DZone®
A DEVADA MEDIA PROPERTY

HtmlUnit: A Quick Introduction

by Avi Yehuda MVB · May. 20, 10 · Java Zone · Not set

HtmlUnit is an open source java library for creating HTTP calls which imitate the browser functionality. HtmlUnit is mostly used for integration testing on top of unit test frameworks such as JUnit or TestNG. This is done by requesting web pages and asserting the results.

Simple Example

```
@Test
public void testGoogle(){
        WebClient webClient = new WebClient();
        HtmlPage currentPage = webClient.getPage("http://www.google.com/");
        assertEquals("Google", currentPage.getTitleText());
}
```

WebClient

- As you can see in the example, the WebClient is the starting point. It is the browser simulator.
- WebClient.getPage() is just like typing an address in the browser. It returns an HtmlPage object.

HtmlPage

- HtmlPage represents a single web page along with all of it's client's data (HTML, JavaScript, CSS ...).
- The HtmlPage lets you access to many of a web page content:

Page source

- You can receive the page source as text or as XML.

```
HtmlPage currentPage =
                webClient.getPage("http://www.google.com/");
String textSource = currentPage.asText();
String xmlSource = currentPage.asXml();
```

HTML Elements

- HtmlPage lets you ability to access any of the page HTML elements and all of their attributes and sub elements. This includes tables, images, input fields, divs or any other Html element you may imagine.
- Use the function **getHtmlElementById()** to get any of the page elements.

```
WebClient webClient = new WebClient();
HtmlPage currentPage = webClient.getPage("http://www.google.com/");
HtmlImage imgElement = (HtmlImage)currentPage.getHtmlElementById("logo");
System.out.println(imgElement.getAttribute("src"));
```

Anchors

- Anchor is the representation of the Html tag *<a href="..." >link</a>*.
- Use the functions **getAnchorByName()**, **getAnchorByHref()** and **getAnchorByText()** to easily access any of the anchors in the page.

```
WebClient webClient = new WebClient();
HtmlPage currentPage = webClient.getPage("http://www.google.com/");
HtmlAnchor advancedSearchAn =
                currentPage.getAnchorByText("Advanced Search");
currentPage = advancedSearchAn.click();
assertEquals("Google Advanced Search",currentPage.getTitleText());
```

Dom elements by XPath

- You can access any of the page elements by using XPath.

```
WebClient webClient = new WebClient();
```

```
HtmlPage currentPage =
        webClient.getPage("http://www.google.com/search?q=avi");

//Using XPath to get the first result in Google query
HtmlElement element = (HtmlElement)currentPage.getByXPath("//h3").get(0);
DomNode result = element.getChildNodes().get(0);
```

Form control

- A large part of controlling your HTML page is to control the form elements:
    - HtmlForm
    - HtmlTextInput
    - HtmlSubmitInput
    - HtmlCheckBoxInput
    - HtmlHiddenInput
    - HtmlPasswordInput
    - HtmlRadioButtonInput
    - HtmlFileInput

```
WebClient webClient = new WebClient();
HtmlPage currentPage = webClient.getPage("http://www.google.com/");

//Get the query input text
HtmlInput queryInput = currentPage.getElementByName("q");
queryInput.setValueAttribute("aviyehuda");

//Submit the form by pressing the submit button
HtmlSubmitInput submitBtn = currentPage.getElementByName("btnG");
currentPage = submitBtn.click();
```

Tables

```
currentPage = webClient.getPage("http://www.google.com/search?q=htmlunit");
final HtmlTable table = currentPage.getHtmlElementById("nav");
for (final HtmlTableRow row : table.getRows()) {
System.out.println("Found row");
    for (final HtmlTableCell cell : row.getCells()) {
       System.out.println("   Found cell: " + cell.asText());
    }
}
```

JavaScript support

- HtmlUnit uses the Mozilla Rhino JavaScript engine.

