Introduction to Java

March 1, 2001

CBRSS and the John M. Olin Institute for Strategic Studies

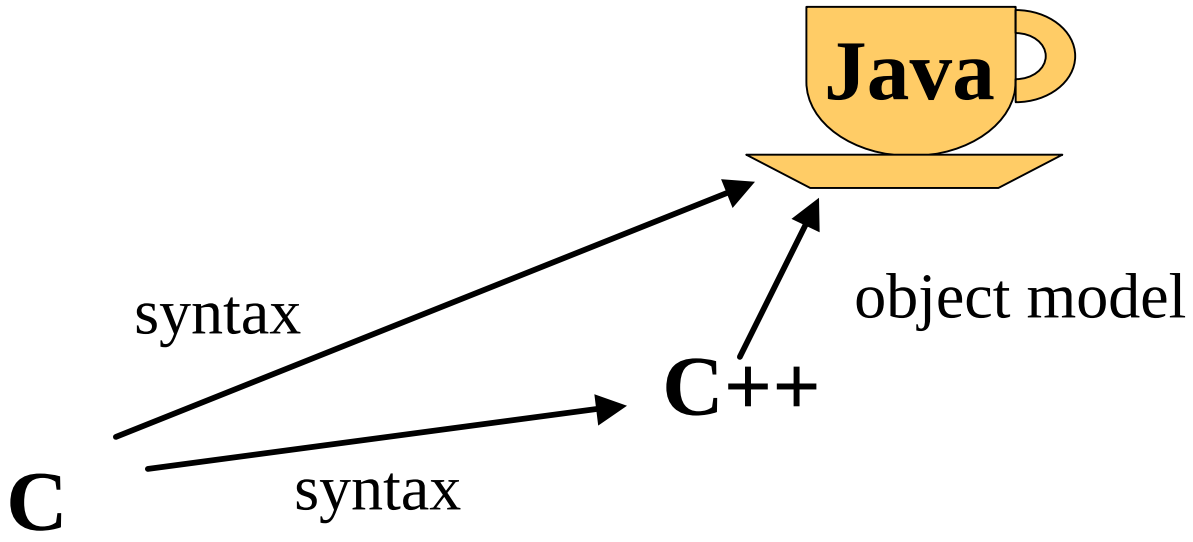
Lars-Erik Cederman

Outline

- Java
  - overview
  - simple structures
  - object-orientation
  - roulette example
- How to compile etc.
- Readings and documentation

Java

- Conceived by Sun in the early 1990s
- Became the new standard for the web thanks to platform-independence

