CMSC433, Fall 2007
Programming Language Technologies and Paradigms

Introduction

Bill Pugh
August 30th, 2007
Important points

• You should have accounts on
  – and have received email about a SurveyMonkey survey

• If you just added the course, I might not have taken care of this
  – please contact me promptly

• Please complete the survey
Course Goal

To make you a better programmer

• Deconstruct relevant programming problems
• Principles and concepts
  – learning specific APIs a side effect, not a primary goal
• Solve them in an object-oriented style, focusing on
  – Reusability
  – Maintainability (clarity)
  – Design
  – (Performance is secondary)
A little about me

• Worked on revising the specification of threads in Java
  – complete rewrite of spec, now part of Java 5 spec

• Lead on FindBugs, open source static analysis tool for finding errors in Java programs
  – 250,000+ lines of code
  – downloaded 400,000+ times, used by many major tech and financial firms

• Spent past two summers working with Google and other bay area tech companies
  – understanding how to incorporate static analysis into software development
Topics

• Review of Java and OO design patterns
  – Regular expressions and patterns
  – TCP Sockets
  – Effective Java
  – Decorator pattern
  – Factory pattern
  – Builder pattern

• Programming Techniques and Tools
  – Testing
  – Mock Objects
  – Assertions/Invariants
  – Source code version control systems
  – Build systems, project automation, continuous integration
  – Static/dynamic tools for code quality/testing
Topics, continued

• Refactoring (1 week)
  – Philosophy
  – Code smells
  – Refactoring patterns

• Security (1 week)
More topics

• Concurrency (3 weeks)
  – Intro to Concurrency
  – Threads
  – Java threads
  – Java memory model
  – Synchronization: locking (synchronized) and signaling (wait/notify)
  – Design forces (safety, liveness/deadlock, performance, reusability)
  – Java 5 concurrency
  – Threaded programming patterns

• Map reduce programming (2 week)
• Distributed Programming (2 week)
• Too many
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

- Six total projects
  - Will extend project templates we provide

- Projects due at 6pm on due date
  - via the submit server
  - You must submit a good-faith effort
    - You can be **failed** for the course if you do not
  - Late submission up to sunrise the next morning
    - Score is multiplied by 0.8
Project grading and class accounts

• We will use the SubmitServer system for project submission and testing
  – Linux lab account for cvs access only
  – Use your own computers or campus accounts for course work
    • see me if this is a problem
• Course grades will be done using grades.cs.umd.edu
  – also used to distribute linuxlab accounts
• All linked from course web page resources
Software

• Will be using:
  – Java 5.0+
  – Eclipse 3.3 IDE
    • Including Clover code coverage plug-in
  – JUnit
  – FindBugs
  – Ant
Marmoset Research Project

• You will be asked to participate in the Marmoset research project
  – gives us permission to study your work on programming projects for research purposes
    • research data is anonymized
• Other than signing a consent form, you don’t have to do anything different to participate
Open Source Contribution Project

• One special project:
  – 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
  – team contribution efforts possible
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
  – 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
Exams

• One midterm

• Final
## Grading

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>% each</th>
<th>% total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homeworks</td>
<td>?</td>
<td>varies</td>
<td>14</td>
</tr>
<tr>
<td>Projects</td>
<td>6</td>
<td>7</td>
<td>42</td>
</tr>
<tr>
<td>Mid-terms</td>
<td>1</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>Final</td>
<td>1</td>
<td>27</td>
<td>27</td>
</tr>
</tbody>
</table>
First homework / Project 0: Binary search

- Implement binary search in an array:
  ```java
  public static int binarySearch(int a[], int x)
  // return i s.t. a[i] == x if such an element exists
  // return -1 otherwise
  ```

- Due next Wednesday, Sept 5th
Why binary search?

• I've heard several people say that when they've asked experienced developers to do this, most have bugs in their implementation

• Heard one person say they use this in job interviews, with similar results
Doing the homework

• Should not take you more than an hour
  – Don't wait until next Wednesday, 5pm to get started

• Try doing it first without any test cases

• submit your project before testing
  – just to satisfy my curiosity; you'll have plenty of opportunities to resubmit it
Discussion and Questions

• Based on early survey results, looks like forums.cs.umd.edu
  – Web-based discussion pages
  – Can post to from off-campus
  – Linked from course web page

• Post questions to projects, pointers to resources, etc.
  – 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 Bill Pugh, pugh at cs.umd.edu
  – 4131 AVW

• TAs: David An
  – Office hours in 1112 AVW

• All hours will be 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/fall2007/cmsc433
Contains:
• Announcements
• Lecture notes
• Project assignments
• Resources
• And more!
Readings for next week

• JUnit
• How web servers work
• Regular expressions
  – meaning of regular expressions (e.g., [0-9]+xx[0-9]+)
  – package java.util.regex.*
  – capturing groups
    • e.g., Pattern.compile("([a-z]+)([0-9]+)")
• TCP Sockets and server sockets
Getting set up

• Download and install Eclipse 3.3
  – either Java or Classic edition
• Add plugins
• Connect to grades.cs.umd.edu, get linuxlab account
• Setup CVS repository on linuxlab within Eclipse
• checkout project p0
  – later today