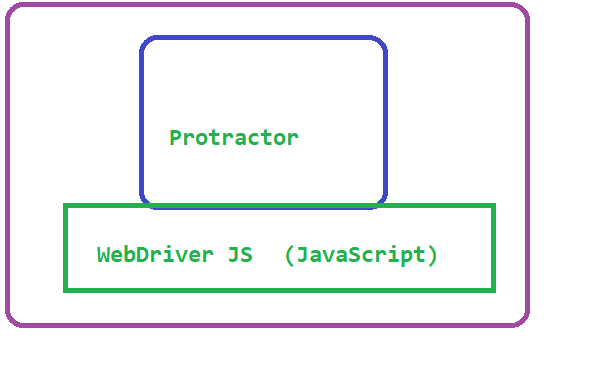
JavaScript and Protractor

What is protractor :

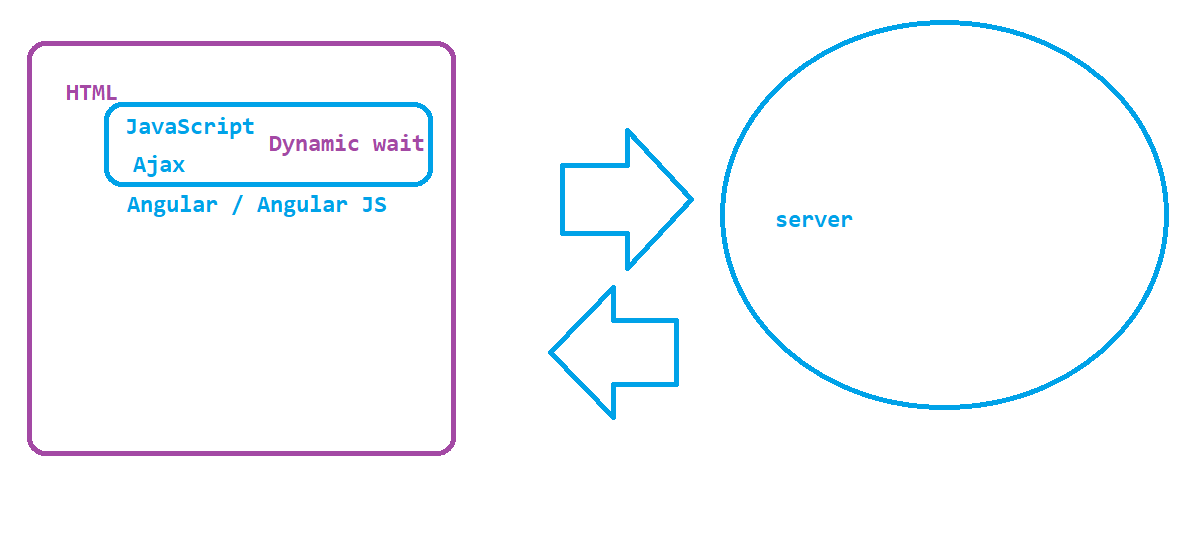
Protractor is a automation tool / framework which is built on top of Webdriver JS (WebdriverJS is a Java script version of selenium )

