C Sc 335 Course Overview

Object-Oriented Programming and Design

Rick Mercer

Main Topics in C Sc 335

1. Java
2. Object-Oriented Programming
3. Object-Oriented Design
4. Technology
5. Object-Oriented Principles
6. Software Development
7. Team Project

1. Java

- Classes and Interfaces
- Exceptions, Streams, Persistence
- Graphical Components
- Event-driven programming
  - Make something happen on a click, mouse motion, window close, checkbox....
- Socket Networking
- Concurrency with Java Threads

2. Object Oriented Programming

- Encapsulation / Modularity
  - keeping data and behavior together
- Inheritance
  - Capture common data and behavior in a class, then let other classes extend it
- Polymorphism
  - via interfaces and inheritance

3. Object-Oriented Design

- Design Guidelines such as
  - Assign a responsibility to the object that has the necessary information, high cohesion, low coupling
- Object-Oriented Design Patterns such as
  - Iterator
  - Strategy
  - Adaptor
  - Decorator
  - Composite
  - Mediator
  - Command
  - Observer
  - Factory

3. OO Design *continued*

- Responsibility Driven Design (RDD)
- Unified Modeling Language (UML)
- Test Driven Design (TDD)
- Refactoring
  - Improving the design of existing code without changing its meaning—make it more readable and maintainable, a few examples:
    - Rename, Extract method , Exit method as soon as possible, Change method signature

4. Technology

- Professional IDE: Eclipse
- Concurrent Versioning System (CVS)
- Use existing frameworks
  - Java's Collection Framework
  - javax.swing, javax.awt
  - java.io
  - java.net

5. Object-Oriented Principles

- The Single Responsibility Principle
- The Open–Closed Principle
- The Dependency Inversion Principle
- The Liskov Substitution Principle
- Favor composition over inheritance
- Encapsulate what varies
- Program to interfaces, not implementations

6. Software Development

- We'll use a mash up of Agile techniques
  - Test Driven Development (TDD)
  - Short iterations
  - Coding standard and collective code ownership
  - Pair programming
  - Frequent build updates
  - Sustainable pace
  - Estimating and planning
  - Retrospectives

7. Team Project

- Great projects have each person developing 50-65 hours each over the final six weeks
  - You can still get very high marks in less time
- Teams of four
- Some rough estimates
  - 15-25 classes
  - A few interfaces
  - 4,000 to 6,000 lines of code (LoC)

No Text Book to buy

- There is no one good textbook for this class
- There will be readings of online content, some views of videos and
- Selected readings are from Safari Books Online
  - You need to be at a UofA computer or establish a Virtual Private Network (VPN) connection on your machine, UofA has a Cisco solution for free
  - You have access to thousands of technical books

Goals

- Understand and use the fundamentals of object-oriented programming: encapsulation, polymorphism, and inheritance
- Understand the relationships between objects, classes, and interfaces
- Build complex systems with at least one that has 15 or more classes that you develop with a team

Goals (continued)

- Learn to work on teams
- Use good practices of programming to develop good object-oriented software
- Become comfortable with event-driven programming and graphical user interfaces
- Use the tools of object-oriented software development
  - Design Patterns, the Unified Modeling Language (UML), unit testing (JUnit), a professional IDE (Eclipse), frameworks, Agile techniques

Goals *continued*

- Value TDD and see how it helps design and provide confidence in correctness
- Write clean code
- Be able to make intelligent design decisions
- Build a project that is better than the sum of the parts (team project is greater than what 1person can do in the same number of person hours)
- Have some fun