

Program szkolenia:

Wzorce architektoniczne - architektura dla projektantów aplikacji i systemów

Informacje ogólne

Nazwa:	Wzorce architektoniczne - architektura dla projektantów aplikacji i systemów
Kod:	Patterns Arch
Kategoria:	Inżynieria oprogramowania
Grupa docelowa:	Projektanci i architekci systemów
Czas trwania:	2 dni
Forma:	50% wykłady / 50% warsztaty

Szkolenie prezentuje wybrane Wzorce Projektowe i Architektoniczne w praktycznym i niepodręcznikowym ujęciu osadzonym w kontekście projektowania aplikacji Webowych, platform, systemów i frameworków. Podczas szkolenia prezentowane są przykłady praktycznego zastosowania zaczerpnięte z rzeczywistych systemów klas: ERP, narzędzia wizualne, systemy rozproszone, serwery.

Podczas szkolenia uczestnicy nabędą zintegrowaną wiedzę na temat zdobyczy nowoczesnej inżynierii oprogramowania pozwalającą im na tworzenie zaawansowanych systemów. Omawiane zagadnienia leżą u podstaw nowoczesnych frameworków i technologii – co zwiększa poziom ich zrozumienia i pozwala na świadome korzystanie. Przedstawiamy techniki łączenie wzorców w struktury wyższego rzędu.

Szkolenie przeznaczone dla projektantów i architektów pragnących poszerzyć swe kompetencje w zakresie standardowych stylów architektonicznych.

Zalety szkolenia:

- » Nowoczesne architektury (CqRS - wspierająca DDD)
- » Szersza perspektywa
- » Dobór klasy rozwiązania do klasy problemu

Program szkolenia:

1. Architektury aplikacji

1.1. Podejście warstwowe

1.1.1. Różnice pomiędzy Layer a Tier

1.1.2. Dobór warstw, kiedy warto rozwarstwiać logikę na aplikacyjną i domenową

1.1.3. Warstwy w kontekście Domain Driven Design

1.2. Podejście Microcernel

1.3. Model-View-Controler

1.3.1. Przegląd podejść do implementacji MVC

1.3.2. Integracja MVC oraz Warstw

1.4. Praktyczne wykorzystanie technik Inversion of Control do budowy frameworków i systemów – na przykładzie Spring lub Seam (do wyboru)

1.4.1. Dependency Injection – podstawowa technika IoC

1.4.1.1. Wykorzystanie zamiast wzorców fabrykujących

1.4.1.2. Budowanie konkretnych Strategii, Dekoratorów itd. w zależności od stanu aplikacji (kontekst, konfiguracja)

1.4.1.3. Otwartość na rozbudowę dzięki wzorcom Strategii

1.4.2. Systemy sterowane zdarzeniami – silniejsza technia IoC

1.4.2.1. Użycie do tworzenie rozszerzalnych architektur opartych o pluginy

1.4.2.2. Użycie do tworzenia skalowalnych systemów wysokiej wydajności (wykorzystanie kolejek, np. JMS)

1.4.2.3. Sagi - Modelowanie złożonych procesów zdarzeniowych

1.4.3. Aspect Oriented Programming

1.4.3.1. Wstęp do AOP

1.4.3.2. Techniki Interceptorów

1.4.3.3. Przykłady zastosowania AOP

2. Command-query Responsibility Segregation – rozszerzona architektura warstwowa

2.1. Wsparcie dla Domain Driven Design

2.2. Rozwiązanie problemów z niedopasowaniem ORM do przeglądu danych w Gridach

2.3. Zorientowanie na skalowanie i rozszerzalność

3. Testability – projektowanie architektur aplikacji zorientowanych na testy

3.1. Dążenie do uruchamiania logiki poza serwerem – zwiększanie produktywności, redukcja czasu używanego na redeploy

3.2. Zagadnienia podatności architektury na testy: problemy i pułapki

3.3. Techniki testowania jednostkowego: dummy, fake, stub, mock

3.4. Narzędzia testowania jednostkowego i integracyjnego (JUnit, Mockito)