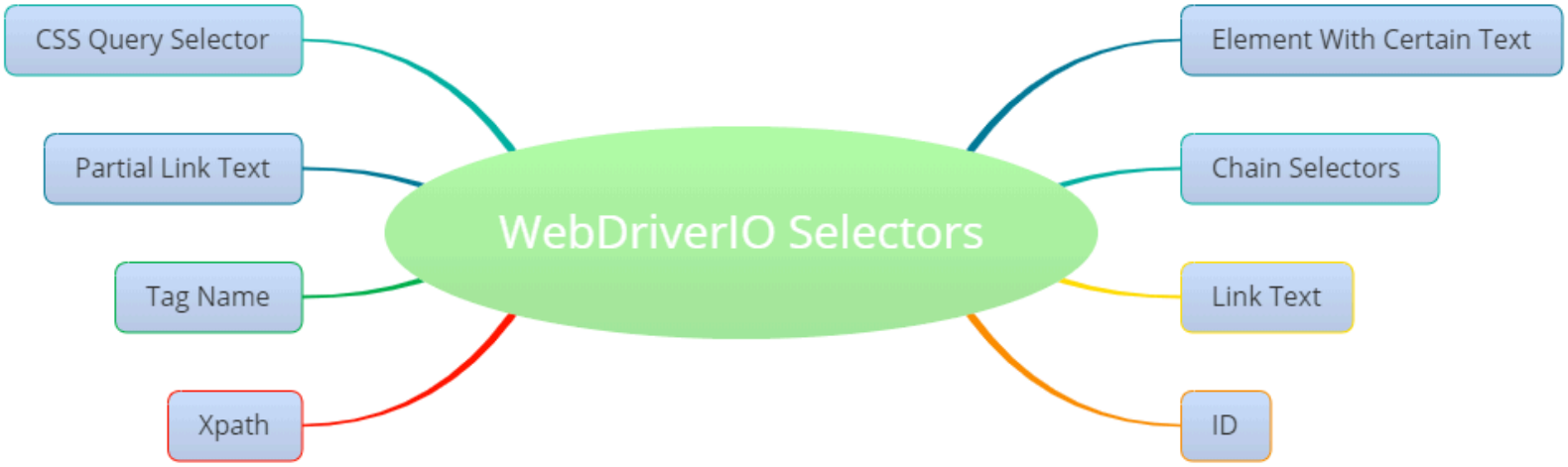
Locators in WebdriverIO

- ➔ Locators are techniques used to find webelement from webpage to perform user actions
- ➔ Locators provide a way to access the HTML elements from a web page.
- ➔ In WebdriverIO, we can use locators to perform actions on the text boxes, links, checkboxes, and other web elements.
- ➔ We can use that locator value while doing the scripting. Each and every web element are presented in HTML tag with open and End tag will be identified through its attributes.
- ➔ Selenium/protractor supports 8 different element locating strategies like id, name, className, tagname, linktext, partiallinktext, css, xpath. We explicitly need to specify which locator we are passing like if we are passing xpath then we need to specify it with By.xpath('locator') while using Selenium/protractor.
- ➔ WebdriverIO has simplified element locating strategies. We don't need to specify whether we are passing xpath or css.
- ➔ Simply we can write browser.click('locator'). WebdriverIO has the intelligence to identify which locator has been passed between xpath and CSS.


WebDriverIO Selectors
CSS Query Selector
Partial Link Text
Tag Name
Xpath
Element With Certain Text
Chain Selectors
Link Text
ID

Tag Name

- Tagname is nothing but the tag used to create the webelement. TagName can also be used for locating elements on-page. $('<tagname />'); method is used for this.
- It is not mandatory to have the closing tags in HTML.

The code which created the Alert button element

```
<input type="button" name="alert" value="Alert" onclick="alertbox();">
```

The way we can locate the element

Syntax:

```
$('<tagname />')
```

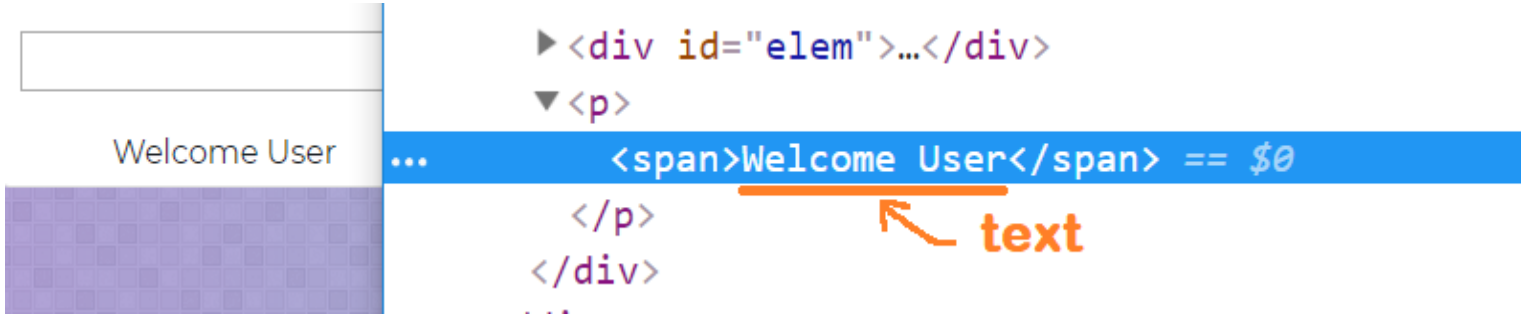
Example for above element

```
$('<input />').click()
```

- When we have a simple page, we might have only one or two elements, In such cases, we might find the elements using Tagname but On complex pages using the Tagname locator might return multiple elements.
- So usage of tagname will be less according to me.

