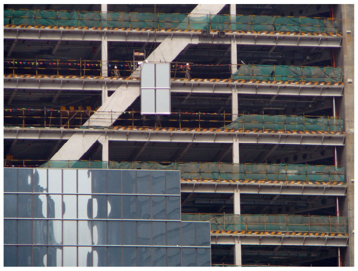
cs205: engineering software
university of virginia
fall 2006

Introducing Java



Don’t forget to email your registration survey – if you haven’t yet, please do it today

David Evans
www.cs.virginia.edu/cs205

Course Announcements

- Assistant Coaches:

  Drew Abott and Dan Marcus

  - Lab hours: after class today, Sunday 7-8:30pm
- Office Hours
  - Posted time conflicts with cs202
  - New time is 10:30-noon on Thursdays
  - My door is almost always open – feel free to stop by outside of office hours

Why so many programming languages?

Fundamental Differences

- All equivalently powerful!
  - *Universal languages*: all capable of simulating each other
- Fundamental differences
  - Expressiveness: how easy it is to describe a computation
  - “Truthiness”: likelihood that a program means what a programmer things it means
  - Safeness: impact of programmer mistakes
- There is usually a conflict between expressiveness and truthiness/safeness

Pragmatic Differences

- Performance of available compilers, interpreters
- Tools available
- Libraries
- Portability
- Availability/cost of programmers

What is Java?

A. Island in Indonesia known for coffee and volcanoes
B. A Programming Language (Java™)
C. A Portable Low-Level Language (JVML)
D. A Platform (JavaVM)
E. A (semi-)successful marketing strategy
   – JavaScript is not related to Java or Java™
F. All of the above

Java History

- 1991: “Stealth Project” formed at Sun
  - Computing for consumer electronics market
- James Gosling tasked with selecting a programming language for project
  - Started with C++, but found inadequate
    - In later classes, we’ll talk about why
  - Developed extensions and **subtractions** that led to new language “Oak”
- 1993: Web arrives
- 1995: Sun releases HotJava browser and Java PL, Netscape incorporated into browser

Buzzword Description

“A simple, object-oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, and dynamic language.”

[Sun95]

Later in the course, we will discuss how well it satisfies these “buzzwords”.

Non-Buzzword Description

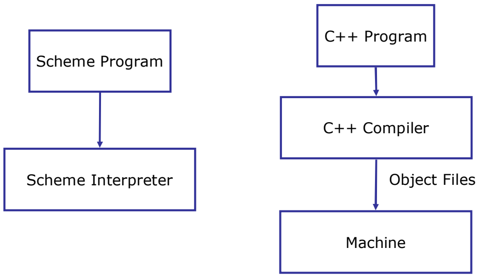
- Java sacrifices expressiveness for safety and “truthiness”
  - A Java program is ~5x larger than the corresponding Scheme or Python program
- Java sacrifices performance for safety and “truthiness”
  - A Java program is ~2x slower than the corresponding C program (but 5x faster than the corresponding Scheme/Python program)

Caveat: these numbers are “guesses” and gross simplifications. Real numbers depend on the program (and programmer!).

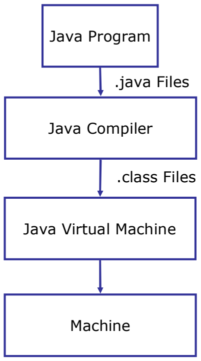
Java Programming Language

- Syntax
  - Similar to C++
  - Designed to be easy for C and C++ programmers to learn
- Semantics (what programs mean)
  - Similar to Scheme
  - Designed to make it easier to reason about properties of programs

