CSCI 136
Data Structures & Advanced Programming

Morgan McGuire

Lecture 2: Java Crash Course

Feb 6, 2017

Administrative Details

- Lab sections will be announced this afternoon
- *Before* lab:
  - Sign in to a CS dept Mac
  - Read the Silver Dollar lab handout online
  - *Design* your Silver Dollar solution
  - Use Boggle design doc as an example of detail
- TA hours start on Wed

Review

- CS136
  - Scalability
  - Analysis
  - Elegance
- Java
  - Static types for variables
  - Common: int, double, boolean, String
  - Lots of semi-colons and curly braces
- Hello.java

Review: Hello.java

```
/*
 * This program prints out a message to the terminal.
 */
public class Hello {

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

Today

1. Sum.java
   - Write a program that adds two integers together
   - Two versions: command-line args and Scanner
2. Object-Oriented Programming (OOP)
   - Classes
   - Members
   - Methods
   - Subclasses

Sum1.java

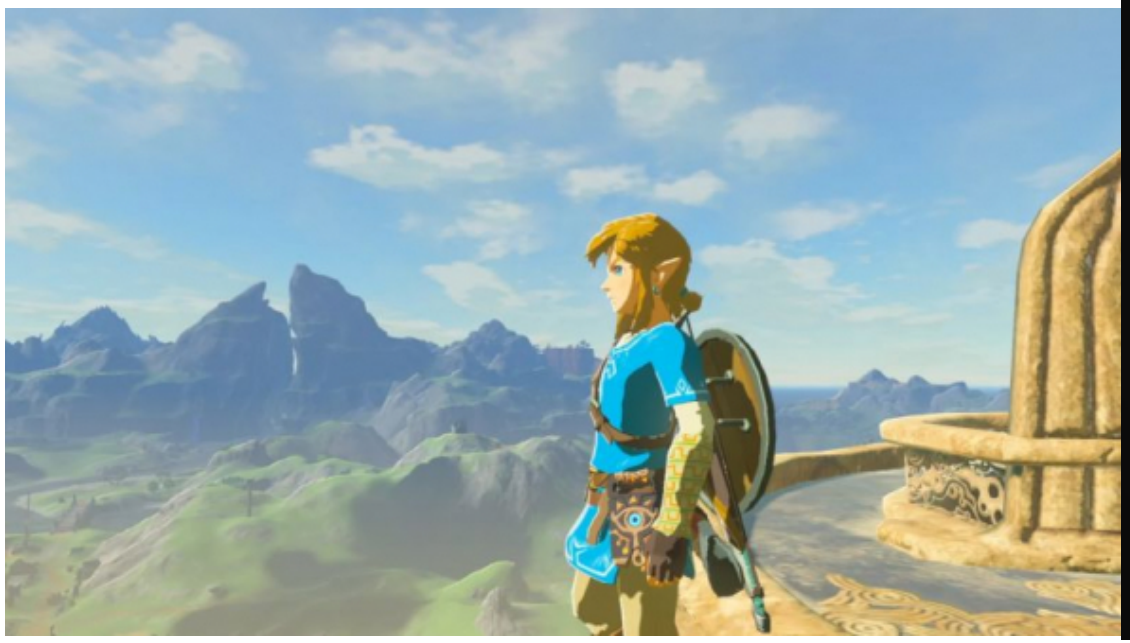
```
/*
 * A program to add together two integers from command line args.
 */
public class Sum1 {

