Special Topic 9.2

Anonymous Classes

An entity is *anonymous* if it does not have a name. In a program, something that is only used once doesn't usually need a name. For example, you can replace

```
Coin aCoin = new Coin(0.1, "dime");
data.add(aCoin);
```

with

```
data.add(new Coin(0.1, "dime"));
```

if the coin is not used elsewhere in the same method. The object `new Coin(0.1, "dime")` is an **anonymous object**. Programmers like anonymous objects, because they don't have to go through the trouble of coming up with a name. If you have struggled with the decision whether to call a coin `c`, `dime`, or `aCoin`, you'll understand this sentiment.

Inner classes often give rise to a similar situation. After a single object of the `RectangleMeasurer` has been constructed, the class is never used again. In Java, it is possible to declare **anonymous classes** if all you ever need is a single object of the class.

```
public static void main(String[] args)
{
   // Construct an object of an anonymous class
   Measurer m = new Measurer()
      // Class declaration starts here
      {
         public double measure(Object anObject)
         {
            Rectangle aRectangle = (Rectangle) anObject;
            return aRectangle.getWidth() * aRectangle.getHeight();
         }
      };

   DataSet data = new DataSet(m);
```

```
        . . .
    }
```

This means: Construct an object of a class that implements the `Measurer` interface by declaring the `measure` method as specified. Many programmers like this style, but we will not use it in this book.