****

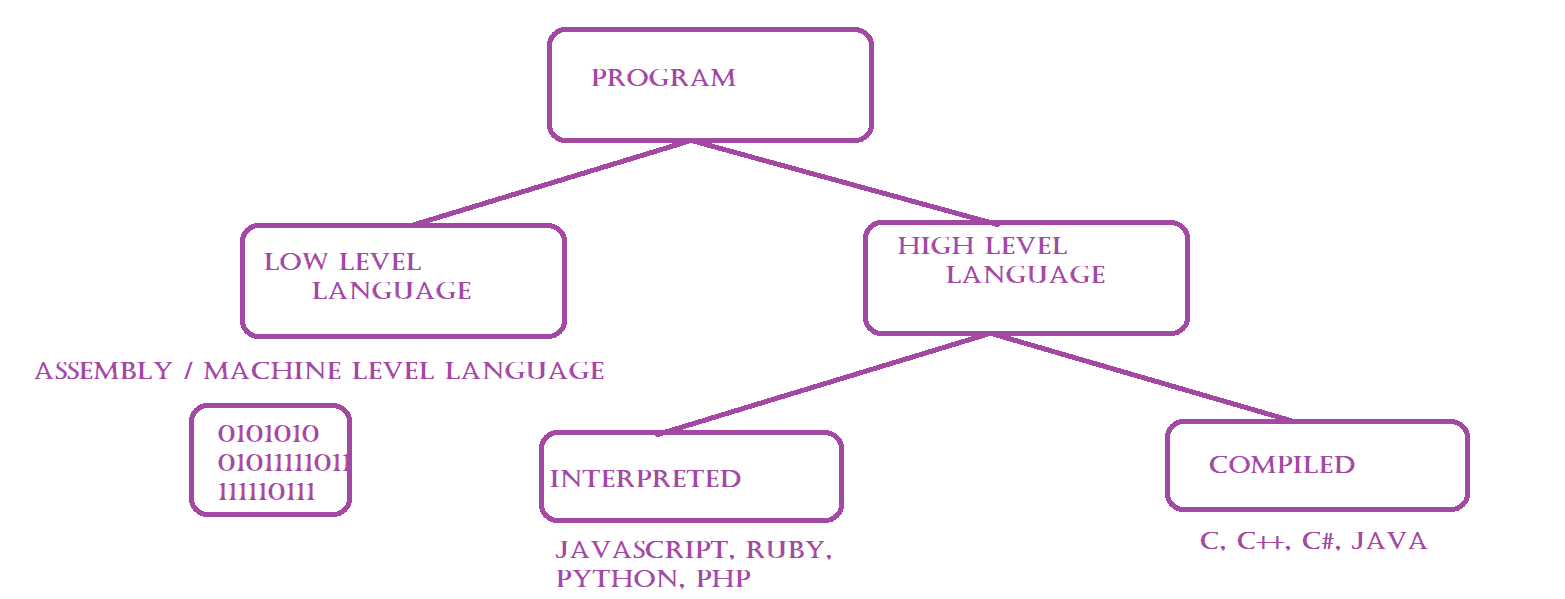
Why Protractor is required :

If the applications are developed using angular or angular JS, then along with normal html attributes (id, name, class name…) we will also have angular specific attributes (ng-repeater, ng-controller etc) which cannot be handled directly in selenium. Also, handling dynamic elements in selenium will be difficult.

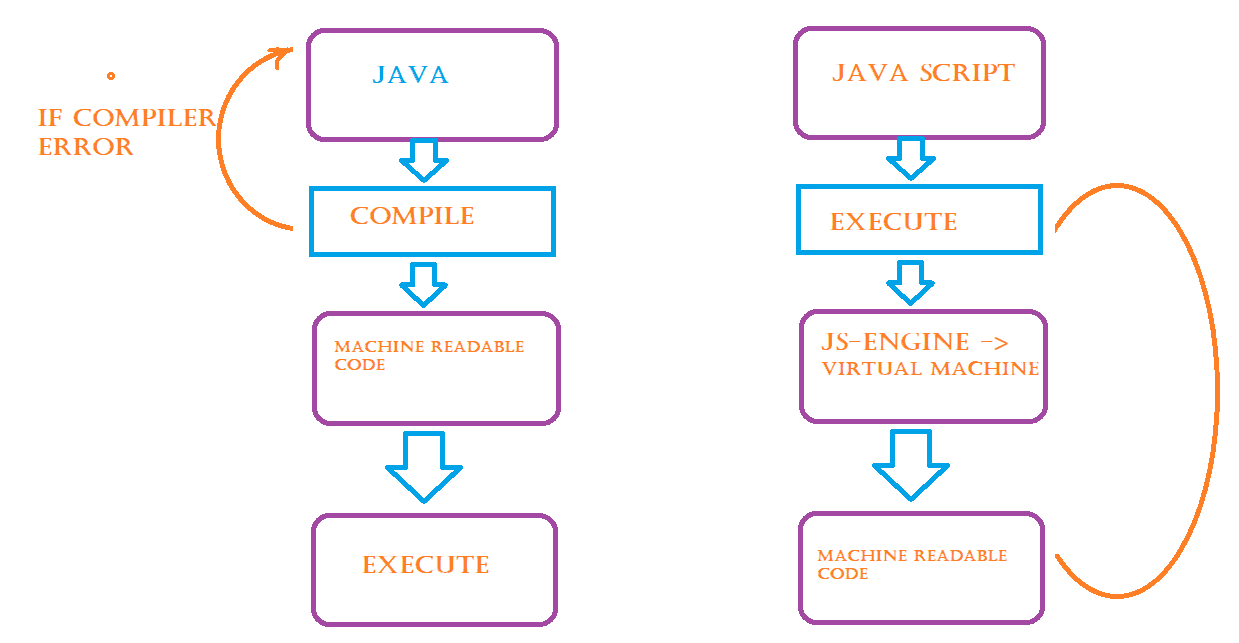
To overcome these 2 drawbacks we go with Protractor

