COMP 215: INTRO TO PROGRAM DESIGN

Prof. Chris Jermaine
cmj4@cs.rice.edu
"Chris"
"Prof Chris"

Prof. Scott Rixner
rixner@rice.edu
???

This Class

- 65% of content: modern programming and program design
  - — The Java programming language will be used (no prior Java assumed)
  - — But this is not a “Java class” per se
  - — Goal is proficiency in modern OO program development
  - — Will fixate on Java a lot (can’t get around that), but will try to stay general, as well

This Class

- 65% of content: modern programming and program design
  - — The Java programming language will be used (no prior Java assumed)
  - — But this is not a “Java class” per se
  - — Goal is proficiency in modern OO program development
  - — Will fixate on Java a lot (can’t get around that), but will try to stay general, as well
- 35% of content: modern algorithms
  - — Modern computer science IS NOT programming
  - — Modern computer science IS algorithmic thinking
  - — We only program because we have to... it’s a necessary evil :-(
  - — In keeping with this view, will try to avoid studying programming in a vacuum
  - — Means we’ll have to cover a lot of algorithms to motivate our programming

What Is "Modern OO Program Development"?

- Despite what you might have heard, OO is no silver bullet
- Simply a paradigm that forces **abstraction** when applied correctly
  - — Unfortunately, very few people apply it correctly!

What Is "Modern OO Program Development"?

- Despite what you might have heard, OO is no silver bullet
- Simply a paradigm that forces **abstraction** when applied correctly
  - — Unfortunately, very few people apply it correctly!
- What is **abstraction**?
  - — Abstraction means taking complicated machinery and wrapping it up in such a way that people can use the machinery w/o understanding the details
  - — Abstraction is all about defining and enforcing **interfaces**

What Is "Modern OO Program Development"?

- Despite what you might have heard, OO is no silver bullet
- Simply a paradigm that forces **abstraction** when applied correctly
  - — Unfortunately, very few people apply it correctly!
- What is **abstraction**?
  - — Abstraction means taking complicated machinery and wrapping it up in such a way that people can use the machinery w/o understanding the details
  - — Abstraction is all about defining and enforcing **interfaces**
- Why is **abstraction** good?
  - — In theory, effort to build/maintain system is linear in system functionality, SLOC
  - — Why? With proper abstraction, everything is local
  - — W/o abstraction, effort is quadratic: $1 + 2 + 3 + ... + n$ is prop. to n^2

The Class Project

- 80% of class grade comes from semester-long project
  - — (Approximately) 10 + 1 Java programming assignments
  - — All except for first fit together to build a document content and retrieval system
  - — System uses machine learning to “understand” the documents, answer queries

The Class Project

- 80% of class grade comes from semester-long project
  - (Approximately) 10 + 1 Java programming assignments
  - All except for first fit together to build a document content and retrieval system
  - System uses machine learning to “understand” the documents, answer queries
- **Abstraction** key to project design
  - Algorithm to “understand” the docs not too complicated (called “LDA”)
  - But requires sophisticated infrastructure to be able to code simply and efficiently
  - If we mix infrastructure and LDA, will have 1000’s of lines of nasty code
  - If we abstract out infrastructure, LDA itself is about 50 lines of code
  - Perhaps 40-50% of the project is building infrastructure to support easy LDA

Now, Onto the Class' Technical Specs

- Textbook
  - — Bruce Eckel's *Thinking In Java*
  - — Either 4th or 5th edition is OK
  - — 4th edition can be downloaded for free from googlecode
  - — Won't teach from the book, but it is a "required" reference

Technical Specs (cont'd)

- Labs
  - — Three lab times per week reserved in Symonds II: Wednesday 1:00-2:30; Thursday 2:30-4:00; Friday 4:00-5:30
  - — Primary purpose is help with assignments (typically due Wed or Fri)
  - — Attendance only required this week (1% of grade)
  - — Will cancel one of three. Turn in your preferences by the end of class...

Technical Specs (cont'd)

- Communication
  - — We have a web page at http://www.clear.rice.edu/comp215/index.shtml
  - — We have a Google group at <omitted to protect us from spam>
  - — You will be invited to join at the email address given on the lab preference sheet
  - — Please post all assignment/exam related questions to this group
  - — That way, everyone benefits from the question/answer
  - — Only personal communications should go to the instructors/TAs

Technical Specs (cont'd)

- Assignments
  - — Turned in electronically (instructions forthcoming)
  - — Bulk of class grade
  - — Probably 11 of them
  - — Will be asked to code to a spec, document and test the code
  - — Though not in that order!!!

Technical Specs (cont'd)

- Late assignments
  - — Assignments always due at 11:55P of due date
  - — Will be sympathetic to timely (7 day advance notice) requests for extensions
  - — Will **never** grant individual extensions within 7 days of due date
  - — One microsecond to 24 hours late: lose 20%
  - — More than 24 hours late: not accepted

Technical Specs (cont'd)

- Academic honesty (**important!**)
  - — All assignments are "semi-individual" efforts... that is, no teams allowed
  - — But collaboration is encouraged
  - — The one classmate-to-classmate interaction **not** allowed is transmission of code (can't transmit or receive any code, no matter the size, no matter the medium)
  - — No outside collaboration/2-way communication is allowed at all, period
  - — Google, other internet resources fine, but follow the "two-line rule"
  - — Get caught, go to the honor council (hate to do it, but want to keep it fair)

Technical Specs (cont'd)

- Exams
  - — 2 exams, one October 19th (tentative), one during final time
  - — 10% of grade each
  - — Open book, open note, computer/iPad/smartphone OK, but no internet

Technical Specs (cont'd)

- Regrades
  - — Feel free to discuss grading with us at your leisure
  - — But want to grovel for points?
  - — Request must be typed, printed out, given to Chris/Scott within 7 days of return of assignment/exam
  - — Should include **all** documentation necessary to evaluate request

Technical Specs (cont'd)

- Regrades
  - — Feel free to discuss grading with us at your leisure
  - — But want to grovel for points?
  - — Request must be typed, printed out, given to Chris/Scott within 7 days of return of assignment/exam
  - — Should include **all** documentation necessary to evaluate request

Questions?

