Chapter 8 – Classes

How to create your own classes

Key Concepts or Terminology

- Encapsulation (wrapping together)
- Attributes (data)
- Methods (behavior)
- Information hiding
- Class
- Instantiation

In C#

"programmers concentrate on creating their own user-defined types called classes"

"We also refer to classes as programmer-defined types"

Describing a Circle

Suppose that we wish to create a “type” Circle that represents an arbitrary circle in the plane.

What does it take to describe a circle?

- A point
- A radius

What constitutes a point?

- An x – coordinate
- A y – coordinate

So three pieces of information are needed for a circle. It can be represented by three double numbers which represent the x-coordinate, the y-coordinate and the radius. Our class will need these three data items.

class Circle

```
class Circle
{
   private xcoord;
   private ycoord;
   private radius;
//other things will go here
}
```

Developing a circle.

A starting point has been established on our web page for the class. We will develop this class to the point that we can:

- declare a circle;
- assign or set the values of a circle;
- show information about the circle;

Throughout the development we need to pay attention to encapsulation and protection. For example, would you ever want a circle with a negative radius?

Add another feature

Suppose we are doing geometry with circles, and we need to know whether or not two circles (already established) intersect each other.

We will set up a boolean valued method that will let one object call the method and test it with another object.

Add a method that computes the area

It will be called by C1.Area( ); where C1 is any instantiated circle.

Properties

Properties can be added to any class. They will have names and represent attributes associated with an object in the class.

Properties can also be used to retrieve data from an object or set data in an object.

Property syntax

public *return-type property-name*
{
    get
    {
        code to get information
    }
    set
    {
        code to set information
    }
}

Write Area as a property

We will illustrate that you can't have a method and a property with the same name.

Also, you will see in the Area property that only a get is used since nothing is being set in the value.

Write a property that allows you to get and set the radius of an instantiated circle.

We will add this to our Circle class. In the set portion, you will see some similarities with the SetCircle method.

When a property is used to set a value, as in

C1.Radius = 3.2;

The value of the expression on the right of the = is computed and is represented by the reserved word value in the property definition.

You do some:

Do the following one at a time and test each by adding new lines of code in Main( )

1. Write a property that will set and get the x – coordinate.
2. Write a property that will set and get the y – coordinate.
3. Write a property that will return the circumference of a circle.