Today's Goals

1. Clock
2. DigitalClock
3. PreciseDigitalClock
4. AnalogClock

Our goals are to understand how methods get inherited and how objects in a hierarchy interact:
- Clock c = new DigitalClock(true);
- AnalogClock ac = new DigitalClock(true);
- PreciseDigitalClock pdc = new DigitalClock(true);
- c.getTime(); ac.getTime(); pdc.getTime();

Clock Hierarchy Diagram

For each of the following, is it **always**, **sometimes**, or **never** true:
- A DigitalClock is a Clock?
  - **Always**
- A DigitalClock is a type of Clock with digital features.
- An AnalogClock is a DigitalClock?
  - **Never**
- AnalogClock's have a face; DigitalClock's don't.
- A PreciseDigitalClock is a DigitalClock?
  - **Always**
- A PreciseDigitalClock is a DigitalClock that includes seconds.
- A DigitalClock is a PreciseDigitalClock?
  - **Sometimes**
- Not all DigitalClocks have seconds, but those that do are PreciseDigitalClocks.
- A Clock is a DigitalClock?
  - **Sometimes**
- Not all Clocks have DigitalClock features, but those that do are DigitalClocks.

Clock Class

```
public class Clock {
  private int hour;
  private int minute;
  public int getMinute() { return this.minute; }
  public int getHour() { return this.hour; }
  public String getTime() { return hour + " " + minute; }
}
```

Output:
```
>> Clock c = new Clock(); // hour = 4, minute = 12
>> System.out.println(c.getTime() + "..." + c.getHour() + "..." + c.getMinute());
4 12...4...12
```

What specializations could we make to Clock?
- An “analog” clock with a face?
- A “digital” clock with military time?
- A clock with seconds?

AnalogClock Class

```
public class AnalogClock extends Clock {
  public static final int NUM_HOURS = 12;
  public static final int NUM_MINUTES = 60;
  public double getHourHandAngle() {
    return 360 * ((double) (this.getHour() % 12) / NUM_HOURS);
  }
  public double getMinuteHandAngle() {
    return 360 * ((double) this.getMinute() / NUM_MINUTES);
  }
  public String getTime() {
    return "Hour Hand: " + this.getHourHandAngle() + ", " + "Minute Hand: " + this.getMinuteHandAngle() + "%";
  }
}
```

AnalogClock vs. Clock

- Is an AnalogClock a Clock?
  - **Always**
- An AnalogClock is a Clock with extra features.

What is different about an AnalogClock?
- It has new methods: getHourHandAngle, getMinuteHandAngle
- It “overrides” getTime to do something different
Notice that Java knows that Clock and DigitalClock are two different classes. If we remove the second and third lines, we get:

```
>> Hour Hand: 180%, MinuteHand: 60%
>> 180
>> 60
```

This doesn’t compile! Java treats PreciseDigitalClock as a new Clock class. Is a PreciseDigitalClock a DigitalClock? Absolutely! A PreciseDigitalClock is a DigitalClock with extra features.

What is different about a PreciseDigitalClock (from a DigitalClock)?
- It has a new constructor
- It has a new field: isMilitaryTime
- It has new methods: getPeriod, isMilitaryTime
- It “overrides” getTime and getHour to do something different

What is different about a PreciseDigitalClock (from a DigitalClock)?
- It is missing the one argument constructor
- It has a new field: second
- It “overrides” getTime to do something different

This doesn’t compile. Clock doesn’t have a getPeriod method!
Mystery Problem #1

```java
public class Snow {
    public void method2() {
        System.out.println("Snow 2");
    }
}
```

```java
public class Rain extends Snow {
    public void method2() {
        System.out.println("Rain " + (PreciseDigitalClock)c6.getTime());
    }
}
```

```java
public class Sleet extends Snow {
    public void method2() {
        System.out.println("Sleet 2");
    }
}
```

```java
public class Fog extends Sleet {
    public void method2() {
        System.out.println("Fog 2");
    }
}
```

```java
public void method3() {
    var2.method1();
}
```

```java
new DigitalClock()
```

```java
new Snow()
```

```java
new Rain()
```

```java
new Sleet()
```

```java
new Fog()
```

```java
var2.method2();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
new DigitalClock()
```

```java
new Snow()
```

```java
new Rain()
```

```java
new Sleet()
```

```java
new Fog()
```

```java
var2.method2();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```

```java
Snow var2 = new Rain();
```

```java
Snow var2 = new Sleet();
```

```java
Snow var2 = new Fog();
```

```java
Snow var2 = new Rain();
```

```java
((Sleet) var2).method2();
```

```java
((Rain) var2).method1();
```
Mystery Problem #4

```java
Snow var2 = new Rain();
var2.method2();
```

OUTPUT
```
>> Rain 2
```

Mystery Problem #5

```java
Snow var2 = new Rain();
((Sleet) var2).method2();
```

OUTPUT
```
>> ClassCastException: *Error*
```

Mystery Problem #6

```java
Snow var2 = new Fog();
((Sleet) var2).method2();
```

OUTPUT
```
>> Sleet 2
>> Snow 2
>> Fog 3
```