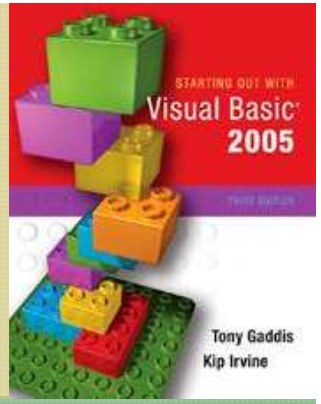
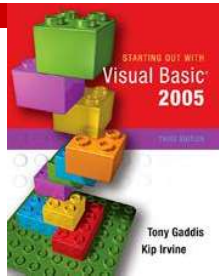


12.7 Introduction to Inheritance

Inheritance Allows a New Class to be Based on an Existing Class

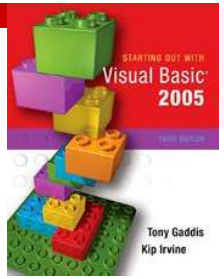
The New Class Inherits the Accessible Member Variables, Methods, and Properties of the Class on Which It Is Based

Why Inheritance?



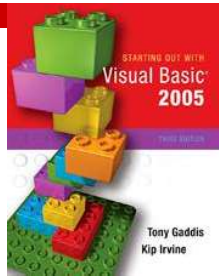
- *Inheritance* allows new classes to derive their characteristics from existing classes
- The Student class may have several types of students such as
  - GraduateStudent
  - ExchangeStudent
  - StudentEmployee
- These can become new classes and share all the characteristics of the Student class
- Each new class would then add specialized characteristics that differentiate them

Base and Derived Classes



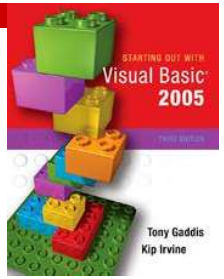
- The *Base Class* is a class that other classes may be based on
- A *Derived Class* is based on the base class and inherits characteristics from it
- Can think of the base class as a parent and the derived class as a child

The Vehicle Class (Base Class)



- Consider a Vehicle class with the following:
  - Private variable for number of passengers
  - Private variable for miles per gallon
  - Public property for number of passengers (Passengers)
  - Public property for miles per gallon (MilesPerGallon)
- This class holds general data about a vehicle
- Can create more specialized classes from the Vehicle class

The Truck Class (Derived Class)

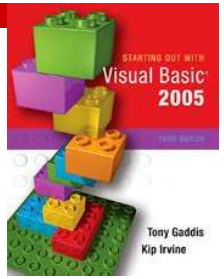


- Declared as:

```
Public Class Truck
    Inherits Vehicle
    ' Other new properties
    ' Additional methods
End Class
```

- Truck class derived from Vehicle class
  - Inherits all non-private methods, properties, and variables of Vehicle class
- Truck class defines two properties of its own
  - MaxCargoWeight – holds top cargo weight
  - FourWheelDrive – indicates if truck is 4WD

Instantiating the Truck Class

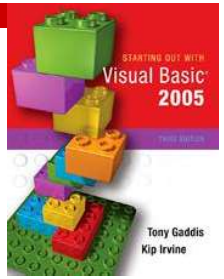


- Instantiated as:

```
Dim pickUp as New Truck()
pickUp.Passengers = 2
pickUp.MilesPerGallon = 18
pickUp.MaxCargoWeight = 2000
Pickup.FourWheelDrive = True
```

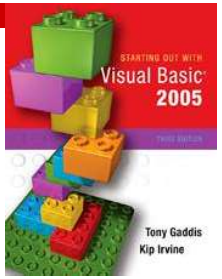
- Values stored in MaxCargoWeight and FourWheelDrive properties
  - Properties declared explicitly by Truck class
- Values also stored in MilesPerGallon and Passengers properties
  - Properties inherited from Vehicle class

