Design Patterns – sinnvoll einsetzen

Bewährte Entwurfsmuster verstehen, auswählen und ohne Pattern-Zoo einsetzen

01 INTRO

Design Patterns beschreiben wiederkehrende Lösungsansätze für typische Entwurfsprobleme. Ihr Wert liegt nicht darin, möglichst viele Muster in einem System unterzubringen, sondern Entwurfsprobleme zu erkennen und geeignete Lösungsprinzipien bewusst anzuwenden.

Der Kurs verbindet klassische Design Patterns mit modernen Software-Engineering-Praktiken. Die Teilnehmer implementieren ausgewählte Muster, vergleichen Alternativen und untersuchen auch Situationen, in denen ein Pattern unnötige Komplexität erzeugt.

DAUER

3 Tage

FORMAT

Präsenz · Live Online · Inhouse

VORKENNTNISSE

Sichere Kenntnisse objektorientierter Programmierung

02 TRAININGSPROGRAMM

Inhalte

Warum Design Patterns?

- wiederkehrende Entwurfsprobleme
- gemeinsame Sprache
- Kontext
- Problem
- Lösung
- Konsequenzen
- Patterns versus Rezepte

Entwurfsprinzipien als Grundlage

- Encapsulation
- Composition
- Polymorphism
- Dependency Inversion
- Open/Closed Principle
- Information Hiding
- Veränderlichkeit isolieren

Strategy

- austauschbares Verhalten
- Polymorphie
- Funktionen als Alternative
- Konfiguration
- Testbarkeit
- typische Einsatzfälle

Factory und Factory Method

- Objekterzeugung
- Entkopplung
- Konfiguration
- Produktvarianten
- Dependency Injection
- Alternativen

Builder

- komplexe Objekte
- optionale Parameter
- Fluent Interfaces
- Validierung
- Test Data Builder
- Spracheigenschaften als Alternative

Adapter und Facade

- inkompatible Schnittstellen
- externe Systeme
- Legacy APIs
- Vereinfachung
- Anti-Corruption Layer
- Abgrenzung der Muster

Decorator und Proxy

- Verhalten ergänzen
- Delegation
- Zugriffskontrolle
- Lazy Loading
- Logging
- technische Querschnittsfunktionen

Observer

- Ereignisse
- Publisher und Subscriber
- lose Kopplung
- Event-Systeme
- Lebenszyklus
- unerwartete Seiteneffekte

Command

- Aktionen als Objekte
- Queue
- Undo
- Logging
- Workflows
- asynchrone Verarbeitung

State und Template Method

- zustandsabhängiges Verhalten
- Zustandsautomaten
- Algorithmusstruktur
- Vererbung
- Komposition
- Alternativen

Patterns kritisch bewerten

- Patternitis
- Overengineering
- sprachspezifische Alternativen
- Framework-Funktionalität
- unnötige Abstraktion
- Kosten eines Patterns
- YAGNI

Pattern-Workshop

- Entwurfsproblem analysieren
- Lösungsvarianten entwickeln
- Pattern auswählen
- implementieren
- Alternative vergleichen
- Konsequenzen bewerten
- Entwurf begründen

03 DURCHFÜHRUNG

Rahmen und Organisation

Dauer	3 Tage
Durchführung	Präsenz oder Live Online, vorzugsweise als Inhouse-Schulung
Schwerpunkt	Inhouse-Schulungen für Unternehmen und Teams
Sprache	Deutsch, englische Fachbegriffe und Dokumentation
Unterlagen	Kursunterlagen, Beispiele und Übungsaufgaben

Technik	Eigener Rechner mit Entwicklungsumgebung für die praktischen Übungen

> **Inhouse-Schulungen**
> Die Übungen können in der Sprache durchgeführt werden, die im Team eingesetzt wird, etwa Java, Python oder C#.

04 ZIELGRUPPE

Für wen ist der Kurs gedacht?

Softwareentwickler und Softwarearchitekten, die Design Patterns nicht nur kennen, sondern situationsgerecht einsetzen möchten.

> **Voraussetzungen**
> Sichere Kenntnisse objektorientierter Programmierung.

05 LERNZIELE

Was Sie nach dem Kurs können

Die Teilnehmer können typische Entwurfsprobleme erkennen, geeignete Patterns auswählen, ihre Konsequenzen beurteilen und unnötige Pattern-Komplexität vermeiden.

06 FAQ

Häufige Fragen

Werden alle Gang-of-Four-Patterns behandelt?

Nein. Der Kurs konzentriert sich auf besonders relevante Muster und vor allem auf das Verständnis ihrer zugrunde liegenden Entwurfsprobleme.

Ist der Kurs sprachabhängig?

Die Konzepte sind weitgehend sprachunabhängig. Übungen können beispielsweise mit Java, Python oder C# durchgeführt werden.

Wann sollte ich kein Pattern verwenden?

Wenn eine einfachere Lösung das Problem ebenso gut löst. Genau diese Entscheidung ist Bestandteil des Kurses.

UC IT Service · Ulrich Cuber · kontakt@uc-it.de
https://www.uc-it.de/coaching/design-patterns/