1

Introduction

Course Objectives

After completing this course, you should be able to:

- Create Java technology applications that leverage the object-oriented features of the Java language, such as encapsulation, inheritance, and polymorphism
- Create applications that use the Collections framework
- Search and filter collections using lambda expressions
- Implement error-handling techniques using exception handling
- Describe Java's new module system
- Identify and address common requirements in migrating older applications to modularity

Course Objectives

- Use Lambda Expression concurrency features
- Implement input/output (I/O) functionality to read from and write to data and text files and understand advanced I/O streams
- Manipulate files, directories, and file systems using the NIO.2 specification
- Understand and apply Java secure guidelines
- Perform multiple operations on database tables, including creating, reading, updating, and deleting, using the JDBC API

Introductions

Meet your classmates and briefly introduce yourself:

- Name
- Title or position
- Company
- Experience with Java programming and Java applications
- Reasons for attending

Audience

The target audience includes those who have:

- Completed the *Java SE Programming I* course or have experience with the Java language and can create, compile, and execute programs
- Experience with at least one programming language
- An understanding of object-oriented principles
- Experience with basic database concepts and a basic knowledge of SQL

Prerequisites

To successfully complete this course, you must know how to:

- Develop applications using the Java programming language
- Use object-oriented programming techniques
- Use primitives and classes commonly found in Java programs, such as arrays, collections, and streams
- Perform basic operating system administration from the command line

Course Roadmap

- **Unit 1: Fast-Track to Object-oriented Programming**
  - **Lesson 1: Course Introduction**
  - **Lesson 2: Java OOP Review**
  - **Lesson 3: Exception Handling and Assertions**
  - **Lesson 4: Java Interfaces**
  - **Lesson 5: Collections and Generics**
- Unit 2:Functional Programming
- Unit 3:Modular Programming
- Unit 4:Streams and Parallel Streams
- Unit 5: Java API Programming and Secure Coding Concepts

Course Roadmap

- Unit 1: Fast-Track to Object-oriented Programming
- **Unit 2:Functional Programming**
  - **Lesson 6: Functional Interface and Lambda Expressions**
  - **Lesson 7: Collections, Streams, and Filters**
  - **Lesson 8: Lambda Built-in Functional Interfaces**
  - **Lesson 9: More Lambda Expressions**
- Unit 3:Modular Programming
- Unit 4:Streams and Parallel Streams
- Unit 5: Java API Programming and Secure Coding Concepts

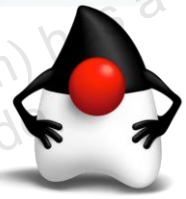
Course Roadmap

- Unit 1: Fast-Track to Object-oriented Programming
- Unit 2:Functional Programming
- **Unit 3:Modular Programming**
  - **Lesson I0: The Modules System**
  - **Lesson 11: Migrating to a Modular Application**
  - **Lesson 12: Services in a Modular Application**
- Unit 4:Streams and Parallel Streams
- Unit 5: Java API Programming and Secure Coding Concepts

Course Roadmap

- Unit 1: Fast-Track to Object-oriented Programming
- Unit 2:Functional Programming
- Unit 3:Modular Programming
- **Unit 4:Streams and Parallel Streams**
- Unit 5: Java API Programming and Secure Coding Concepts

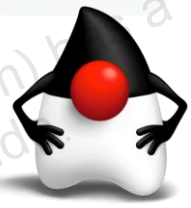
- **Lesson I3: Concurrency**
- **Lesson 14: Parallel Streams**
- **Lesson 15: Terminal Operations: Collectors**
- **Lesson 16: Creating Custom Streams**



Course Roadmap

- Unit 1: Fast-Track to Object-oriented Programming
- Unit 2:Functional Programming
- Unit 3:Modular Programming
- Unit 4:Streams and Parallel Streams
- **Unit 5: Java API Programming and Secure Coding Concepts**

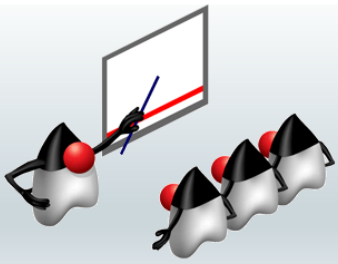
- **Lesson I7: I/O Fundamentals and NIO2**
- **Lesson 18: Java Secure Coding Concepts**
- **Lesson 19: JDBC**
- **Lesson 20: Localization**



