

# CS601: Principles of Software Development

Interfaces.  
Comparable.

Olga A. Karpenko

# Quiz 1

- On Canvas
- Access Code: HASH

# Announcements

- Lab0 is due on Friday night
- Possibility: Sep 9, programming "bootcamp"  
– will you come?
- Visualizations on Canvas:

<http://cs.usfca.edu/~galles/visualization/>

# Abstract methods

- You can declare a method but not define it
- Called an “abstract method”

```
public abstract void draw(int size);
```

# Interfaces

- *Interface* is a collection of abstract methods and constants
  - In Java 8, interfaces can also have *default* methods
- Establishes methods that a class will implement
- You do *not* have to use the keyword *abstract* for methods
  - Because all methods in an interface are abstract unless they are declared as **default**

# Interfaces

```
public interface Doable {  
    public void doThis();  
    public int doThat();  
    public void doThis2(double value, char ch);  
}
```



**A semicolon immediately  
follows each method header**

# Interfaces


- An interface cannot be instantiated
- Methods in an interface have public visibility
- A class formally implements an interface by:
  - stating so in the class header
  - providing implementations for every abstract method in the interface

# Interfaces

```
public class CanDo implements Doable
{
    public void doThis()
    {
        // whatever
    }

    public void doThat()
    {
        // whatever
    }

    // etc.
}
```



Each method listed  
in Doable is  
given a definition



# Example

```
public interface Moveable() {  
    public void move();  
}
```

# Example

```
public class AlienX implements Moveable {  
    private double x, double y;  
    public void move() {  
        x += 2;  
    }  
}  
  
public class AlienY implements Moveable {  
    private double x, double y;  
    public void move() {  
        y -= 10;  
    }  
}
```

# Interfaces

- A class can implement multiple interfaces
- The interfaces are listed in the `implements` clause
- The class must implement all methods in all interfaces listed in the header

```
class ManyThings implements interface1, interface2
{
    // all methods of both interfaces
}
```

# Default Methods in Interfaces

- New feature in Java 8
- Provide definition of some methods
- Example: interface Speaker, class Creature (see code)

# Java Standard Library Interfaces

# Interfaces

- The Java API contains many helpful interfaces
- The `Comparable` interface contains `compareTo`
  - used to compare two objects
- The `String` class implements `Comparable`
  - So we can put strings in lexicographic order

# Comparable

- Any class can implement Comparable
  - to provide a mechanism for comparing objects of that type

```
MyClass obj1, obj2;
```

```
// TODO: ... initialize obj1, obj2
```

```
if (obj1.compareTo(obj2) < 0)
```

```
    System.out.println ("obj1 is less than obj2");
```

# Comparable

- The value returned from `compareTo` should be:
  - negative if `obj1 < obj2`,
  - 0 if `obj1 == obj2`
  - positive if `obj1 > obj2`



# Comparable

- It's up to the programmer to determine what makes one object less than another
- Examples:
  - Compare students based on GPA
  - Compare Employees based on salary
  - Compare books based on titles

# Example

- See classes `Student` and `StudentExample`
- We can compare students based on the GPA or based on the name