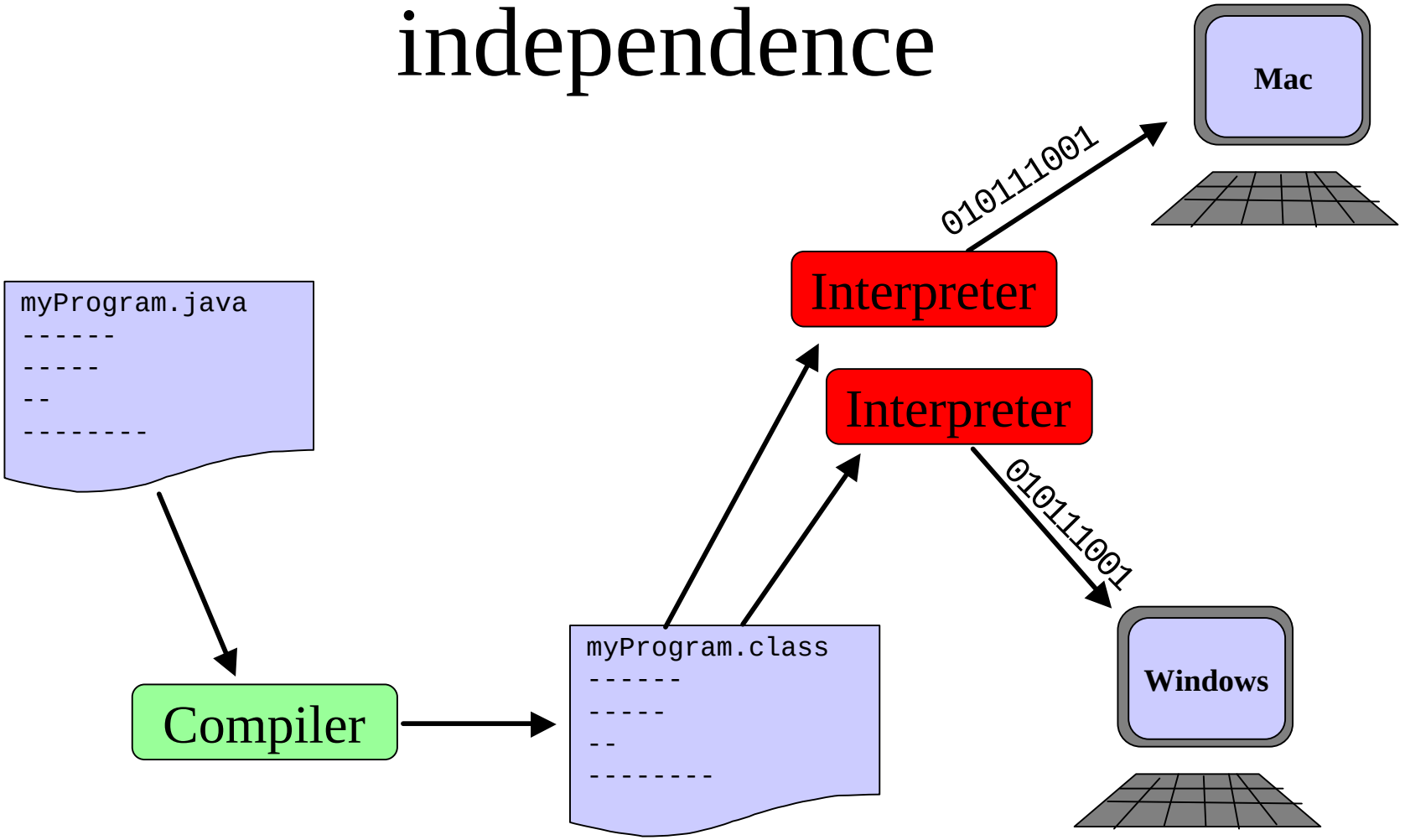


Some Java Buzzwords

- Robust
- Portable
- Object-oriented

(see Schildt Module 1)

The Java solution to platform-independence



C-like syntax

```
int i;
int age;
int drinks;

drinks = 0;
age = 18;

if (age > 20) {
  for (i = 0; i<10; i++)
    drinks++;
}
```

Example

```
class Example {

  public static void main (String args[]) {

    System.out.println("Java drives the Web.");

  }
}
```

Example2

```
class Example2 {
  public static void main (String args[]) {
    int var1;
    int var2;

    var1= 1024;

    System.out.println("var1 contains " + var1);

    var2 = var1/ 2;

    System.out.print("var2 contains var1/ 2");
    System.out.println(var2);
  }
}
```

CheckerBoard: Desired output

```
BWBWBWBW
WBWBWBWB
BWBWBWBW
WBWBWBWB
BWBWBWBW
WBWBWBWB
BWBWBWBW
WBWBWBWB
```

CheckerBoard: Code

```
class CheckerBoard {
  public static void main (String args[]) {
    boolean black = true;
    for (int i=0; i<8; i++) {
      for (int j=0; j<8; j++) {
        if (black)
          System.out.print("B");
        else
          System.out.print("W");
        black = !black;
      }
    black = !black;
    System.out.println();
    }
}
```

