How Selenium WebDriver works:

Let’s start with remote WebDriver.

JsonWireProtocol is the base of WebDriver architecture.

It is the backbone of WebDriver and understanding the protocol helps us understand how Selenium works.

The selenium client works by sending request to selenium server through the wired protocol, which is a REST based service. The client and server can remain in two separate machines connected over a network, be it LAN or WWW.

We can find the documentation of JsonWireProtocol in the below website:

<https://code.google.com/p/selenium/wiki/JsonWireProtocol>

also there is <http://www.w3.org/TR/webdriver/>

I am using few tools in this session.

Browsers: chrome, Firefox.

Browser plugins: Postman for chrome and firebug for Firefox.

The simple way to start Selenium WebDriver is by downloading the Selenium stand alone server and using the command “java -jar selenium-server-standalone-2.47.1.jar” from command prompt or terminal of your machine.

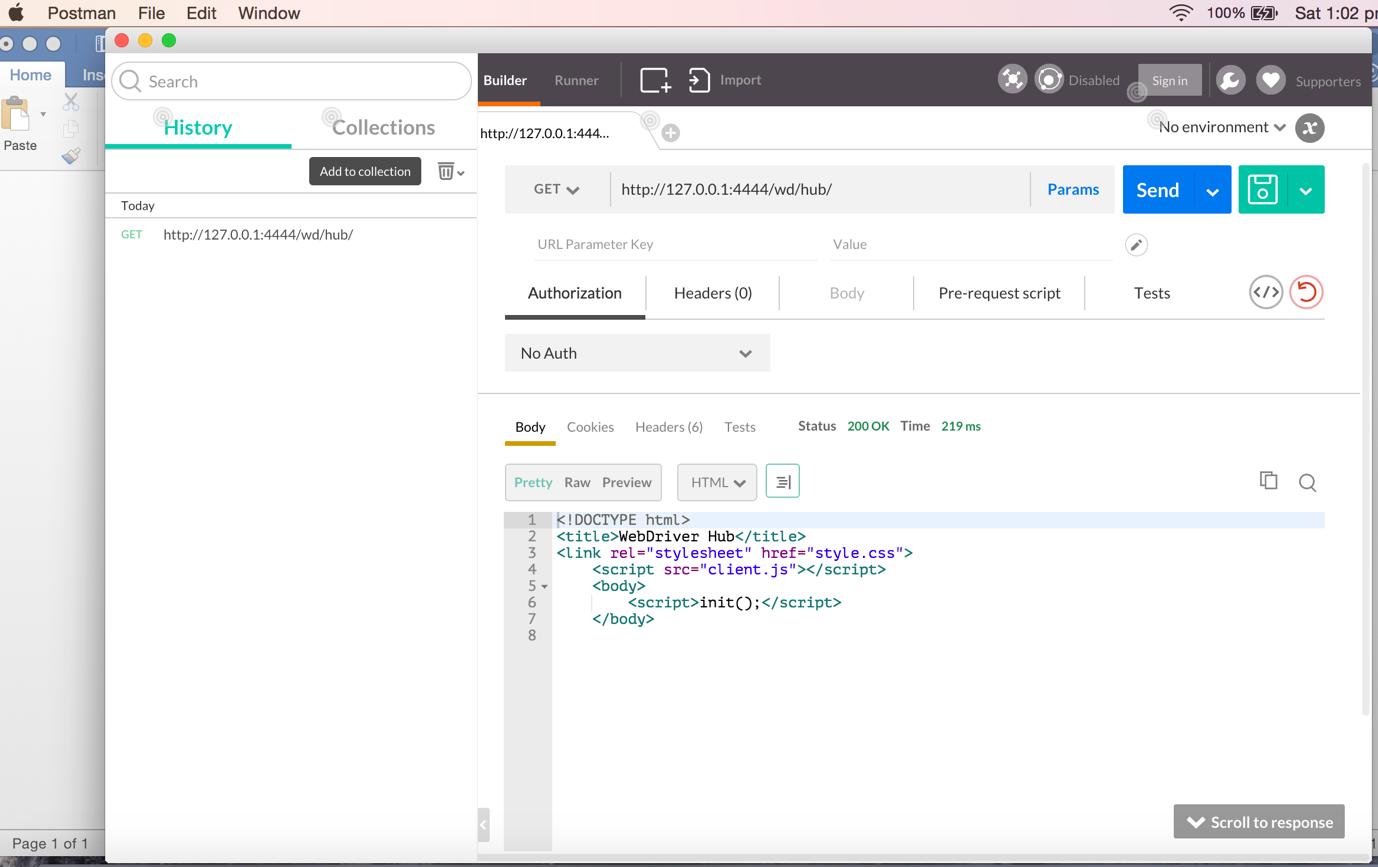
Eg:

