Chapter 15
Assembling a Three-Tier Windows Application

Objectives
In this chapter, you will:
- Review three-tier design
- Combine a problem domain class, a data access class, and a GUI
- Use multiple GUls and add an instance to a database
- Use GUls with multiple problem domain classes to interact with a database

Reviewing Three-Tier Design
- Three-tier design
  - An architecture for structuring OO applications
  - The three tiers
    - GUI classes
    - Problem domain classes
    - Data access classes
- In three-tier design
  - User interacts with GUI class instances
  - GUI instances interact with problem domain classes and instances
  - Problem domain classes and instances interact with data access classes
  - Data access classes handle storing and retrieving data from files or databases

Reviewing Three-Tier Design
- Benefits of three-tier design
  - System maintenance is simplified
  - Applications can be distributed across the network
  - Three-tier design lends itself to client-server architectures
Three-Tier Design and the Development Process

- OO system development is done iteratively
  - Some analysis, some design, and some implementation occurring before continuing with more analysis, more design, and more implementation

Three-Tier Design and the Development Process

- OO system analysis
  - Define the requirements for the system
    - Use cases
      - Each use case is expanded into multiple scenarios
      - Scenarios define the functions the system provides
    - Problem domain classes
      - Initially defined in terms of
        - Attributes
        - A few key methods
      - Additional details are added during each iteration

Three-Tier Design and the Development Process

- OO system design
  - Add details that specify how the system will be physically implemented
    - Examples
      - User interface design
      - Physical design of the database
  - OO system implementation
    - Write code to define
      - Problem domain classes
      - GUI classes
      - Data access classes

Three-Tier Design and the Development Process

- Combining a PD class, a DA class, and a GUI
  - First three-tier example
    - Involves
      - Customer problem domain class from Chapter 13
      - Customer data access class from Chapter 13
      - FindCustomer GUI from Chapter 11
        - FindCustomer GUI retrieves information about a specific customer

Three-Tier Design and the Development Process

- Combining a PD class, a DA class, and a GUI
  - Involves
    - Customer problem domain class from Chapter 13
    - FindCustomer GUI from Chapter 11
  - Example
    - FindCustomer GUI retrieves information about a specific customer
Reviewing the Customer Problem Domain Class

- Customer problem domain class from Chapter 13
  - Used by this example
  - Includes
    - Four class methods
      - Initialize
      - Terminate
      - GetAll
      - Find
    - Three instance methods
      - AddNew
      - Update
      - Delete

Reviewing the Customer Data Access Class

- CustomerDA class
  - Implements the details associated with the data access methods in the Customer class
  - Throws exceptions if
    - A new customer already exists
    - A customer that is to be deleted or updated cannot be found

Updating the FindCustomer GUI

- FindCustomer GUI
  - Introduced in Chapter 11, Example 4
  - Used for the final tier of this example
  - Requires only a few minor modifications to work with the Customer and CustomerDA classes
Updating the FindCustomer GUI

• Required modifications (continued)
  – btnUpdate_Click method
    • Invokes the Update method of the Customer class
      – To update customer information in the database
    • The Enabled property of the CustomerPhone text box is set to False
      – Reason: Customer phone number is the primary key for the customer table

Updating the FindCustomer GUI

• Required modifications (continued)
  – btnClose_Click method
    • Terminates the connection to the database
      – Reason: This example does not include a main menu GUI

Using Multiple GUls and Adding an Instance to the Database

• Second three-tier example
  – Uses multiple GUls
  – Adds an instance to the database using a data access class
    – Based on Example 4 in Chapter 11
    – Contains a main menu, which allows the user to
      • Find a customer
      • Add a new customer

Using Multiple GUls and Adding an Instance to the Database

• Classes involved
  – FindCustomer GUI from previous example in this chapter
    • Modified slightly to work with the main menu
  – AddCustomer GUI from Chapter 11
    • Modified slightly to work with the data access classes
  – MainMenu GUI from Chapter 11
    • Modified slightly to work with the data access classes

• Classes involved (continued)
  – Custom exceptions
    • NotFoundException
    • DuplicateException
  – Customer class defined in the previous example
    • Does not have to be modified
  – CustomerDA class defined in the previous example
    • Does not have to be modified
Using Multiple GUIs and Adding an Instance to the Database

Reviewing the MainMenu GUI

- MainMenu GUI class of Chapter 11
  - Only two minor changes are needed for this example
    - Invokes the Initialize method of the Customer class
      - To establish the connection to the database
    - Invokes the Terminate method of the Customer class
      - To terminate the connection to the database