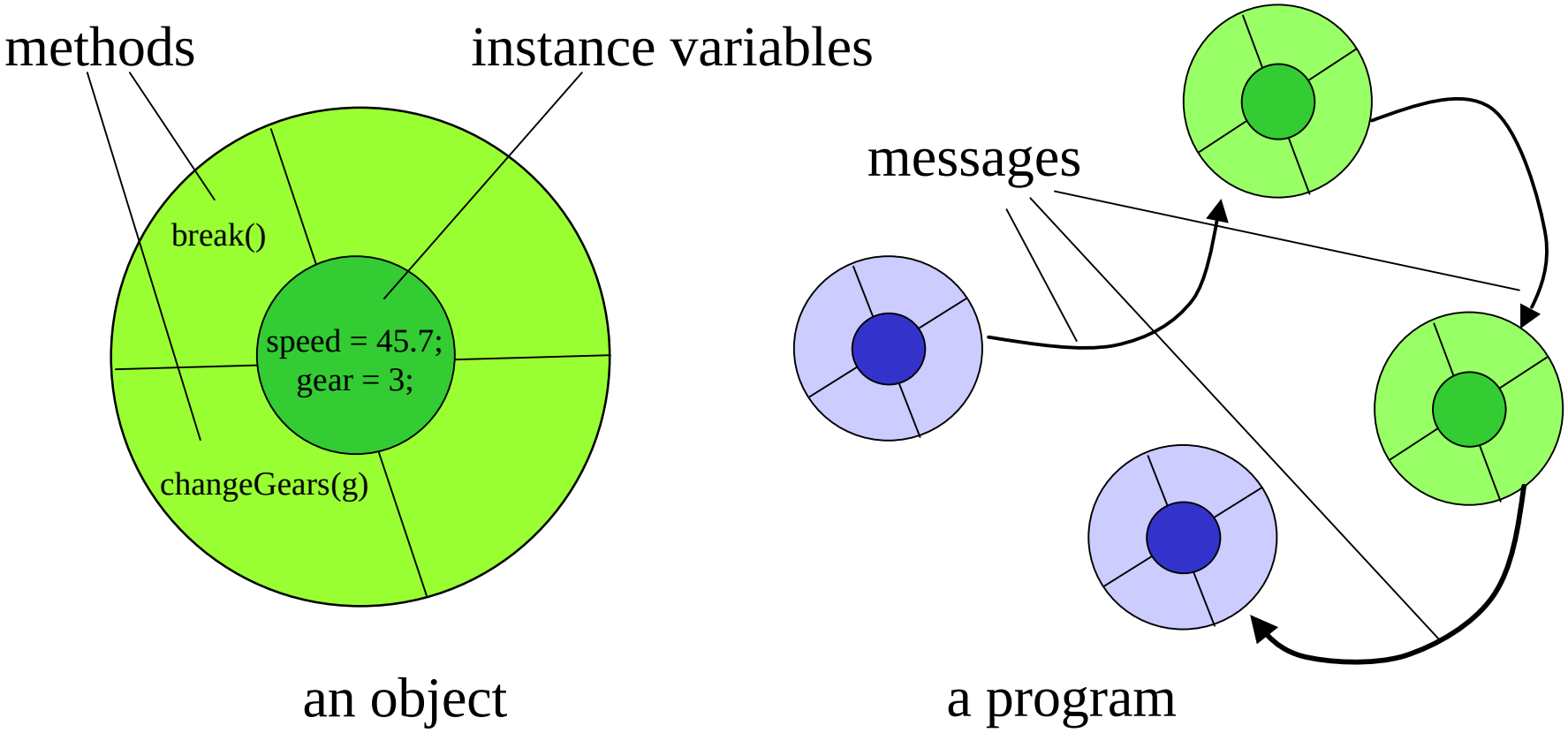
Simple datatypes

- int a = 12;
- double d = 3.14;
- char c = 'X';
- boolean b = false;

Operators: +, -, *, /, ==, >, <, &&, ||, ++, --

More in Schildt Module 2!

