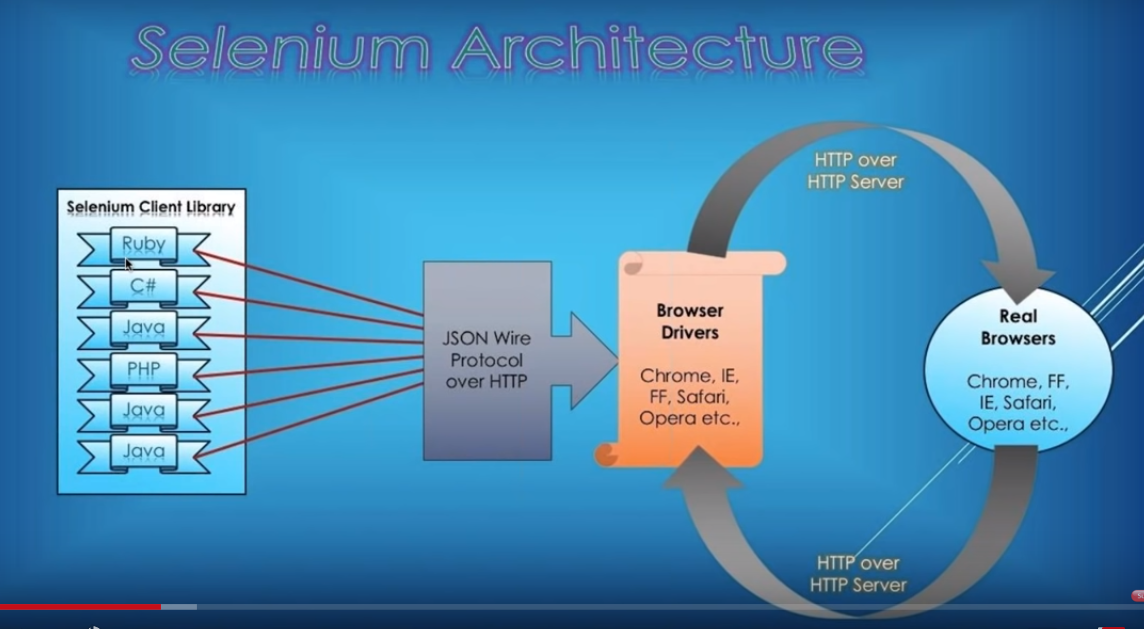
\*\*Selenium Architecture



Selenium Client Library send request to Json server over HTTP, it will send to browser driver, it will send request to Real browser. Every thing will happen through Rest API, Rest API is a web service.

When selenium library sent request, internally json payload will create

Selenium provides multiple platform supported.

Request sent from selenium library to finaly real browser, if the statement is passed it returns some pass result to browser driver, if fails send some error report number.

<https://github.com/SeleniumHQ/selenium/wiki/JsonWireProtocol>

In this link we can find different session commands or API’s how they internally called. And also u can find how we can send parameters in post call

Demo:

Place chrome driver, selenium server in one location

Open command prompt, go to that location

Type: java -jar selenium server-3.13

Wait until it gets “selenium server up and running” on some port number

Go to chrome open localhost: port number available in cmd prompt

Then click on create session, select browser chrome, then chrome will open

U can click on screenshots then screnshot will take for chrome, you can see the session command which is used internally in cmd prompt.

Now open postman, to get the current session

1. Select Get

<http://localhost:7777/wd/hub/s>essions , click on sent (up to hub copy from chrome localhost url)

it gives how many current sessions are there in the form of json(key, value)pair

1. Select GET

If you want the capabilities type <http://localhost:7777/wd/hub/session/sessionid> (sessionid given by chrome)

Eg: http://localhost:7777/wd/hub/sessions/ 5dc0d4552ef709c222555b22e831cc89

1. Select Post – API for url

<http://localhost:7777/wd/hub/session/sessionid/url>

Eg: http://localhost:7777/wd/hub/sessions/ 5dc0d4552ef709c222555b22e831cc89/url

Select Body🡪 select raw🡪 select json(app..) at Text

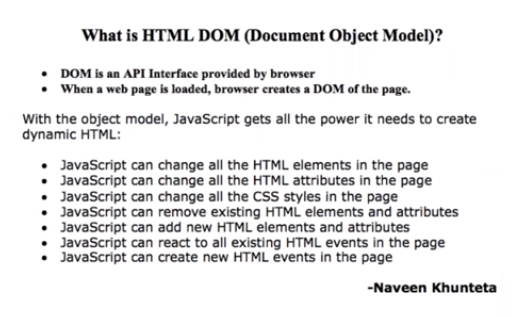
Write there as – passing one parameter

{

“url” : “http://google.com”

