

Selenium Monitor

What's the story behind the Selenium Monitor?

The SeleniumMonitor similar was created to handle tracking sequences on websites that rely heavily on Javascript.

The monitor uses the Selenium API to run regression testing through the browser on the supplied url. The only requirement is that you have installed Firefox 3.6 installed on the instance you are running OpenNMS

Requirements

- Firefox 3.6 installed on the same instance as OpenNMS is running

* Firefox 3.6 is the version I personally tested, try other versions at your own risk.

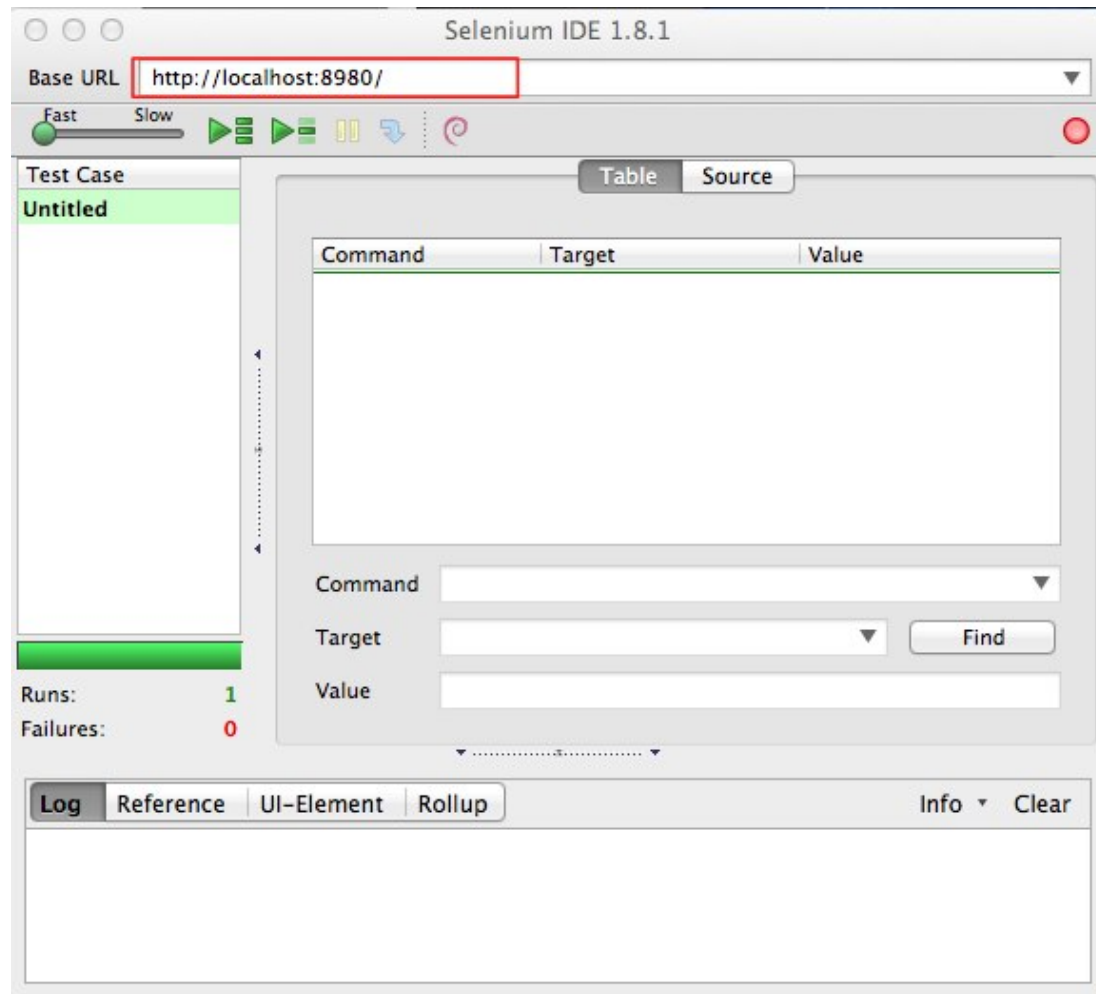
Install the Selenium IDE

- Install the Selenium IDE add-on for Firefox

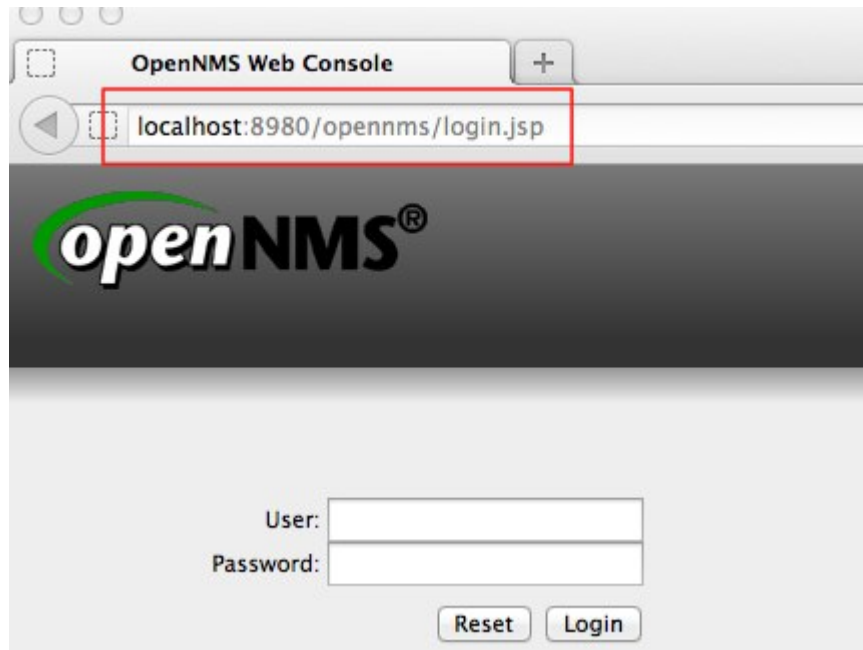
- <http://seleniumhq.org/download/>

*** Because Firefox decided to do these rapid releases the Selenium IDE works on the latest Firefox, version 13, but the Selenium Monitor currently has not been updated to run on that version. I know its a pain to have two instances of Firefox installed but currently this is how it works.

