Today ...

- Course overview
- Java overview
- Start on Java

Readings

- Elements of Style, ch: 1–4
- Core Java, ch 3: 35–61 (mostly review)

Homework

- HW 1 out (due next Thurs.)
Course overview

We’ll cover four main topics:

Java  ... but this isn’t just a Java class
  • assume you know C++
  • we’ll move pretty fast

OOP concepts  within the context of Java
  • inheritance
  • interfaces/abstract classes
  • generics
  • garbage collection
  • exception handling

GUIs  the “event-driven” part
  • various widgets (buttons, text fields, lists, menus, etc.)
  • layout managers
  • IDEs

OO Modeling
  • minimal UML
  • principles/techniques for designing OO programs
  • software design patterns
Course Logistics

Me:

- email: bowers@gonzaga.edu
- office: Herak 309C
- office hours: 2-3pm or by appointment

Course webpage:

- http://www.cs.gonzaga.edu/~bowers/courses/cpsc224

Email list:

- Google group
- I'll sign you up (see handout)

Textbook (required): Core Java, Volume 1, Prentice Hall

- other readings and handouts
Grades:

- 20% Assignments
- 10% Group project
- 20% Quizzes
- 30% 3 mid-semester exams
- 20% Comprehensive final

Standard letter grades: A (90-100), B (80-89), C (70-79), D (60-69), F (< 60)

Attendance

- 4 absences may result in a V letter grade (GU policy)
- Emergencies
  - notify me asap if you are going to miss a class (prior to class if possible)
  - may be able to arrange special schedule in certain cases
**Group project**

Grading

- about a 4-week long programming project
- a final project report & presentation
- self evaluation
- evaluation by your team members

**Quizzes**

Short (5-10 min), weekly or more frequent

Why quizzes?

1. examples of questions I tend to ask on exams
2. gives me feedback on how folks are doing
3. gives you feedback on how you are doing

**Assignments**

We’ll have eight or so individual assignments

Hard copy:

- due in class on the due date
- must include a cover sheet
must include a design doc

Submitting your programs:

• use the class submission site
• only need to submit your source code

Formatting and commenting part of your grade!

Lab Classes

We’ll meet once a week in the lab (Herak 223)

Tentatively scheduled for Tuesdays

Will start in the lab on Tuesday, Jan. 31st (not next Tues.)
Java Intro

The plan ...

- Quickly introduce main ideas and features of Java
- Focus on differences between Java and C++
- Gradually move into more advanced topics

Java history

- Started at Sun around 1991 (James Gosling)
- Initial goals: language to run on “devices”
  - limited memory and power
  - deal with multiple computer architectures
  - similar syntax to C++ (most popular language at the time)

The result: new language + Java Virtual Machine (JVM)

- source code typically compiled to architecture-specific (assembly) language
- instead, Java code compiled to an intermediate language (\textbf{“bytecode”})
- each architecture (OS) has its own JVM program
- a device’s JVM executes the bytecode
- Java: Write once, run everywhere
- C++: Write once, compile everywhere
Java (the JVM) has morphed today into a platform (or “framework”)

- many languages now compile to Java bytecode
  - Ruby, Python, Scala, Groovy, C, Clojure, ...
  - (some) can call Java libraries and vice versa

- Idea adopted by MS .NET
  - Common Language Infrastructure/Runtime (CLI, CLR)
  - Common Intermediate Language (basically bytecode)

- C# and Java are very similar languages

Some Java features

- syntax a lot like C++
  - object oriented
  - static type system (types checked at compile-time)
  - Java has a bit more streamlined syntax

- garbage collection

- no pointers! ... references instead

- large set of standard libraries (packages)

- slightly different class model (e.g., single inheritance)

- Java still most popular programming language
Q: Is this Java or C++ code?

```java
int sumOfFirstNInts(int n) {
    int result = 0;
    for(int i = 1; i <= n; i++)
        result = result + i;
    return result;
}
```

You already know a lot of Java syntax!