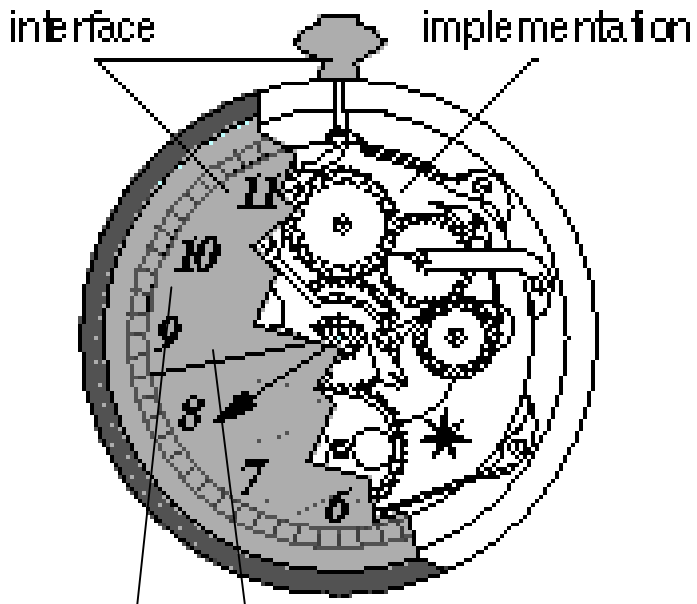
Object Oriented Programming



an object

a program

Interface vs. implementation

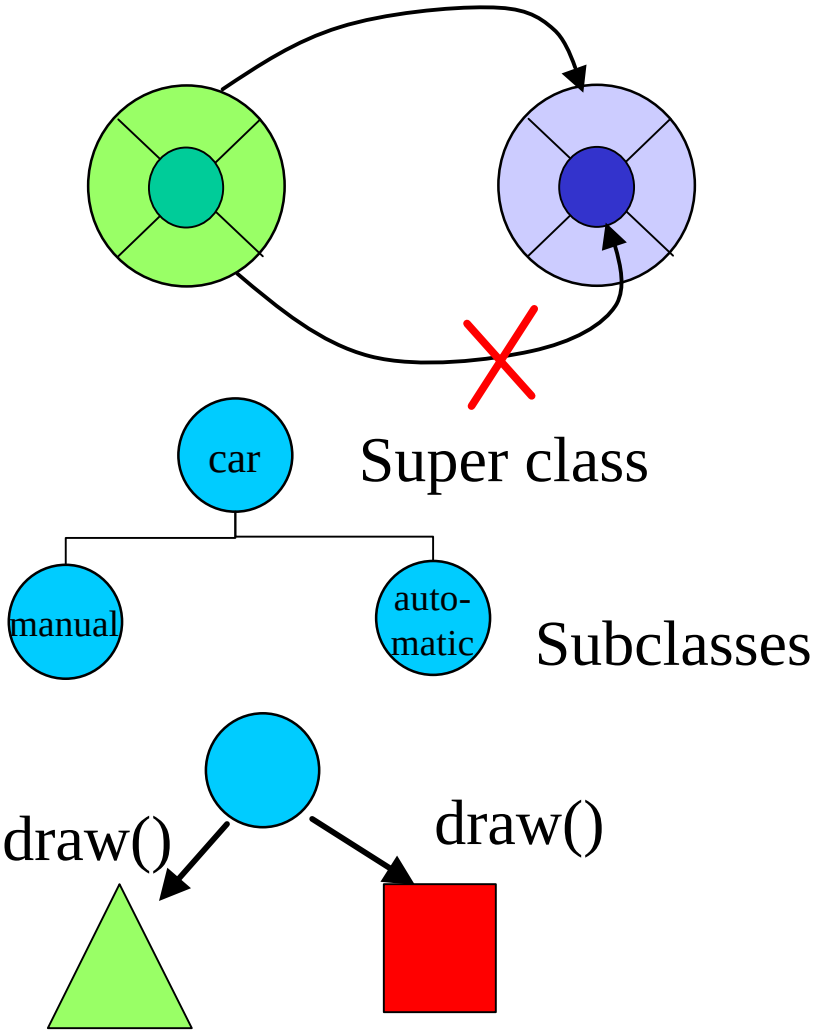


Objects hide their functions and data

User only has to be familiar with the interface of an object, not its implementation

The three principles of OOP

- Encapsulation
  - Objects hide their functions (**methods**) and data (**instance variables**)
- Inheritance
  - Each **subclass** inherits all variables of its **superclass**
- Polymorphism
  - Interface same despite different dat1types



Class declarations

```
class Car {
  public boolean isManual;
  private double speed;

  public void start (double s) {
    speed = s;
  ;

  public void stop() {
    speed = 0.0;
  }

  public double getSpeed() {
    return speed;
  }
}
```

