Developing Applications With the Java SE Platform

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. Students use graphical user interface (GUI) design principles and network communications capabilities to code a functional Java application that interacts with a networked database server. The blended approach of designing and developing programs for applications has been clearly emphasized in this course. New features that blend with the special IDE GUI building capabilities, such as Matisse, are covered. The Java Management Extensions (JMX) and the JUnit tool are also covered. 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) 5.5.

Duration: 5 Days

Prerequisites

To succeed fully in this course, students should be able to:

- Develop applications by using the Java programming language
- Understand basic Unified Modeling Language (UML) diagrams
- Understand basic Structured Query Language (SQL) statements
- Understand how to implement interfaces and handle Java programming exceptions
- Use object-oriented programming techniques
- Understand GUI design
- Understand basic Transmission Control Protocol/Internet Protocol (TCP/IP) communication
- Program with sockets or streams
- Understand the monitoring framework that is provided by Java
- Develop testing methodologies and test procedures

Skills Gained

Upon completion of this course, students should be able to:

- Apply Model View Controller (MVC) design pattern to create reusable classes
- Implement unit testing using JUnit

- Implement a program from the ground up that could be used in a commercial intranet application
- Develop classes to connect programs to Structured Query Language (SQL) database systems using the core aspects of the Java Database Connectivity (JDBC) application programming interface (API)
- Organize and set up the GUI generation and event handling to support a Java technology project
- Implement the Logging API to generate log messages in GUI
- Create two-tier and three-tier Java technology applications
- Create a multithreaded server
- Create remote objects using Java Remote Method Invocation (Java RMI)

Course Content

Module 1 - 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

Module 2 - 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

Module 3 - Implement Unit Testing

- Develop unit testcases using JUnit
- Execute Unit testcases
- 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

Module 4 - 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

Module 5 - 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

Module 6 - 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

Module 7 - 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

Module 8 - 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

Module 9 - 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

Module 10 - 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

Module 11 - 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