# Object-Oriented Programming
# Course Overview

Ewan Klein

Inf1 :: 2008/09

Outline
Organizational Issues
Learning Java
Dive into Java







## Who and What

Lecturer: Ewan Klein

Teaching assistant: Joshua Ritterman

Tutors: Similar to FP

Newsgroup: Same as last semester

Webpage: `http://www.inf.ed.ac.uk/teaching/courses/inf1/oop`





## Textbooks

- Kathy Sierra & Bert Bates (2005) *Head First Java*, 2nd Edition, O'Reilly.
  This is the official text for the course. Available online via Safari.
- Sharon Zakhour et al (2006) *The Java Tutorial: A Short Course on the Basics*, 4th Edition, Addison-Wesley.
  More dense, and targetted at readers with more programming experience. Optional. Available via Safari and on Sun website.

## Course HOWTO

- Come to the lectures
  - but the lecture slides are not a textbook …
- Do the assigned reading from *Head First Java*
  - but you can't learn a language just by reading about it …
- Do the weekly Lab Exercises
  - Make use of the demonstrators.
  - ≈ FP's Tutorial exercises
  - Web-CAT — next week: intended to give you on-the-spot feedback.
  - Lab fortnight — starts today!
- Go to the Tutorials (from Week 3)
  - Follow-up / further support for Lab Exercises.
  - Opportunity to discuss problems.
- Ask questions, give feedback, get help
  - Newsgroup
  - Tutors
  - TA
  - Class reps





## Assessment

Lab exercises: In general, not graded. Except

Class Test:

- Not an in-class written test.
- Consists of one of the lab exercises.
- Released Week 3, 30th Jan
- To be submitted on 6th Feb.
- Worth 5% of total marks.
- Mainly to show that you're keeping up with the course.

Exam:

- Programming exam (the other 95%), scheduled as part of normal exam diet.
- Mock programming exam scheduled for Week 9.





## Learning Outcomes (the official story)

1. Name, explain and apply the core concepts and constructs used in imperative and object-oriented programming.
2. Given a detailed design, develop a working program that implements the design.
3. Develop small programs, or components of larger ones, or modify existing ones, to solve clearly defined programming problems.
4. Given a clearly described component, develop a test set and test code for the component. Use code review and debugging tools to identify the location of a fault in an erroneous program.
5. Run and analyse a given program; describe how well it works compared to its specification, or identify ways in which it fails.
6. Apply basic tools to aid in developing programs (e.g., IDE, version control).

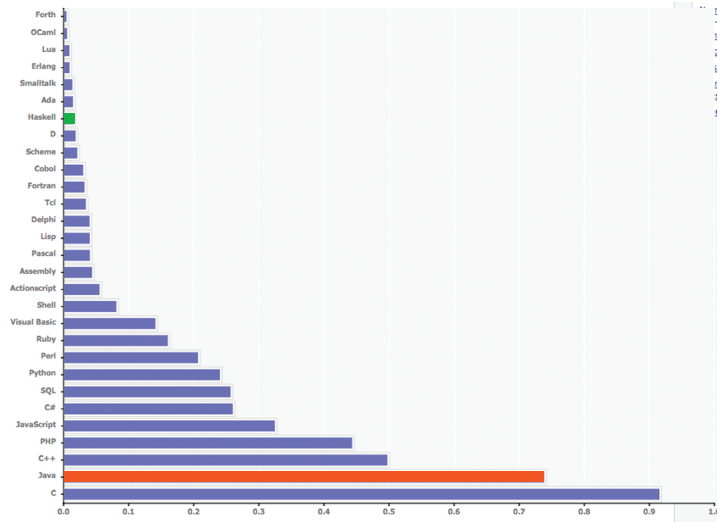




## Some Questions

- Why another programming language?
- Why Java?

## Java popularity (http://www.langpop.com/)




Data based on Yahoo pagecount, Craigs List (jobs), Amazon books, Freshmeat OS projects,





## Some Questions

- Why another programming language?
- Why Java?
- How does Java compare to Haskell?
  - Lots of differences will become apparent.
  - Both use static typing.
  - Java doesn't have type inference, and generally more verbose.
  - Statements and operators vs. classes and objects — presentation has to follow a kind of spiral, not a straight line.





