BO·TT·EGA
IT minds

Program szkolenia:

Shift-left w Testowaniu Mikroserwisów na platformie Java

Informacje:

Nazwa:	**Shift-left w Testowaniu Mikroserwisów na platformie Java**
Kod:	**ms-test**
Kategoria:	Warsztaty eksperckie Microservices
Odbiorcy:	liderzy techniczni, architekci, developerzy
Czas trwania:	3 dni
Forma:	30% wykłady/70% warsztaty

Celem szkolenia jest nauka strategii zapewnienia wysokiego poziomu bezpieczeństwa wytwarzanego produktu przy zmniejszeniu kosztu testów programistycznych, automatycznych i manualnych. Uczestnikami są programiści mierzący się ze złożonością budowania i testowania architektury mikroserwisów pragnący poszerzyć swoją wiedzę o strategie pomocne w pragmatycznym podejściu do tego zagadnienia. Szkolenie systematyzuje rozbudowane zagadnienie do elastycznej strategii dającej bezpieczeństwo oraz stabilność.

Mnogość ruchomych elementów i skomplikowanie architektury mikroserwisów powoduje że znane dotąd strategie pracy z kodem i testami wymagają dedykowanego podejścia. Takiego które zagwarantuje wysoką jakość produktu i sprawną współpracę zespołów nie powiększając jednocześnie kosztu wytworzenia i utrzymania zarówno dla testów wewnątrz pojedynczego mikroserwisu jak i kontraktów oraz interakcji pomiędzy nimi.

Uczestnicy szkolenia:

- zrozumieją przyczyny i skutki problemów z wyborem poziomu testu do celu który powinien realizować
- poznają receptury i kryteria doboru i utrwalą strategie testowania dla napotkanych problemów
- rozpoznają wzorce i antywzorce spotykane na każdym poziomie testów automatycznych

Zalety szkolenia:

- Zagadnienia architektury systemu i testów wspierającej testowalność kodu
- Najlepsze wzorce i praktyki testów mikroserwisów i testów kontraktowych
- Aspekty modularyzacji oprogramowania i testów

BO·TT·EGA
IT minds

Szczegółowy program:

1. Problemy w testowaniu mikroserwisów

- 1.1. Czym jest Architektura Mikroserwisów
  - 1.1.1. Odpowiedzialność serwisów i całego systemu
  - 1.1.2. Odniesienie poziomu testów do poziomu abstrakcji modelu C4
- 1.2. Zyski i koszty rozpraszania
- 1.3. Odpowiedzialność zespołów za serwisy i jakość / testy
- 1.4. Wzorce integracji i kontraktów
  - 1.4.1. Komunikacja Synchroniczna i Asynchroniczna
  - 1.4.2. Różnicowanie relacji między Producentem a konsumentem (OpenHost, Conformist, Consumer-Supplier)
  - 1.4.3. Różnicowanie wzorców komunikatów: Komendy, Zdarzenia i Kwerendy

2. Shift-left: Strategie redukcji kosztów testów

- 2.1. Poziomy testów
- 2.2. Racjonalny dobór zakresu testu do poziomu abstrakcji
- 2.3. Posługiwanie się językiem wymagań biznesowych
- 2.4. Uzasadnienie i koszt dla automatyzacji

3. Perfekcyjna domena dzięki testom jednostkowym

- 3.1. Na jakie pytania odpowiadają testy jednostkowe
- 3.2. Prawidłowy model jako jednostka testowania
- 3.3. Granice zaufania do testów jednostkowych
- 3.4. Świadome korzystanie z Test Doubles
- 3.5. Testowanie narzędzi i klas pomocniczych

4. Testy komponentowe

- 4.1. Poprawne formułowanie hipotez do sprawdzenia

BO·TT·EGA
IT minds

4.2. Obserwowalne zachowania gwarantowane w testach

4.3. Strategie testowania API publicznych i wewnętrznych

4.4. Testowanie punktów integracji i infrastruktury

5. Testy API i GUI

5.1. Zakres i odpowiedzialność testu

5.2. Rola logiki w testach API

5.3. Strategie testowania API publicznych i wewnętrznych

6. Testy Kontraktowe

6.1. Wzorce i antywzorce dla zakresu testów kontraktowych

6.2. Zmienność środowisk testowych i stabilizacja kontraktów

6.3. Gwarantowanie stałości kontraktów przez testy

7. Testowanie na produkcji

7.1. Jak zagwarantować bezpieczeństwo działania i przeprowadzić w pełni wiarygodne testy

7.2. Observability dla pewności działania po wdrożeniu

8. Wzorce i antywzorce w testowaniu mikroserwisów

8.1. Nadużywanie Test Doubles a ich pragmatyczne wykorzystanie

8.2. Wzorce na powtarzalne operacje na repozytoriach

8.3. Wzorce na czytelne sekcje `given`

8.4. Sposoby na wywołania API w sekcji `when`

8.5. Ukrywanie złożoności w sekcjach `then`

8.6. Wzorce przygotowywania systemu do testów

8.7. Wzorce i antywzorce nazewnictwa testów

8.8. Rola frameworka w testach

8.9. Budowa Testu - najlepsze praktyki

8.10. Dodaję nowy feature: kiedy i jaki test napisać

BO·TT·EGA
IT minds

8.11. Pokrycie kodu testami