Creating a Selenium Test with the Selenium IDE



Set the base url for the test



Navigate to <http://localhost:8980>

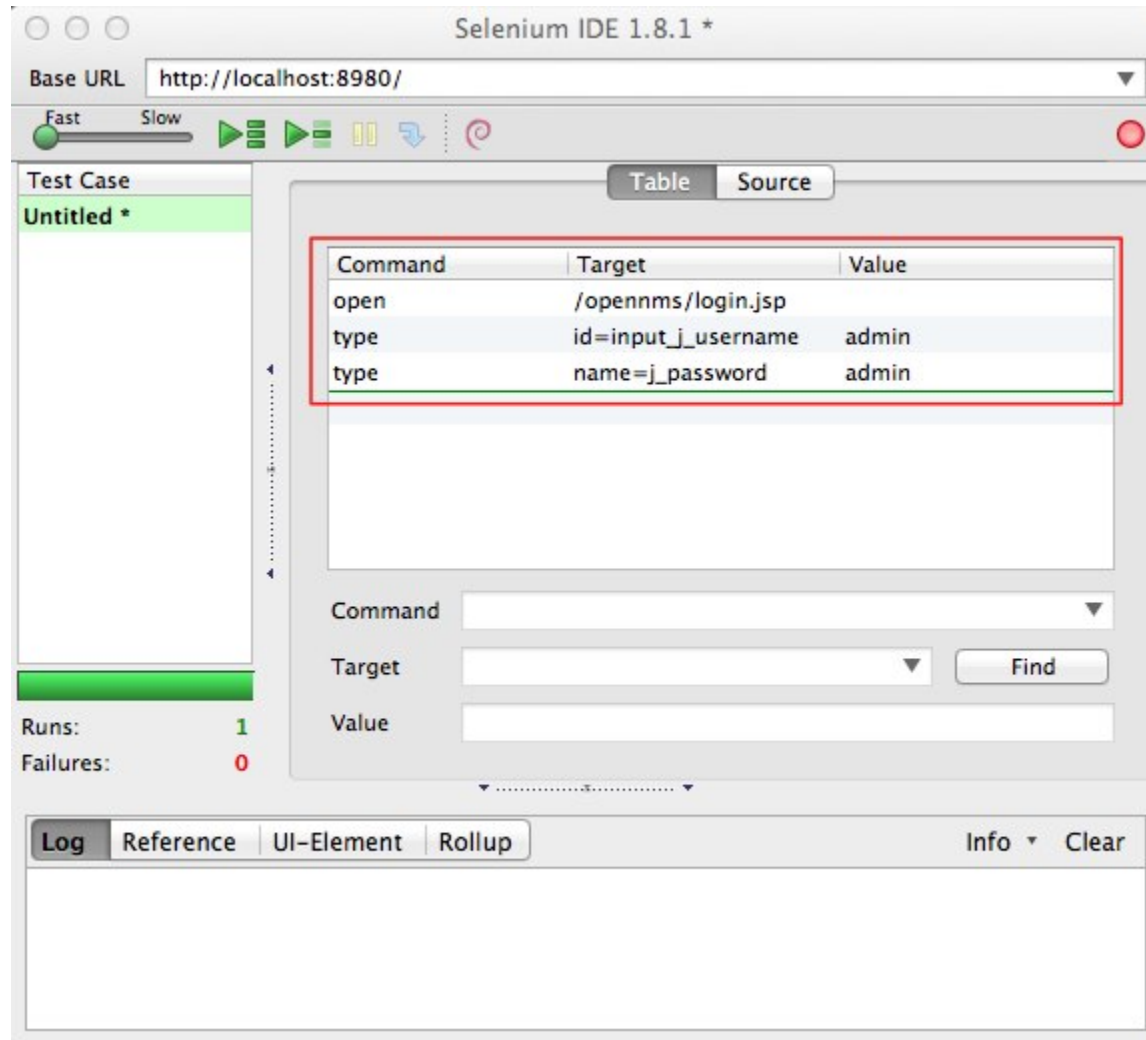


openNMS[®]

User:

Password:

