

## Program szkolenia:

# Testowanie akceptacyjne i end-to-end aplikacji web przy użyciu narzędzia Cucumber i języka Ruby w metodyce BDD

## Informacje:

<b>Nazwa:</b>	<b>Testowanie akceptacyjne i end-to-end aplikacji web przy użyciu narzędzia Cucumber i języka Ruby w metodyce BDD</b>
<b>Kod:</b>	<b>RoR-cucumber</b>
<b>Kategoria:</b>	Ruby on Rails
<b>Grupa docelowa:</b>	developerzy testerzy
<b>Czas trwania:</b>	3 dni
<b>Forma:</b>	30% wykłady / 70% warsztaty

---

Szkolenie służy nabyciu konkretnych umiejętności w zakresie technik BDD w oparciu o narzędzie Cucumber i język Ruby.

Rozpoczniemy od krótkiego repetytorium z języka Ruby, idei BDD, architektury testów, przeglądu wzorców, dobrych i złych scenariuszy, by przejść do praktycznego przećwiczenia przedstawionych koncepcji. Będziemy pisać testy automatyczne do przygotowanej wcześniej aplikacji webowej, która udostępnia zarówno API jak i webowy interfejs użytkownika.

## Zalety szkolenia:

- Nauka poprzez ćwiczenia praktyczne
- Poznanie pełnego procesu wytwarzania oprogramowania w oparciu o Behavior Driven Development
- Szybkie tworzenie w pełni testowalnych aplikacji internetowych

## Szczegółowy program:

### 1. Język Ruby

1.1. Filozofia

1.2. Różnice pomiędzy dynamicznie a statycznie typowanymi językami programowania

1.3. Podstawowe konstrukcje języka (Metody, Klasy, Moduły, Bloki, Sterowanie, Wyrażenia regularne, Yield)

1.4. Podstawowe typy (Teksty, Liczby, Tablice, Mapy)

1.5. Programowanie zorientowane obiektowo

1.6. Metaprogramowanie

1.7. Podejście funkcyjne

1.8. Idiomy języka Ruby

### 2. Behavior Driven Development

2.1. Zalety bliskiej współpracy z klientem

2.1.1. Rola dostawcy, rola klienta w testach akceptacyjnych

2.2. Tworzenie aplikacji podejściem BDD

2.3. Techniki tworzenia scenariuszy akceptacyjnych

2.3.1. Podejście deklaratywne zamiast imperatywnego

2.3.1.1. Unikanie pisania "skryptów klikania"

2.3.1.2. Raczej: "co ma być" niż "co trzeba zrobić"

2.3.2. Odporność scenariuszy na zmiany systemu

2.3.3. Wady kruchych scenariuszy

2.4. Podejście dwuwarstwowe

2.4.1. Warstwa Flow - User Story

2.4.2. Warstwa Automatykacji interakcji z systemem

2.5. Narzędzia i wzorce

2.5.1. Cucumber - najlepsze praktyki

2.5.2. Capybara – testowanie GUI

2.5.3. Page Object - antywzorzec powodujący powstawanie kruchych testów

2.5.4. Feature Object - bezpieczna alternatywa dla Page Object

2.5.5. Technika ujednociania testów wykonywanych poprzez GUI i Servisy –  
Agenty

### 3. Warsztaty

3.1. Instalacja środowiska

3.2. Ruby Mine IDE

3.3. Tworzenie testów akceptacyjnych i e2e aplikacji webowej

3.3.1. Zapoznanie z aplikacją którą będziemy testować

3.3.2. Testowanie automatyczne na poziomie API

3.3.3. Testowanie automatyczne na poziomie GUI