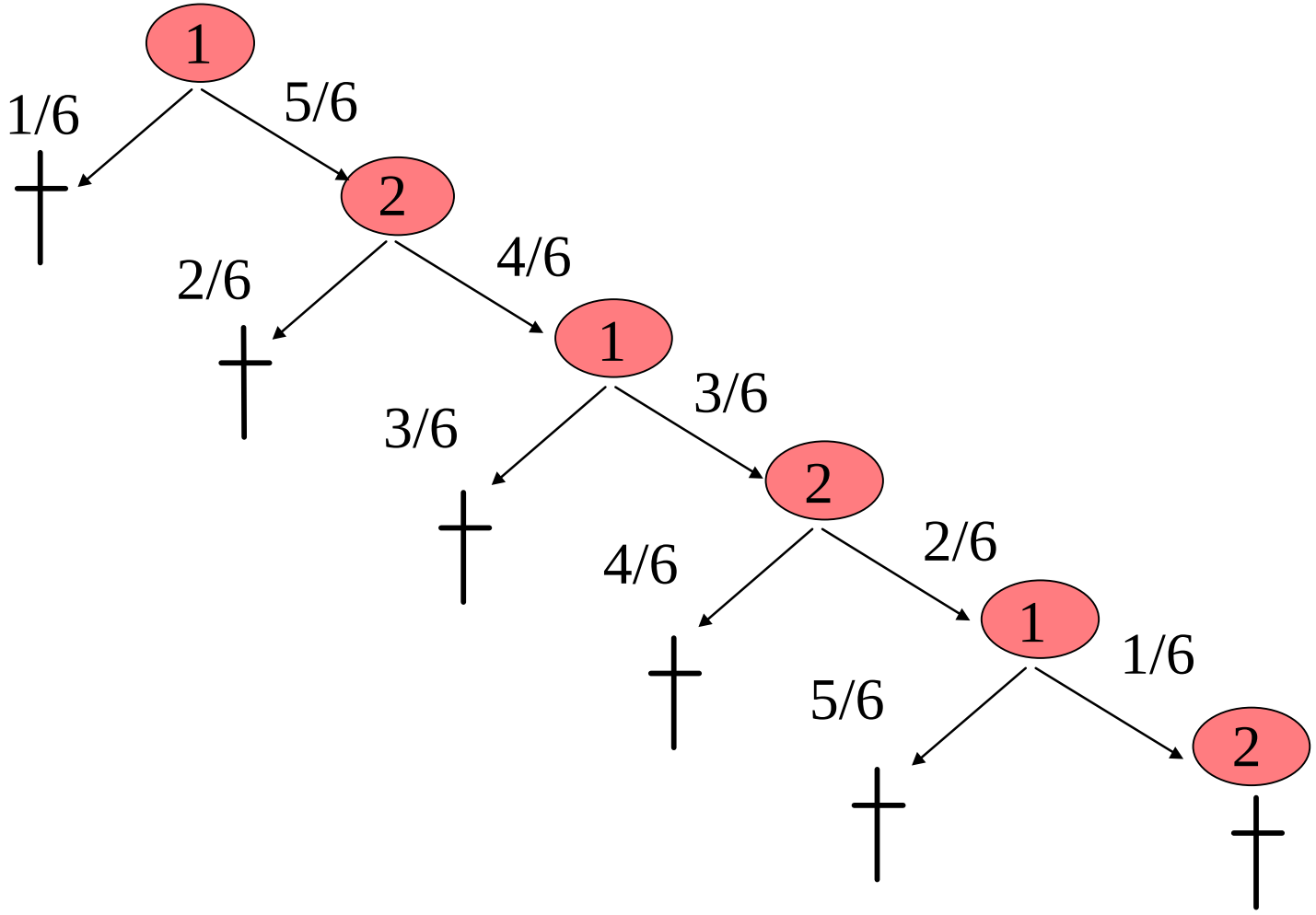
Class declarations

```
class Manual extends Car {
  private int gear;

  public void shiftGears(int g) {
    gear = g;
  }
}
```