****

PROGRAMMING DEFINITION

****

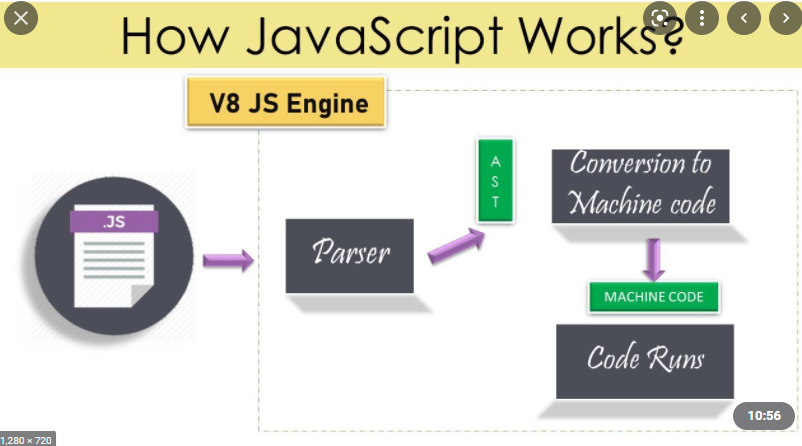
HIGHLEVEL DIFFERENCE BETWEEN INTERPRETED AND COMPILED PROGRAMMING



Java Script :

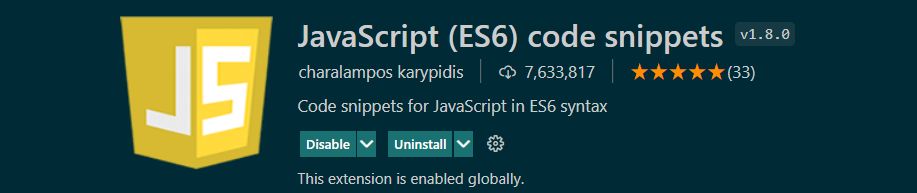
## Javascript is a High level and Interpreted Programming Language, which is introduced in the year 1995 by Bredan Eich

Initially JS is developed as a programming language for WEB but now we can use it in non-browser environments also like node.js

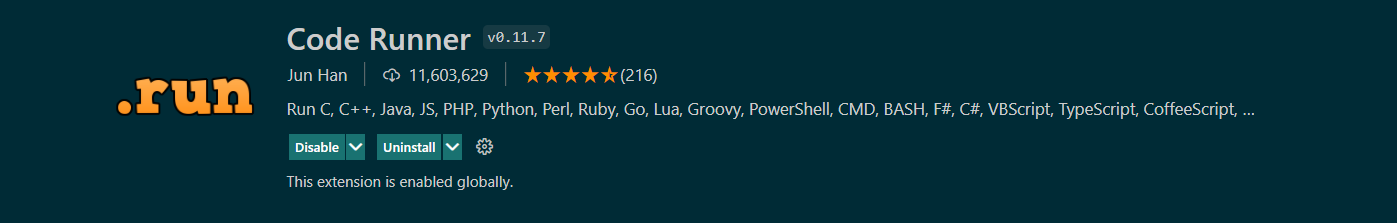


Visual Studio Code Editor

* Download the software from <https://code.visualstudio.com/download>
* Follow the instructions to install the software
* Select File -> Open Folder -> Browse and select the folder (workspace)
* Use Extension option  ( 4th from left panel ) to install the plugin
* Install Javascript code snippet



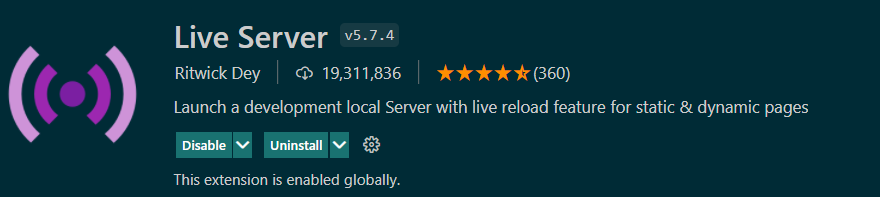
* To Run Java Script code install code runner plugin