Element with Complete Text :

- We can identify elements with text, most of the time span, label, div are the elements which we will find using the text. The format is element=



Syntax:

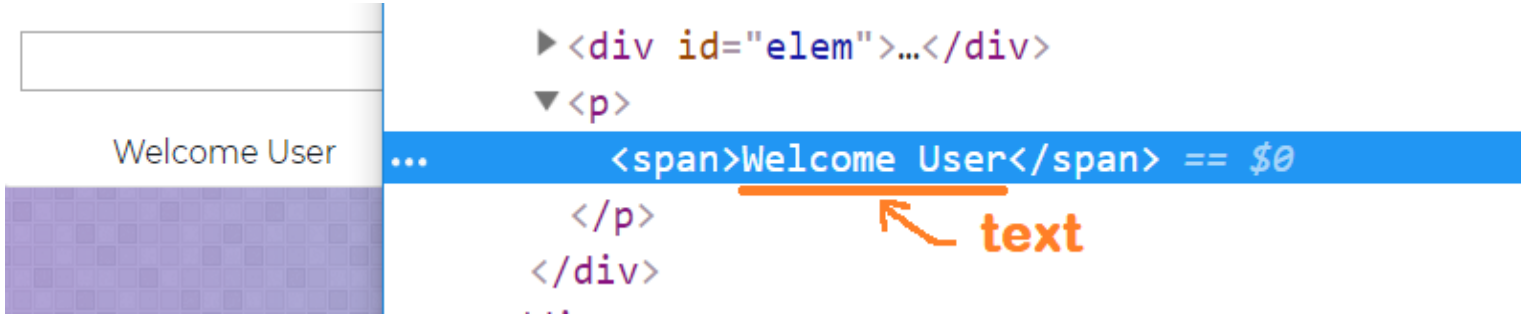
$('tagname=complete text');

Locator for above element

$('span=Welcome User').getText() // Welcome User

Element with partial Text

- Sometimes there will be situations where you might have the partial changing text and partial static text, like Welcome Harray, Welcome Tom or something like that. WebdriverIO provides a way to handle these kinds of scenarios using partial text.



Syntax:

```
$('tagname*=Part text');
```

Locator for above element

```
$('span=Welcome').getText() // Welcome User
```

Link with Complete Text

- Link text is nothing but the text present in the anchor tag <a>. To identify an anchor element with its visible text, we need to use =. The locator would be (eg: =bing).

```
<a href="https://support.tech">Support Tech </a>
```

The code for locating the above element is :

Syntax:

```
$('=Complete Link text');
```

Locating above element and getting attribute

```
const link = $('=Support Tech');
console.log(link.getAttribute('href')); // outputs: "https://support.tech"
```

Link with Partial Text

- We can find the link element using partial text present in the link text.

Syntax:

```
$('*=Partial Link text');
```

Locating above element and getting attribute

```
const link = $('*=Tech');
console.log(link.getAttribute('href')); // outputs: "https://support.tech"
```

Id

- **We can find the web element by using id attribute as**

  Syntax:

  ```
  $('#value of id attribute');
  ```

  Eg.

  ```
  //identify element with id then click
  $("#firstname").click();
  ```

Name

- **We can find web element by using name attribute as**

  Syntax:

  ```
  $('[name=”value of name attribute”]');
  ```

  Eg.

  ```
  //identify element with id then click
  $(‘[name=”firstname”]).click();
  ```

Class

- **We can find web element by using class / classname attribute as**

  Syntax:

  ```
  $('.value of class attribute');
  ```

  Eg.

  ```
  //identify element with class then click
  $(".firstname").click();
  ```

What is XPath

- Xpath is a locator technique used to find web elements from a webpage using XML.
- Xpath is nothing but an XML path, and the developer used XPath to validate XML files.
- HTML also follows the same structures as XML so that we can apply XPath to HTML pages as well along with webdriberIO webdriver.
- Xpath is nothing but sting expression which used to find the element(s) along with webdriberIO Webdriver, and other automation tools

Types

1. Absolute XPath
2. Relative XPath

Relative Xpath -

Xpath = //tagName[@attribute = attribute's value]

➔ // - points to any node in the webpage

➔ tagName - tag name - the name which is present after the < (angular bracket)

➔ attribute - whatever is present inside < and > bracket except tagname is an attribute

➔ attribute's value - it is corresponding value to the attribute

1. Xpath with Tagname -

We can write Xpath based on Tagname, which is very simple.

```javascript
Syntax for Xpath with Tagname : //tagName
<html>
      <body>
            <div id="pancakes">
                  <button type="button">Blueberry</button><br><br>
            </div>
      </body>
</html>
```

In the above code, there is a button present under div. we can write the XPath with tagname :

//button

2. Simple Xpath

- **Xpath with Attribute:**

  //tagName[@attribute='attribute value']

- **Xpath with multiple Attribute:**

  We can add n number attributes in one XPath itself

  //tagName[@attrib='attrib value'][@attrib2='attrib2 value']...

- **Xpath with Attribute and Index:**

  We can use index along with attribute: but the index will be useful only when matches are under a single parent

  //tagName[@attribute='attribute value'][index]

- **text() function in Xpath**

  //tagname[text()='value']

3. Contains() function in Xpath

- contains() function helps user to fins the element with partial values, or dynamically chaning values, contains verifies matches with portion of the value
- contains function :

    //xpath[contains(@attribute, 'attribute value')]

    //xpath[contains(text(), 'attribute value')]