

RobotFramework

Techniki zaawansowane

Cele szkolenia

Przedstawienie możliwości rozszerzenia funkcjonalności **Robot Framework** w testowaniu aplikacji za pomocą bibliotek Python i integracji z Java..

Umiejętności

Dzięki szkoleniu uczestnik:

- Pozna dobre praktyki, które powinno się stosować w programowaniu testów:
 - unikanie powtórzeń
 - refactoring
- Stworzy dobrą architekturę test framework'a
 - łatwe utrzymanie
 - łatwe rozszerzanie
 - wsparcie wielu środowisk
- Pozna sposoby i strategie uruchamiania testów w procesie CI

Profil uczestników

Szkolenie kierowane jest do testerów automatycznych, którzy chcą rozwinąć swoje umiejętności w wykorzystaniu narzędzia Robot Framework.

Uczestnicy szkolenia powinni znać narzędzie Robot Framework oraz język Python i narzędzia wykorzystywane w budowaniu projektów w języku Java

Przygotowanie uczestników

Dobra znajomość Robot Framework - tworzenie i uruchamianie testów, podstawowa znajomość języka programowania Python i Java

Szczegółowy program szkolenia

1. Wprowadzenie
 - 1.1. Kilka słów o automatyzacji testów - dlaczego warto?
 - 1.2. Alternatywne edytory - Atom
2. Budowanie test frameworka
 - 2.1. Pisanie czytelnego kodu

- 2.1.1. Zasada DRY
- 2.1.2. Refactoring
- 2.2. Dobra architektura (4 warstwy)
 - 2.2.1. Scenariusz testowy
 - 2.2.2. Dane testowe i środowiskowe
 - 2.2.3. Warstwa akcji użytkownika
 - 2.2.4. Warstwa interakcji z systemem
- 2.3. Przykłady wykorzystania 4 warstwowej architektury
 - 2.3.1. Łatwa podmiana rodzaju danych testowych we wszystkich scenariuszach (dane losowe, dane statyczne) - na przykładzie aplikacji TodoMVC i SeleniumLibrary
 - 2.3.2. Jeden test - wiele środowisk - na przykładzie różnych implementacji aplikacji TodoMVC i SeleniumLibrary
 - 2.3.3. Przełączanie testu pomiędzy warstwami UI i Rest API - na przykładzie aplikacji Todoist i SeleniumLibrary / RESTinstance
- 3. Rozszerzanie możliwości
 - 3.1. Listener'y
 - 3.1.1. Prosty debugger / logger
 - 3.1.2. Raportowanie wyników do zewnętrznego systemu przy pomocy RestAPI
 - 3.1.3. Automatyczne raportowanie błędów (na przykładzie GitHub'a)
 - 3.1.4. Dodatkowe asercje
 - 3.2. Tworzenie bibliotek keywordów (Python)
 - 3.2.1. Rozszerzanie istniejących na przykładzie Selenium i algorytmu do trawersowania elementów strony
 - 3.2.2. Własne biblioteki na przykładzie Rest API i pythonowej biblioteki Requests
 - 3.2.3. Podejście do asercji w bibliotekach
 - 3.2.4. Pluginy
 - 3.3. Obsługa dockera
 - 3.3.1. Wykorzystanie biblioteki Docker SDK for Python
 - 3.3.2. Startowanie, restartowanie i zatrzymywanie
 - 3.3.3. Modyfikacja plików konfiguracyjnych wewnątrz dockera, na przykładzie konfiguracji serwera Nginx
 - 3.4. Środowisko Java
 - 3.4.1. Integracja z Mavenem
 - 3.4.2. Tworzenie bibliotek keywordów (Java)

Metoda realizacji szkolenia

Live coding: zaczynamy od zera a kończymy posiadając w pełni funkcjonalny i skalowalny projekt z dobrą architekturą, który może posłużyć jako baza dla kolejnych implementacji.

Liczba dni, liczba godzin szkoleniowych

3 dni, 21 godzin szkoleniowych

Ścieżka rozwoju po szkoleniu

[RobotFramework User guide](#)