}

200 = success code

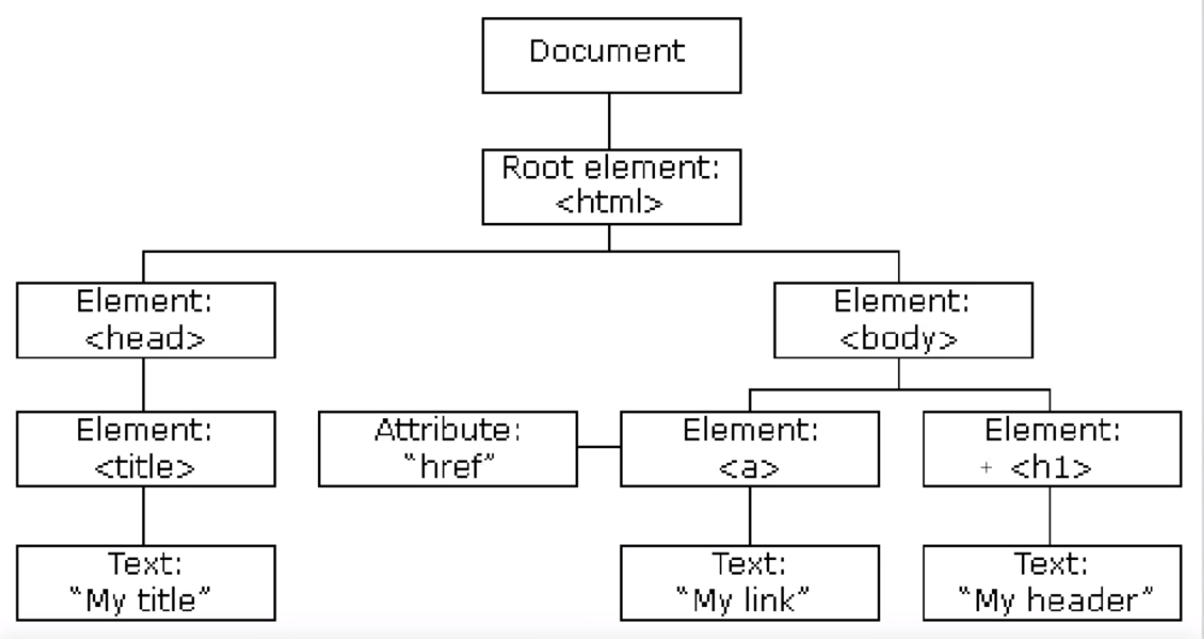


3 types DOM

1. HTML DOM – standard model for html documents
2. XML DOM -- standard model for xml documents
3. CORE DOM – for all different types of documents, standard module

Selenium interacts with HTML DOM. In HTML DOM we have 3 properties

1. Get all the properties of elements or change properties of elements or play with properties of elements
2. Can provide methods or access different methods
3. Can provide events or perform different events



DOM – DOCUMENT OBJECT MODEL

1. Which it follows W3C Standards - World Wide Web Consortium

(whenever you launching your application its compulsory to follow w3c)

1. DOM is a kind of platform or interface – which is available in the form of API’s
2. DOM structure will change from browser to browser and also versions

Eg: working on IE6, IE7 but not in IE10 viseversa because there may be updated or delete some changes

1. <http://facebook.com> is a site , have to work compatible on different browsers like chrome, safari, Firefox, IE. For that we will do cross browser Testing.

That means fb should compatible with Interface API DOM provided by chrome, firefox, safari. So developers will write code compatible related

1. Backend and webservices remain same without changes
2. For compatible, they will write Javascript code for all browsers
3. If sometimes page was not load properly, It is because of browser not with application
4. W3c gives typical CRUD API’s

