CS 3230 Spring 2014 Syllabus
Internet Multimedia Services and Applications Using Java

Instructor Info
Jeff Compas
email: jeffozozo@gmail.com

Prerequisites

CS 2420 (which has prerequisites of CS 1400 [Java] and CS 1410 [C++])

You should have a good working knowledge of programming methods and feel comfortable writing programs to solve problems.

Grading

Weighting: Scale:

Programs: 50%  93-100% A  73-76% C
Mid-Term Exam 1: 15%  90-92% A-  70-72% C-
Mid-Term Exam 2: 15%  87-89% B+  67-69% D+
Final Exam: 20%  83-86% B  63-66% D
80-82% B-  60-62% D-
77-79% C+  <= 59% F

Important Dates

Midterm #1: Testing Center Feb 13-15; Volume I, Chapters 1-6, 11
Midterm #2: Testing Center April 3-5, Volume I, Chapters 7-9, 13-14, Vol II pp 370-442
Final: April 24 (To be confirmed)
Students are responsible for knowing the exam dates and the testing center hours and policies

Text

The text for the course is in two volumes. We’ll be doing most of our work in Volume I, but you’ll want both because they are great references for future courses:

Core Java, Volume 1: Fundamentals
Eighth Edition
Cay Horstmann and Gary Cornell
Publisher: Prentice Hall
Pub. Date: Sep 21, 2007
Copyright: 2008

Core Java: Volume II, Advanced Features
Eighth Edition
Cay S. Horstmann and Gary Cornell
ISBN: 978-0-13-235479-0
Publisher: Prentice Hall
Pub. Date: April 18, 2008
Copyright: 2008
Programming Assignments

The best way to learn how to program is to write programs. There will be approximately one programming assignment per week though some assignments will be larger and students will be given two weeks to complete them. There will be a published assignment schedule that will outline the due dates of the assignments as well as a detailed specification for each assignment.

Programming assignments are intended to help you learn the material. If you do the assignment yourself, solve the problems and understand the solutions, you will do great on the tests. If you don’t do your own work, copy your friend or get solutions off the internet, you will be short-changing yourself at the least, and you may be guilty of plagiarism at the worst. For this class (and for life in general) the solution to the problem is less important than knowing how to arrive at the solution.

Programs are submitted via WSU Online and are due at 11:59 p.m. on the due date. The specifications for the assignments will contain instructions as to how you should submit the assignment. These instructions will need to be followed completely in order to receive full credit for the assignment.

Late Assignments will be accepted only under extreme, unavoidable, extenuating circumstances. After the due date of the assignment, the instructor reserves the right to refuse acceptance of any work.

Mid-Term Tests

There will be two mid-term tests designed to gauge your knowledge of the programming principles we have learned to that point. This means that the tests will be mostly programming problems. The best way to prepare for these tests is to do the programming assignments and review the example programs.

The tests will be made available in the testing center for several days. We will have a test prep-review the class period before the test, and we will review the test in the class period following.

Comprehensive Final

The final will contain material from the entire semester. It will also be a programming-style test.

Class Sessions

Class sessions will be formatted with a lecture for the first hour, and a “lab style” or hands-on activity with some demonstration for the second hour. We may periodically have an open class where the instructor will focus on demonstrating solutions to specific problems and answer your questions.

Attendance is strongly encouraged. Lectures will not be recorded and information will be introduced in class that will appear on tests. You may not be able to get that information anywhere else. A common “rule of thumb” for allocating study time is 2 to 3 hours of study per credit hour (i.e., 8 - 12 hours per week for CS 3230). Some people find that this is not enough for a programming class. By now, you should have a rough idea of how long it takes YOU (it is different for everyone) to complete programming assignments and you should plan accordingly.

Any student requiring accommodations or services due to a disability must contact Services for Students with Disabilities (SSD) in room 181 of the Student Services Center. SSD can also arrange to provide course materials (including this syllabus) in alternative formats if necessary.
This syllabus is subject to change at any time. Alterations made in class or on the course web site supersede this document. Please see the website for current information.

**Honesty & Fair Use:**

Each student is expected to maintain high standards of honesty and ethical behavior. Each assignment must represent the student’s own, best effort. You are encouraged to study together and to work together on the assignments. This means that it is okay to discuss algorithms, syntax, and Java in general with others. You may also get ideas and code fragments from books or from the Internet. However, you may not copy whole methods, classes, files, or programs from someone else, from a book or from the Internet; nor may you exchange or share code in any electronic format - including code from previous semesters. If this is or any other dishonesty is demonstrated, you will fail the course and may face University disciplinary action. If you have any questions about what is acceptable and what is not acceptable, you may ask the instructor.

Please do not ask students (past or present) to see copies of their assignments. Please do not give other students (current or future) copies of your assignments.

Plagiarism will result both in a failing grade and university sanctions. The university expels students if plagiarism is proven the second time.

**Course Objective**

You will learn the Java programming language this semester, which is frequently used for Internet and multimedia programming. Java has, in a few short years, become one of the most popular and most successful programming languages of all time. Its success and growth has paralleled and has been driven by the success and the growth of the Internet. It is an extremely versatile language that runs on countless platforms and can be found in virtually every computing environment. Java really is an appropriate topic for an Internet class, a topic that will easily fill an entire semester.

The primary goal of this class is to help you learn how to solve problems and how to code solutions in Java. This entails several sub-goals. I want to help you understand: (a) how the object-model works; (b) how to solve problems and create models before you attempt to code the solution in any language; (c) how to debug a program; and (d) to practice building appropriate mental models of computer/language systems that will help you understand the tasks on which computer scientists work.

The text for the course is *Core Java*, which is an appropriate description of what you will learn. At the conclusion of the course, you will understand the “core” of the Java language. You will learn in this class those “core” features of the Java language that are used regardless of what area of specialized Java programming you may wish to pursue. This includes programming within the object model as well as advanced features such as graphics, GUIs, networking, and multithreading. Following are the objectives that we will undertake together as a class this semester.

At the conclusion of the course you will:

1. Understand the fundamental or core concepts of the Java language; specifically, those parts of the language which are generally used regardless of the problem domain.

2. Understand the Object-Oriented model and its relationship to and implementation in the Java programming language; specifically, you will understand the components and concepts:
   - classes, objects, and instantiation
   - relationships: inheritance, association, aggregation, composition, and dependency
   - attributes and methods, and their relationship to encapsulation and abstraction
   - abstract classes
   - polymorphism
3. Understand the physical organization of the Java language system and the relationship of this organization with the Java Development Kit (JDK), used to create Java programs:
   • .java and .class files
   • the Java Virtual Machine (JVM) and how it relates to applications and to applets
   • how multi-class programs are assembled into programs (including executable JAR files)
   • how to use an Integrated Development Environment (IDE) to edit and debug java programs.

4. Understand and use interfaces and inner classes

5. Understand and use Java’s event delegation model (i.e., be able to create programs based on the event-driven programming model)

6. Be able to write graphical programs

7. Be able to write Graphical User Interface (GUI) programs
   • AWT
   • Swing
   • Applications
   • Applets
   • Java’s GUI event interfaces

8. Understand and be able to use event handling, including Java’s event classes

9. Understand and be able to use exception handling

10. Understand and be able to use Java’s input and output philosophy and classes

11. Understand and be able to use Java’s networking classes

12. Understand the concept of multi-tasking in general and Java’s multithreading features specifically, and be able to write complex, multithreaded programs.