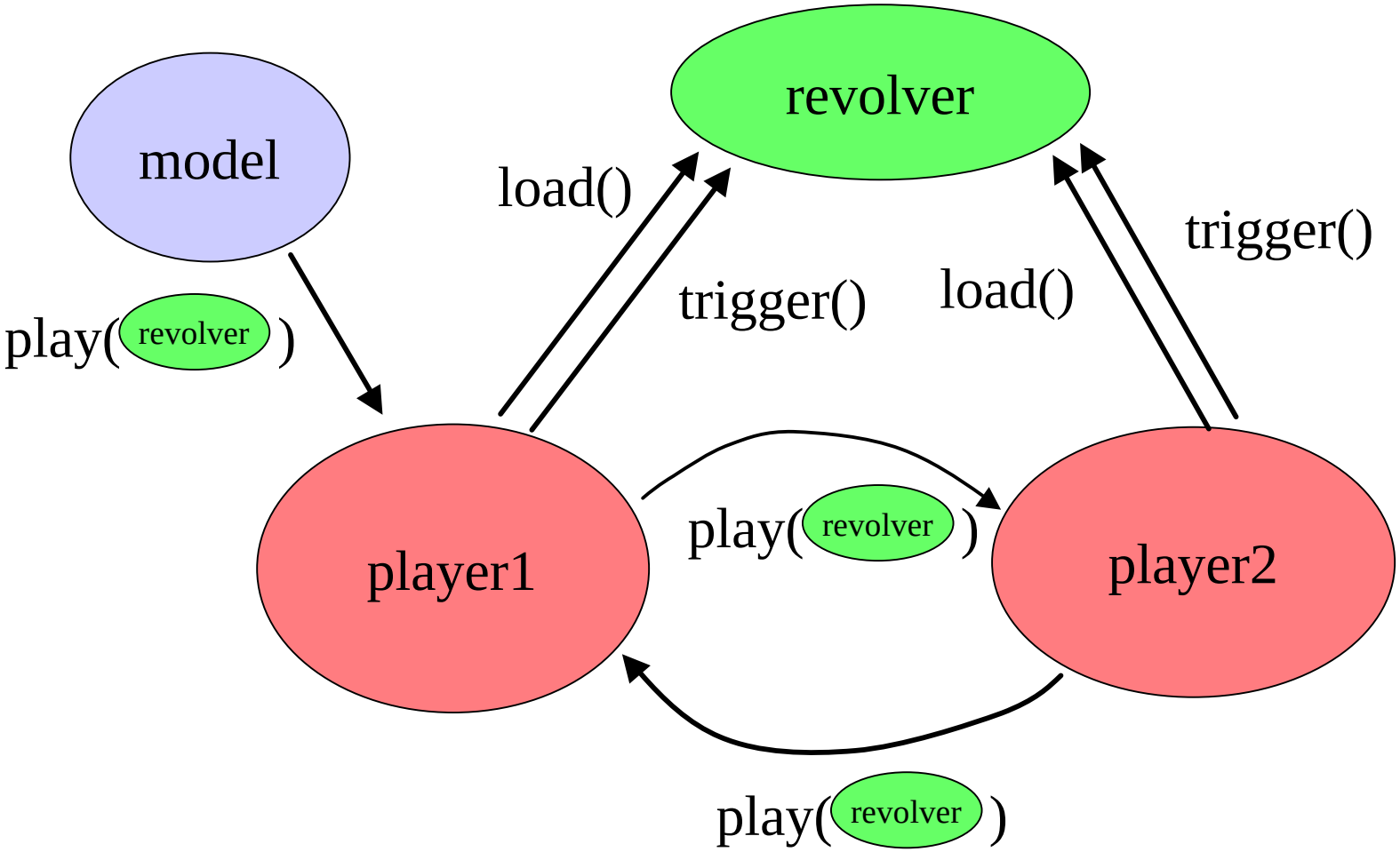
Creating and using objects

```
class Driver {
  Car myCar;

  public void buyCar(boolean isManual) {
    if (isManual)
      myCar = new Manual();
    else
      myCar = new Car();
    myCar.isManual = isManual;
    }
  }

  public void drive() {
    if (myCar.isManual)
      myCar.shiftGears(1);
    myCar.start(45.0);
    myCar.stop();
  }
}
```

