« Outils de gestion des tests fonctionnels »

Les solutions de tests fonctionnels sont très variées sur le marché, regroupés selon certains critères notamment la licence, le langage de script, les technologies supportées et les fonctionnalités proposées.

1. Selenium

Selenium est un ensemble d'outils **open source** implémentés en **JavaScript** permettant d'automatiser les navigateurs web à des fins de tests. Selenium peut s'exécuter sur différents navigateurs et systèmes d'exploitation.

Selenium supporte une variété de langages de programmation et de scripts notamment C#, Java, JavaScript, PHP, Python et Ruby. Il offre la fonctionnalité record-playback qui permet d'enregistrer un scénario de tests de point de vue utilisateur et du le ré exécuter. Il est aussi possible de convertir le scénario enregistré en code.

Selenium dispose d'une large communauté d'utilisateurs et de développeurs permettant une contribution importante et des mises à jours régulières.

Afin de supporter le mieux les applications web complexes et modernes où des éléments peuvent changer sans que la page est rechargée, on a intégré WebDriver à Selenium 2.0 (alias Selenium WebDriver). WebDriver est un ensemble d'APIs offrant une interface de programmation simple et brève pour les tests fonctionnels.

Avec Selenium 2.0, le projet Selenium était responsable de fournir les drivers/extensions pour chaque navigateur. Maintenant avec la sortie de Selenium 3.0, la majorité des navigateurs fournissent leurs propres implémentations ce qui donne une meilleure expérience de test.

A partir de la version 47.0.1, FireFox propose son propre webDriver geckodriver. Le webDriver de Chrome est chromedriver et celui de Internet Explorer est IEDriverServer.

Selenium ne possède pas de fonctionnalité intégrée pour la génération des rapports, mais il y a des plugins qui pourraient ajouter cette fonctionnalité comme TestNG.

```
@Before
public void setUp() throws Exception {
   // resultList.add("Step Id, Action,Expected Result, Actual Result, Test Result");
    System.setProperty("webdriver.chrome.driver", "C:\\drivers\\chromedriver.exe");
  driver = new ChromeDriver();
  baseUrl = "http://localhost/";
  driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
  //resultList.add("1, Open Browser,Browser should open, Browser Opened, Pass");
}

@Test
public void testJunit1() throws Exception {
  driver.get(baseUrl + "/portal/intranet/home");
  driver.manage().window().maximize();
  driver.findElement(By.id("username")).clear();
  driver.findElement(By.id("username")).sendKeys("root");
  Thread.sleep(2000);
  driver.findElement(By.id("UIPortalLoginFormControl")).clear();
  driver.findElement(By.id("UIPortalLoginFormControl")).sendKeys("gtn");
  driver.findElement(By.cssSelector("button.button")).click();
  //resultList.add("2, login,login should be done, login done, Pass");
  driver.findElement(By.cssSelector("li.navItemSelected > a > span")).click();
  driver.findElement(By.cssSelector("a.spaceIcon.avatarMini > span")).click();
  driver.findElement(By.id("documents")).click();
}

@After
public void tearDown() throws Exception {

    //pdfUtility.WriteTestResultToPdfFile("TestResult.pdf", resultList);
    Thread.sleep(4000);
    driver.quit();
```

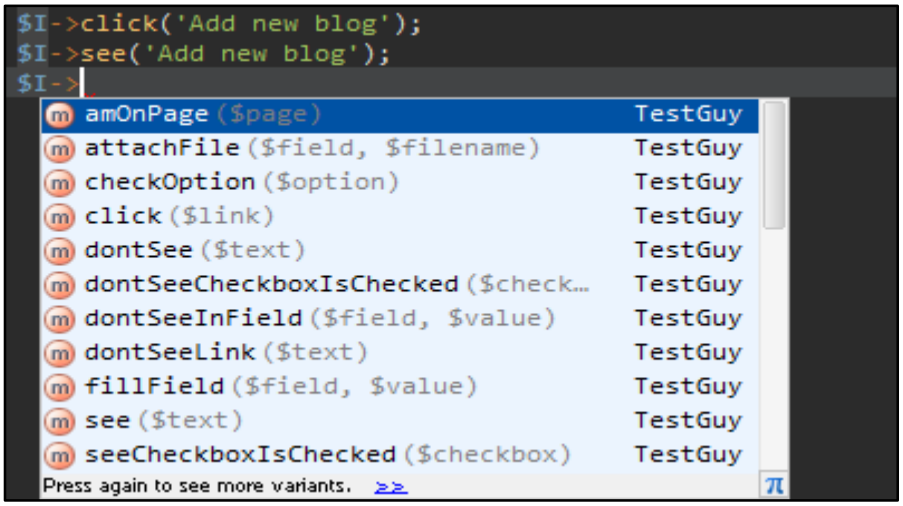
2. Codeception

Codeception est un full-stack framework moderne de tests **open source**. Il est écrit en **PHP**, basé sur **PHPUnit** et inspiré par l'approche **BDD**.

Codeception offre une nouvelle approche pour l'écriture des **tests d'acceptation**, les tests fonctionnels et même les tests unitaires.

Codeception est un outil open source qui profite d'une large communauté et des mises à jours régulières ainsi qu'une documentation riche et complète.

Les actions sont définies de point de vue utilisateur permettant une utilisation facile.



Il permet également de :

- Générer les rapports de tests en HTML, XML ou JSON.

- Rechercher les données requises, nettoie et remplie la base de données après chaque exécution.

- Exécuter des tests multi session.

- Interconnecter des plugins d'intégration continue comme **Jenkins**.

- Tester les scénarios soit par **PHPBrowser** soit par **Selenium WebDriver**.

- Visualiser les détails d'exécution et prendre des captures d'écran.

