Lightning Java Review

(or, perhaps, Introduction)

Overview

- Some facts
- Methods
- Variables
- Inheritance
- Creating objects (instances)
- Exceptions

Some facts

- Almost all processing is java is accomplished by sending messages to objects
- Objects are instances of classes
- Class definitions include variables and methods, both are referred to as members
- There are two type of methods; ordinary methods and constructors
- Members may belong to either classes (by use of static) or instances

Methods

- In the body of an instance method, there is a hidden parameter called "this", it is a reference to the object which was sent the message that invoked the method
- The syntax of a method definition is:
<returntype> <name> ([<parameters>]) <body>,
unless the method is a constructor; then its name is the same as the class it is in and it has no type before that name.
- If a method returns nothing, its type is `void`, otherwise it must end with `return <expression>;`, where `<expression>` has a type compatible with the return type.

Variables

- there are 4 common kinds of variables
  - instance
  - class
  - method
  - parameter (!)
  - what are two other kinds?
- Danger! The most local variable with a particular name is used; i.e. you can shadow a variable by accidently giving another variable (including parameters) the same name.

Inheritance

- One of the two ways to reuse software
- Which method gets invoked?
  - example of class and superclass where a message sent to aClassObject is fielded by the superclass, which includes a line like this.doit() and both the class and the superclass have doit() methods defined
- Chains of super() sometimes do most of the work of the object

Creating objects

- Every object must be created by sending new to the appropriate class
- new Foo(), is really Foo.Foo(), i.e. the Foo() message is sent to the Foo class (classes are in fact objects).
- Special syntax for invoking other constructors from a constructor
  - must come first
  - use super(), for the superclass or this(), for the default constructor for this class

Exceptions

- null pointer exceptions
  - Almost always means you are sending a message to a null pointer
  - forgetting to initialize an object is the easiest (and most common way to do this)

- another debugging tool
  - you can catch and handle Exceptions
  - when all else fails, you can use a try-catch block to find a bug