192-168-1-3:Downloads rakeshprabhakaran$ java -jar selenium-server-standalone-2.47.1.jar

This will start the server in localhost “4444” port of your machine.

<http://127.0.0.1:4444/wd/hub/>

<http://127.0.0.1:4444/wd/hub/>



Now let’s create a new session:

As per the protocol document:

**/session**

**POST /session**

Create a new session. The server should attempt to create a session that most closely matches the desired and required capabilities. Required capabilities have higher priority than desired capabilities and must be set for the session to be created.

**JSON Parameters:**

desiredCapabilities - {object} An object describing the session's desired capabilities.

requiredCapabilities - {object} An object describing the session's required capabilities (Optional).

**Returns:**

{object} An object describing the session's capabilities.

**Potential Errors:**

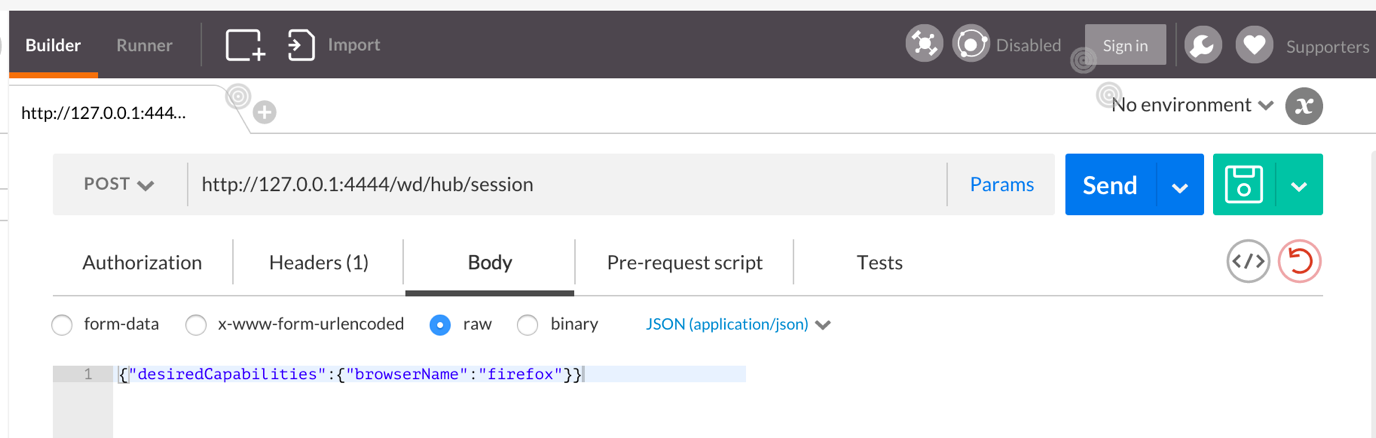
SessionNotCreatedException - If a required capability could not be set.