Reviewing the AddCustomer GUI

- AddCustomer GUI
  - First introduced in Chapter 11
  - Modifications for this example
    - btnAdd_Click method
      - Invokes the AddNew method of the Customer class
        (rather than the CustomerData class)
        - Allows AddCustomer to work with data access classes

Updating the FindCustomer GUI

- FindCustomer GUI class used in the previous example
  - Modifications required for this example
    - The statement that invokes the Initialize method is removed
      - In this example, the MainMenu GUI establishes the connection to the database
    - The statement that invokes the Terminate method is removed
      - In this example, the MainMenu GUI terminates the connection to the database

Using GUIs with Multiple Problem Domain Classes

- Third three-tier example
  - Combines multiple GUIs with multiple problem domain and data access classes
  - Contains
    - GUIs seen in this chapter and in Chapter 11
      - MainMenu
      - FindCustomer
      - AddCustomer
      - AddBoat

Using GUIs with Multiple Problem Domain Classes

- Third three-tier example (continued)
  - Contains
    - Classes from Chapter 14
      - Customer
      - CustomerDA
      - Boat
      - BoatDA
      - CustomerAndBoatDatabaseConnect
Using GUIs with Multiple Problem Domain Classes

Reviewing the Customer and Boat PD and DA Classes From Chapter 14

- This example uses the chapter 14 versions of the following classes without modifications
  - Customer problem domain class
  - Customer data access class
  - Boat problem domain class
  - Boat data access class
  - CustomerAndBoatDatabaseConnect class

Reviewing the Customer and Boat PD and DA Classes From Chapter 14

- Association relationship between Customer and Boat
  - Multiplicity: one-to-one in both directions
  - Established by including
    - A Boat reference variable in Customer
    - A Customer reference variable in Boat
    - Accessor methods for the Boat and Customer references

Reviewing the Customer and Boat PD and DA Classes From Chapter 14

- CustomerAndBoatDatabase.mdb from Chapter 14
  - A relational database
  - Includes two tables:
    - CustomerTable
    - BoatTable
      - Foreign key: CustomerPhoneNo
        - Allows the two tables to be joined
      - Used without modifications in this example

Modifying the MainMenu GUI

- MainMenu GUI
  - Opens and closes the connection to the database
  - Constructor
    - Invokes the Initialize method of the CustomerAndBoatDatabaseConnect class
      - To establish the common connection
    - Shares this connection by passing it to the Initialize methods of
      - Customer class
      - Boat class

Modifying the MainMenu GUI

- MainMenu GUI (continued)
  - btnClose_Click procedure
    - Closes the connection to the database
Modifying the FindCustomer GUI

- FindCustomer GUI from the previous example
  - Modifications required for this example
    - A Boat reference variable is included in the class definition
      - Reason: FindCustomer must now recognize the association between Customer and Boat
    - lblBoatInfo label is added to the btnFind_Click method
      - lblBoatInfo label holds information about the customer’s boat

Reviewing the AddCustomer GUI from Chapter 11

- AddCustomer GUI presented in Chapter 11
  - Used without modifications in this example
  - Validates customer information entered by the user
  - Creates a corresponding instance of Customer
  - Passes the newly created Customer instance to the AddBoat GUI as an argument
    - AddBoat GUI
      - Captures information about the customer’s boat
      - Adds the customer information to CustomerTable
      - Adds the boat information to BoatTable

Modifying the AddBoat GUI

- AddBoat GUI from chapter 11
  - Constructor: receives a Customer reference variable in the parameter list
  - Validates the boat information entered by the user
  - Creates a corresponding instance of Boat
  - Establishes the two-way association between the customer and boat instances

Modifying the AddBoat GUI

- In this three-tier example, AddBoat must now
  - Add the boat record to BoatTable
  - Add the associated customer record to CustomerTable

Modifying the AddBoat GUI

- Modifications needed for this example
  - AddSailboat method
    - Creates an instance of the Boat class, rather than an instance of Sailboat
    - Invokes the AssignBoatToCustomer method of the Boat class
  - AddPowerboat method
    - Creates an instance of the Boat class, rather than an instance of Powerboat
    - Invokes the AssignBoatToCustomer method of the Boat class

Summary

- Three-tier design is a strategy for creating OO applications that are easy to maintain
- Three-tier design provides a framework that lends itself to iterative development
- In iterative development, some analysis, some design, and some implementation are completed and then the process repeats with more analysis, more design, and more implementation
Summary

• First example: demonstrated one GUI, one PD class, one DA class, and a database with one table
• Second example: used multiple GUIs (including MainMenu, AddCustomer, and FindCustomer), one PD class and one DA class
• Third example: used multiple GUIs, two PD classes with an association relationship, two DA classes, and a database with two tables