CMSC433 - Programming Language Technologies and Paradigms

Introduction
Course Goal

• 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
Event-Based Programming*

- We’ll look at systems with
  - A single thread of control
  - Implementing multiple tasks
  - On a single machine
- Implementation technology
  - java.nio.*
  - Android
• Let’s hear from you
Course 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
Optional 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 (plus one ungraded one)
• 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 use the submit server system for project submission
- 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
Software

• The TA and I will mostly be using:
  – Java 1.7
  – Eclipse 4.2 (Juno)
  – Junit 4+

• If you can, please bring your laptop to class so we can examine source code examples together
Exams

- One midterm: March 13\textsuperscript{rd}
- Final: Monday, May 15\textsuperscript{th}, 8:00 am-10:00 am
<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>25</td>
<td>25</td>
</tr>
<tr>
<td>Final</td>
<td>1</td>
<td>25</td>
<td>25</td>
</tr>
</tbody>
</table>
• Will use Piazza for our class forums
• 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: Khoa Ha, khoaha@terpmail.umd.edu
- Office hours in 4103 AVW
- Office hours posted on web page
Excused Absences

• See syllabus for more details
• 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 may **not** be sufficient
I will use **various methods** to provide you with course information.
- Class webpage
- Piazza forum
- Email

It is **your responsibility** to stay up to date.
A Single-Threaded Logging Server

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

• Get the files from the class website’s Lectures page.
• Organization
  – **Utils**
    • DataRecord.java
    • LoggingServerCore.java
    • MsgHandler.java
  – **Client**
    • ClientSimulator.java
  – **Server**
    • SingleThreadedServer.java
Take Home Assignment

• Download the code from the Lectures page of the class website
• Read and understand how it works
• Run the 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?
• Be prepared to talk about your observations in the next class session