3. Windmill

Windmill est un framework de tests web **open source** qui offre une automatisation complète et une forte capacité de débogage. Il supporte la majorité des navigateurs et est disponible pour Windows, Mac OS X et Linux.

Windmill est développé en Python et **JavaScript**. Les tests peuvent être écrits en Python ou **JavaScript**. Il propose aussi la fonctionnalité **record-playback**. Il suffit juste d'enregistrer un scénario de test, le modifier et le ré-exécuter moyennant une interface.

Ses atouts sont:

- Un ensemble riche de commandes pour interagir avec l'application.

- Génération des rapports de tests.

- Support de SSL.

Cependant, Windmill n'est plus maintenu activement, il a été éclipsé par Webdriver/Selenium 2.

4. Sahi

Sahi est un outil disponible en deux versions – **open source** et Pro - qui permet l'automatisation des tests des applications web. Il est destiné aux testeurs en permettant une manipulation facile des applications web complexes avec beaucoup de contenu Ajax.

Sahi offre **record & playback**, identification intelligente des objets, un simple langage de script, attente automatique des événements tels que les événements **JavaScript** et la **génération de rapports**.

Il permet à l'utilisateur des outils à la fois puissants et simples pour accomplir des tests sur une variété de navigateurs et systèmes d'exploitation.

Certaines fonctionnalités sont offertes uniquement avec la version Pro notamment la customisation et la sauvegarde des rapports de tests dans la base de données, les captures d'écrans et l'automatisation des tests.

Contrairement à la version Pro qui bénéficie de mises à jours régulières, la version open source n'est pas activement maintenue.

```
_setValue(_textbox("user"), "test");
_setValue(_password("password"), "secret");
_click(_submit("Login"));
_setValue(_textbox("q"), "2");
_setValue(_textbox("q[1]"), "1");
_setValue(_textbox("q[2]"), "1");
_click(_button("Add"));
_click(_cell("Rs.200[1]"));
_assertExists(_textbox("total"));
_assert(_isVisible(_textbox("total")));
_assertEqual("1150", _textbox("total").value);
_click(_button("Logout"));
```

5. Behat

Behat est un **BDD** framework **open source** d'automatisation des tests pour **PHP**, son code source utilise fortement les composants de **Symfony 2**. Il est très adapté au développeurs PHP.

Behat est extensible de façon que chaque fonctionnalité peut être améliorée ou remplacée. Il profite de nombreuses extensions régulièrement mises à jours par ses contributeurs.

Les tests sont écrits en **Gherkin**, un langage spécifique créé pour décrire le comportement. Comme YAML et Python, Gherkin est un langage whitespace-oriented qui utilise l'indentation pour définir la structure.

```
Feature: Serve coffee
  In order to earn money
  Customers should be able to
  buy coffee at all times

  Scenario: Buy last coffee
    Given there are 1 coffees left in the machine
    And I have deposited 1 dollar
    When I press the coffee button
    Then I should be served a coffee
```

Gherkin est disponible dans plusieurs langues permettant ainsi d'écrire des user stories en utilisant des mots de la langue courante. Par exemple nous pouvons utiliser le mot Fonctionnalité au lieu de *Feature*.

6.Cucumber

Cucumber est un BDD framework open source d'automatisation des tests pour JAVA, son code source est écrit en ruby.

Il utilise les bibliothèques de selenium pour l'exécution des tests.

Les tests sont écrits en Gherkin et développés en java

exoTest.feature | ExoLoginTest_StepDef.java | TestRunner.java

```
Feature: exo application testing
Scenario: exo Login Test
Given user opens exo login page
Then user enters loginId
Then user enters password
And user clicks on signin button
And close the browser
```

```
public class ExoLoginTest_StepDef {

    WebDriver driver=null;
    @Given("^user opens exo login page$")
    public  void open_login_page() {

        System.setProperty("webdriver.chrome.driver", "C:\\drivers\\chromedriver.exe");
        driver = new ChromeDriver();
        driver.get("http://localhost:8080");
    }
    @Then("^user enters loginId$")
    public void user_enters_login(){
        driver.findElement(By.id("username")).sendKeys("root");
    }
```

exoTest.feature | ExoLoginTest_StepDef.java | TestRunner.java

```
package CucumberTest.CucumberTest;

import org.junit.runner.RunWith;

import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;

@RunWith(Cucumber.class)
@CucumberOptions(features="src\\test\\exoTest.feature")
public class TestRunner {

}
```

7. Étude comparative

Afin de distinguer les meilleurs outils, nous avons sélectionné des critères que nous avons jugé essentiels.

	Selenium	Codeception	Windmill	Sahi	Behat	Cucumber
Open source	Oui	Oui	Oui	Oui avec version *Pro*	Oui	Oui
Multi-navigateurs	Oui	Oui	Oui	Oui	Oui	Oui
Langage de développement	Java	PHP	Python	PHP	PHP	ruby
Langage de script/test	JavaScript PHP Python C# Java Ruby	PHP	JavaScript Python Ruby	JavaScript	Gherkin	Gherkin + java
Communauté active et mises à jour régulières	Importante	Importante	Faible	Faible	Bonne	Bonne
Record-playback	Oui	Oui	Oui	Oui	No	Non
Identification intelligente des objets	Non	Non	Non	Oui	Non	Non
Attente des événements (Ajax)	Oui	Oui	Non	Oui	Non	Non
Génération de rapports de tests personnalisés	Oui	Oui	Oui	Version *Pro* seulement	Plugin	Plugin
Exportation des rapports de tests	Oui	Oui	Non	Version *Pro* seulement	Plugin	
Plugins de solutions complémentaires	Oui	Oui	Oui	Version *Pro* seulement	Plusieurs	
Support SSL	Oui	Oui	Oui	Version *Pro* seulement	Non	