## Static Modifier

**As we discussed previously, static means, "belongs to the class." We can have static variables of a class and static methods in a class.**

**The former are very, very infrequently used. Students often mistakenly call these "global variables" and use them to avoid learning the details of parameter passing.**

**Here is an example of a declaration of a static variable. It would be declared outside of any method, but inside of a class:**

**private static int value;**

**This variable would then belong to the class. To refer to it, (which can only be done inside of the class), you just use the term value. Only one copy of value would exist no matter how many instances of the class are created.**

## Static Methods

**We have already discussed these in length. Basically, since a static method resides in a class, the standard way to call one is to use the syntax:**

**classname.methodname(parameters here)**

**If you are within the same class then you can omit the classname. Since these just belong to the class, they DON'T operate on an object, and no object has to be created to use one. For the most part, a static method is very similar to a regular function in C.**

**The best way to see the effect of a static variable is an example:**

```
public class statictest {

  private int num;
  private static int value = 0;

  public statictest(int x) {
    num = x;
    value++;
  }

  public int numObjectsCreated() {
    return value;
  }

  public void increment() {
    num++;
  }

  public String toString() {
    return "num = "+num+" value = "+value;
  }

}
```

**Test this with the following main method in another class:**

```
public static void main(String[] args) {
  statictest first = new statictest(3);
  statictest second = new statictest(10);
  first.increment();
  System.out.println(first);
  System.out.println(second);
}
```

## Fraction Class Example

**For the remainder of the lecture we will extend the Fraction class. We will talk about designing a class and go through the process of determining reasonable methods to add to a class, etc.**

## String Class Example

**We will also build a String Class from scratch. We will design ours differently than Java's!**