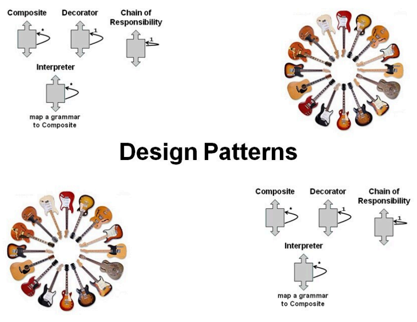
Design Patterns

Duration: 3 days **Price: € 1999**

Open Schedule



Audience Course Design Patterns

The course Design Patterns is intended for experienced developers and software architects with knowledge of object oriented programming and systems analysis who want to apply Design Patterns when designing these systems.

Prerequisites Course Design Patterns

Knowledge of an object-oriented programming language like C++, C#, or Java and experience with object oriented analysis and design with UML is required.

Realization Training Design Patterns

The concepts are treated according to presentation slides. The theory is illustrated with demos of patterns in C++, C# and Java. There are exercises in design problems where patterns are applied. The course material is in English. The course times are from 9.30 up and to 16.30.

Certification Design Patterns

Participants receive an official certificate Design Patterns after successful completion of the course.

Content Course Design Patterns

In the course Design Patterns you will learn how design patterns can be applied to the object oriented design of systems.

Design Patterns Intro

After an introduction about the role that design patterns play and how they can be used to realize the non-functional requirements of systems, attention is paid to how design patterns are described and cataloged.

Architectural Role

Also the role of design patterns in the architecture of applications is discussed and the various categories of design patterns that are distinguished.

Creational Patterns

In the module Creational Patterns the Factory patterns and the Builder, Prototype and Singleton pattern are discussed. You learn out of which classes, relationships, responsibilities and cooperations a typical design pattern solution can consist.

Structural Patterns

Next in the module the Structural Patterns the Adapter, Composite, Bridge, Decorator, Proxy and Flyweight pattern are discussed. You will learn the consequences of applying the patterns, the benefits and possible disadvantages in terms of time and space considerations and how to decide on the use of a particular pattern.

Behavioral Patterns

Next in the module Behavioral Patterns the Chain of Responsibility, Interpreter, Iterator, Mediator, State and Strategy patterns are discussed.

Architectural Patterns

Finally the module Architectural Patterns considers certain patterns that are involved in the architectural structure of software including Operating Systems and Frameworks. This module focuses on the Layer pattern, the Micro Kernel pattern and the Model View Controller (MVC) pattern.

Modules Course Design Patterns

Module 1 : Intro Design Patterns	Module 2 : Creational Patterns	Module 3 : Structural Patterns
What is a design pattern? Describing design patterns Reuse through design patterns Structure of patterns Classification of patterns Catalog of Design Patterns Creational Patterns Structural Patterns Behavioral Patterns Sample design patterns Selecting Design Patterns Solving problems with design patterns	Factory Patterns Factory Method Pattern Connect parallel class hierarchies Abstract Factory Pattern Concrete Class Isolation Promoting Consistency Builder Pattern Controlling the build process Prototype Dynamic configuration Singleton Pattern Controlled access	Adapter Pattern Pluggable Adapters Composite Pattern Sharing Components Decorator Pattern Lots of little objects FaÇade Pattern Reducing client-subsystem coupling Flyweight Pattern Reducing number of instances Proxy Pattern Copy-on-write
Module 4 : Behavioral Patterns	**Module 5 : Architectural Patterns**	
Chain of responsibility Command Pattern Interpreter Pattern Iterator Pattern Mediator Pattern Memento Pattern Observer Pattern State Pattern Strategy Pattern Template Pattern	Architectural patterns versus design patterns Patterns for real-time software Layers Pipes and Filters Blackboard Broker Model-View-Controller Presentation-Abstraction-Control Microkernel Reflection	