Chapter 9
Implementing Association Relationships

Objectives
In this chapter, you will:
• Identify association relationships on Bradshaw Marina’s class diagram
• Associate VB .NET classes in a one-to-one relationship
• Add functionality to the Boat class

Objectives
In this chapter, you will:
• Associate Dock and Slips in a one-to-many relationship
• Add the Boat and Customer classes to the Slip example
• Create and use an association class—Lease

Identifying Association Relationships on Bradshaw Marina’s Class Diagram
• Association relationships
  – Show how instances of classes are associated, or connected, to each other
  – Indicate that the system requires information about these associations
  – Can be used to navigate from instance to instance following the association

Identifying Association Relationships on Bradshaw Marina’s Class Diagram
• In a class diagram
  – An association relationship
    • Appears as a line connecting classes
    • Multiplicity of the association
    • Indicated by numbers written at both ends of the line

Identifying Association Relationships on Bradshaw Marina’s Class Diagram
• Association relationships in the Bradshaw class diagram:
  – A customer owns a boat
  – A boat is owned by a customer
  – A boat is assigned to a slip
  – A slip contains a boat
  – A dock contains many slips
  – A slip is attached to a dock
  – A slip is leased to a customer (Lease is an association class)
  – A customer leases a slip (Lease is an association class)
Identifying Association Relationships on Bradshaw Marina’s Class Diagram

**Lease class**
- An example of an association class
  - Exists because of the relationship between a slip and a customer
  - In the class diagram
  - Shown as a class connected by a dashed line to an association between Slip and Customer

**Associating VB .NET Classes in a One-to-One Relationship**
- Association relationship between Customer and Boat
  - Has two “directions”
    - A customer owns a boat
    - A boat is owned by a customer
  - Each direction of the relationship
    - Must be defined separately by the system developer
    - Must be handled separately with VB .NET

**Associating VB .NET Classes in a One-to-One Relationship**
- To implement an association relationship in VB .NET
  - Use a reference variable as an attribute of a class
    - A reference variable points to an actual instance

**Associating VB .NET Classes in a One-to-One Relationship**
- A Customer reference variable
  - Points to a Customer instance
  - If added as an attribute in the Boat class, each Boat instance can
    - Point to a Customer instance
    - Invoke the customer’s methods
Associating VB .NET Classes in a One-to-One Relationship

• A Boat reference variable
  – Points to a Boat instance
  – If added as an attribute in the Customer class, each Customer instance can
    • Point to a Boat instance
    • Invoke the boat’s methods

Modifying the Customer Class

• Modifying the Customer class implements one direction of the association relationship
  – From Customer to Boat
• theBoat attribute
  – A Boat reference variable
  – Added to the Customer class
  – Used to implement a one-to-one association relationship with the Boat class

Modifying the Customer Class

• Accessor methods
  – SetBoat
    • Accepts a Boat reference as a parameter
    • Assigns the reference to the attribute
  – GetBoat
    • Returns the Boat reference

Modifying the Boat Class

• Modifying the Boat class implements the other direction of the association relationship
  – A boat is owned by a customer
• theCustomer attribute
  – A Customer reference variable
  – Added to the Boat class
  – Used to implement a one-to-one association relationship with the Customer class

Modifying the Boat Class

• Accessor methods
  – GetCustomer
    • Returns the Customer reference
  – SetCustomer
    • Accepts a Customer reference as a parameter
    • Assigns the reference to the attribute
Adding Functionality to the Boat Class

- Functionality of classes that have association relationships can be increased
- AssignBoatToCustomer custom method
  - Added in Boat
  - Establishes the association between Boat and Customer in both directions in one step

Adding Functionality to the Boat Class

- Mandatory association between Boat and Customer
  - Possible because
    - Bradshaw Marina does not want to keep information about a boat if its owner is not a customer
  - Done by
    - Adding a Customer reference parameter to the Boat constructor

Adding Functionality to the Boat Class

- Modified Boat TellAboutSelf method
  - Returns information about
    • Boat
    • Customer
  - Possible because
    • A boat must be associated with a customer

