TP2 Selenium IDE

1. Objectifs

Effectuer des tests d'intégration d'applications Web

Savoir utiliser Selenium IDE, enregistrer des tests, compléter des tests, etc.

2. Installation et premier test

Selenium est une suite logicielle permet d'effectuer des tests d'intégration d'applications Web. Selenium est basé sur Javascript et permet de manipuler le contenu d'une page (complétion des champs, click sur bouttons, etc.), d'effectuer des assertions sur des pages Web. Selenium est donc capable de manipuler la structure DOM (vous connaissez oui non ? si non, demandez !!), qu'elle soit connue (utilisation d'identifieurs statiques) ou non. 2 frameworks sont possibles:

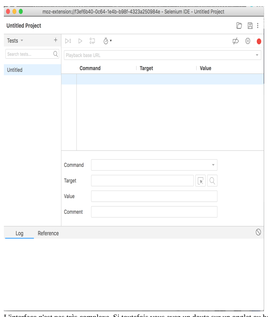
1. Selenium IDE qui est un plugin Firefox qui permet de faire du record and replay et de compléter avec des assertions les cas de tests enregistrés. Il est également possible de rentrer des cas de test complets à la main.
2. Selenium WebDriver qui permet de construire des cas de test uniquement via du code, code qui peut être écrit en Java, en PHP, etc.

Le fonctionnement de Selenium a été repris par d'autres outils, comme par exemple Robotium pour tester des applications Web (c'est exactement le même concept).

Dans ce TP, on va s'intéresser à Selenium IDE.

A

Pour installer Selenium IDE, lancez Firefox, allez sur ce lien et téléchargez Selenium IDE. Après un redemarrage, vous aurez un nouveau bouton et un nouveau lien dans le menu Outils. Lancez Selenium, vous obtenez cette fenêtre :



L'interface n'est pas très complexe. Si toutefois vous avez un doute sur un onglet ou bouton, regardez la doc !!!!

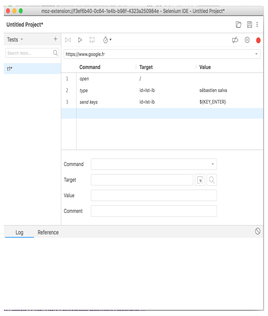
B

Nous allons créer un premier cas de test. Faites: *click "+" Add new test case* . Lancez l'enregistrement en cliquant sur le boutton rouge.

Puis sur firefox, entrez *www.google.fr* et dans le texte "votre nom prénom". Revenez sur Selenium IDE et arretez l'enregistrement (click boutton rouge).

Vous pouvez ensuite rejouer le test, à différentes vitesses.

Le cas de test est donné sous forme de tableau composé de commandes. Dans l'exemple ci-dessous, j'ai appuyé sur une touche pour faire la recherche mais j'aurai aussi pu cliqué sur le boutton de la page :



On a ici quelques commandes:

- open permet d'ouvrir un document donné en argument
- type indique que l'on ajoute du texte dans un élément HTML
- send keys permet de simuler une touche. Click correspond à l'action click sur un élément

La liste des commandes "classiques" est donnée ll.

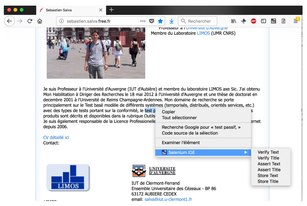
Les commandes sont de type:

- actions: telles que click, select. Ces actions ont souvent un suffixe *"AndWait"* pour ajouter une attente sur l'action (ex: attendre qu'une page se charge).
- accesseurs: actions sur l'état de l'application et sur des variables (stockage),
- assertion (et vérifications)

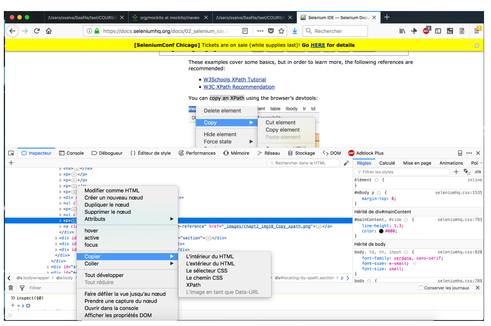
C

Le test précédent n'est pas réellement un test. Il manque les assertions !!
Il est possible de les ajouter à la main, MAIS il est également possible, pendant l'enregistrement, de faire un click droit et d'utiliser les commandes offertes.