    public static void main(String args[]) {
        int a = Integer.valueOf(args[0]);
        int b = Integer.valueOf(args[1]);
        System.out.println("Answer is " + (a + b));
    }
}
```

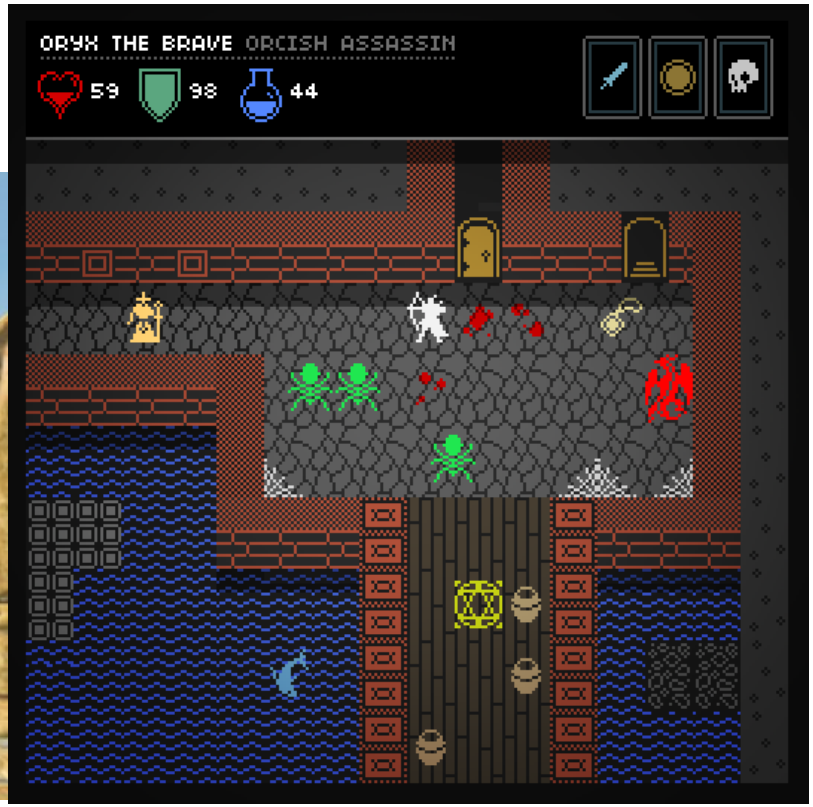
Sum2.java

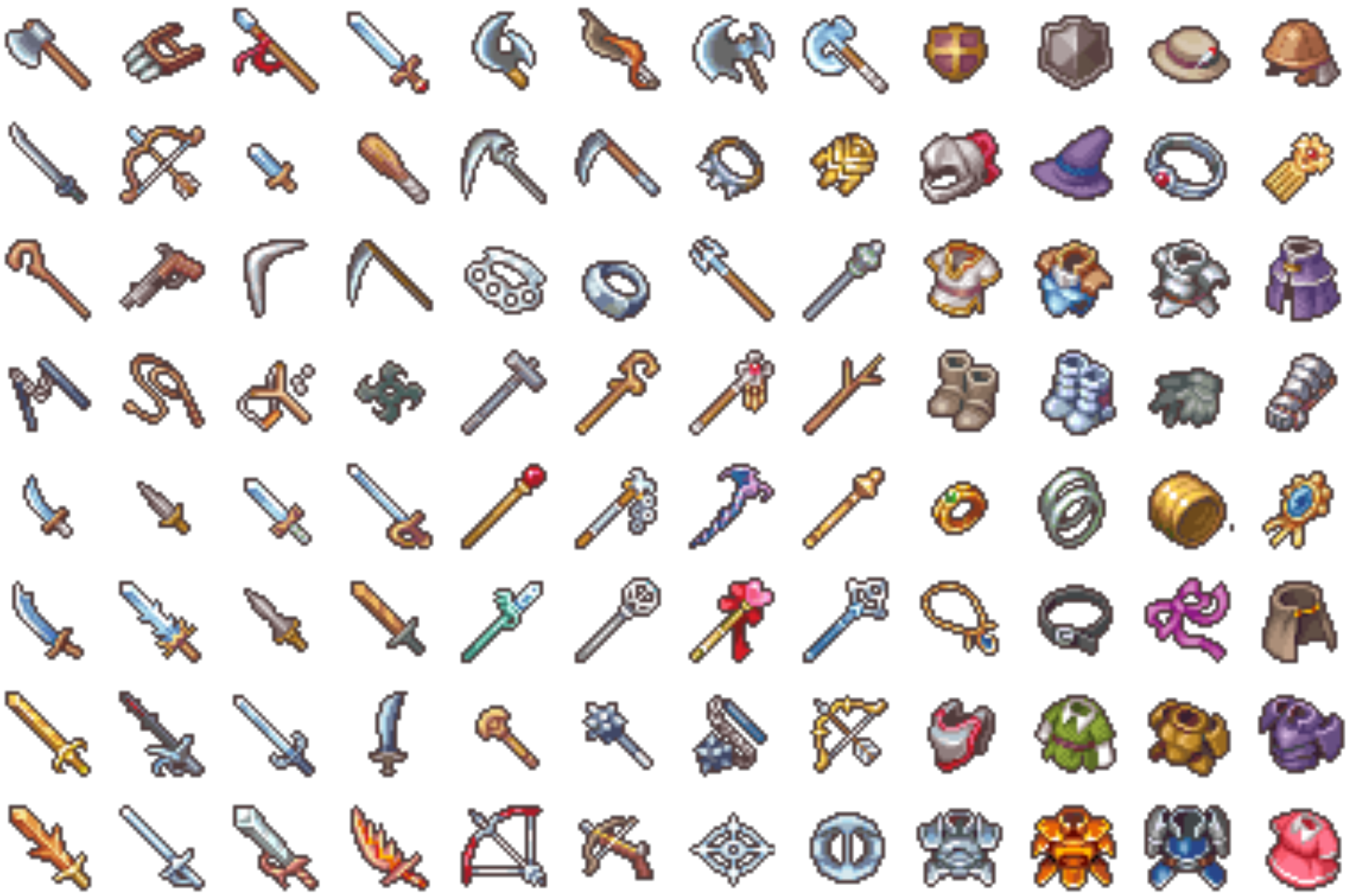
```
import java.util.Scanner;

