Slide 1

Administrivia

- *Please* do not reboot the machines in this room (HAS 340); people rely on their being available for remote access.

  Also be careful not to inadvertently shut them down when trying to log off.

  If a previous user has left the machine's screen locked, you can use use control-alt-backspace to restart the graphical subsystem.

- Linux accounts for new students should have been created, with passwords mailed to Trinity e-mail addresses. To change your password, `yppasswd` from the command line.

  If you're new to Linux and/or its command line: See "Useful links" page (here). for links to relevant information.

- Homework 1 design due next Thursday.

Slide 2

A Little More About Homework 1

- Did you start reading the project description? Questions we should talk about now (briefly)?

- You're not committing yourself to anything at this point, but try to be as detailed as you can — so I can try to spot potential trouble. Also good to think in terms of a basic design (not too ambitious) plus extras. Keep in mind that what you do has to fit into an existing framework. (That's actually one of the pedagogical goals.)

- What you will actually turn in is HTML documentation of your planned game's main class — put it in your `Local/HTML-Documentation` and send me mail saying "ready to be graded". (Complete instructions in homework writeup.)

Slide 3

“Object Orientation”?

- A “programming paradigm” — contrast with procedural programming, functional programming, etc.
- No accepted-by-all definition, but most definitions mention encapsulation:
  - Data and functionality grouped together into “objects”.
  - Some data/functionality is hidden.
- Origins in simulation/modeling, where the goal is to model complex systems consisting of many (real-world) objects.

Slide 4

What’s An Object?

- Object — set of data (attributes) and associated functions (methods, behaviors, operations) that can act on data.
- Objects interact by calling each other’s methods, or by sending each other messages.
- Often makes sense to have many similar objects — hence “classes”.

Slide 5

What’s a Class?

- Can be thought of as a blueprint for objects of a given type; individual objects are “instances” of the class.
- Defines attributes and methods each object will have (instance variables/methods), attributes and methods shared by all objects of a class (class variables/methods).
- Public interface — attributes and methods visible from outside the class.

Slide 6

Java and Object Orientation

- Java is not purely object-oriented — also includes “primitive types” for efficiency — but it’s much more strongly object-oriented than a hybrid language such as C++.
- Java programs consist of definitions of classes. (No free-standing functions like the ones in C.)
- Java variables (except primitives) are references to objects, classes define types.
- Classes, attributes, methods have varying “visibilities” (from public to private).

Slide 7

Program Structure

- In Java, everything (variables and code) is part of a class. Typically have only one class per source code file (exception is inner/nested classes — more about them later).
- Any class can have a `main` method that can be launched by the runtime system (more about that later).

Slide 8

Defining a Class

- Each class is like a blueprint for objects of a particular kind, and can include:
  - Variables — instance (one copy per object) or static (one copy shared by all objects).
  - Methods — similar to C functions, but can be static or non-static (“instance methods”). Instance methods are “invoked on an object’.
  - Classes (more later).
- Variables and methods can be `public` or `private`. Good practice to define as private, except for constants that need to be used outside the class.

Slide 9

Tools

- Java programs are text, so you can write them with a text editor and compile and run them from the command line. (In fact I often do.)
- However, many professional programmers use an IDE (Integrated Development Environment), so we will too. We will use Eclipse, which is a free open-source tool written in Java, so you should be able to install a copy on your home machine if you like. (Versions seem to be available for Windows, Linux, and Mac OS X.)

Slide 10

Example(s)

- Let's write a "hello world" program.
- We'll use Eclipse to
  - Define a project, a package, and a class with a `main` method.
  - Compile and run.
  - Generate HTML documentation.

Slide 11

Minute Essay

- Was there anything today that was particularly unclear?
- If you have programmed in Java before, what tool(s) did you use?