Introduce 'advanced' features earlier
  - 'Produce' more efficient programmers
- Pedagogical help through style guidance
  - Mistakes are caught only during marking
  - Hard to weed out 'bad style'
  - Use available tools to create automated style guide

KenyaEclipse feature overview

- Ported functionality
  - Compilation errors highlighted 'as-you-type'
  - Switching between Kenya/Java
  - Running & Debugging
- New functionality
  - Automated Style Guidance
  - Code completion proposals
  - Variable reference/occurrence highlighting
  - Basic refactoring (renaming)

Tool Demonstration

- Getting started
- Running & Debugging
- Some ‘advanced’ features
- Configurable Style Guidance Module
  - Step-by-step guidance and resolution
  - Omitted ‘break’ statements
  - Metric style measures
  - Shadowed constants

Approach Criticism - Benefits

- Improvements to teaching
  - Integrated guidance and help
  - Stylistic errors caught early and explained comprehensively
  - Introduction to readily available, yet often undiscovered, IDE tools
  - Familiarisation with production-level environment

Approach Criticism - the other side

- Too much automation?
  - Know the basics (command line compilation)
  - Students start relying on tools for correction
    - Like spell checking or calculators
- Tool introduction not gradual enough?
  - Information overload
  - ‘Right feature at the right time’
- Loss of independence
  - Students may bind themselves to Eclipse
  - Choose the best tool for each situation

Conclusions

- IDE (Eclipse) provides excellent tool base
  - Teaching tool creation much simplified
  - Re-use available technology/techniques
  - No need to ‘re-invent the wheel’
- BUT
  - Bear in mind who the target audience is
  - Tools cannot easily replace good teachers

Future Work

- Style Guidance
  - Compiled style patterns rather than classes
  - Generic for commercial languages
- As a Teaching Platform?
  - Adapt tool to student’s level
  - Explicit supervisor control of features
    - Control available IDE features
  - Dynamic programming language
    - Control language features (much harder)

Questions ?

All project material -
http://www.doc.ic.ac.uk/~tt101/kenya