CIF2013 – Java Crash Course

Lecture 1

Instructor: Lee Maclin

Group web site

- Subscribe – Send email to:
  - cif2013-subscribe@yahoogroups.com
- Group home page
  - http://groups.yahoo.com/group/cif2013
- Post message
  - cif2013@yahoogroups.com

Purpose of this class

- Primary purpose: To teach software development in a financial context
- Secondary purpose: To introduce financial concepts that are best discussed in terms of computing. Examples:
  - High frequency data and trading
  - Back-testing and simulation
  - Order book mechanics
- This is not a class for learning basic Java. You can do that on your own

Books

See *links* section on group site

Why no single text book?

Look at the syllabus!
We draw on too many disciplines.

Can I really learn basic Java in 24 hours?

- You can learn basic Java in 12 hours
- You will spend the next five years improving your programming skills
- In this class, we will cover what the book tells us should be covered in 7 hours
- Best way to learn Java: Two passes through the book
  - First pass: Skim
  - Second pass: Do the examples
- If you are new to computing, do it this weekend or you will fall behind when the classes start

Why Java?

- Fast execution – Very close to C++
  - Worst case scenario: Slower execution than C++ by factor of 2
    - http://scribblethink.org/Computer/javaCbenchmark.html
- Faster development time than C++ by a factor of 5!
  - For most financial applications, development time is more important than running time.
  - For applications where that is not true, you can always optimize with native code
- Easier to learn – Takes less time to make developers productive
- Widely used – Experiment?

HelloWorld.java

- Uploading HelloWorld.java
  scp HelloWorld.java maclin@math1.cims.nyu.edu:HelloWorld.java
- Logging in to your Courant account
  ssh -l maclin math1.cims.nyu.edu
- Editing code: HelloWorld.java
  vi HelloWorld.java
- Compiling code
  javac HelloWorld.java
- Running code
  java HelloWorld

Logging in from various platforms

- You can come here and work on the machines in the labs (not recommended)
- You can log in from a Linux / Unix / Mac command line as follows: ssh –l maclin math1.cims.nyu.edu
- Note: Do not use the above machine! Use the machine NYU computing services tells you to use
- You can get a Unix like shell for your Windows machine – Might as well be Git

Git shell for Windows

- Do a Google search for git-1.7.4
  - Git-1.7.4-preview20110204.exe
- Install it
- What you get:
  - perl
  - git
  - ssh
  - scp
  - ..more
- Example of using scp to move files to your Courant account: scp <from> maclin@math1.cims.nyu.edu:<to>

Integrated Development Environments

- IntelliJ IDEA
  - http://www.jetbrains.com/idea/
  - What Eran uses. Probably your best choice.
- Eclipse
  - www.eclipse.org
  - What I use (out of habit)
- NetBeans
  - The instructions for installing NetBeans are in the appendix of your Java book
- On your own computer, in addition to an IDE, you will have to install the Java runtime environment (JRE), and the Java Development Kit (JDK)

Get info on Courant's machines here

http://cims.nyu.edu/webapps/content/systems/userservices/netaccess/tunnel

Starting a project in your IDE

- We will make two source directories
  - src
  - junit
- Add junit to your libraries
- Create a package called ‘lecture1’ in source directory ‘src’
- Create a package called ‘lecture1’ in source directory ‘junit’
- Setting command line arguments for HelloWorld.java

Running and debugging

- Running and stopping
- Debugging
- Stepping through execution
- Setting breakpoints
- Setting conditional breakpoints
- Examining variables
- Evaluating expressions

A series of short examples

- Variables.java
- Conditionals.java
- Loops.java
- ComparisonsAndEquality.java
- DoubleComparator.java
- ArraysExample.java

Passing arguments to methods

- All basic data types – int, float, double, etc. – are passed by value. In other words, the contents of the variables are copied onto the stack. That is why making a change to the new variables has no effect on the old ones
- For complex data types – objects or arrays, the value that is pushed onto the stack is a reference to the object
  - The contents of the original data may be changed via this reference
  - Assigning a value to this reference has no effect on the original data

More examples

- ArgumentsToMethods.java
- ReturningAValue.java