- HtmlUnit uses the Mozilla Rhino JavaScript engine.
- This lets you the ability to run pages with JavaScript or even run JavaScript code by command.

```
ScriptResult result = currentPage.executeJavaScript(JavaScriptCode);
```

- By default JavaScript exceptions will crash your tests. If you wish to ignore JavaScript exceptions use this:

```
webClient().setThrowExceptionOnScriptError(false);
```

- If you would like to turn off the JavaScript all together, use this:

```
currentPage.getWebClient().setJavaScriptEnabled(false);
```

HTTP elements

URL

```
WebClient webClient = new WebClient();
HtmlPage currentPage =
        webClient.getPage("http://www.google.co.uk/search?q=htmlunit");
URL url = currentPage.getWebResponse().getRequestSettings().getUrl()
```

Response status

```
WebClient webClient = new WebClient();
HtmlPage currentPage = webClient.getPage("http://www.google.com/");
assertEquals(200,currentPage.getWebResponse().getStatusCode());
assertEquals("OK",currentPage.getWebResponse().getStatusMessage());
```

Cookies

```
Set cookies = webClient.getCookieManager().getCookies();
for (Cookie cookie : cookies) {
        System.out.println(cookie.getName() + " = " + cookie.getValue());
}
```

Response headers

```
WebClient webClient = new WebClient();
HtmlPage currentPage =
        webClient.getPage("http://www.google.com/search?q=htmlunit");

List headers =
        currentPage.getWebResponse().getResponseHeaders();
        currentPage.getWebResponse().getResponseHeaders();
for (NameValuePair header : headers) {
        System.out.println(header.getName() + " = " + header.getValue());
}
```

Request parameters

```
List parameters =
        currentPage.getWebResponse().getRequestSettings().getRequestParameters();
for (NameValuePair parameter : parameters) {
        System.out.println(parameter.getName() + " = " + parameter.getValue());
}
```

Making assertions

- HtmlUnit comes with a set of assetions:

```
   assertTitleEquals(HtmlPage, String)
   assertTitleContains(HtmlPage, String)
   assertTitleMatches(HtmlPage, String)
   assertElementPresent(HtmlPage, String)
   assertElementPresentByXPath(HtmlPage, String)
   assertElementNotPresent(HtmlPage, String)
   assertElementNotPresentByXPath(HtmlPage, String)
   assertTextPresent(HtmlPage, String)
   assertTextPresentInElement(HtmlPage, String, String)
   assertTextNotPresent(HtmlPage, String)
   assertTextNotPresentInElement(HtmlPage, String, String)
   assertLinkPresent(HtmlPage, String)
   assertLinkNotPresent(HtmlPage, String)
   assertLinkPresentWithText(HtmlPage, String)
   assertLinkNotPresentWithText(HtmlPage, String)
   assertFormPresent(HtmlPage, String)
   assertFormNotPresent(HtmlPage, String)
   assertInputPresent(HtmlPage, String)
   assertInputNotPresent(HtmlPage, String)
   assertInputContainsValue(HtmlPage, String, String)
   assertInputDoesNotContainValue(HtmlPage, String, String)
```

- You can still of course use the framework's assertions. For example, if you are using JUnit, you can still use assertTrue() and so on.
- Here are a few examples:

```
WebClient webClient = new WebClient();
HtmlPage currentPage =
      webClient.getPage("http://www.google.com/search?q=htmlunit");

assertEquals(200,currentPage.getWebResponse().getStatusCode());
assertEquals("OK",currentPage.getWebResponse().getStatusMessage());
WebAssert.assertTextPresent(currentPage, "htmlunit");
WebAssert.assertTitleContains(currentPage, "htmlunit");
WebAssert.assertLinkPresentWithText(currentPage, "Advanced search");
assertTrue(currentPage.getByXPath("//h3").size()>0); //result number
assertNotNull(webClient.getCookieManager().getCookie("NID"));
```

See also

- Product main page
- Download page
- Jar dependencies

Download example

from http://www.aviyehuda.com/

Like This Article? Read More From DZone

Free DZone Refcard

Java Performance Optimization

Topics:

Opinions expressed by DZone contributors are their own.

ABOUT US
About DZone
Send feedback
Careers

ADVERTISE
Developer Marketing Blog
Advertise with DZone
+1 (919) 238-7100

CONTRIBUTE ON DZONE
MVB Program
Zone Leader Program

CONTACT US
600 Park Offices Drive
Suite 150
Research Triangle Park, NC 27709
support@dzone.com
+1 (919) 678-0300