Kod szkolenia: **CH/TDD**

Tytuł szkolenia: **Test-Driven Development w C#**

Dni: 3

Opis:

Adresaci szkolenia

Szkolenie adresowane jest do programistów C#, chcących podnieść swoje umiejętności tworzenia czystego, testowalnego kodu. Do udziału w warsztatach zapraszamy też znających język C# architektów i testerów, którzy chcą poznać zalety tworzenia kodu z pomocą testów.

Cel szkolenia

Uczestnicy czują się pewnie stosując TDD w codziennej pracy swojego zespołu. Uczestnicy znają i potrafią zastosować techniki pozwalające na tworzenie kodu wysokiej jakości.

W szczególności uczestnicy dowiedzą się czym różnią się poszczególne rodzaje testów. Poznają i przyswoją sobie cykl pracy TDD, nauczą się projektować oprogramowanie pod względem testowalności i tworzyć czytelny kod. Poznają biblioteki ułatwiające stosowanie TDD oraz umożliwiające testowanie na różnych poziomach.

Mocne strony szkolenia

Szkolenie prowadzone jest w formie warsztatów. Uczestnicy przyswajają wiedzę w najskuteczniejszy możliwy sposób – praktykując TDD podczas serii ćwiczeń. Po szkoleniu czują i rozumieją, czym TDD jest i jak je zastosować w pracy.

Wymagania

Od uczestników szkolenia wymagana jest umiejętność programowania w języku C#.

Parametry szkolenia

3*8 godzin (3*7 godzin netto) wykładów i warsztatów (z wyraźną przewagą warsztatów).

Wielkość grupy: maks. 8-10 osób.

Program szkolenia:

Podstawy TDD

- Jak TDD może Ci pomóc?
- Stosowanie cyklu Red-Green-Refactor
- Bezpieczne refaktoryzowanie kodu pokrytego testami
- Pełne wykorzystanie możliwości IDE (VisualStudio + ReSharper) w celu wspierania procesu TDD (generowanie kodu, refaktoryzacje automatyczne, skróty klawiaturowe)

Podstawy testowania jednostkowego

- Co to jest test jednostkowy?
- Możliwości NUnit* - asercje, testy parametryzowane, testy wyjątków
- Czytelne asercje z wykorzystaniem Fluent Assertions*
- Pisanie czytelnych, utrzymywalnych testów
- Nazewnictwo testów

Mechanika TDD

- Utrzymanie rytmu TDD
- Wybór następnego testu do zaimplementowania
- Częste pułapki w stosowaniu TDD

Testy współpracujących obiektów

- Używanie dublerów testowych (mocków) do izolowania klas testowanych
- Różnice pomiędzy mockami a stubami
- NSubstitute* jako wygodne narzędzie generowania mocków w locie

Test-drive'owanie pełnej aplikacji

- TDD w warstwie dostępu do danych z NHibernate*
- TDD w GUI z ASP.NET MVC*

- Integrowanie wszystkich warstw

Projektowanie obiektowe

- Testowalność - projektowanie aplikacji pod kątem łatwego pisania testów
- Heurystyki projektowania obiektowego (zasady SOLID, Inversion of Control, Dependency Injection, powiązania, spójność)

Wprowadzenie do pracy z kodem odziedziczonym

- Jak zrefaktoryzować kod, aby dało się go pokryć testem (rozcinanie zależności / Dependency Breaking)
- Zrozumienie kodu i zabezpieczenie miejsca zmiany dzięki testom charakteryzacyjnym

* na życzenie klienta może być wykorzystany inny framework/biblioteka