Programming Systems

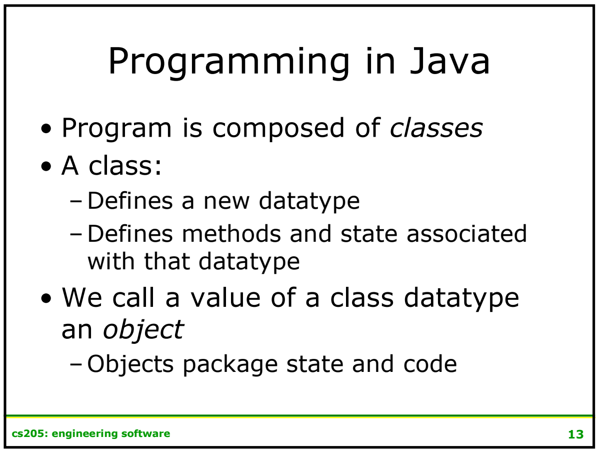


Java VM



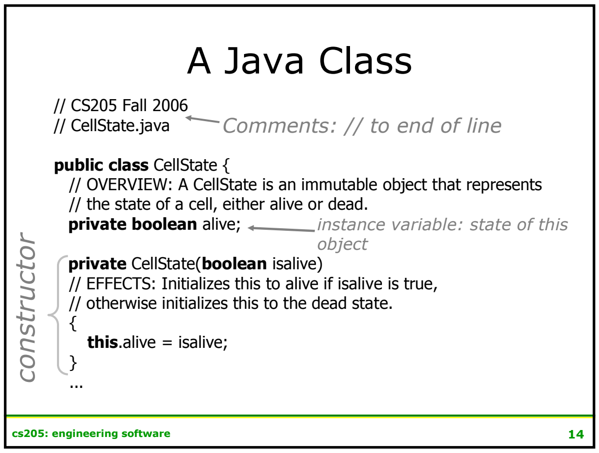
Why use a virtual machine?

- Portability
  - If you can implement a Java VM on your machine, then you can run all Java programs
- Security
  - A VM can limit what programs can do to the real machine
- Simplicity
  - VM instructions can be simpler than machine instructions



Programming in Java

- Program is composed of *classes*
- A class:
  - Defines a new datatype
  - Defines methods and state associated with that datatype
- We call a value of a class datatype an *object*
  - Objects package state and code



A Java Class

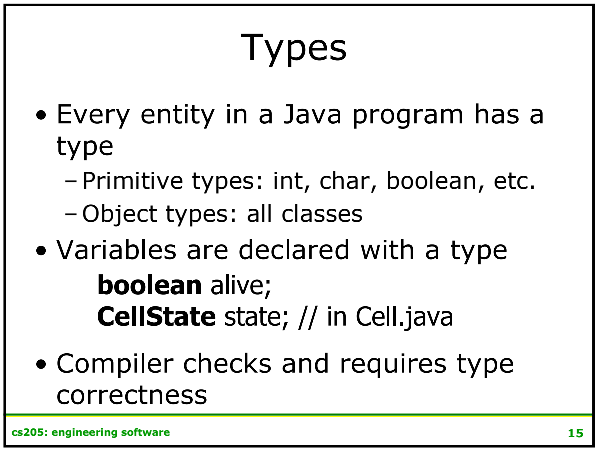
```
// CS205 Fall 2006
// CellState.java
```

Comments: // to end of line

```
public class CellState {
   // OVERVIEW: A CellState is an immutable object that represents
   // the state of a cell, either alive or dead.
   private boolean alive;
```

instance variable: state of this object

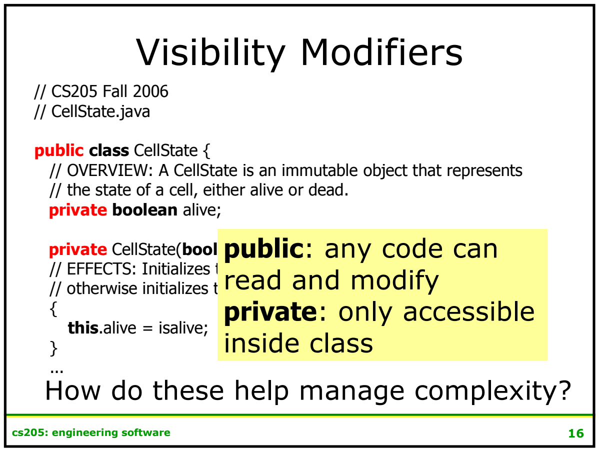
```
   private CellState(boolean isalive)
   // EFFECTS: Initializes this to alive if isalive is true,
   // otherwise initializes this to the dead state.
   {
      this.alive = isalive;
   }
   ...
```



Types

- Every entity in a Java program has a type
  - Primitive types: int, char, boolean, etc.
  - Object types: all classes
- Variables are declared with a type

  boolean alive;
  CellState state; // in Cell.java
- Compiler checks and requires type correctness



