SZKOLENIE ŚREDNIO ZAAWANSOWANE

Dobre praktyki programowania obiektowego w języku C++

CPP/OOBP

Czas trwania: 3 dni (24h)

Zapoznanie i przećwiczenie dobrych praktyk, zasad i wzorców programowania obiektowego w C++

Cele szkolenia

- Poznanie zasad SOLID
- Poznanie wybranych wzorców GRASP
- Poznanie wybranych wzorców Bandy Czworga (Gang of Four)
- Przećwiczenie wybranych idiomów programowania w C++

Zalety

- Praktyka przed teorią - wszystkie szkolenia technologiczne prowadzone są w formie warsztatowej. Konieczna teoria jest wyjaśniana na przykładzie praktycznych zadań
- Konkretne umiejętności - w ramach każdego szkolenia rozwijamy praktyczne umiejętności związane z daną technologią i tematyką
- Nauka z praktykami - wszyscy trenerzy na co dzień pracują w projektach, gwarantuje to dostęp do eksperckiej wiedzy i praktycznego know-how

Dla kogo?

- Szkolenie skierowane jest do programistów, projektantów i architektów, którzy pragną poznać, pogłębić lub usystematyzować swoje wiadomości na temat dobrych praktyk, zasad i wzorców programowania obiektowego w C++

Wymagania

- Uczestnicy muszą posiadać umiejętność programowania w języku C++

Program

1. Zasady SOLID
    a. Zasada jednej odpowiedzialności (The Single Responsibility Principle)
        -Wstęp, definicja, odpowiedzialności
        -Warsztaty problemowe
        -Zasady refaktoryzacji
        -Wybrane wzorce Bandy Czworga (GoF) w kontekście zasady

b. Zasada otwarte/zamknięte (The Open/Closed Principle)
-Wstęp, definicja, odpowiedzialności
-Warsztaty problemowe
-Zasady refaktoryzacji
-Wybrane wzorce Bandy Czworga (GoF) w kontekście zasady
c. Zasada podstawienia Liskov (The Liskov Substitution Principle)
-Wstęp, definicja, odpowiedzialności
-Warsztaty problemowe
-Zasady refaktoryzacji
-Wybrane wzorce Bandy Czworga (GoF) w kontekście zasady
d. Zasada segregacji interfejsów (The Interface Segregation Principle)
-Wstęp, definicja, odpowiedzialności
-Warsztaty problemowe
-Zasady refaktoryzacji
-Wybrane wzorce Bandy Czworga (GoF) w kontekście zasady
e. Zasada odwrócenia zależności (The Dependency Inversion Principle)
-Wstęp, definicja, odpowiedzialności
-Warsztaty problemowe
-Zasady refaktoryzacji
-Wybrane wzorce Bandy Czworga (GoF) w kontekście zasady

2. Wzorce Bandy Czworga (GoF)
a. Wzorce konstrukcyjne
-Wzorce: Singleton, Factory Method, Abstract Factory, Prototype, Builder
-Warsztaty problemowe
b. Wzorce strukturalne
-Wzorce: Adapter, Bridge, Composite, Decorator, Façade, Flyweight, Proxy
-Warsztaty problemowe
c. Wzorce behawioralne
-Wzorce: Chain of Responsibility, Command, Iterator, Mediator, Memento, Observer, State, Strategy, Template Method, Visitor
-Warsztaty problemowe

3. Wzorce GRASP
a. Low Coupling, High Cohesion
b. Information Expert, Creator, Controller
c. Polymorphism, Pure Fabrication
d. Indirection, Protected Variations
e. Warsztaty problemowe

4. Idiomy programowania w C++
a. Zarządzanie zasobami
-Idiomy: Release Return, Move Constructor, Resource Acquisition Is Initialization, Scope Guard
-Warsztaty problemowe
b. Zarządzanie pamięcią
-Idiomy: Intrusive Reference Counting, Non-intrusive Reference Counting, Const auto_ptr, Checked Delete, Concrete Data Type, Copy and Swap

-Warsztaty problemowe

c. Optymalizacja pamięci i przetwarzania
   -Idiomy: Shrink to Fit, Clear and Minimize, Non-throwing Swap, Erase-Remove, Boost Mutant, Computational Constructor, Copy on Write, Empty Base Optimization
   -Warsztaty problemowe

d. Typy i bezpieczeństwo typów
   -Idiomy: Type Safe Enum, Type Selection, Type Generator, Traits, Capability Query, Coercion by Member Template, Mixin from Above, Int to Type
   -Warsztaty problemowe

e. Konstrukcja i inicjalizacja
   -Idiomy: Construction Tracker, Construct on First Use, Base from Member, Runtime Static Initialization Idiom
   -Warsztaty problemowe

f. Polimorfizm
   -Idiomy: Interface Class, Inner Class, Virtual Friend Function, Polymorphic Exception, Virtual Constructor, Calling Virtuals During Initialization, Polymorphic Value Types
   -Warsztaty problemowe