

Object-Oriented Concepts and Design in UML

Duration

2 days

Instructor

Sam Rostam

Class Limit

20 students

Prerequisite

None

Price

On-site:

Please contact SPC for pricing (contact information on page 2)

Public Training:

\$995 (2 days)

*Discount available for early registration

Materials Provided

- Student manual containing the course slides
- Student handouts with class exercises and class studies

Object Orientation has become the predominant paradigm for virtually all modern software development. In this course, participants learn OO concepts and how Unified Modeling Language (UML) is used to represent Objects, Classes, Components, Relationships, Architectural Design and the supporting diagrams.

This seminar will provide the participants with a rich understanding of Object-Orientation stressing the strengths of Object-Oriented approach. An overview of Software Engineering best practices and some of the most popular Design Patterns will be discussed as well.

In this seminar participants will learn to:

- List major phases and workflows of the iterative, incremental lifecycle of projects
- Describe OO concepts such as Abstraction, Encapsulation, Inheritance and Polymorphism
- Describe Strengths of Object Oriented approach
- Understand the value of OO in implementing and maintaining large software systems
- Define class behaviors, using polymorphism and other design techniques
- Design details of class attributes, operations and relationships
- Describe some of the Object-Oriented Design Patterns
- Identify various design patterns in a given problem
- Analyze and Create Static Models using UML
- Identify and Analyze Dynamic Models using UML

Intended Audience

This seminar is intended for software developers, programmers and analysts who are familiar with and experienced in software development methodologies who will be using object orientation and UML in upcoming projects.



TRAINING

Object Oriented Concepts and Design in UML

Instructor

Sam is Sun certified as a Java Enterprise Instructor and Scalable Internet Architect and Trainer for Forte. He has taught courses at Fortune 1000 companies across North America. Most recently as a Senior Consultant at Sun Microsystems Palo Alto CA, he focused on Internet, eBusiness & Enterprise Systems Architecture in Java and Forte. Engagements included projects at Motorola, TransCanada Pipeline, Bank of America, HP, US West, Airborne Express, US Air force, GM and Applied Materials.

Sam holds certifications on adult learning and effective teaching techniques as well as Sun/iPlanet certifications on eBusiness infrastructure and systems integration using Java, LDAP and Portal. He holds an MSc from SFU and studied in a PhD program at UBC.

With over 12 years experience as a trainer, an educator, software engineer and an enterprise systems consultant, Sam brings a unique perspective for effective learning. His teaching experience includes BCIT, SFU, TechBC and UBC.

For more information on this or other SPC Springboard courses, please visit www.spcspringboard.com or e-mail SPC at info@spc.ca

Software Productivity Center
Suite 460—1122 Mainland Street
Vancouver, BC V8M 4T8
www.spc.ca

Toll Free: 1.877.548.1948 Fax: 604.689.0141

Vancouver: 604.662.8181 Toronto: 416.885.0512

Outline

Best practices of Software Engineering

- Why Develop software iteratively
- Advantages of Component-based architecture
- Visual modeling
- Verify software quality
- Control changes

Introduction to Unified Modeling Language (UML)

- Overview of UML
- Why use UML
- Static and Dynamic
- Modeling with UML

Introduction to Object Orientation

- Basic Principles of Object Orientation
- Basic Concepts of Object Orientation
- Strengths of Object Orientation

Object Oriented Languages

- Bridging Modular and Procedural Languages to OO languages
- Modern trends in OO languages

Objects and Large Software Systems

- Major challenges
- Object Orientation to rescue

Requirements Overview

- Requirements Artifact
- Concepts in Use-Case Modeling

Analysis and Design Overview

- Analysis versus Design
- Use-case realization
- Workers and their Responsibilities

Use-Case Analysis

- Find Classes from Use-Cases Behavior
- Describe Responsibilities
- Describe Attributed and Associations
- Finding Relationships

Architectural Analysis & Design

- Identify Key Concepts
- Architectural and Design Patterns



TRAINING

