INNOSOFT GULF
Leading Edge IT Training

DUBAI KNOWLEDGE VILLAGE

Java

Developing Java SE Applications

Duration

5 days

Course Description

The Developing Applications with the Java SE Platform course provides students with practical experience in designing a vertical solution for a distributed, multi-tier application. The course takes students through the process of designing a multi-tier application in a case study approach – requirements gathering, analysis and design, and development of the key components of the application. Students will design the application with a Model-View-Controller (MVC) pattern, implement testing with JUnit, create a Graphical User Interface (GUI) that supports logging, implement database connections with JDBC, create both client and server components, implement threading to enabling scaling of your application and use Remote Method Invocation (RMI) to communicate between objects on your client and server components.

The course features the Java Platform, Standard Edition 6 (Java SE 6) technology and utilizes the Java SE Development Kit 6 (JDK 6) product. The students perform the course lab exercises using the NetBeans Integrated Development Environment (IDE).

Course Topics

Introduce the BrokerTool Application

- Explain the problem statement of the BrokerTool application
- Creating and populating the StockMarket Database
- Executing SQL Statements on the StockMarket Database

Apply the Model View Controller (MVC) Design Pattern

- Explain design patterns
- Explain the MVC design pattern
- Analyze how the MVC design pattern can be used in applications
- Add MVC Interaction Code

Implement Unit Testing

- Develop unit test cases using JUnit
- Execute Unit test cases
- Open the InfoTool Project
- Prepare JUnit Test Cases for the InfoTool Project
- Analyze the JUnit Test Cases of the InfoController class of the InfoTool Project
- Create and Analyze Test Methods Inside InfoToolTest.java File
- Create a TestSuite of all the Test Cases of the InfoTool Project

Design the BrokerTool Application

- Apply the MVC design pattern
- Begin the analysis and design of the project under study
- Develop a build plan for the project
- Create the MVC Participants
- Establish the BrokerTool MVC Baseline

Implement the Java Database Connectivity (JDBC) API

- Describe the JDBC API
- Explain how using the abstraction layer provided by the JDBC API makes a database front end portable across platforms
- Describe the five major tasks involved with the JDBC programmer's interface
- State the requirements of a JDBC driver and its relationship to the JDBC driver manager
- Describe the data access objects (DAO) pattern and its applicability to a given scenario
- Identify the Workflow and Object Interactions
- Implement a Database-Connected Broker Model by Using the DAO Pattern

Create Graphical User Interfaces (GUI)

- Apply the principles of good GUI design
- Design and implement a GUI for the project using Matisse
- Apply the Composite Design pattern to build the BrokerTool GUI
- Use JTable and JTabbedPane classes in your application to build a sophisticated GUI
- Add AllCustomerTablePanel to the Palette Window and drag-and-drop to the BrokerGui Class

- Create the CustomerPanel Class, add to the Palette Window and drag-and-drop to the BrokerGui Class
- Change the Order of the Tabs
- Compile and Test the BrokerGui Class

Handle GUI Events

- Implement a view class
- Implement a controller class
- Create the BrokerTool view Class
- Create the BrokerTool Controller Class
- Compile and Testing the BrokerGui Class
- Add Event Handling Functionality

Log Messages in GUI

- Use the logging API
- Examine a logging example
- Write a custom handler
- Set filters to a particular handler
- Create the Custom Handler Class

Implement Multiple-Tier Design

- Compare the BrokerTool two-tier design with the three-tier design for the same application
- Explain how you can use the Java technology package, java.net to implement networking applications
- Demonstrate how to use the Command design pattern in the application
- Apply the Strategy design pattern to create reusable code
- Describe how you can implement the network client
- Describe how you can implement the network server

Implement Advanced Multiple-Tier Design

- Use the new Java concurrency APIs to create a multithreaded server
- Examine a thread pool
- Identify integrity problems in multithreaded servers
- Create a Generic Network Client Class

Communicate With Remote Objects Using Java RMI

- Create remote objects
- Use Java RMI to create a multi-tier application
- Deploy a Java RMI Implementation of the BrokerModel Interface
- Create a Java RMI Implementation of the BrokerView Interface