Effectively Using Java Packages And Features

Duration: 35 hours

Prerequisites: Java programming experience and an understanding of object-oriented design principles. The course Java Programming or equivalent knowledge provides a solid foundation.

Course Description: This intermediate level course is intended for programmers who already have a fundamental understanding of Java programming and some experience writing code. It provides additional insights and details regarding some of the more advanced and useful capabilities contained in the Java Programming Language and it's associated packages. Topics include reflection and JavaBeans, Java type safety enhancements, the Java Collections Framework, Java Database Connectivity (JDBC), multithreading, inner classes, lambda expressions and networking.

Comprehensive hands on exercises are integrated throughout to reinforce learning and develop real competency.

Students Will Learn

- Reflection Using the Class Object
- Invoking Methods Using Reflection
- Methods Available on the Class Object
- Customizing BeanInfo
- Type Safety Enhancements
- Collections Framework
- Simple Arrays and Arrays of Objects
- Legacy Container Classes
- Data Sharing Among Threads
- Coordination and Controlling Threads
- Synchronizing Threads
- Working with JDBC
- Overview of java.net Package
- Inner and Nested Classes
- Lambda Expressions

Overview

JavaBeans, Reflection, and Introspection

- JavaBean Requirements
- Determining Type of an Object
- Reflection Overview and Uses
- Reflection Issues
- Reflection Using the Class Object
- Invoking Methods Using Reflection
- Methods Available on the Class Object
- Creating a New Instance
- Introspection
- Customizing `BeanInfo`

Type Safety Enhancements

- Annotations
  - Standard Annotations
  - User Defined Annotations
  - Reflection Annotation Information
- The `enum` Data Type
- Generics
- Autoboxing
- Methods Having Variable Parameters
- Assertions

Invoking Methods Using Introspection
- Using Reflection to Access Properties
- Review of `equals` Method

Collections Framework

- What is the Collections Framework
- Simple Arrays and Arrays of Objects
- Legacy Container Classes
- Collections Framework Overview
- Collections Interfaces
  - Lists
  - Sets
  - Queues
- Map Interfaces
- Interface Implementations
  - Lists
  - Sets
  - Queues
  - Maps
- Iterators

JDBC

- What is JDBC
- JDBC Drivers
- Accessing the Database
  - Loading Driver
  - Connecting to Data Source
  - Creating Statements
  - Executing Statements
- Processing Result Sets
  - Cursor Positioning
  - Column Retrieval
  - Updating Result Sets
- Connection Pooling
- Processing Errors and Warnings
- Using Prepared Statements
- Using Stored Procedures
- Metadata
  - Result Set Metadata
  - Database Metadata
- Transaction Processing
- Isolation Levels
- SQL Batches

Threads

- Definition of a Thread
- Creating Threads
- Naming Threads
- Data Sharing Among Threads
  - Local Data
  - Instance Data
  - Class Data
  - `volatile` Keyword
- Thread States
- Thread Priority
  - Minimum. Maximum, Normal Priorities
  - Preemptive Thread Priority
  - `setPriority` Method
  - `getPriority` Method
- Piping Data Between Threads
- Coordination and Controlling Threads
- Synchronizing Threads
  - Why Synchronization is Needed
  - Producer/Consumer Example
  - `synchronized` Keyword
  - Thread Synchronization Methods

Networking

- Overview of `java.net` Package
- Format of URL
- `URL` Class
- `URLConnection` Class
- `InetAddress` Class
- Definitions
  - Client
  - Server
  - Port
  - Socket
- TCP/IP Protocols
- Socket Class
  - Writing Client Side Applications
  - Read From Socket
  - Write to Socket
- `ServerSocket` Class
  - Writing Server Side Applications
  - Daemon Servers
  - Multi-Threaded Servers

Inner and Nested Classes

- What Are Inner Classes
  - Benefits of Inner Classes
  - Terminology
  - Restrictions
  - Syntax
- Types of Inner Classes
  - Member Inner Class
  - Local Inner Class
  - Anonymous Inner Class
  - Nested Top Level Class
- Lambda Expressions
  - Functional Interfaces
  - Default Interface Methods
  - Static Interfaces Methods
  - Lambda Expression Uses
  - Lambda Expression Syntax
  - Method and Constructor References

Related Bootcamp

Track	Duration	Price
Advanced Java Developer	4-course track 5-course track 6-course track 7-course track 8-course track	$4,800 $6,000 $7,200 $8,400 $9,600

Contact Us

Address: 1 Village Square, Suite 3 Chelmsford, MA 01824

Phone: 978.250.4983

Mon - Thur: 9 am - 5 pm EST
Fri: 9 am - 4 pm EST

E-mail: info@developer-bootcamp.com