Associating Docks and Slips: A One-to-Many Association Relationship

- Association relationship between Slip and Dock
  - Two directions of the relationship
    • Association relationship between Slip and Dock
      - One-to-one
        » A slip is attached to a dock
    • Association relationship between Dock and Slip
      - One-to-many
        » A dock contains many slips

Associating Docks and Slips: A One-to-Many Association Relationship

- To implement
  - A one-to-one association relationship between Slip and Dock
    • Use the technique used to implement the Customer and Boat associations
  - A one-to-many association relationship between Dock and Slip
    • Requires that a dock instance have reference variables for more than one slip
      - Use an ArrayList in the Dock class to hold many Slip reference variables

Associating Docks and Slips: A One-to-Many Association Relationship

- ArrayList class
  - Can be used to instantiate a container that can hold many reference variables
  - Has methods which can be used to
    • Add Slip references
    • Retrieve Slip references
Introducing the Dock Class

- Dock class
  - Four previous attributes
    - An ID number
    - A location
    - Two Boolean variables indicating
      - Whether the dock has electricity
      - Whether the dock has water
  - Fifth attribute
    - slips
      - An ArrayList
      - Implements the one-to-many association relationship

- Dock class constructor
  - Sets values for the four attributes
  - Instantiates the new ArrayList
  - Assigns the new ArrayList to the slips attribute

- GetSlips
  - A getter method
  - Returns the slips reference variable

Introducing the Dock Class

- AddSlip custom method
  - Used to add a slip to the dock
  - Invoked by the Slip AddSlipToDock method
  - Contains the slips.Add(aSlip) statement
    - Used to add a Slip reference to the ArrayList referenced by the variable slips

Associating the Slip Class with Dock

- Modifications to the Slip class to implement the mandatory one-to-one association relationship
  - theDock attribute
    - A Dock reference variable
    - Accessor methods for the Dock reference attribute

Associating the Slip Class with Dock

- Modifications to the Slip class (Continued)
  - Modified constructor
    - Expects a Dock reference parameter
      - Result: When a slip is instantiated, it must be associated with a dock
    - Invokes the slip’s AddSlipToDock method
      - AddSlipToDock method establishes the association in both directions by
        - Invoking the dock’s AddSlip method using the Me keyword

- A TellAboutSelf method
  - Returns information about
    - The slip
    - The slip’s dock
Adding the Boat and Customer Classes to the Slip Example

- Association relationship between Slip and Boat
  - Modifications to the Slip class
    - A Boat reference attribute
    - Accessor methods
      - GetBoat
      - SetBoat
  - Modifications to the Boat class
    - A Slip reference attribute
    - Accessor methods
    - AssignBoatToSlip method

Adding the Boat and Customer Classes to the Slip Example

- To include the Customer class in this example
  - Boat class does not need to be modified
    - It is already designed to associate with a customer
  - Given a Customer reference
    - Can navigate to
      - The customer's boat
      - The boat's slip
      - The slip's dock

Creating and Using an Association Class – Lease

- Lease class
  - An association class
    - A Lease is like an association between a customer and a slip, but with attributes for
      - Start date
      - End date
      - Amount of lease, and so on
  - Subclasses
    - AnnualLease
    - DailyLease

Creating and Using an Association Class – Lease

- No modifications needed for the Boat class
  - Reason: Boat is not directly associated with Lease
- Modifications to the Customer class
  - A Lease reference attribute
  - Accessor methods
Creating and Using an Association Class – Lease

- Modifications to the Slip class
  - A Lease reference attribute
  - Accessor methods
  - LeaseAnnualSlip method
    - Creates an AnnualLease instance

Summary

- Association relationships are shown on the class diagram as lines connecting classes
- Association relationships are implemented in two directions that must be considered separately
- Association relationships can be optional or mandatory
- One-to-one association relationships are implemented by including a reference variable as an attribute in one class that points to an instance of another class

Summary

- One-to-many association relationships are implemented using an ArrayList that contains a collection of reference variables
- Association relationships can be directly navigated by writing one statement with multiple method calls that are executed from left to right
- An association class exists because of a relationship between two classes, such as Lease between Slip and Customer