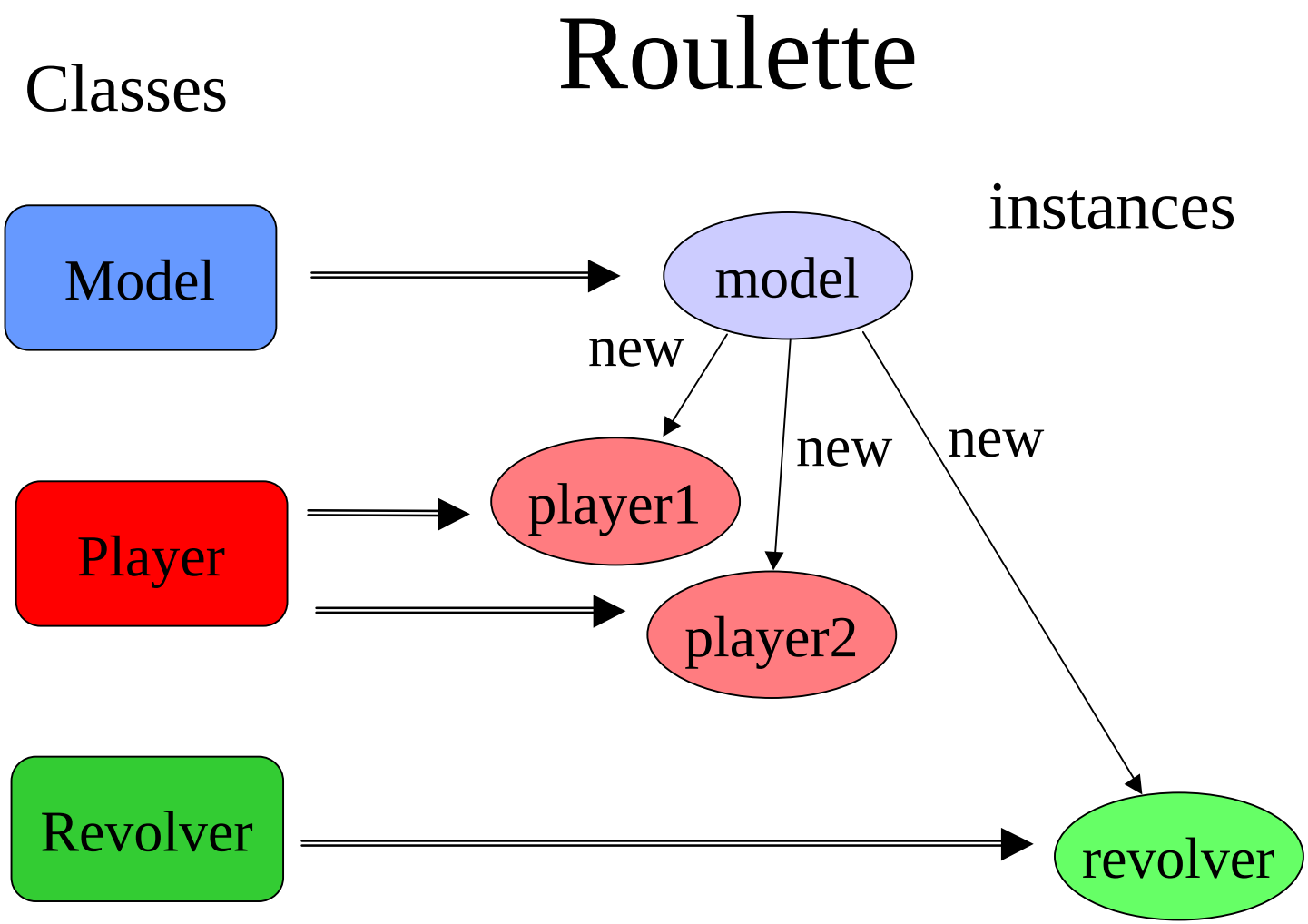
Example: Russian Roulette



Web of message calls...



Roulette


Classes
instances
Model
Player
Revolver
model
player1
player2
revolver
new
new
new

Roulette: Model

```
import java.util.Random;

public class Model {
  public static void main(string[] args) {
    int numIter = 20;
    int counter = 0;
    Random generator = new Random();

    for (int i=0; i<numIter; i++) {
      generator.setSeed(i);
      Revolver revolver = new Revolver(generator);
      Player player1= new Player(1);
      Player player2 = new Player(2);

      player1.setOther(player2);
      player2.setOther(player1);

      int outcome = player1.play(revolver);
      if (outcome == 1)
        counter++;
    }
    System.out.println("1 dies with probability " +
                       (double)counter/(double)numIter);
  }
}
```

one game

Roulette: Player

```
public class Player {
  int id;
  Player other;

  public Player(int i) {
    id = i;
  }

  public void setOther(Player o) {
    other = o;
  }

  public int play(Revolver r) {
    r.load();
    boolean dead = r.trigger();
    if (dead)
      return id;
    else
      return other.play(r);
  }
}
```

Roulette: Revolver

```
import java.util.Random;

public class Revolver {
  int bullets;
  Random generator;

  public Revolver(Random g) {
    generator g;
    bullets = 0;
  }

  public void load() {
    bullets++;
  }

  public boolean trigger() {
    return generator.nextInt(6)+1<= bullets;
  }

}
```

Russian Roulette

0: 12
1: 1212
2: 12
3: 12
4: 1
5: 12
6: 12121
7: 1212
8: 12
9: 1212
10: 121212
11: 121
12: 12
13: 121
14: 12
15: 1212
16: 1212
...

1's prob. of dying =
0.521605...

1,000 replications:
0.522

Documentation

- Schildt, H. *Java2: A Beginner's Guide*
- Eckel, B. *Thinking in Java*
- http://java.sun.com/ (select "Java Tutorial")
- http://www.borland.com/techpubs/jbuilder/