Chapter 3
Object Interaction

3.11

Internal & External Method Calls

3.10.1 Internal Method Calls

A class definition typically includes *fields*, *constructors*, and *methods*.

Each object is an *instance* of that class. Two objects from the same class will have the same fields, constructors, and methods.

The state of each object, however, is usually different.

3.10.1 Internal Method Calls

Consider a class, Rectangle, that defines a rectangle, with the following features:

- int fields for *length* and *width*
- a constructor to initialize length and width
- a method to calculate the area

3.10.1 Internal Method Calls

Suppose we created a new Rectangle object, setting the length to 5 and the width to 10.

If we call, or *invoke*, the area() method, the result will be 50 (length times width).

This is an internal method call. The Rectangle method area() is called from within a Rectangle object. The area calculation uses the length and width fields from that object.

3.10.2 External Method Calls

Now suppose we have a Shapes class, with the following code:

```
// fields
private Rectangle box1;
private Rectangle box2;

// as part of constructor
box1 = new Rectangle(5, 10);
box2 = new Rectangle(2, 2);
```

3.10.2 External Method Calls

When we create an object using the Shapes class, it will automatically include two Rectangle objects as well.

The Shapes class probably doesn't have an area() method of its own. What if we want the area of one of the rectangles?

3.10.2 External Method Calls

It is possible to call, or *invoke*, the method of one class from within an object of a different class.

From our Shapes object, we can call the area() method using "dot notation".

```
    System.out.println( box1.area() );
```

Since box1 is a Rectangle object, the method area() is available.

3.10.2 External Method Calls

When making an external call using dot notation, it is important to understand that the method will be invoked using the state information of the specified object.

```
System.out.println( box1.area() );   // outputs 50
System.out.println( box2.area() );   // outputs 4
```

Since each box has its own state information for length and width, each call will calculate a different area.

Assigned Work

Read Chapter 3: 3.11.1 to 3.11.3

Complete exercises: 3.28 to 3.30
(and catch up on previous work)

Take up selected previous exercises

Record your answers in a text document or OpenOffice document