Type in the admin user name and password in the web ui



Check out the Selenium IDE and verify that it has captured the login correctly

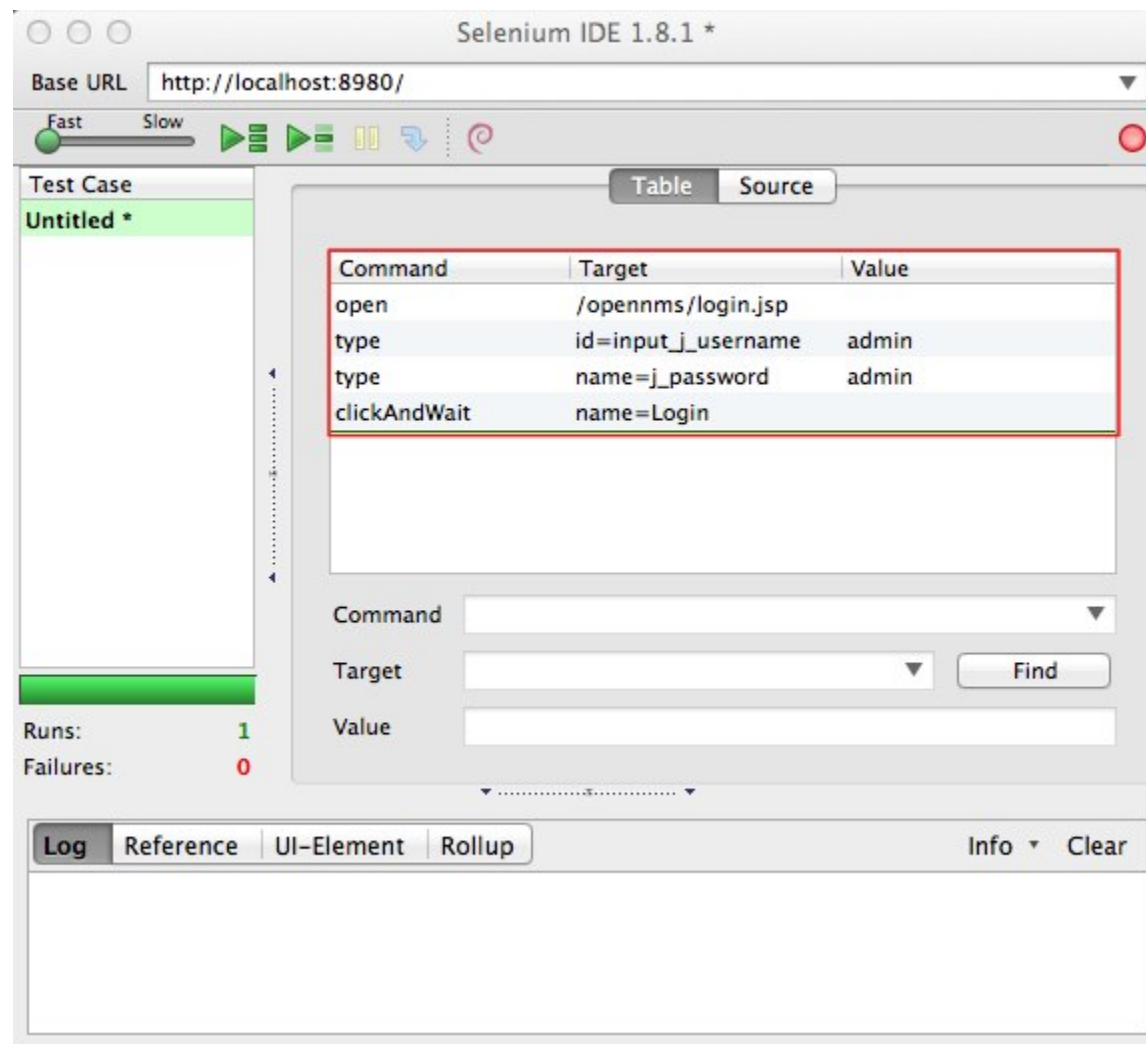


openNMS[®]

User:

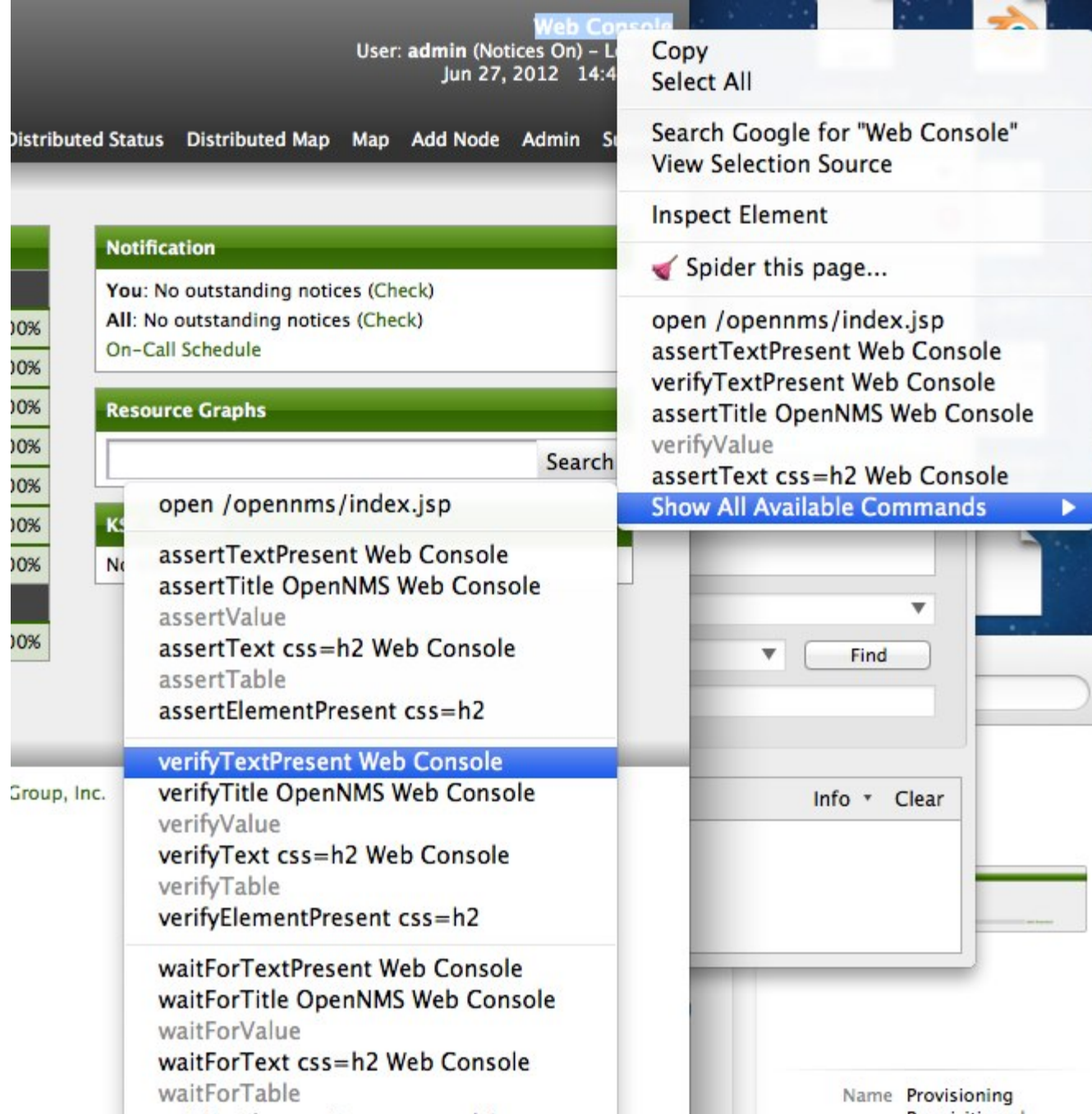
Password:

