Tytuł szkolenia: Projektowanie oparte na domenie (Domain Driven Design)

Kod szkolenia: MS-DDD

Wprowadzenie

Projektowanie oparte na domenie (ang. Domain Driven Design) jest zarówno metodologią, jak i metodyką tworzenia oprogramowania, zakładającą tworzenie systemów informatycznych w oparciu o ciągle ewoluujący model biznesowy (tzw. domenę). *Projektowanie oparte na domenie*, oprócz szerokiego zestawu wskazówek i wzorców dla programistów, projektantów i architektów oprogramowania aż po menedżerów projektu, oferuje sposoby sprostania dwóm największym stojącym przed zespołami tworzącymi systemy informatyczne: komunikacją (w ramach zespołu oraz ze światem zewnętrznym), oraz kodem odziedziczonym. Zespół postępujący zgodnie z zasadami *projektowania opartego na domenie* potrafi nie tylko im sprostać, ale również uczynić z nich swoje mocne strony.

Adresaci szkolenia

Szkolenie przeznaczone jest dla wszystkich osób zaangażowanych w tworzenie systemów informatycznych od strony techniczej oraz biznesowej, m. in. Programistów, Projektantów oprogramowania, Architeków oprogramowania, Specjalistów z dziedziny UX, Managerów projektów bądź osób zarządzających pracą zespołu.

Cel szkolenia

Celem szkolenia jest zapoznanie uczestników z głównymi założeniami*projektowania opartego na domenie* oraz umożliwienie im wdrożenia jego elementów w projektach nad którymi pracują. Podczas szkolenia od podstaw tworzona będzie przykładowa aplikacja w technologi Microsoft.Net.

Po zakończeniu szkolenia aktywny uczestnik:

- Rozumie ideę efektywnego komunikowania się w ramach zespołu oraz poza nim przy pomocy *języka powszechnego*.
- Potrafi wyróżnić *domenę* oraz *infrastrukturę* w ramach systemu informatycznego. Potrafi również zaprojektować system informatyczny w oparciu o *architekturę warstwową*.
- Potrafi zidentyfikować i wdrożyć wzorce projektowe stosowane w domenie.
- Rozumie ideę tworzenia *deklaratywnego kodu*, zaszywania w domenie *reguł biznesowych* oraz dokumentowania projektu poprzez kod.
- Rozumie ideę *kontekstów*, potrafi efektywnie integrować bądź wydzielać kod odziedziczony i niezależne aplikacje z systemu nad którym pracuje.
- Potrafi wyróżnić i wdrożyć różnie podejścia do wspólnej pracy kilku zespołów w ramach tego samego projektu bądź z tym samym kodem.

Czas i forma szkolenia

- 14 godzin (2 dni x 7 godzin), w tym wykłady i warsztaty praktyczne.

Plan szkolenia

1. Co to jest Projektowanie oparte na domenie?
    a. Co to jest domena?
        i. Domena
        ii. Infrastruktura
        iii. Ekspert domenowy
    b. Co to jest model?
        i. Model
        ii. Reprezentacja/instancja modelu
    c. Architektura warstwowa.
        i. Wzorzec projektowy “sprytny interfejs” (smart UI)
        ii. Podział oprogramowania na warstwy
        iii. Warstwa danych
        iv. Warstwa domeny (modelu, logiki biznesowej)
        v. Warstwa aplikacji
        vi. Warstwa interfejsu
        vii. Luźne połączenia między warstwami
    d. Język powszechny i komunikacja w zespole.
        i. Język powszechny
        ii. Dokumentowanie oprogramowania
        iii. Słownik z najważniejszymi pojęciami z języka powszechnego.
    e. Tworzenie oprogramowania w procesie iteracyjnym.
        i. Ewolucja modelu z naiwnego do zaawansowanego poprzez ciągłą naukę.
        ii. Dlaczego dokumentowanie modelu nie jest zalecane?

2. Z jakich elementów składa się domena.
    a. Co to jest wzorzec projektowy? Przypomnienie.
    b. Byt
        i. Co to jest byt?
        ii. Co to jest tożsamość?
        iii. Cykl życia bytu.
        iv. Byty proste (POCO lub POJO)
    c. Wartość
        i. Co to jest wartość?
        ii. Czym różni się wartość od bytu?
        iii. Wartości nie posiadają cyklu życia.
    d. Usługa
        i. Co to jest usługa?
        ii. Kiedy powinno się stworzyć usługę?
    e. Moduł
        i. Co to jest moduł?
        ii. Jak nie dzielić oprogramowania na moduły?
        iii. Jak dzielić oprogramowanie na moduły?
    f. Agregat
        i. Co to jest agregat?
        ii. Co to jest korzeń agregatu?
        iii. Korzeń agregatu jako jedyny umożliwia dostęp do obiektów w ramach agregatu oraz dodawanie i usuwanie obiektów w ramach agregatu.
        iv. Korzeń agregatu odpowiada za sprawdzanie ograniczeń wewnątrz agregatu podczas zmiany jego stanu.
        v. Nie każdy obiekt zawierający kolekcję obiektów jest agregatem! Różnica pomiędzy agregacją i prostą relacją.
    g. Repozytorium
        i. Co to jest repozytorium?
        ii. Czy agregat jest repozytorium? Czy repozytorium jest agregatem?
        iii. Kiedy używać repozytorium?
    h. Fabryka
        i. Co to jest fabryka?
        ii. Kiedy powinno się używać fabryki?
3. Jak budować zaawansowany model?
    a. Co to jest przełom i co robić kiedy nastąpi?
        i. Co to jest przełom?
        ii. Co robić kiedy nastąpi przełom?
        iii. Dlaczego opłaca się wykorzystywać przełomy?
    b. Jak zdobywać/poszerzać wiedzę o domenie?
        i. Eksperci domenowi
        ii. Literatura fachowa
    c. Modelowanie nietrywialnych obiektów w domenie.
        i. Modelowanie ograniczeń
        ii. Modelowanie procesów, wzorzec projektowy “Strategia”, proces: od linijki kodu po osobną klasę
        iii. Wzorzec projektowy “Specyfikacja”
    d. Pisanie elastycznego kodu.
        i. Deklaratywne nazewnictwo metod i interfejsów
        ii. Metody bez skutków ubocznych
        iii. Asercje
        iv. Klasy z najmniejszą możliwą liczbą powiązań
    e. Używanie wzorców projektowych w domenie.

4. Dzielenie modelu między zespołami.
    a. Co to jest kontekst i jak wyróżniać konteksty?
        i. Mapa kontekstów
    b. Metody dzielenia modelu z innym zespołem.
        i. Konformizm
        ii. Wspólne jądro
        iii. Warstwa ochronna
        iv. Osobne ścieżki
        v. Zespół “dostawca” i zespół “klient”
        vi. Ciągła integracja
    c. Metody publikowania modelu.
        i. Model otwartego hosta
        ii. Język publikowany