Decisions

The Plan

- Decisions at the Basic Level
  - if if/else
  - What is equality?
- Decisions at the Game/Graphics Level
  - When do things collide?
- Go over/write several programs
  - RoundOffError.java
  - BoundingCircle.java
  - BoundingCircleTest.java
  - BoundingBox.java
  - BoundingBoxTest.java

If Statements

- Have seen two coding forms
  - if(boolean expression) {
    do something
  }
  - if(boolean expression) {
    do something
  } else {
    do some alternative
  }
- Often logical (boolean) expression asks about equality
  - Why can this be a problem?

String Equality

- Look at stringEqualsTest
  String one="happy day"
  String two="happy"+" day"
  System.out.println("Don't use == to compare Strings")
  System.out.println("Test A: comparing "+one+" and "+two+" ");
  if(one==two)
    System.out.println("same object");
  else
    System.out.println("different object");
  if(one.equals(two))
    System.out.println("same contents");
  else
    System.out.println("different contents");
String Equality

- Look at `stringEqualsTest` (continued)
  
  ```java
  two="happy day";
  System.out.println("Test B: comparing " + one + " and " + two);
  if(one==two)
      System.out.println("same object");
  else
      System.out.println("different object");
  if(one.equals(two))
      System.out.println("same contents");
  else
      System.out.println("different contents");
  ```

Floating (double) Equality

- Look at `floatingEqualsTest`

  ```java
  System.out.println("Don't use == to compare floating point numbers");
  double x = Math.sqrt(13);
  if(x*x==13)
      System.out.println("same");
  else
      System.out.println("different: " + x*x + "!=13");
  ```

  Look at `RoundOffError` in code directory

Game Level Decisions

- Video Game Level Domain: What are we trying to do
- Decide on collision (intersection)
  - Bullet with target
  - Two major objects
  - Beam (line) with target
- Potentially Very Difficult Problem
  - Imagine Complex shaped Space Ship
    - does bullet miss or just hit that fin?
- Different approaches available
  - First decide *Exact or Approximate*
    - For many games, especially fast moving, short cuts work
  - Exact solution costly:
    - difficult code
    - computer time demands result in sluggish game

Approximate Solutions

- Approximate Shape of object
  - Bounding Box
  - Bounding Circle
- Design code to detect intersection of two rectangles

  ```java
  public class BoundingBox {
      double x, y, width, height
      public BoundingBox(double px, py, w, h) {
          x = px;
          y = py;
          width = w;
          height = h;
      }
  }```
Rectangle Intersection

```java
public boolean intersect(double px, double py,
           double w, double h) {

    // write in class...

}

public boolean isPointIn(double px, double py){

    // write in class...

}
```

Approximate Solutions

- When is a bounding circle better than a bounding rectangle?
- Design code to detect intersection of two circles

- Note that much of this is done for you already if you choose the correct class

Exact Solutions

- Sometimes you can’t approximate
- See java.awt.geom
  - However, consider costs (even if you don't have to code)
  - Just because it's done for you doesn’t mean it won’t take time

- Using constructive area geometry you can build complex shapes

- Look at API for classes that define intersection, etc.