Wzorce projektowe

CZAS TRWANIA
3 dni

ABSTRAKT
Szkolenie ma za zadanie zapoznać uczestników z tematyką wzorców projektowych. W odróżnieniu od innych nie pokazujemy wzorców używając syntetycznych książkowych przykładów, a prawdziwych biznesowych przypadków. Szkolenie jest prowadzone w postaci warsztatów z dużym naciskiem na praktyczną ich formę.

W ciągu tych kilku dni uczestnicy będę rozwiązywać praktyczne problemy takimi jak np.: system rabatowy, wyliczanie Vatu, dokument stanowy, ticket błędu, liczenie zdolności kredytowej, system regulacji temperatury, system do walidacji. W każdym przypadku będziemy się zastanawiali nad testowalnością rozwiązania i tworzyli odpowiednie zestawy testów.

Szkolenie jest prowadzone z użyciem języka C#, choć wszystkie koncepcje są łatwo przenośne na inne platformy. Jeżeli uczestnik zna komplementarne narzędzie może bez problemu uczestniczyć w szkoleniu i dużo z niego wynieść.

ZAGADNIENIA

Kod wysokiej jakości
- OOD
  - Klasa abstrakcyjna czy interfejs? Jakie są różnice, które kiedy używać?
- GRASP
  - W szczególności fundamentalne zasady Low Coupling i High Cohesion.
- SOLID
  - 5 zasad skutecznej strukturyzacji kodu z przykładami i antyprzykładami.

Wzorce
- Decorator
  - Jak dynamicznie rozszerzyć zachowanie innego obiektu?
  - Jak wykorzystać kontener DI do automatycznego składania decoratorów?
- Composite
  - Jak obsłużyć wiele obiektów jako jeden?
- Strategia
  - Jak umożliwić zmianę działania systemu w zależności od zewnętrznych warunków?
  - Jak zbudować system, który będzie wspierał różne stawki podatkowe w różnych krajach?
  - Współpraca z kontenerem DI
- Specyfikacja
  - Jak zenkapsulować logiczne warunki w reużywalne algorytmy?
- Template method i iterator
  - Wstęp do programowania funkcyjnego
    - Delegaty, metody anonimowe, lambdy
  - Iterator w .Net
    - Nieskończone sekwencje
    - Strumieniowanie

- Factory method i abstract factory
  - Miejsce przejęcia najgorszej formy zależności - tworzenia obiektów
  - TypedFactory w kontenerze DI
- Chain of responsibility
- Interpreter i Visitor
  - Silne typowanie w .Net'cie
    - INotifyPropertyChanged
  - Realne przykłady biznesowe: system liczący zdolność kredytową
- State
  - Alternatywny podejście do obsługi maszyny stanów.
  - Kiedy wzorzec się nie sprawdza?
- Singleton
  - Dlaczego singleton jest antywzorcem? Alternatywy
  - Kiedy naprawdę warto użyć singletonu?
- Observer i mediator
  - Jak skomunikować się z obiektem nie znając go?
- Proxy
- Facada
  - Rozwarstwienie warstwy biznesowej na aplikacyjną i domenę biznesową
- Bridge
- Flyweight
  - Techniczny wzorzec optymalizacji użycia zasobów
- Prototype i builder
- Command
  - Sposób na odłożenie operacji w czasie
  - CQRS jako przykład użycia wzorca commend
    - Zyski z takiej konstrukcji kodu

Wstęp do lokalizowania komponentów

- ServiceLocator
- Dependency Injection
  - Castle Windsor jako przykład kontenera
  - Co, gdzie i jak rejestrować w kontenerze DI
    - Kod
    - Konwencja
    - Atrybuty
    - Interfejsy
    - Xml
- Zdarzenia
  - Systemu oparte na komunikatach
- Programowanie aspektowe
  - Jak dodać logowanie, bądź profilowanie do każdej metody klas serwisowych?
  - Jak automatycznie zaimplementować INotifyPropertyChanged?

Testowalność, a wzorce

- Testy jednostkowe
  - NUnit albo MsTest jako narzędzie do testów automatycznych
  - NSubstitute jako narzędzie do mockowania zależności
- Testy integracyjne
- Strategie testowania
  - Co testować, kiedy testować, jak testować?

KONTAKT

W celu omówienia szczegółów i rezerwacji terminu, skontaktuj się z nami:
kontakt@macmichal.pl
tel. 513 959 379