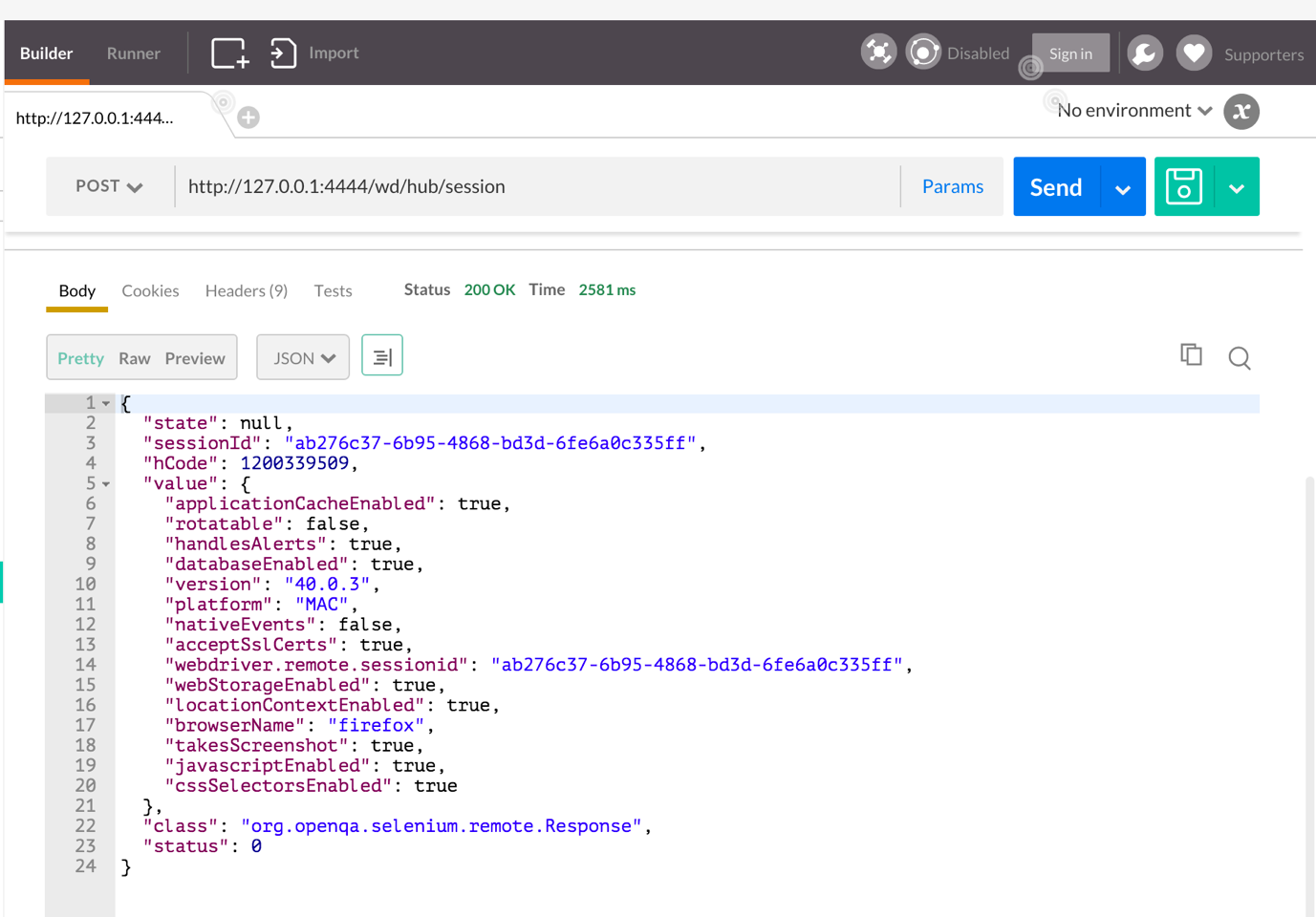
Let’s create the session for browser ‘firefox’, the capabilities required are as below,

I was able to understand this by intercepting the create request from Firefox:

|  |
| --- |
| {"desiredCapabilities":{"browserName":"firefox"}}  To find it I loaded the url  <http://127.0.0.1:4444/wd/hub/>  in a browser and clicked on the create session button, meanwhile I used firebug’s ‘net’ tab to view the JSON message to the server. |
|  |

Let’s replicate a session create post request through the “postman” plugin of chrome.