Sur chaque page, vous pouvez faire un click droit, pour avoir une liste de commandes disponibles:



Malheureusement, tout ne fonctionne pas encore (cette version est récente et est mise à jour continuellement) . Vous devrez souvent entrer des cibles (target) et valeurs à la main. Pour vous aider, vous pouvez utiliser le debugger de Firefox.



Une fois que vous avez vos valeurs xpath ou css vous pouvez les ajouter à la main dans Selenium sous *target*.

Par exemple, si vous voulez utilisez xpath, vous devrez mettre dans Selenium xpath= ** un exemple** /html/body/div[4]/footer/p/a[1].

Avec css, css=** un exemple** img.responsive-img@href

Refaites un cas de test pour le site de test Google Gruyere. C'est un site exemple simple qui permet à un utilisateur de gérer des snippets. Auparavant lancez votre instance de site avec: http://google-gruyere.appspot.com/start et gardez cette instance.

Ici, je vous demande de créer un cas de test pour:

1. ouvrir le site,
2. vérifier que le titre est bien 'Gruyere: Home",
3. créer un compte toto/toto
4. vérifier que le contenu de la page obtenue affiche *Account created*.

2. Faire des cas de test plus complexes

A

[A tester, en cours de dev]Complétez le cas de test précédent pour vérifier que la page home contient les mots "cheesiest" et "brie". Pour l'instant, vous ne pouvez pas utiliser la commande *assertText*. Il faut également utiliser des expressions régulières. Vous pouvez en attendant utiliser ceci.

B

Ajoutez une assertion pour vérifier si "cantal" est dans la page Home.
Votre cas de test précédent doit retourner FAIL . A ce stade, le cas de test stoppe son exécution.
On peut toutefois utiliser d'autres commandes, à la place des assert, pour vérifier des propriétés et continuer le test, en utilisant les commandes de la famille *verify*.
Modifiez votre cas de test précédent en utilisant verifyText à la place de assertText. En relancant votre test, vous observerez qu'il s'exécutera jusqu'au bout.

C

Implantez un nouveau cas de test permettant de vous authentifier et d'ajouter un snippet. N'oubliez pas de faire les assertions.

D

On veut maintenant vérifier que sur la page Home que le snippet existe. Faites le cas de test.

Le problème est que ce cas de test est statique, il faut en créer un pour chaque nouvel enregistrement (bof bof). Utilisons des variables pour alléger le code et génériser.

Une variable avec le mot clé store. Voir ici.
Je vous invite à utiliser *execute script* qui permet d'utiliser du javascript pour manipuler les variables. Exemple:

	Command	Target	Value
1	*store*	1	x
2	*execute script*	${x}=parseInt(${x},10)+2; return ${x}	x
3	*echo*	${x}	
4	//		

Modifiez votre cas de test précédent pour stocker dans une variable le nom entré dans la page *registration* et vérifier que ce nom est présent dans la page *administration* en utilisant cette variable.

Si le tmeps le permet, amusez vous à entrer un snippet "`<script type='application/javascript'>alert('moi');</script>`". Que fait-on ? Si le résultat attendu n'apparait pas, c'est que votre navigateur bloque le script. REgardez ici

3. Suites de test

Si ce n'est pas encore fait, enregistrez vos cas de test !
Par défaut, l'ensemble des cas de test chargés sont dans l'onglet gauche, vous pouvez lancez un cas de test ou bien tous les cas de test.
Mais vous pouvez aussi construire une suite de test, en utilisant la Vue TestS uite

Créez une suite de test pour vos cas de test précédents.

4. Test d'attributs

Selenium IDE offre la possibilité de tester l'état de propriétés d'éléments de pages (le style, le href, le title, etc.). Il faut employer la commande store attribute avec ce genre d'attribut *xpath=//body/div[3]/div[1]/div[1]/div[1]/div/ol/li[3]/a@href*

Créez un cas de test permettant de vérifier que le lien Homepage de cheddar pointe vers https://images.google.com/?q=cheddar+cheese

5. Boucles

Après s'être connecté, on veut tester que le click des Homepage amene sur une nouvelle page. Au lieu de tester lien par lien, on peu utiliser une boucle lien par lien avec le xpath de type "/html/body/div[3]/table/tbody/tr[**XXXX**]/td[3]/a[2]".

La boucle se fait avec les mots clés while et end.