Lesson Format

Lecture / Student Guide (50%)

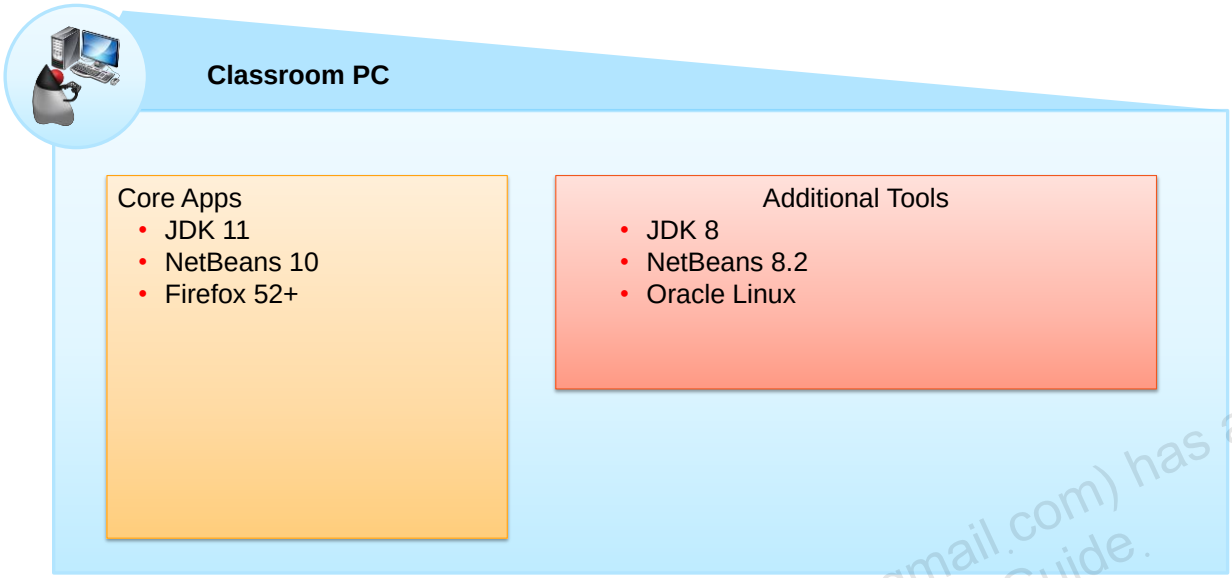
- Traditional slides
- Sample Code
- Demos
- Short quizzes



Practices / Activity Guide (50%)

- Hands-on learning
- Work with Java code
- Intended for the OU Practice Environment

Practice Environment



The following products are preinstalled for the lesson practices:

- **JDK 11**
  - **JLink**
  - **JShell**
  - **JDeprscan**
- **Firefox 52+**
- **NetBeans 10 rc3**.
- **Oracle Linux** is Oracle's enterprise implementation of Linux. It's compatible with RedHat Linux.

How Do You Learn More After the Course?

- In the Oracle Learning Library, there is a list of resources that you can use to learn more about Java programming. Look for the collection on the oracle.com/oll/java page.
- *Oracle Learning Library:*
  - http://www.oracle.com/goto/oll

Additional Resources

Resource	Website
Education and Training	http://education.oracle.com
Product Documentation	http://www.oracle.com/technology/documentation
Product Downloads	http://www.oracle.com/technology/software
Product Articles	http://www.oracle.com/technology/pub/articles
Product Support	http://www.oracle.com/support
Product Forums	http://forums.oracle.com
Product Tutorials	http://www.oracle.com/technology/obe
Sample Code	http://www.oracle.com/technology/sample_code

The table in the slide lists web resources where you can obtain additional information about Java.

Additional Resources

Resource	Website
Java Documentation	https://docs.oracle.com/javase
API Documentation	https://docs.oracle.com/javase/10/docs/api/index.html

The table in the slide lists web resources where you can obtain additional information about Java.

Summary

In this lesson, you reviewed the course objectives and the tentative class schedule. You met your fellow students, and you saw an overview of the computer environment that you will use during the course.

Enjoy the next five days of *Java SE Programming II*

Practice 1: Overview

This practice covers the following topics:

- 1-1: Logging in to Oracle Linux
- 1-2: Opening Terminal Windows in Oracle Linux
- 1-3: Verifying the Version of Java
- 1-4: Opening a Text File in Oracle Linux
- 1-5: Starting NetBeans and Opening a Project