

## Program szkolenia:

# Projektowanie i modelowanie obiektowe oraz funkcyjne

### Informacje:

<b>Nazwa:</b>	<b>Projektowanie i modelowanie obiektowe oraz funkcyjne</b>
<b>Kod:</b>	<b>Craft-OOD i OOA</b>
<b>Kategoria:</b>	Wzorce i Craftsmanship
<b>Grupa docelowa:</b>	developerzy architekci
<b>Czas trwania:</b>	3 dni
<b>Forma:</b>	50% wykłady / 50% warsztaty

Podczas szkolenia uczestnicy mogą skonfrontować szkolne rozumienie obiektowości z tym czym na prawdę ona jest. Szkolenie skupia się na rozwiązywaniu realnych problemów z proejktów poprzez poprawne stosowanie technik obiektowych i funkcyjnych.

Koncepcje rozwiązań są wzbogacone o wybrane Wzorce Projektowe w praktycznym i niepodręcznikowym ujęciu osadzonym w kontekście aplikacji biznesowych. Wszystkie wzorce są ilustrowane przykładami zastosowania w modelowaniu logiki aplikacji i logiki biznesowej aplikacji enterprise.

Szkolenie przeznaczone dla programistów, projektantów i architektów tworzących oprogramowanie klasy biznesowej, pragnących poszerzyć swe kompetencje w zakresie profesjonalnych technik programistycznych zwiększających jakość kodu i projektu. Zdobyta wiedza przekłada się w praktyczny sposób na produktywność mierzoną w szerszej perspektywie czasu.

### Zalety szkolenia:

- Skupienie na kontekście aplikacji biznesowych
- Wybór jedynie użytecznych wzorców oraz technik
- Realne przykłady

## Szczegółowy program:

### 1. Model i cel modelu

1.1. Narzędzie do komunikacji

1.2. Baza wiedzy

### 2. Poznawanie biznesu, wymagań i celów

2.1. Modeling Whirlpool

2.2. CRC

2.3. BDD

### 3. Efektywne techniki wizualizacji i dokumentowania

3.1. C4

3.2. BDD

### 4. Uczucie się a praca programisty

### 5. Architektura aplikacji umożliwiająca programowanie obiektowe

5.1. 4 Layers

5.2. Ports and Adapters

### 6. Realizacja wymagań w formie biblioteki biznesowej

### 7. Projektowanie API: simple and powerful

### 8. Rozszerzalność, jak się przygotować na to czego nie da się przewidzieć

### 9. OOP, OOD, OOA

9.1. Poprawne rozumienie paradygmatu OO

9.1.1. Antywzorce i typowe błędy

9.1.1.1. Model anemiczny

9.1.1.2. Struktury danych

9.1.1.3. Dziedziczenie ze względu na wspólne atrybuty

9.1.1.4. Klasy zamiast atrybutów

9.1.1.5. Modelowanie ról przez dziedziczenie

9.1.2. Obiekt vs struktura vs funkcja cs procedura

9.2. Techniki wspomagające myślenie obiektowe

9.2.1. SOLID

9.2.2. GRASP

9.2.3. RDD

9.2.4. DDD

9.2.4.1. Building Blocks

9.3. Kiedy nie stosować OO

## 10. Composition over inheritance, dekompozycja zachowań

10.1. Nieksiążkowe przykłady biznesowe

## 11. Inversion of control, to więcej niż DI

11.1. Events, AOP

## 12. Praktyczne wzorce projektowe w przykładach typowych problemów

12.1. Strategy jako plugin

12.2. Dekorowane strategie do walidacji

12.3. Role object to modelowania ról i uprawnień w systemie

12.4. Extension object

## 13. Elementy funkcjonalne, które robią różnice

## 14. Immutability w praktyce

## 15. Wzorce w testowaniu

15.1. Assembler/Builder

15.2. Assert Object

15.3. Object Mother

## 16. Modelowanie testów, a cele testów

### 16.1. Podejście strategiczne

16.1.1. Co testujemy: unit, komponent, moduł, system

16.1.2. Po co testujemy: perfekcja, akceptacja, regresja, postęp prac