CS 211: Casting, Equals Methods

Chris Kauffman

Week 5

Logistics

Deadlines

- Lab 5: Monday (today)
- P3: Due Sunday

Reading: Inheritance

- Building Java Programs Ch 9
- Lab Manual Ch 7

Goals Today

- Equality
- Dynamic Dispatch

Exam 1 Schedule

Mon 2/27	Equals, Dispatch
Wed 3/1	Abstract Classes
	Lab Quiz
Sun 3/5	Project 3 Due
Mon 3/6	Review
Wed 3/8	Exam 1
Mon 3/13	Spring Break

Everyone has `equals()` and `toString()`

```
package java.lang;
public class Object{ ... }
```

Class Object is the root of the class hierarchy. Every class has Object as a superclass. All objects, including arrays, implement the methods of this class.

```
public String toString()
```

Returns a string representation of the object.

```
public boolean equals(Object obj)
```

Indicates whether some other object is "equal to" this one.

```
int a[] = {1,2,3}, b[] = {1,2,3};
System.out.println( a.equals(b) ); // ??
```

Checking type at run time: `instanceof`

`X instanceof Y`

- A keyword/syntax construct
- `true` if `X` has `Y` as an ancestor - *X is a Y*
  - `Mascot` is a `Duck`, `Duck` is a `Animal`, `Animal` is a `Object`
- `false` otherwise

Casting: Trust me, javac

```
Object o = new Coord(1,2);
System.out.println(o.row);   // Compile error
Coord c = (Coord) o;         // Trust me, it's a Coord
System.out.println(c.row);   // Voila!
```

- What can go wrong with casting: `(Coord) o`
- Try it interactively:

```
Object o = new String("hi");
Coord c = (Coord) o;
```

- What about the following. . .

```
Object x = new Coord(1,2);
Object y = new Coord(1,2);

System.out.println( x.equals(y) );
```

The most common case of casting

Compare current object like `Coord` to arbitrary other `Objects`

`Coord` Class Methods

```
// Are coordinates equal
public boolean
equals(Coord c){
  return
    this.row==c.row &&
    this.col==c.col;
}
// Compare arbitrary object
public boolean
equals(Object o){
  if(o instanceof Coord){
    Coord that = (Coord) o;
    return
      this.row==c.row &&
      this.col==c.col;
  }
  else{ return false; }
}
```

Equals works great now

```
Object x = new Coord(1,2);
Object y = new Coord(1,2);
System.out.println( x.equals(y) );
```

But what about...

```
Object w = new Coord(1,2);
Object z = new Coord3D(1,2,3);

System.out.println( w.equals(z) );
System.out.println( z.equals(w) );
```

(Hint: damn...)

Note on @Override

Annotating methods with `@Override` which are intended to override a parent method notifies the compiler to check for danger.

A Subtle Bug

```
@Override
public boolean equals(Coord other){
  if(other==null ||
     !(other instanceof Coord)){
    return false;
  }
  Coord that = (Coord) other;
  return
    this.row==that.row &&
    this.col==that.col;
}
```

Compiler Output

```
> javac Coord.java

Coord.java:17: error:
method does not override or
implement a method from a
supertype
  @Override
  ^
1 error
```

Exercise: ScreamWriter Equality

Implement `equals(Object o)` for `ScreamWriter`

```
public class ScreamWriter
extends PrintWriter
{
  private boolean highVolume;
  public ScreamWriter(OutputStream o){
    super(o);
    this.highVolume = true;
  }
  public ScreamWriter(File f)
  throws Exception{
    super(f);
    this.highVolume = true;
  }
  public ScreamWriter(String filename)
  throws Exception{
    super(filename);
    this.highVolume = true;
  }
  public void toggleVolume(){
    this.highVolume = !this.highVolume;
  }

  public void println(String s){
    String output = s;
    if(this.highVolume){
      output = output.toUpperCase();
    }
    super.println(output);
    this.flush();
  }

  // Override Equals: returns true if
  // the argument o is another
  // ScreamWriter and has the same value
  // for highVolume
  public boolean equals(Object o){
    ???
  }
}
```

Optional Exercise: Equality Gets Trickier

What is printed on the right based on `equals()` definitions?

```
public class Coord {
  public boolean equals(Object o){
    if(!(o instanceof Coord)){
      return false;
    }
    Coord that = (Coord) o;
    return
      this.row==that.row &&
      this.col==that.col;
  }
}
public class Coord3D extends Coord{
  public boolean equals(Object o){
    if(!(o instanceof Coord3D)){
      return false;
    }
    Coord3D that = (Coord3D) o;
    return
      this.row==that.row &&
      this.col==that.col &&
      this.height == that.height;
  }
}
```

```
Coord   a = new Coord(1,2);
Coord3D b = new Coord3D(1,2,3);

Coord   c = new Coord(10,12);
Coord3D d = new Coord3D(10,12,14);

System.out.println( a.equals(c) );
System.out.println( a.equals(b) );
System.out.println( b.equals(a) );
System.out.println();

System.out.println( c.equals(d) );
System.out.println( d.equals(c) );
System.out.println();

String s = "(1,2)";
System.out.println( s.equals(a) );
System.out.println( a.equals(s) );
```