Object-Oriented Analysis and Design Using UML

Duration: 5 Days

What you will learn

This Object-Oriented Analysis and Design Using UML training teaches you how to effectively use object-oriented technologies and software modeling as applied to a software development process. Expert Oracle University instructors present one practical, complete, object-oriented analysis and design (OOAD) road map from requirements gathering to system design.

Learn To:

- Use object-oriented technologies.
- Use Unified Modeling Language 2.2.
- Perform object-oriented analysis and design.
- Follow a software development process using an OO software project.
- Create a system design (the Solution model) supporting the functional requirements (FRs).

Benefits to You

By enrolling in this course, you'll experience the benefits of using the widely adopted graphical modeling language—the Unified Modeling Language (UML) version 2.2. Use this to help communicate concepts and decisions, understand the problem and proposed solution and manage complexity of artifacts describing the problem and proposed solution. Furthermore, you'll develop a deeper understanding of the patterns and frameworks that help build more flexible and re-usable software components.

Course Structure

This course is structured to follow a generic form of software development process that focuses on the analysis and design aspects as applicable to an OO software project. This generic process can be easily adapted to specific processes, which are discussed later in the course. Expert Oracle University instructors will present a pragmatic approach to object-oriented (OO) software development following proven OO technologies, principles and patterns as applicable to OO languages like the Java(TM) programming language.

People Who Will Benefit from this Course

System architects, software engineers, systems analysts and designers responsible for the conception and creation of object-oriented software applications will find the most value in taking this course. Architects responsible for the conception and creation of object-oriented software applications can also benefit from this educational investment.
Related Training

Required Prerequisites

Understand object-oriented concepts and methodology

Demonstrate a general understanding of programming, preferably using the Java programming language

Understand the fundamentals of the systems development process

Course Objectives

Describe the object-oriented software development process

including object-oriented methodologies and workflows

Gather system requirements through interviews with stakeholders

Analyze system requirements to determine the use cases and domain model of the problem domain (the Requirements model)

Create a system architecture (the Architecture model) supporting the nonfunctional requirements (NFRs) and development constraints

Create a system design (the Solution model) supporting the functional requirements (FRs)

Course Topics

Examining Object-Oriented Concepts and Terminology
Describe the important object-oriented (OO) concepts
Describe the fundamental OO terminology

Introducing Modeling and the Software Development Process
Describe the Object-Oriented Software Development (OOSD) process
Describe how modeling supports the OOSD process
Describe the benefits of modeling software
Explain the purpose, activities, and artifacts of the following OOSD workflows (disciplines): Requirements Gathering, Req
Justify the need for a Use Case diagram
Identify and describe the essential elements in a UML Use Case diagram
Develop a Use Case diagram for a software system based on the goals of the business owner
Develop elaborated Use Case diagrams based on the goals of all the stakeholders
Recognize and document use case dependencies using UML notation for extends, includes, and generalization
Describe how to manage the complexity of Use Case diagrams by creating UML packaged views

**Creating Use Case Scenarios and Forms**
Identify and document scenarios for a use case
Create a Use Case form describing a summary of the scenarios in the main and alternate flows
Describe how to reference included and extending use cases.
Identify and document non-functional requirements (NFRs), business rules, risks, and priorities for a use case
Identify the purpose of a Supplementary Specification Document

**Creating Activity Diagrams**
Identify the essential elements in an Activity diagram
Model a Use Case flow of events using an Activity diagram

**Determining the Key Abstractions**
Identify a set of candidate key abstractions
Identify the key abstractions using CRC analysis

**Constructing the Problem Domain Model**
Identify the essential elements in a UML Class diagram
Construct a Domain model using a Class diagram
Identify the essential elements in a UML Object diagram
Validate the Domain model with one or more Object diagrams

**Transitioning from Analysis to Design using Interaction Diagrams**
Explain the purpose and elements of the Design model
Identify the essential elements of a UML Communication diagram
Create a Communication diagram view of the Design model
Identify the essential elements of a UML Sequence diagram
Create a Sequence diagram view of the Design model

**Modeling Object State Using State Machine Diagrams**
Model object state
Describe the essential elements of a UML State Machine diagram

**Applying Design Patterns to the Design Model**
Define the essential elements of a software pattern
Describe the Composite pattern
Describe the Strategy pattern
Describe the Observer pattern
Describe the Abstract Factory pattern

**Introducing Architectural Concepts and Diagrams**
Distinguish between architecture and design
Describe tiers, layers, and systemic qualities
Describe the Architecture workflow
Describe the diagrams of the key architecture views
Select the Architecture type
Create the Architecture workflow artifacts

**Introducing the Architectural Tiers**
Describe the concepts of the Client and Presentation tiers
Describe the concepts of the Business tier
Describe the concepts of the Resource and Integration tiers
Describe the concepts of the Solution model

**Refining the Class Design Model**
Refine the attributes of the Domain model
Refine the relationships of the Domain model
Refine the methods of the Domain model
Declare the constructors of the Domain model
Annotate method behavior
Create components with interfaces

**Overview of Software Development Processes**
Explain the best practices for OOSD methodologies
Describe the features of several common methodologies
Choose a methodology that best suits your project
Develop an iteration plan

**Overview of Frameworks**
Define a framework
Describe the advantages and disadvantages of using frameworks
Identify several common frameworks
Understand the concept of creating your own business domain frameworks

**Course Review**
Review the key features of object orientation
Review the key UML diagrams
Review the Requirements Analysis (Analysis) and Design workflows