



Les applications Java modernes utilisent très souvent une base de donnée et des interfaces web dynamiques. Il faut donc gérer l'interface avec une base de données et la communication avec un serveur et des pages webs. L'aspect Base de données est souvent mis en oeuvre via JDBC et le second via des servlets et JSP (JavaServerPage).

Ce TP est consacré à voir comment utiliser le logiciel *httpunit* pour faire des tests sur les pages webs que peut générer une application Java. Nous nous limiterons à écrire des tests permettant de vérifier les composants d'une page, pas aux test effectifs d'une application permettant d'écrire/modifier/afficher.

1 Récupérer *httpunit*

Le site sur lequel se trouve *httpunit* est <http://httpunit.sourceforge.net/>

Récupérer les sources et installer *httpunit* dans un répertoire personnel. Vérifier que l'installation est correcte en lançant *ant run-example*. Cet exemple accède la page du site *meterware*, récupère le lien sur *httpunit* et compte le nombre de lien de cette page.

2 Principe d'utilisation

Httpunit peut s'utiliser dans un programme ou bien pour écrire des tests unitaires. Il fournit des classes et fonctions permettant d'accéder un site via son uri et de d'accéder aux composants de sa page.

- Créer un objet *wc* type *WebConversation*.
- Ensuite on peut interagir avec des sites locaux (fichier sur la machine) ou distants :
 - donner un uri de type *http ://....* fera appel au protocole http pour accéder à un site (hébergé sur une autre machine en général).
 - donner un uri de type *file ://localhost/....* amènera sur un fichier local.

Quelques classes et méthodes utiles :

- *WebConversation*
`getResponse(WebRequest r)`
- *WebRequest*
`setParameter(String name, String value)`
- *WebResponse*
`getElementNames(), getElementWithID(String id), getElementWithName(String n), getForms(), getFormWithID(String id), getFormWthName(String n)`
- *WebForm*
`submit(), getAction(), setParameter(String param, string s),getButtons(), setCheckBox(String n, Boolean state)`

- *WebLink*
`asText()`, `click()`, `getURLString()`

3 Vérifier une page web

Quelques bases (à compléter en lisant la doc)

- `wc.getResponse(String)` donne un objet de type `WebResponse` qui correspond à l'adresse donnée en argument. C'est une page formée de texte structurée éventuellement en *table*, *form*,... contenant des liens, images,...
- `getText()` récupère le texte de l'objet
- `getTables()[i]` donne la *i*ème table de l'objet et lui appliquer la méthode `asText()` retourne le texte.

4 Ecrire des tests

Il s'agit d'apprendre à effectuer des tests sur des pages ou des fichiers.

1. Récupérer le fichier *BasicExampleTest.java* sur le site du cours et modifier le fichier utilisé en mettant votre page personnelle. Epliquer pourquoi le rattrapage d'exception contient une clause *fail()*
2. Copier le fichier

```
http://www.lif.univ-mrs.fr/~lugiez/Enseignement/  

Master2/FIABILITE1/fiabilite1.html
```

dans sur votre machine dans le répertoire *public.html*. Ecrire des tests pour

- Vérifier que nombre maximal de rangées est 3.
 - Vérifier que le premier argument de la première rangée est *Cours/TD*, celui de la deuxième est la chaîne vide et celui de la troisième est *TP*
3. Ecrire un test pour vérifier que la page `http://tomcat.apache.org/index.html` contient un lien avec le nom *apache* et que la page obtenue en suivant ce lien contient xxx liens (xxx une valeur à récupérer via le test).
 4. Le type d'un formulaire est *WebForm* et le tableau récupérant ces formulaires est obtenu par `getForms()`. Ecrire un test pour vérifier le premier formulaire situé en `http://www.lif.univ-mrs.fr/~lugiez/public_html/Enseignement/
Master2/FIABILITE1/Progs/testform.html`
en testant
 - que l'avis émis dans le champ texte de nom *Avis* est "Tres bon",

- que l’option est FSI,
 - que le champ Reussi est coché.
- (Voir le cookbook de la distribution *Httpunit* pour voir comment faire)

5 Travail à rendre

Toutes les classes de Test réalisés ainsi que les résultats obtenus (à rajouter en commentaire du fichier). Le résultat sera mis dans un fichier archive .zip dont le nom est

`NOM DU CHEF DU PROJET-identifiant du projet.zip`

- Ce fichier contiendra les sources des classes les réponses aux questions étant sous forme de commentaire.
- L’en-tête du fichier contiendra les noms des membres du projets en identifiant clairement le responsable ainsi qu’une description succincte du travail demandé.
- L’identifiant de ce projet est tpweb.