HtmlUnit

Técnicas de documentación y validación

TUDAI - UNICEN

¿Qué es HtmlUnit?

- HtmlUnit es un navegador web sin interfaz de usuario escrito en Java.
- Permite la manipulación de alto nivel de sitios web a partir de otro código Java, incluyendo el llenado y envío de formularios y el hacer click en hiperenlaces.
- Proporciona acceso a la estructura y los detalles de las páginas web recibidas.
- La versión 2.0 incluye muchas mejoras nuevas, como una puesta en práctica de las características de la implementación W3C DOM Java 5, un mejor soporte XPath, y un mejor manejo de HTML incorrecto, además de diversas mejoras en JavaScript, mientras que la versión 2.1 se centra principalmente en afinar algunos problemas de rendimiento reportados por los usuarios.

Características

- Soporte para HTTP y HTTPS y para cookies.
- Soporte para submit methods POST, GET, etc.
- Es *wrapper* de páginas HTML y provee acceso a los datos de la misma.
- Permite navegar por el DOM model de HTML
- Soporte para Proxy
- Soporte JavaScript.
- Es posible combinarlo con JUnit para armar test automatizados.

Introducción

```
@Test
public void homePage() throws Exception {
    try (final WebClient webClient = new WebClient()) {
        final HtmlPage page = webClient.getPage("http://www.google.com");
        Assert.assertEquals(page.getTitleText(),"Google");

        final String pageAsXml = page.asXml();
        Assert.assertTrue(pageAsXml.contains("<body class=\"searchform\">"));
    }
}
```

Imitando a un navegador específico

- ```
  @Test
  public void homePage_Firefox() throws Exception {
      try (final WebClient webClient = new WebClient(BrowserVersion.FIREFOX_52)) {
          final HtmlPage page = webClient.getPage("http://google.com.ar");
          Assert.assertEquals(page.getTitleText(),"Google");
      }
  }
  ```
  ```

Encontrando un elemento específico

- ```
  @Test
  public void getElements() throws Exception {
      try (final WebClient webClient = new WebClient()) {
          final HtmlPage page = webClient.getPage("http://www.google.com");
          final HtmlDivision div = page.getHtmlElementById("id_del_elemento");
      }
  }
  ```
  ```

Encontrando un elemento específico

- Una forma simple de encontrar elementos es por un tipo específico.

```
@Test
public void getElements() throws Exception {
    try (final WebClient webClient = new WebClient()) {
        final HtmlPage page = webClient.getPage("http://some_url");
        NodeList inputs = page.getElementsByTagName("nombre");
        final Iterator<E> nodesIterator = nodes.iterator();
        // iterar
    }
}
```

Encontrando un elemento específico

- **XPath** es la forma sugerida para hacer búsquedas complejas en el DOM.

```
@Test
public void getElements() throws Exception {
   try (final WebClient webClient = new WebClient()) {
       HtmlSpan spam = (HtmlSpan)  page.getFirstByXPath("//*[@id=\"result_box\"]/span");
 }
}
```

XPath (XML Path Language) es un lenguaje que permite construir expresiones que recorren y procesan un documento XML. La idea es parecida a las expresiones regulares para seleccionar partes de un texto sin atributos (plain text). XPath permite buscar y seleccionar teniendo en cuenta la estructura jerárquica del XML.

Usar un proxy

- try (final WebClient webClient = new WebClient(BrowserVersion.CHROME, "proxy.exa.unicen.edu.ar", 8080)).

  El browser Cliente puede ser de distinto tipo: Chrome, firefox, etc...

Veamos un ejemplo

```
@Test
public void searchInGoogle() throws Exception {
try (final WebClient webClient = new WebClient(BrowserVersion.CHROME,"proxy.exa.unicen.edu.ar",
8080))
{
        HtmlPage page = webClient.getPage("http://www.google.com");
        Assert.assertEquals(page.getTitleText(),"Google");
        System.out.println(page.getTitleText());
        final HtmlTextInput textField = (HtmlTextInput) page.getElementById("lst-ib");
        textField.setValueAttribute("Facebook");
        final HtmlSubmitInput searchButton = (HtmlSubmitInput) page.getElementByName("btnK");
        page = searchButton.click();
        System.out.println(page.asText());
}}
```