Software Development

CS2500: Syllabus

M.R.C. van Dongen

September 19, 2011

1 Main Topics

This section provides a brief overview of the main topics of CS2500. The course is divided into three parts:

Basic Java: Students learn object oriented programming in Java. Most of the time will be spent on this.

Software development: Students learn principles of good OO software development.

Design: Students study useful design patterns and design principles. They will learn to analyse and compare different approaches and solutions to solving a problem. In addition, they will learn the difference between structural, behavioural, and creational design patterns.

2 Reading List

The course is based on the following book:

Required: *Head First* Java, *by Kathy Sierra and Bert Bates,* O'Reilly, ISBN: *978-0-596-00712-6.*

We will use some material from the following book:

Optional: *Head First* Design Patterns, *by Eric and Elisabeth Freeman,* O'Reilly, ISBN: *978-0-596-00920-5.*

Some of the course material is based on the following two books. If you like programming and want to learn more, then I'm more than happy to recommend them.

Recommended: *Effective Java,* by Joshua Bloch, *Addison-Wesley,* ISBN: 978-0-321-35668-0. Learn good principles of Java programming from the horse's mouth. Josh Bloch designed, implemented, and maintained many of the Java platform libraries. Here he shares 57 "nuggets" — rules and code examples showing what works, what doesn't, and how to use the language and its libraries to best effect.

> "I sure wish I had had this book ten years ago. Some might think that I don't need any Java books, but I need this one."
>
> — James Gosling, Fellow and Vice President, Sun Microsystems, Inc.

Optional: *Java Generics,* by Maurice Naftalin and Philip Wadler, *O'Reilly,* ISBN: 978-0-596-52775-4. This book is a bit technical, but it is a great source of information about generics.

3 Teaching Methods

There will be three weekly lectures:

Monday: 2pm–3pm in WGB G15;

Wednesday: 2pm–3pm in WGB G08; and

Friday: 11am–12noon in WGB G14.

There will be weekly labs:

Tuesday: 4pm–6pm in WGB G24.

4 Assessment

The assessment of this course are as follows:

Total marks: 300;

Written exam: 1×3 hour exam: 225 marks;

Continuous assessment: 75 marks: 11 assignments (5 marks each) and 1 in-class test (20 marks).

5 Learning Outcomes

5.1 Basic Java

Besides learning about basic compiler usage, which is *not* examinable for the written exam, the learning outcomes for `Java` are as follows:

- Classes and objects;
- Variables, types, and declarations;
- Arrays; Loops and branching;
- Attributes and methods: state and behaviour;
- Object creation and garbage collection;
- Wrapper classes and autoboxing;
- Encapsulation and visibility modifiers;
- `Math` and `Java` libraries;

- Inheritance; polymorphism; Overriding versus overloading;
- Is-a versus has-a versus owns-a: extends versus aggregates versus composes;
- Constructor chaining;
- Delegation, aggregation, and composition;
- Abstraction; Abstract classes and interfaces; Static methods;
- Enumerated types (in-depth);
- Risky behaviour: exceptions; files and I/O;
- Graphical User Interfaces (GUIs) and GUI component classes;
- `Swing` library; Action events and event listeners; Inner classes;
- Generics and collections; Commonly used collection types (sets, lists, and maps).

5.2 Software Development

You will learn the following about good practice of OO software development.

- `JavaDoc` and guidelines for program layout and presentation;
- `JUnit` testing;
- Releasing your code;
- Version control.

5.3 Design Principles

A *design principle* is a basic tool or technique that can be applied to designing or writing code to make that code more maintainable, flexible, or extensible. You will learn useful design principles and how to use them.

5.4 Design Patterns

In addition, they will learn some of the following design patterns and how to classify these design patterns.

Strategy: Defines a family of algorithms, encapsulates each one, and makes them interchangeable.

Observer: Defines a one-to-many dependency between a *subject* and *subscribers*, so that if the subject changes, all its subscribers are notified automatically. (`Observer-Listener` in `Swing`.)

Factory: Separates object creation from the class that defines the object (avoids `new`) and hides the details of object creation behind a well-defined interface.

Singleton: Ensures that a class has (at most) one instance, and provides a global point of access to the instance.

Iterator: Provides sequential access to members of aggregate object without exposing their representation.

6 First Lectures

We won't study the book in the first couple of lectures. Instead we'll study basic computer science algorithms, including:

- Searching and sorting, including: linear search, binary search, bubblesort, quicksort, and radix sort.
- Hashing. In computer science, *hashing* refers to the operation of turning a data structure into a number (and `int`). Ideally, random data structures should result in random `int`s. An important application of hashing is improving search.
- Bloom filters. A *Bloom filter* is a space-efficient probabilistic data structure which is used to determine whether a given item is a member of a set. An interesting application is the recognition of malicious web sites at 9.6 bits/malicious URL. This lets you store 1 000 000 malicious sites with 18 Mb. (See for example `http://blog.alexyakunin.com/2010/03/nice-bloom-filter-application.html`.)

We'll also study some software engineering techniques, including:

- developing an algorithm in a top-down fashion, and
- using methods.

7 Contact Details

Lecturer	Marc van Dongen
Room	WGB 1-74
Phone	+353 (0)21 4205903
Electronic mail	`dongen@cs.ucc.ie`
Snail mail	Computer Science Department University College Cork Western Road Cork

8 Acknowedgements

The picture in the presentation and some of the text in the literature section are from Amazon, O'Reilly, and Wikipedia.

References

[Bloch, 2008] Joshua Bloch. *Effective Java*. Addison-Wesley, 2008.

[Freeman and Freeman, 2005] Eric Freeman and Elisabeth Freeman. Head First *Design Patterns*. O'Reilly, 2005.

[McLaughlin *et al.*, 2007] Brett D. McLaughlin, Gary Pollice, and David West. Head First *Object-Oriented Analysis & Design*. O'Reilly, 2007.

[Naftalin and Wadler, 2009] Maurice Naftalin and Philip Wadler. *Java Generics*. O'Reilly, 2009.

[Sierra and Bates, 2004] Kathy Sierra and Bert Bates. Head First *Java*. O'Reilly, 2004.