BO·TT·EGA
IT minds

Program szkolenia:

Testowanie obciążeniowe

Informacje:

Nazwa:	**Testowanie obciążeniowe**
Kod:	**craft-test-load**
Kategoria:	Testowanie automatyczne
Grupa docelowa:	testerzy developerzy
Czas trwania:	2 dni
Forma:	50% wykłady / 50% warsztaty

Podczas szkolenia uczestnicy poznają techniki testowania wydajności i obciążenia systemów.

Podczas warsztatów praktycznych uczestnicy posiądą umiejętność używania popularnych narzędzi opensource, przygotowania i realizacji kompleksowego procesu testowania wydajności oraz analizy i interpretacji wyników.

Zalety szkolenia:

- Narzędzia automatyzacji
- Najlepsze wzorce i praktyki
- Kompleksowy proces

BO·TT·EGA
IT minds

Szczegółowy program:

1. Wprowadzenie

- 1.1. Proces testowania wydajności
- 1.2. Zrozumienie sposobu testowania wydajności
- 1.3. Podejścia do procesu testowania wydajności
  - 1.3.1. zbieranie wymagań i ustalenie celów testów
  - 1.3.2. wybieranie sposobów pomiaru i kryteriów akceptacji
  - 1.3.3. określenie profilu ruchu
- 1.4. Koszty testów wydajnościowych

2. Identyfikacja celów wydajności i celów biznesowych

- 2.1. Przygotowanie strategii testów wydajnościowych

3. Infrastruktura i architektura, co musi być częścią testu?

- 3.1. Platforma docelowa vs platforma testowa, najczęściej popełniane błędy

4. Projektowanie testów z wykorzystaniem JMeter oraz SOAPUI

- 4.1. Definiowanie obciążenia i profilu ruchu
- 4.2. Symulowanie transakcji, wykorzystanie narzędzi nagrywających ruch
- 4.3. Symulacja obciążenia systemu, z wykorzystaniem narzędzi Unix (m.in. stress)
- 4.4. Analiza czynników wpływających na obciążenie systemu
- 4.5. Rodzaje testów wydajności, m.in. czasy reakcji, wykorzystanie zasobów, itp.

5. Proces

- 5.1. Przygotowanie scenariuszy i danych do testów
- 5.2. Narzędzia do generowania danych do testów wydajności na przykładzie Databene Benerator
- 5.3. Konfigurowanie infrastruktury testowej / Architektura
- 5.4. Procedura wykonania testów, zbieranie i analiza danych

BO·TT·EGA
IT minds

6. Monitorowanie

6.1. wykorzystania zasobów systemu (IO,CPU,RAM) z wykorzystaniem narzędzi Unix

6.2. JVM z wykorzystaniem VisualVM i dostępnych rozszerzeń

6.3. zachowania serwera aplikacyjnego z wykorzystaniem JavaMelody

7. Analiza

7.1. Zachowania systemu

7.2. Typowe scenariusze (IO, CPU, RAM, aktywność GC, lockcontention)

7.3. Analiza wąskich gardeł z wykorzystaniem narzędzi dostępnych w Java SDK

7.4. Analiza z wykorzystaniem profilerów i Eclipse Memory Analyzer Tool

8. Automatyzacja testów wydajnościowych

9. Zarządzanie i rozwój scenariuszy testowych