Course Goal

To make you a better programmer

- Deconstruct relevant programming problems
- Solve them in an object-oriented style, focusing on
  - Reusability
  - Maintainability (clarity)
  - Design
  - (Performance is secondary)
Approach

• Will use the Java programming language exclusively
  – But the ideas apply to other languages too

• Sequential object-oriented (OO) programming
  – Basic principles and tools, and
  – Design patterns to improve reusability and reliability

• Concurrent OO programming
  – Emphasis: shared-memory multi-threading (Java Thread class)
  – Also distributed message passing (e.g., RMI)
Topics

• Java review, new features in Java 5.0/6.0
• Programming techniques and tools
  – Specifications and testing
• Design patterns
  – And refactoring
• Concurrency
  – Concurrent programming in Java
  – Design patterns
• Distributed programming
• Special topics
  – Possibilities include refactoring, security, event-based programming, reflection, memory management, XML, ...?
Interaction
  – This is your course: what do you want to learn?

Discussion
  – Not just professor/TA to student, but student to student, with regard to ideas, techniques, and solutions

Learn by doing
  – If you don’t put effort into the programming projects, you will learn very little
Additional reference materials

- Lots of resources
  - many on-line and free
- Will be pointed out during semester
- Find your own
  - If you copy code from any resource, acknowledge it
• Five total projects
  – Will sometimes extend project templates we provide, but generally will write from scratch

• Focus on networked applications
  – Encourages modular, abstract design
  – Admits natural use of concurrency and distribution
  – Relevant in our connected society
Project Submission

- Projects due at Midnight on due date
  - By Unix time of day
  - You must submit a good-faith effort
    - You can be **failed** for the course if you do not
  - Late submission up to 9am the next morning
    - Score is multiplied by 0.7 (it is generally not in your best interest to submit late)
  - Only last submission will be graded!
Project grading and class accounts

- We will use the SubmitServer system for project submission
- Linux lab accounts available
  - Can use your own campus accounts for course work
- Course grades and accounts will be managed using grades.cs.umd.edu
  - All linked from course web page resources
Software

• Will be using:
  – Java 1.5+
  – Eclipse 3.1+ IDE (optional)
  – JUnit
Open Source Contribution Project

- One special project: (grade included in final exam)
  - make a contribution to a large open source software project
  - large meaning 40,000+ lines of code

- Everyone has to pick something different
  - could be different contributions to same project
A simple contribution: bug fix

• Find a large Java App
  – Download it, build it, run it.
• Run FindBugs over it
• Understand, document code defect
• Write test cases
• Fix defect
• Submit your work to the project
More aggressive contributions

• Find a problem report in a bug database
  – Figure out what the defect is
  – Document and fix, as before

• Add a feature to an open source project
  – e.g., “Mozilla’s new HTML editor has support for ftp, but not sftp -- add sftp support” -- Jeff Hollingsworth
  – Plenty of stuff for FindBugs
  – Ask around (faculty, others)
Grading of open source project

- Project intended to get your feet wet with real software
- Grade not based on size of contribution, but on how seriously you take it
- Just blasting email to the developers list (“Hey, line 45 of FooBar.java contains a bug”) won’t count for much
- For overachievers, prizes for anyone who does a significant contribution
# Grading

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>% each</th>
<th>% total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projects</td>
<td>5</td>
<td>10</td>
<td>50</td>
</tr>
<tr>
<td>Mid-terms</td>
<td>1</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Final</td>
<td>1</td>
<td>30</td>
<td>30</td>
</tr>
</tbody>
</table>
Discussion and Questions

• Web forum
  – Web-based discussion pages
  – Can post to from off-campus
  – Linked from course web page

• Post questions, pointers to resources, test cases.
  – Will be monitored by professor and TA
  – Don’t cross the line! Help on ideas of projects; never post code or pseudocode that gives away the exact approach.
Office Hours

• Professor Adam Porter, aporter@cs.umd.edu
  – 4125 AVW
• TA: Ferhan Ture, fture@cs.umd.edu
  – Office hours in 1112 AVW
• All hours posted on web page
  – Or set up an appointment
Excused Absences

• Religious holidays or other personal conflicts
  – Let us know *as soon as you get the project*

• Medical and other emergencies
  – Must provide documentation stating what dates/times you were incapacitated
  – Self reporting is *not* sufficient
http://www.cs.umd.edu/class/fall2008/cmsc433
Contains:
• Announcements
• Lecture notes
• Project assignments
• Resources
• And more!