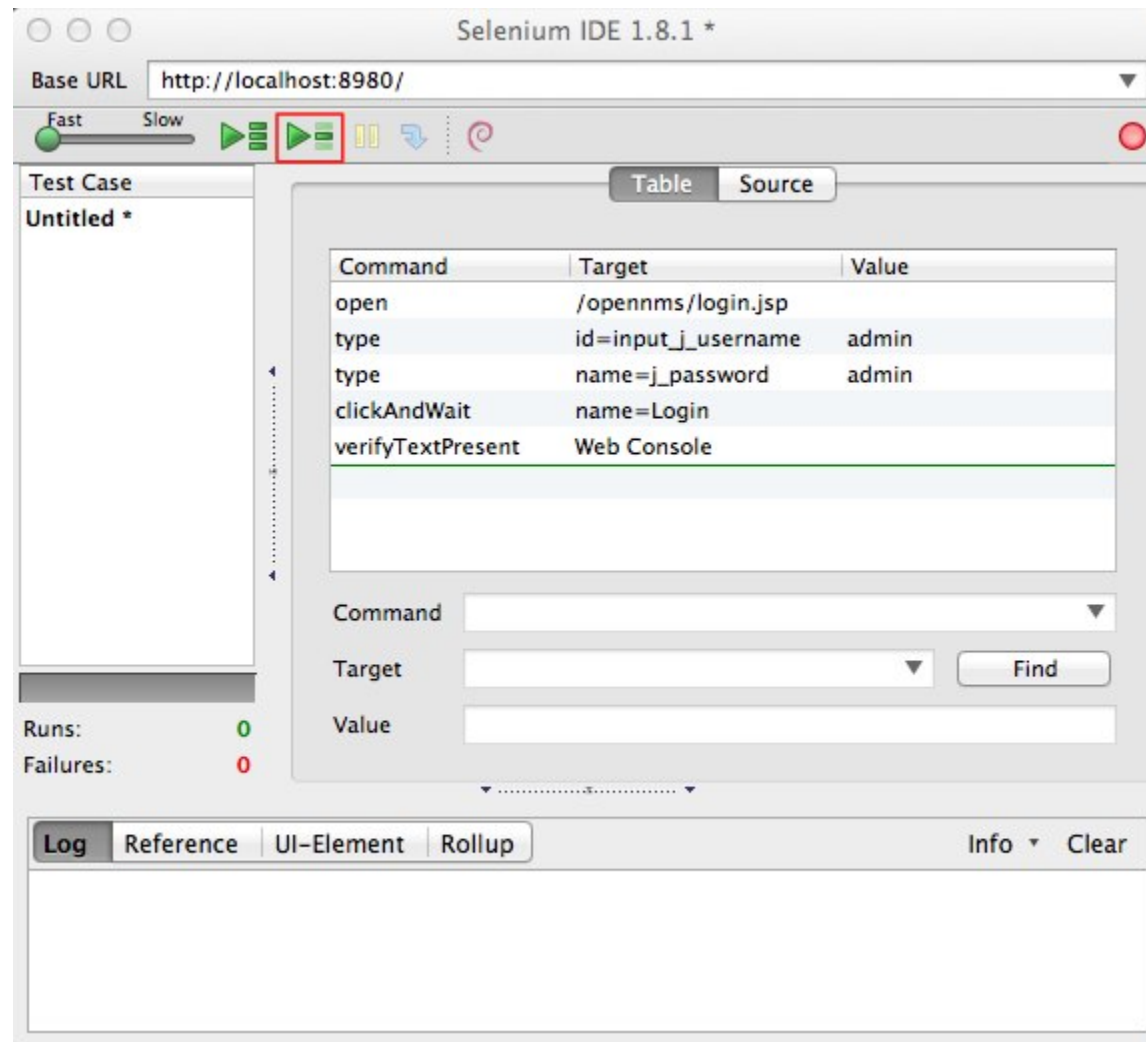
Click login



Look Selenium IDE recorded our action,
Woot!!

