

# 4

## Introduction to Classes and Objects

© 2006 Pearson Education, Inc. All rights reserved.

### OBJECTIVES

In this chapter you will learn:

- What classes, objects, and methods are.
- How to declare a class and use it to create an object.
- How to implement a class's behaviors as methods.
- How to call an object's methods to make the methods perform their tasks.

© 2006 Pearson Education, Inc. All rights reserved.

### 4.2 Classes, Objects, Methods, Properties and Instance Variables

- Class provides one or more methods
  - Method represents task in a program
  - Describes the mechanisms that actually perform its tasks
  - Hides from its user the complex tasks that it performs
  - Method call tells method to perform its task
- Classes contain one or more attributes
  - Specified by instance variables
  - Carried with the object as it is used

© 2006 Pearson Education, Inc. All rights reserved.

```
1 // Fig. 4.1: GradeBook.cs
2 // Class declaration with one method.
3 using System;
4
5 public class GradeBook
6 {
7     // display a welcome message to the GradeBook user
8     public void DisplayMessage()
9     {
10         Console.WriteLine( "Welcome to the Grade Book!" );
11     } // end method DisplayMessage
12 } // end class GradeBook
```

The method header for DisplayMessage()

© 2006 Pearson Education, Inc. All rights reserved.

### 4.3 Declaring a Class with a Method and Instantiating an Object of a Class

- Class GradeBook
  - Class declarations include:
    - Access modifier
      - Keyword **public** is an access modifier
    - Keyword **class**
    - Pair of left and right braces
  - Method declarations
    - Keyword **public** indicates method is available to the public
    - Keyword **void** indicates that there is no return type
    - Access modifier, return type, name of method and parentheses comprise method header
  - Naming convention is to capitalize the first letter of every word

© 2006 Pearson Education, Inc. All rights reserved.

```
1 // Fig. 4.2: GradeBookTest.cs
2 // Create a GradeBook object and call its DisplayMessage method.
3
4 public class GradeBookTest
5 {
6     // Main method begins program execution
7     public static void Main( string[] args )
8     {
9         // create a GradeBook object and assign it to myGradeBook
10        GradeBook myGradeBook = new GradeBook();
11
12        // call myGradeBook's DisplayMessage method
13        myGradeBook.DisplayMessage();
14    } // end Main
15 } // end class GradeBookTest
```

Use class instance creation expression to create object of class GradeBook

Call method DisplayMessage using GradeBook object

Welcome to the Grade Book!

© 2006 Pearson Education, Inc. All rights reserved.

### 4.3 Declaring a Class with a Method and Instantiating an Object of a Class (Cont.)

- **Class GradeBookTest**
  - Any class that contains a **Main** method can be used to execute an application
  - **static** method is special because it can be called without first creating object of the class
  - C# is extensible
    - Programmers can create new classes
  - Class instance creation expression
    - Keyword `new`
    - Then name of class to create and parentheses
  - Calling a method
    - Object's name, then dot separator (`.`)
    - Then method's name and parentheses

© 2006 Pearson Education, Inc. All rights reserved.

### 4.4 Declaring a Method with a Parameter

- **Method parameters**
  - Additional information passed to a method
  - Supplied in the method call with arguments
- **Parameters specified in method's parameter list**
  - Part of method header
  - Uses a comma-separated list

© 2006 Pearson Education, Inc. All rights reserved.

```
1 // Fig. 4.4: GradeBook.cs
2 // Class declaration with a method that has a parameter.
3 using System;
4
5 public class GradeBook
6 {
7     // display a welcome message to the GradeBook user
8     public void DisplayMessage( string courseName )
9     {
10         Console.WriteLine( "Welcome to the grade book for " +
11                             courseName );
12     } // end method DisplayMessage
13 } // end class GradeBook
```

Method header  
DisplayMessage that takes a courseName argument of type string

© 2006 Pearson Education, Inc. All rights reserved.

```
1 // Fig. 4.5: GradeBookTest.cs
2 // Create GradeBook object and pass a string to
3 // its DisplayMessage method.
4 using System;
5
6 public class GradeBookTest
7 {
8     // Main method begins program execution
9     public static void Main( string[] args )
10    {
11        // create a GradeBook object and assign it to myGradeBook
12        GradeBook myGradeBook = new GradeBook();
13
14        // prompt for and input course name
15        Console.WriteLine( "Please enter the course name: " );
16        string nameOfCourse = Console.ReadLine(); // read a line of text
17        Console.WriteLine(); // output a blank line
18
19        // call myGradeBook's DisplayMessage method
20        // and pass nameOfCourse as an argument
21        myGradeBook.DisplayMessage( nameOfCourse );
22    } // end Main
23 } // end class GradeBookTest
```

Call ReadLine method to read a line of input and assigns it to nameOfCourse

Call DisplayMessage with an argument

Please enter the course name:  
CS101 Introduction to C# Programming

Welcome to the grade book for  
CS101 Introduction to C# Programming!