Tutorial

Objectif:

Utiliser l’outil “Selenium” et Eclipse pour tester l’interface utilisateur d’un site/appli web.

Etapes :

Lancer Eclipse, puis créer un nouveau projetMaven:

*File -> New->Other->Maven->Maven project.*

Unefenêtreapparaît :

Cocher les cases *Create a single project*et *Use default workspace location*, puis continuer sur la page suivante.

Completer ensuite les champs avec les valeurssuivantes:

**GroupId:** *TestRecetteIntegration.com*

**ArtefactId** *Selenium\_Maven\_Eclipse*

**Name:** *MonProjetSelenium*

Enfin, terminer en cliquantsur le bouton*Finish*.

Accéder à l’URL« *seleniumhq.org/download/maven.html* » et suivre les premières instructions afin de compléter le fichier *pom.xml*.

Aller dans le fichier pom.xml de votre projet.

Attention ! Il est nécessaire d’ajouter les balises <dependancies> autour de la dépendance ajoutée :

<dependencies>  
[Code obtenu sur seleniumhq.org]  
</dependencies>

Création de notre classe de test :

Créer une nouvelle classe dans *src/test*.

Vérifier que le package sélectionné est *src/test/java.*

Contenu de la classe :

**publicclass**premierTest {

@Test

**publicvoid**startWebDriver() {

//On crée ici un driver pour firefox

WebDriver driver = **new**FirefoxDriver();

//On envoi le driver vers l’url spécifié

driver.navigate().to("http://seleniumsimplified.com/");

// On vérifie que le titre de la page correspond bien

Assert.*assertTrue*(driver.getTitle()

.startsWith("Selenium Simplified"));

driver.close();

}

}

Ne pas oublier d’ajouter JUnit4 et les imports nécessaires.

**import**junit.framework.Assert;

**import**org.junit.Test;

**import**org.openqa.selenium.WebDriver;

**import**org.openqa.selenium.firefox.FirefoxDriver;

Ce test très simple devrait parfaitement fonctionner. Il ne fait que vérifier que le titre de la page appelée commence bien par « SeleniumSimplified ».

Il est donc possible de tester toutes sortes d’élément de l’interface utilisateur.

Pour ceci, utiliser les propriétés de l’objet *WebDriver*.

On peut également test que le titre ne commence pas par une certaine valeur :

Assert.*assertFalse*(driver.getTitle().startsWith("Ne commence pas par"));

On va maintenant vérifier le contenu de l’url :

Assert.*assertTrue*(driver.getCurrentUrl().equals(

"http://seleniumsimplified.com/"));

Nous allons voir, comment accéder à un élément de la page :

WebElement myDynamicElement = driver.findElement(By.*id*("post-1248"));

Ici on accede à un article de la page.

On peut maintenant manipuler l’élément :

Assert.*assertTrue*(myDynamicElement.getText().startsWith(

"Selenium WebDriver getAttribute nuance"));

L’article doit normalement commencer par la phrase ci-dessus.

Nous allons effectuer les tests suivants sur un autre site :

driver.navigate().to("http://www.w3schools.com/html/html\_css.asp");

On accède ensuite à un élément par sa classe :

myDynamicElement = driver.findElement(By.*className*("tryitbtn"));

Comme l’élément est un bouton, on peut cliquer dessus :

myDynamicElement.click();

Lorsque l’on lance nos tests, on peut vérifier si la bonne page est appelé par l’appuie sur le bouton.