/*
 * A program to add together two numbers read from the terminal.
 */
public class Sum2 {

    public static void main(String args[]) {

        Scanner in = new Scanner(System.in);

        System.out.print("Give me a number: ");
        int a = in.nextInt();
        System.out.print("Give me another number: ");
        int b = in.nextInt();

        System.out.println("Answer is " + (a + b));
    }
}
```

Program Design

- State (nouns) ➔ member variables
- Compution (verbs) ➔ methods

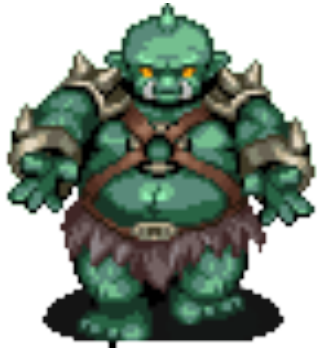






RPG Class Hierarchy Example

- Entity *(extends Object)*
  - position
- Item **extends** Entity
  - weight
  - size
- Club **extends** Item
  - damage
  - attack()
- Emerald **extends** Item
- Monster **extends** Entity …



Equality

- **a == b** if **a** and **b** are the same object
- There is only one instance of each number
- “hello ” + “world” != “hello world”
- String.equals
- Add .equals to your own classes:
  - Example: Emerald.equals()
  - Using instanceof and then casting
  - Calling super methods
  - Beware of floating-point roundoff

Summary

- Members for state
- Methods for computation
- Constructor: special initializing computation

- **Extend** classes to specialize
- **this**, **super**
- **instanceof** and Casting
- **==** versus .equals(), floating-point equality

Next Time

- **Protecting** our abstractions
- Separating **interface** from implementation
- Accessors
- **Arrays**
- Lab 1: Silver Dollar

Object-Oriented Programming

- Objects are building blocks of software

- Programs are collections of objects
  - Cooperate to complete tasks
  - Represent "state" of the program
  - Communicate by sending messages to each other

Object-Oriented Programming

- Objects can model:
  - Physical items - Dice, board, card, dictionary
  - Concepts - Date, time, words, relationships
  - Processing - Sort, search, simulate

- Objects contain:
  - **State** (instance variables, members, records)
    - Attributes, relationships to other objects, components
      - Letter value, grid of letters, number of words
  - **Computation** (methods, functions, callbacks)
    - addWord, lookupWord, removeWord
  - Accessor and mutator methods: state as computation
  - Constructors: computation to initialize state