Overriding Properties and Methods



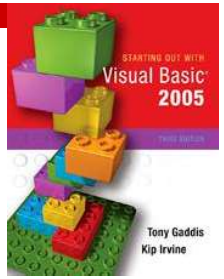
- Sometimes a base class property procedure or method does not work for a derived class
  - Can *override* base class method or property
  - Must write the method or property as desired in the derived class using same name
- When an object of the derived class accesses the property or calls the method
  - VB uses overridden version in derived class
  - Version in base class is not used

Property Override Example



- Vehicle class has no restriction on number of passengers
- But may wish to restrict the Truck class to two passengers at most
- Can override Vehicle class Passengers property by:
  - Coding Passengers property in derived class
  - Specify *Overridable* in base class property
  - Specify *Overrides* in derived class property

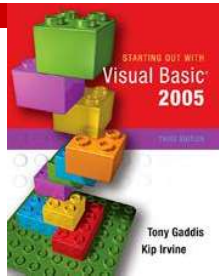
Overridable Base Class Property



- Overridable keyword added to base class property procedure

```
Public Overridable Property Passengers() As Integer
    Get
        Return intPassengers
    End Get
    Set(ByVal value As Integer)
        intPassengers = value
    End Set
End Property
```

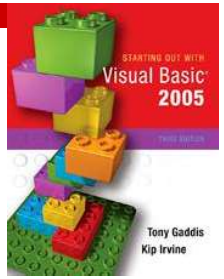
Overridden Derived Class Property



- Overrides keyword and new logic added to derived class property procedure

```
Public Overrides Property Passengers() As Integer
   Get
      Return MyBase.Passengers
   End Get
   Set(ByVal value As Integer)
      If value >= 1 And value <= 2 Then
         MyBase.Passengers = value
      Else
         MessageBox.Show("Passengers must be 1 or 2", _
            "Error")
      End If
   End Set
End Property
```

Overriding Methods



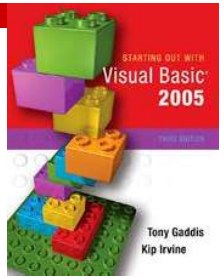
- Overriding a method is similar to a property
- Specify Overridable and Overrides keywords
- An overridable base class method

```
Public Overridable Sub ProcedureName()

Public Overridable Function ProcedureName() As DataType
```

- An overriding derived class method

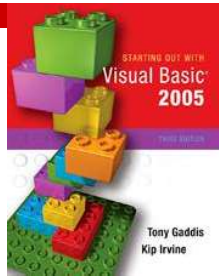
```
Public Overrides Sub ProcedureName()

Public Overrides Function ProcedureName() As DataType
```

Overriding the ToString Method



- Every programmer created class is derived from a built-in class named *Object*
- Object class has a method named ToString which returns a fully-qualified class name
- Method can be overridden to return a string representation of data stored in an object

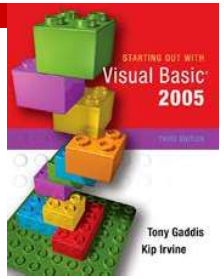
ToString Override Example



- Object class ToString method is Overridable
- Vehicle class might override the ToString method as shown below

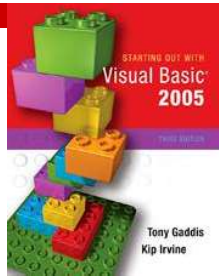
```
' Overriden ToString method
Public Overrides Function ToString() As String
    ' Return a string representation
    ' of a vehicle.
    Dim str As String

    str = "Passengers: " & numPassengers.ToString & _
       " MPG: " & mpg.ToString
    Return str
End Function
```

Base & Derived Class Constructors



- A constructor (named New) may be defined for both the base class and a derived class
- When a new object of the derived class is created, both constructors are executed
  - The constructor of the base class will be called first
  - Then the constructor of the derived class will be called

Protected Members



- In addition to *Private* and *Public*, the access specifier may be *Protected*
  - *Protected* base class members are treated as public to classes derived from this base
  - *Protected* base class members are treated as private to classes not derived from this base

- Tutorial 12-5 provides an opportunity to work with base and derived classes