SpiralTrain
developer courses

Object Oriented Analysis and Design

Audience Course Object Oriented Analysis and Design
The course Object Oriented Analysis and Design is intended for developers and architects who want to learn object oriented analysis and design techniques and UML to design systems.

Prerequisites Course Object Oriented Analysis and Design
To join the course Object Oriented Analysis and Design knowledge of the basic principles of object orientation is required and experience in object oriented software development is desirable.

Realization Training Object Oriented Analysis and Design
The subject matter is treated on the basis of presentation slides. During the course two case studies are developed from requirements to design. A modern UML tool is used to draw UML diagrams in it. The course material is in English. The course times are from 9.30 up and to 16.30.

Certification Object Oriented Analysis and Design
Participants receive an official certificate Object Oriented Analysis and Design after successful completion of the course.

Duration: 5 days   **Price: € 2999**

Open Schedule

Object Oriented
Analysis and Design

Content Course Object Oriented Analysis and Design

In the course Object Oriented Analysis and Design you will learn the object oriented ways of thinking and techniques to analyze, design and model a software system as a collection of cooperating objects. The UML language runs as a central thread through the course.

Iterative and Incremental Development
After an introduction and review of the key object oriented concepts and principles, the modern system development principle of iterative and incremental development is discussed.

Requirements Gathering and Uses Cases
Next attention is paid to how the requirements of a system can be analyzed and how the typical forms of system use can be described with uses cases.

Domain Modeling
After an overview of UML, it is discussed how a domain model can be established, how the various objects can be distinguished together with their attributes and relationships, and what information they exchange.

Interaction Modeling
Attention is paid to how responsibilities can be assigned to objects and how these can be translated and made visible with interaction modeling using sequence and collaboration diagrams and state charts. The various patterns that can be used in this process are also discussed.

Packages and Subsystems
Part of the subject matter is also how the translation of the analysis model to a design class model can take place, including the design of a logical architecture with packages and subsystems and the mapping to code.

Architectural Design
The course also considers aspects of architectural design that are dealt with using component and deployment diagrams.

Design Patterns
Finally the focus is on the importance of design patterns to implement standard solutions.

Modules Course Object Oriented Analysis and Design

Module 1 : Software Process	Module 2 : Requirements Analysis	Module 3 : Use Case Modeling
Software Development Process	Understanding Requirements	Use Cases and Actors
Software Development Phases	Vision Documents	Identifying Actors
Good Software Characteristics	Requirement Analysis Activities	System Context Diagram
Iterative and Incremental Development	Requirement Types	Identifying Use Cases
Requirements Capturing	Functional Requirements	Use Case Diagram
Requirements Analysis Process	Non-Functional Requirements	Use Case Modeling Steps
System Design	Requirements Determination	High Level Use Cases
Test Driven Development	Requirements Classification	Alternative Paths
Waterfall Model	Conflicting Requirements	Scenarios
Evolutionary Development	Requirements Risks	Generalizations
Unified Process	The glossary	include and extends

Module 4 : UML Overview	Module 5 : Domain Modeling	Module 6 : Use Case Realization
What is UML?	Why Domain Modeling?	Realizing Requirements
UML Diagrams	Conceptual Classes	System Behavior
Use Case View	Noun Identification	Input Events and Operations
Logical View	Physical and Conceptual Objects	System Sequence Diagrams
Component View	Types of Classes	Derivation from Use Case
Deployment View	Domain Analysis Classes	Postconditions
Notes and Adornments	Finding Associations	Class Responsibilities
Stereotypes	Multiplicity and Associations	Class Collaborations
Tagged Values	Generalization and Specialization	Interaction Diagrams
Constraints	Aggregation and Composition	Sequence Diagrams
System Sequence Diagrams	Finding Attributes	Grasps Design Patterns

Module 7 : Statecharts	Module 8 : Design Model	Module 9 : Architectural Design
State Machine Concepts	Transition to Design	System Partitioning
State Machine Diagram	From Requirements to Design	Large Scale Element Collaboration
Event Driven Behavior	Class Design Diagrams	Layers and Packages
State Machines and Objects	The Design Model	Simple Logical Architecture
Object Behavior	Design Aspects	Consider Coordination Layer
Objects and Threads	Design Model Characteristics	Web Application Architecture
Passive and Active Objects	Mapping to Code	Consider MVC Architecture
Entry and Exit Actions	Packages	Package Dependencies
Internal Transitions	Package Design	Clustering
State Activities	Packaging Guidelines	Vertical Scaling
Guards	Data Access Class Packages	Horizontal Scaling
History States	Subsystems	Physical Architecture

Module 10 : Applying Design Patterns
What are Patterns?
Creational Patterns
Behavioral Patterns
Structural Patterns
Architectural Patterns
Singleton
Abstract Factory
Factory Method
Reducing Dependencies
Observer Pattern
Adapter Pattern
FaÇade pattern
Proxy Pattern