## A Programming Language is …

- a language!
  - words
  - phrases
  - sentences
  - idioms
- How do you learn a new language?
- algorithms
- software design





## The Object-oriented Approach

- Java is one of several object-oriented programming languages.
- Objects are reusable software components that model real-world items.
- How we think about the world (e.g., people, animals, cars, *etc.*)

## A First Example

**Hello World!**

```
public class HelloWorld {

    public static void main (String[] args) {
        System.out.println("hello, world");
        // now for a hackerese version
        System.out.println("|-|3LLO, \\/\\/ORLD");
    }
}
```

- Confused? Overwhelmed?
- Don't panic — it's normal.





## Creating a New Class

1. All Java code sits inside a class.
2. By convention, class names are capitalized and in 'CamelCase'.
3. Each class goes into a file of its own.
4. So, use a text editor (e.g., emacs) to create a file called `HelloWorld.java`.
5. The name of the file has to be the same as the name of the class, and suffixed with `.java`.

**At the terminal**

```
% emacs HelloWorld.java
```





## A First Example

**Declare a (public) class**

```
public class HelloWorld {

    public static void main (String[] args) {
        System.out.println("hello, world");
        // now for a hackerese version
        System.out.println("|-|3LLO, \\/\\/ORLD");
     }
}
```

- Basic form of a class definition.
- Class definition enclosed by curly braces.





## A First Example

**Declare the `main()` method**

```
public class HelloWorld {

    public static void main (String[] args) {
        System.out.println("hello, world");
        // now for a hackerese version
        System.out.println("|-|3LLO, \\/\\/ORLD");
     }
}
```

- We need a `main()` method to actually get our program started.
- All our other code is invoked from inside `main()`.
- The `void` modifier means that the method doesn't return a value.
- The argument of the method is an array of `String`s; this array is called `args`.

## A First Example

Print a string to standard output

```
public class HelloWorld {
    public static void main (String[] args) {
        System.out.println("hello, world");
        // now for a hackerese version
        System.out.println("|-|3LLO, \\/\\/ORLD");
     }
}
```

- `System.out.println("hello, world");` — this is a statement to be executed.
- Statements must be terminated with a semi-colon (`;`).
- The argument is the string to be printed.
- Strings must be demarcated by double quotes (not single quotes, or doubled single quotes).





## Compiling

- The program needs to be compiled before it can be executed.
- Use the `javac` command in a terminal.

At the terminal

```
% javac HelloWorld.java
```

- If there's a problem, the compiler will complain.
- If not, the compiled code will be produced as a file `HelloWorld.class`.

At the terminal

```
% file HelloWorld.class
HelloWorld.class: compiled Java class data, version 49.0
```





## Running the Program

- Now that we have compiled code, we can run it.
- Use the `java` command in a terminal.

At the terminal

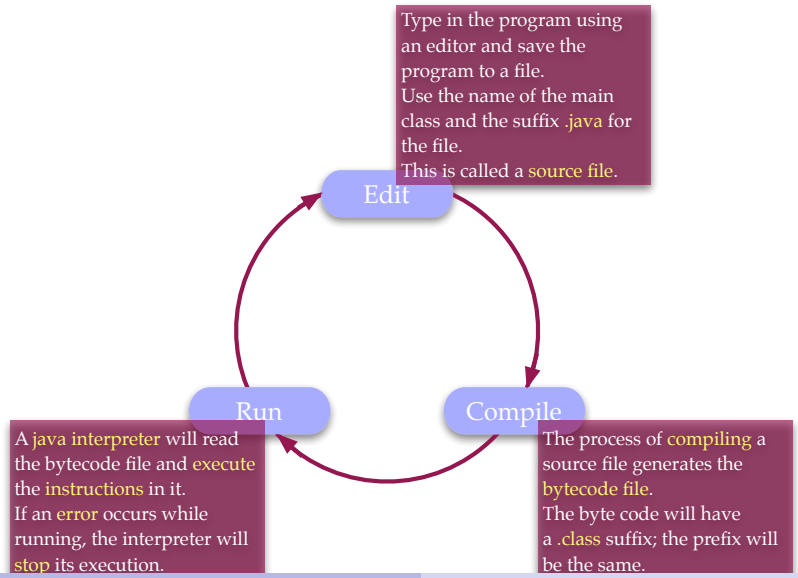
```
% java HelloWorld
hello, world
|-|3LLO, \/\/ORLD
```

- Hooray, it worked!
- Note that we omit the `.class` suffix in the run command.





## Edit-Compile-Run Cycle



## Edit-Compile-Run Cycle

- The program needs to be compiled before it can be executed.
- If you edit a program, you need to compile it again before running the new version.





## This Week's Reading

**Head First Java**

Read Chapter 1 (pp. 26)