Syllabus-related subjects I forgot?

On To Java!

- “Java”... what is it? What’s the big idea?
  - — OO programming language first released in 1995 (wow, pretty old already!)
  - — Unique in that it was designed to run on a “virtual machine” (**JVM**)
  - — To run a Java program, someone must have JVM installed on their machine
  - — You write your program, software called **compiler** translates into Java **bytecode**
  - — You give your bytecode to someone who wants to run your program
  - — JVM on their machine **interprets** that bytecode
  - — **Interprets** = uses underlying hardware to take actions spec’d by bytecode

On To Java! (cont'd)

- Why is the JVM such a big deal?
  - — Offers true portability
  - — Same bytecode runs on a Mac, PC, Unix server, smartphone, anything with a JVM
  - — Plus, ensures a set of key algs, data structures, and I/O capabilities are there to use
  - — All bundled in the "Java Class Library"... comes with the JVM
  - — Back in '95, a typical PL might have a target-specific compiler + some low-level libraries (such as libc for C/C++ I/O)
  - — Needed hash table? You wrote your own, asked your friends, or searched the web
  - — Using Java? Just use the hash table that comes with the JVM
  - — Hard to overstate significance of this change!

Compilation vs. Interpretation

- Java is a “compiled” language
  - — Since entire program is translated into low-level bytecode, all at once
  - — Then the bytecode is run
- Python is an “interpreted” language
  - — Python interpreter actually executes Python
- Why might we prefer a compiled language?

Compilation vs. Interpretation

- Java is a “compiled” language
  - — Since entire program is translated into low-level bytecode, all at once
  - — Then the bytecode is run
- Python is an “interpreted” language
  - — Python interpreter actually executes Python
- Why might we prefer a compiled language?
  - — Often far faster to execute (though use of JVM does hurt a bit in the case of Java)
  - — Can catch many errors at compile time

So Is Java A Good Language?

- The bad stuff:
  - — It’s going on 20 years old, and showing its age a bit
  - — Some stuff (such as “generics”) are arguably not done correctly
  - — It’s a big and complicated language (so our class is quite front-loaded!)

So Is Java A Good Language?

- The bad stuff:
  - — It's going on 20 years old, and showing its age a bit
  - — Some stuff (such as "generics") are arguably not done correctly
  - — It's a big and complicated language (so our class is quite front-loaded!)
- The good stuff:
  - — It's the most modern of the widely-used languages
  - — The JVM
  - — It's somewhat difficult to really screw up in Java (compared to C, for example)

So Is Java A Good Language?

- The bad stuff:
  - — It's going on 20 years old, and showing its age a bit
  - — Some stuff (such as "generics") are arguably not done correctly
  - — It's a big and complicated language (so our class is quite front-loaded!)
- The good stuff:
  - **— It's the most modern of the widely-used languages**
  - — The JVM
  - — It's somewhat difficult to really screw up in Java (compared to C, for example)
- Why are we using it?
  - — For me, comes down to pragmatics
  - — Meanwhile, we can sit and hope for a better alternative! C# anyone?

Writing Your First Java Program

- You'll have to do this to "pass" the first lab
  - More explicit instructions will be given at that time!
  - But most should be able to do this at home, with little difficulty

Writing Your First Java Program

- You'll have to do this to "pass" the first lab
  - — More explicitly instructions will be given at that time!
  - — But most should be able to do this at home, with little difficulty
- First, you'll need to download a "JDK"
  - — Contains compiler, JRE (JVM, object library), and a bunch of other stuff
  - — The one you want comes from Sun/Oracle (just Google "JDK")
  - — You'll want version 6

Writing Your First Java Program

- You'll have to do this to "pass" the first lab
  - More explicitly instructions will be given at that time!
  - But most should be able to do this at home, with little difficulty
- First, you'll need to download a "JDK"
  - Contains compiler, JRE (JVM, object library), and a bunch of other stuff
  - The one you want comes from Sun/Oracle (just Google "JDK")
  - You'll want version 6
- Then, you'll want an "IDE"
  - Allows you to manage the source files associated with a program
  - Typically has support for debugging, testing, editing source code
  - Standard open source IDE is called "Eclipse"
  - We'll use one targeted towards "Java 101" users called DrJava

Once You've Installed the JDK and DrJava

- Fire up DrJava
  - — Check "Full Java" under "Language Level"
- Then type in your first program:

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

- Press "compile" and then "run" and watch it go...

What Does This Program Do?

```
// A “class” is a bundle of data plus functionality (methods)
// A “program” consists of a set of classes
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

What Does This Program Do?

```
public class HelloWorld {

    // This is a “static” method called “main”
    // “static” means that it is shared by all objects of
    //   type HelloWorld
    // “public” means it can be called from outside the
    //   class
    // In every program, a class must have a public, static
    //   method called “main” that’s invoked at startup
    // “args” is the list of parameters passed via the
    //   command line when the program is run
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

What Does This Program Do?

```
public class HelloWorld {
          public static void main(String[] args) {

       // “System” is a special class in the SCL
       // It has a data item (aka, a “member”) called “out”
       //   that corresponds to your console
       // “out” has a method called “println” that accepts a
       //   string and outputs it
       System.out.println("Hello World");
   }
}
```

Questions?