Introduction
To help you become a better programmer

• Introduce advanced programming technologies
• Deconstruct relevant programming problems
• Solve them using the advanced technologies
Topics

• Concurrency
• Distributed programming
• Possible special topics
  – Event-based programming
  – Android
Concurrency

• We’ll look at systems with
  – Multiple threads of control
  – Implementing multiple tasks
  – On a single machine

• Implementation technology
  – java.lang.Thread
  – java.util.concurrent.*
  – Other supporting classes
Distributed Programming

• We’ll look at systems with
  – Multiple threads of control
  – Implementing multiple tasks
  – On multiple machines

• Technology
  – Java RMI
  – Hadoop / Google MapReduce
Other Special Topics

• Let’s hear from you
Style

• 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
Textbook

- Download & investigate source code examples
  - www.javaconcurrencyinpractice.com
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
Projects

• Five projects
• You will write projects mostly from scratch
• I encourage you to write and share your own test cases
Project Submission

- Projects due at Midnight (23:59:59) 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 0.8 (it is generally not in your best interest to submit late)
  - Only last submission will be graded!
Project Grading and Class Accounts

- Will try an alternative to the SubmitServer system for project submission
  - More on this in a later lecture
- Linux lab accounts will be 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
• The TA and I will mostly be using:
  – Java 1.6
  – Eclipse 3.7 (Indigo)
  – Junit 4+
• If you can, please bring your laptop to class so we can examine source code examples together
Exams

- Two midterms: March 6\textsuperscript{th} & April 19\textsuperscript{th}
- Final: Monday, May 14\textsuperscript{th}, 8:00 am-10:00 am
### Tentative Grading Scheme

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>% each</th>
<th>% total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projects</td>
<td>5</td>
<td>8</td>
<td>40</td>
</tr>
<tr>
<td>Mid-terms</td>
<td>2</td>
<td>15</td>
<td>30</td>
</tr>
<tr>
<td>Final</td>
<td>1</td>
<td>30</td>
<td>30</td>
</tr>
</tbody>
</table>
Discussion and Questions

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

• Post questions, pointers to resources, test cases.
  – Will be monitored by professor and TA
  – Use good judgment. Discussion is fine, but never post code or pseudo-code that gives away exact solution approaches
Office Hours

• Professor Adam Porter, aporter@cs.umd.edu
  – 4125 AVW

• TA: Emre Sefer, esefer@cs.umd.edu
  – Office hours in 1112 AVW

• Office hours posted on web page
  – http://www.cs.umd.edu/class/spring2012/cmsc433
Excused Absences

• Religious holidays or other personal conflicts
  – Let us know *as soon as you can*

• 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/spring2012/cmsc433
Contains:
• Announcements
• Lecture notes
• Project assignments
• Resources
• And more!
A Single-Threaded Logging Server

- Logging server
  - Accepts records from client
  - Writes record to client-specific file
Let’s Look at the Code

• **Organization**
  – **Utils**
    • DataRecord.java
    • LoggingServerCore.java
    • MsgHandler.java
  – **Client**
    • ClientSimulator.java
  – **Server**
    • SingleThreadedServer.java
Ungraded Assignment

• Download the code
  – Will be posted publicly by Friday
• Read and understand how it works
• Run this code and observe its performance
• Assuming the ClientSimulator is fixed
  – What factors account for the program’s running time?
  – What possibilities might exist to speed things up?