C – ADD the elements

R - GET the elements

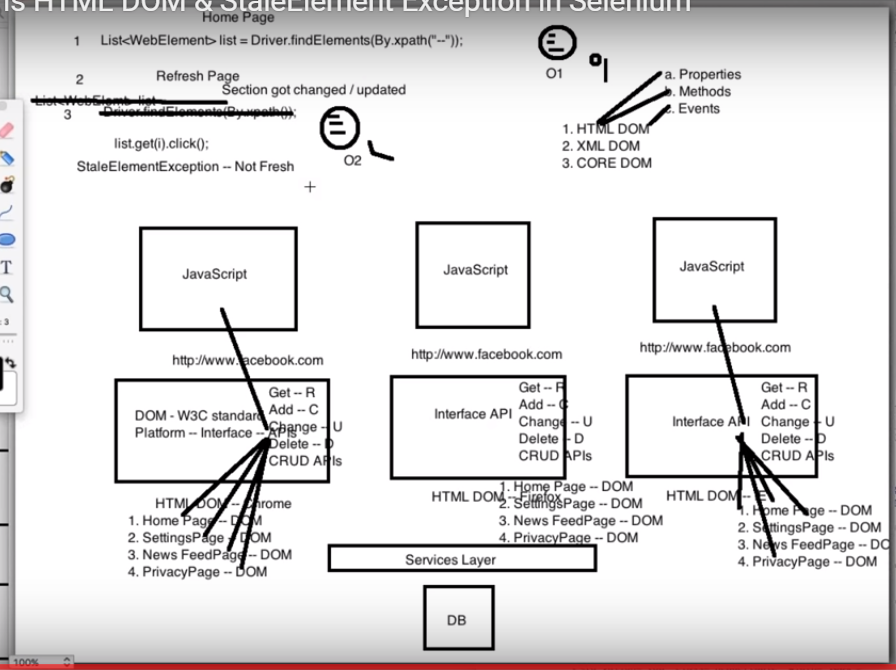
U – CHANGE the elements

D – DELETE the elements

1. Now, UI developers will interact with those CRUD
2. DOM is defined Page wise, DOM will load at run time when you open particular page
3. Importance of Javascript here, now By using JS we can interact with DOM and can execute those pages
4. DOM will loaded along with page loading on browser. If you refresh also again it will loaded with page
5. If u store some elements in a list and if you perform refresh, then perform click or some operations on that elements . It will throw StaleElementException, bcoz DOM object has been changed or refreshed
6. To overcome this you need to get latest list again after refresh

\*\* DOM is a interface API, is provided by specific browser, which provide different CRUD API’s

\*\* DOM will provide power to Javascript to create dynamic HTML pages do some CRUD OPERATIONS



Dynamic Xpath:

 **Dynamic XPath** is used to locate exact attribute or decrease the number of matching nodes/result from a webpage and following **XPath** expressions can be used for the same.

At times, XPath may change dynamically and we need to handle the elements while writing scripts. Standard way of writing xpath may not work and we need to write dynamic XPath in selenium scripts. Let’s see different way of writing dynamic XPath in Selenium with examples:

1. [Using Single Slash](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#SINGLESLASHXPATH)
2. [Using Double Slash](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#DOUBLESLASHXPATH)
3. [Using Single Attribute](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#SINGLEATTRIBUTEXPATH)
4. [Using Multiple Attribute](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#MULTIPLEATTRIBUTEXPATH)
5. [Using AND](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#ANDXPATH)
6. [Using OR](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#ORXPATH)
7. [Using contains()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#CONTAINSXPATH)
8. [Using starts\_with()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#STARTSWITHXPATH)
9. [Using text()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#TEXTXPATH)
10. [Using last()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#LASTXPATH)
11. [Using position()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#POSITIONXPATH)
12. [Using index()](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#INDEXXPATH)
13. [Using following xpath axes](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#FOLLOWINGXPATH)
14. [Using preceding xpath axes](https://www.softwaretestingmaterial.com/dynamic-xpath-in-selenium/#PRECEDINGXPATH)

**contains():**It is used to identify an element, when we are familiar with some part of the attributes value of an element.

Syntax

//<HTML tag>[contains(@attribute\_name,'attribute\_value')]

or

//\*[contains(@attribute\_name,'attribute\_value')]

Eg:

//input[contains(@id,'Email')

or

//\*[contains(@id,'Email')]

or

//input[contains(@name,'Email')]

or

//\*input[contains(@name,'Email')]

**starts-with():**It is used to identify an element, when we are familiar with the attributes value (starting with the specified text) of an element.

Syntax:

//<HTML tag>[starts-with(@attribute\_name,'attribute\_value')]

or

//\*[starts-with(@attribute\_name,'attribute\_value')]

Eg:

//input[starts-with(@id,'Ema')]

or

//\*[starts-with(@id,'Ema')]

**text():** This mechanism is used to locate an element based on the text available on a webpage

Syntax

//\*[text()='New look for sign-in coming soon']

**last():**Selects the last element (of mentioned type) out of all input element present

To identify the element (last text field ) “Your current email address”, we could use the below xpath.

findElement(By.xpath("(//input[@type='text'])[last()]"))

To identify the element “Year”, we could use the below xpath.

**[last()-1]** – Selects the last but one element (of mentioned type) out of all input element present

findElement(By.xpath("(//input[@type='text'])[last()-1]"))

**position():**Selects the element out of all input element present depending on the position number provided

In below given xpath, [@type=’text’] will locate text field and function [position()=2] will locate text filed which is located on 2nd position from the top.

findElement(By.xpath("(//input[@type='text'])[position()=2]"))

or

findElement(By.xpath("(//input[@type='text'])[2]"))

**following:**By using this we could select everything on the web page after the closing tag of the current node

//\*[@id='FirstName']

To identify the input field of type text after the FirstName field, we need to use the below xpath.

//\*[@id='FirstName']/following::input[@type='text']

Here I used, following xpath axes and two colons and then specified the required tag (i.e., input)

To identify just the input field after the FirstName field, we need to use the below xpath.

//\*[@id='FirstName']/following::input

**preceding:**Selects all nodes that appear before the current node in the document, except ancestors, attribute nodes and namespace nodes

xpath of the LastName field is as follows

//\*[@id='LastName']

To identify the input field of type text before the LastName field, we need to use the below xpath.

//\*[@id='LastName']//preceding::input[@type='text']"

Here I used, preceding xpath axes and two colons and then specified the required tag (i.e., input).