Session log from command prompt in which server is running:

14:19:07.154 INFO - Creating a new session for Capabilities [{browserName=firefox}]

14:19:09.726 INFO - Done: [new session: Capabilities [{browserName=firefox}]]

and the JSON response from server is:

{

"state": null,

"sessionId": "ab276c37-6b95-4868-bd3d-6fe6a0c335ff",

"hCode": 1200339509,

"value": {

"applicationCacheEnabled": true,

"rotatable": false,

"handlesAlerts": true,

"databaseEnabled": true,

"version": "40.0.3",

"platform": "MAC",

"nativeEvents": false,

"acceptSslCerts": true,

"webdriver.remote.sessionid": "ab276c37-6b95-4868-bd3d-6fe6a0c335ff",

"webStorageEnabled": true,

"locationContextEnabled": true,

"browserName": "firefox",

"takesScreenshot": true,

"javascriptEnabled": true,

"cssSelectorsEnabled": true

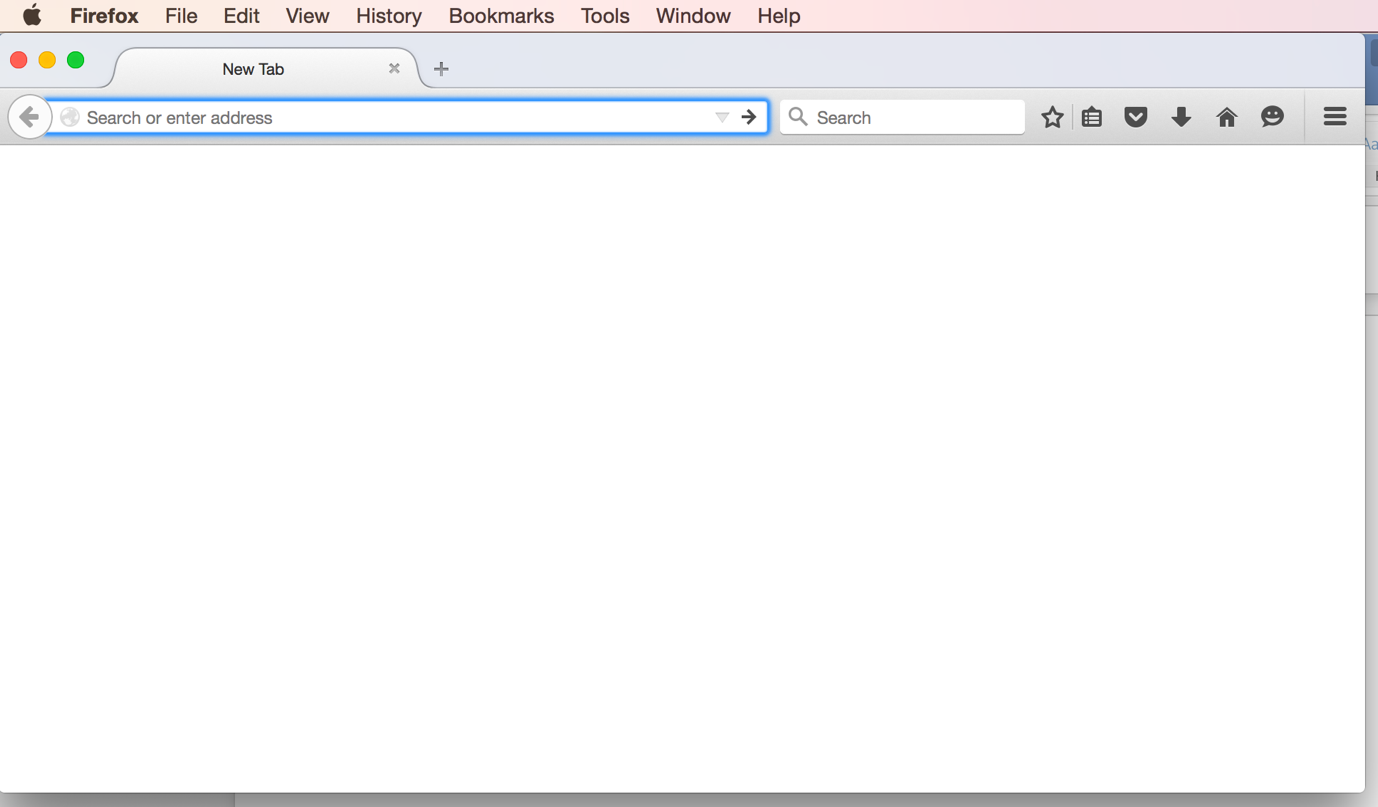
},

"class": "org.openqa.selenium.remote.Response",

"status": 0

}

Now I have a new Firefox opened up in my machine :