Visibility Modifiers

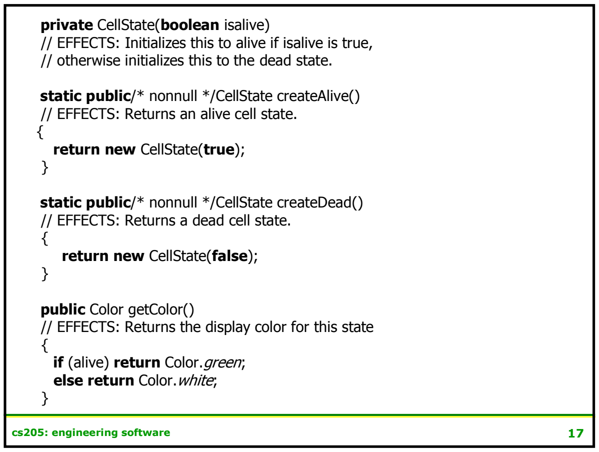
```
// CS205 Fall 2006
// CellState.java

public class CellState {
   // OVERVIEW: A CellState is an immutable object that represents
   // the state of a cell, either alive or dead.
   private boolean alive;

   private CellState(bool
   // EFFECTS: Initializes t
   // otherwise initializes t
   {
      this.alive = isalive;
   }
   ...
```

public: any code can read and modify
private: only accessible inside class

How do these help manage complexity?

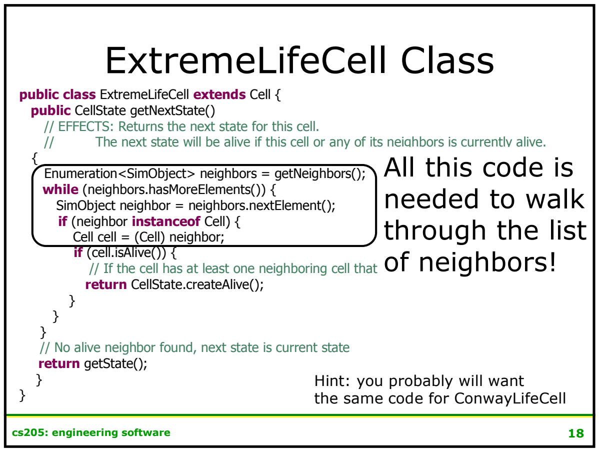


```
private CellState(boolean isalive)
// EFFECTS: Initializes this to alive if isalive is true,
// otherwise initializes this to the dead state.

static public/* nonnull */CellState createAlive()
// EFFECTS: Returns an alive cell state.
{
   return new CellState(true);
}

static public/* nonnull */CellState createDead()
// EFFECTS: Returns a dead cell state.
{
   return new CellState(false);
}

public Color getColor()
// EFFECTS: Returns the display color for this state
{
   if (alive) return Color.green;
   else return Color.white;
}
```

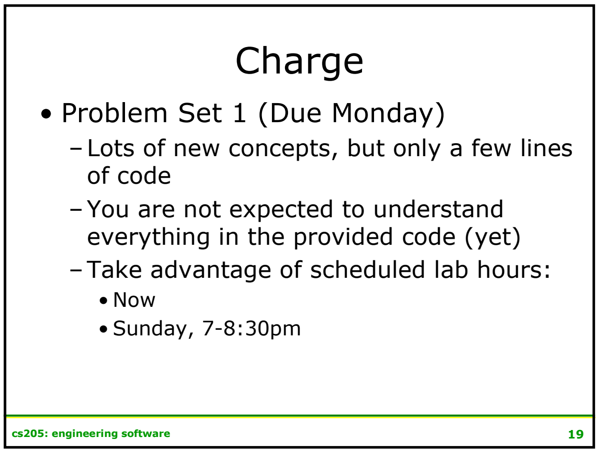


ExtremeLifeCell Class

```
public class ExtremeLifeCell extends Cell {
  public CellState getNextState()
    // EFFECTS: Returns the next state for this cell.
    //         The next state will be alive if this cell or any of its neighbors is currently alive.
  {
    Enumeration<SimObject> neighbors = getNeighbors();
    while (neighbors.hasMoreElements()) {
      SimObject neighbor = neighbors.nextElement();
      if (neighbor instanceof Cell) {
        Cell cell = (Cell) neighbor;
        if (cell.isAlive()) {
          // If the cell has at least one neighboring cell that
          return CellState.createAlive();
        }
      }
    }
    // No alive neighbor found, next state is current state
    return getState();
  }
}
```

All this code is needed to walk through the list of neighbors!

Hint: you probably will want the same code for ConwayLifeCell



Charge

- Problem Set 1 (Due Monday)
  - Lots of new concepts, but only a few lines of code
  - You are not expected to understand everything in the provided code (yet)
  - Take advantage of scheduled lab hours:
    - Now
    - Sunday, 7-8:30pm