Right click on the text
"Web Console" and
select the
verifyTextPresent
command





Run your test in Selenium IDE make sure it works.

Exporting the Selenium IDE test

To get your Selenium IDE tests into OpenNMS you will need to export it as a JUnit4 (Webdriver) test.

*Save the file with .groovy extension not a java extension.

Updating the Selenium Groovy test

There are a few lines of code you will need to update in the Groovy file you just exported to get this working in OpenNMS.

- Change the package
- Create a Constructor
- Add an import
- Add timeout code if necessary
- Adjust the assert code

All in all its very minor.

Changing the package

Change the package of the Groovy file from

```
package com.example.tests;
```

to:

```
package selenium;
```

Create a Constructor

Add the following code above the setup method.

```
public OpennmsSeleniumExample(String url, int timeoutInSeconds) {  
    baseUrl = url;  
    timeout = timeoutInSeconds;  
}
```

The name of the method must match the name of the Groovy file. Indicated by the highlighted text.

Adding the Import

You will need to add the following import.

```
import java.util.concurrent.TimeUnit;
```

Adding the Fields

You will need to add a timeout field and change a line of code in the test to use the timeout we set in the constructor.

Add a timeout field at top of the class under the baseUrl field of type int.

```
private String baseUrl = "";  
private int timeout = 30;
```

Change the Assert code

Sometimes the SeleniumIDE when exporting to a format doesn't know how to create the assert and we need to help it out. Probably the only bummer.

In the Unit Test add the following code at the bottom of the method.

```
assertEquals("Web Console", driver.findElement(By.tagName("h2")).getText());
```

Configuring OpenNMS for the Selenium Monitor

- Edit the poller-configuration.xml
 - Add the Selenium Monitor

```
<service name="OpenNMS-Site-Login" interval="300000" user-defined="false" status="on">
  <parameter key="retry" value="1" />
  <parameter key="timeout" value="3" /> <!-- timeout in seconds -->
  <parameter key="rrd-repository" value="/opt/opennms/share/rrd/response" />
  <parameter key="rrd-base-name" value="selenium-service" /> <!-- change name accordingly -->
  <parameter key="ds-name" value="selenium-service" /> <!-- change name accordingly -->
  <parameter key="base-url" value="http://${ipAddr}" /> <!-- using monitored service's ip address-->
  <parameter key="selenium-test" value="OpennmsSeleniumExample.groovy" />
</service>
```

Add the monitor service.

```
<monitor service="OpenNMS-Site-Login" class-name="org.opennms.netmgt.poller.monitors.SeleniumMonitor" />
```

Customizing the settings in Poller config

base-url

You have two options when it comes to setting the base-url, you can either set a static base-url or you can set it to use the monitored service's ip address to monitor.

Static url

```
<parameter key="base-url" value="http://google.com" />
```

Monitored Service IP address

```
<parameter key="base-url" value="http://${ipAddr}" />
```

Using https

```
<parameter key="base-url" value="https://${ipAddr}" />
```

Adding a port

```
<parameter key="base-url" value="http://${ipAddr}:8080" />
```

selenium-sequence

This property is the filename of the Groovy JUnit test sequence you want to run. The default directory for selenium sequences is **\$OPENNMS_HOME/etc/selenium** .

```
<parameter key="selenium-test" value="OpennmsSeleniumExample.groovy" />
```

Finally the Good part

- Provisioning a node for the Selenium Monitor
 - Add an http detector with a same name as the service name in our poller-configuration.xml
- Watch the selenium monitor do its job.

Provisioning the node

- Create a provisioning group.
 - Under Foreign Source I removed all detectors and added a new HttpDetector, removing the other detectors is optional. The only requirement is that you name the detector the same name as the service name in poller-configuration.xml



Home / Admin / Provisioning Requisitions / Edit Foreign Source Definition

Foreign Source Name: MyTestGroup

Done

Scan Interval 1d

Detectors Add Detector

•   name OpenNMS-Site-Login class org.opennms.netmgt.provision.detector.simple.HttpDetector [Add Parameter]

Complete the Provisioning process

- Add a node for your system.
- Add the 127.0.0.1 interface
- Add the OpenNMS-Site-Login service to the interface
- Exit and sync the provision group.

**Now watch the SeleniumMonitor
work**