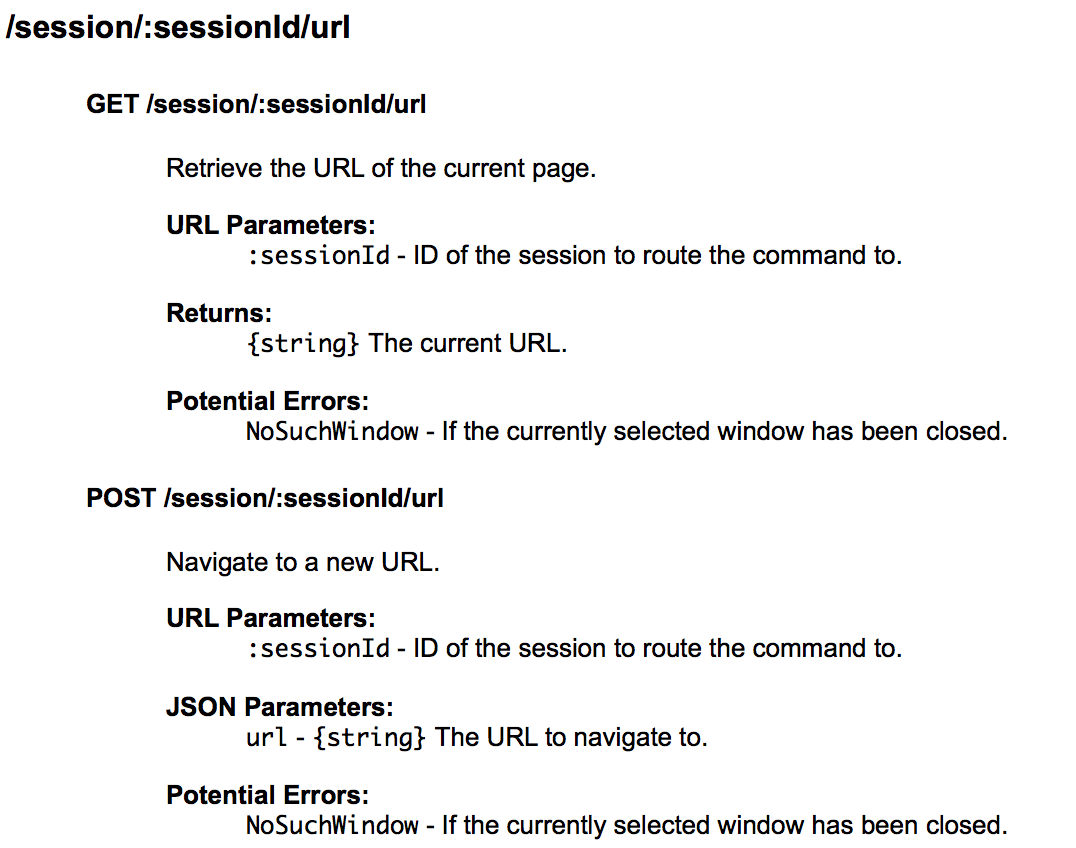
And any new commands that I will send now to the WebDriver will get executed in this browser. Any request that I want to get executed in this browser should be given with the session id for this browser session.

We can get the session id from the JSON response very easily:

"sessionId": "ab276c37-6b95-4868-bd3d-6fe6a0c335ff",

That was one great step, now let’s open an application to test in this session we created.

Going back to the documentation in google code site for get command:



We need to do a post request to navigate to the URL with URL parameter as the session id and JSON parameter as the URL of the application.