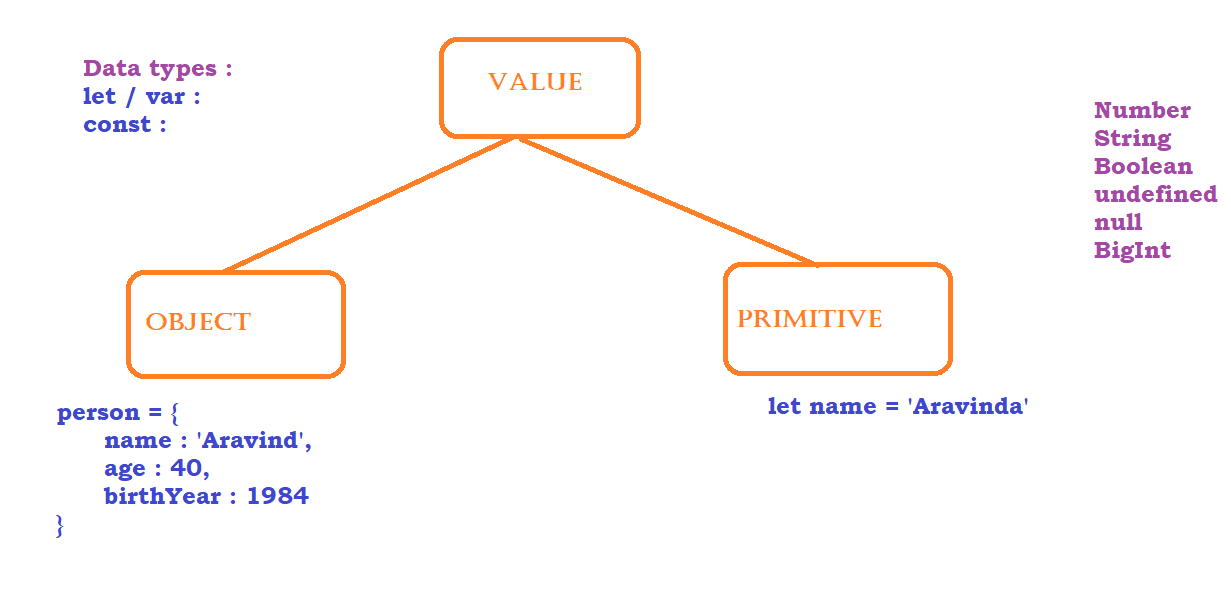
# Before running the program we should make sure we have installed node js software on the machine

* <https://nodejs.org/en/download/>
* open command prompt and test the installation by executing below commands
  + node –v
  + npm –v

to execute javascript on browser



Variables



Javascript supports 2 data types, let and const. based on the value stored inside the variables JS will decide the class variable belongs to. for ex,

1. Number
2. String
3. Boolean
4. undefined
5. null
6. Bigint

NOTE : AVOID USING **VAR** AS IT IS THE OLD WAY OF DECLARING VARIABLES

only difference between VAR and LET is, var is a function scope whereas let is a block scope.

Operators in JS

# Arithmetic Operators

### Addition -> +

### Subtraction -> -

### Multiplication -> \*

### Division -> /

### Module -> %

### Exponential -> \*\*

### Increment -> ++

# Assignment Operators

### Assign = x = y

### Add and assign += x+=y x = x + y

### Subtract and assign -= x-=y x = x - y

### Multiply and assign \*= x\*=y x = x \* y

### Divide and assign /= x/=y x = x / y

### Module and assign %= x%=y x = x % y

### Exponential and assign \*\*= x\*\*=y x = x \*\* y

# Comparison Operators

### check equal Value ==

### check equal Value along with type ===

### check Not equal Value !=

### check equal Value along with type !==

### Greater than >

### Greater than or equal >=

### Less than <

### Less than or Equal <=

### Ternary Operator ?:

# Logical Operators

### Logical AND && x == 5 && y == 5

### Logical OR || x == 1 || y = =5

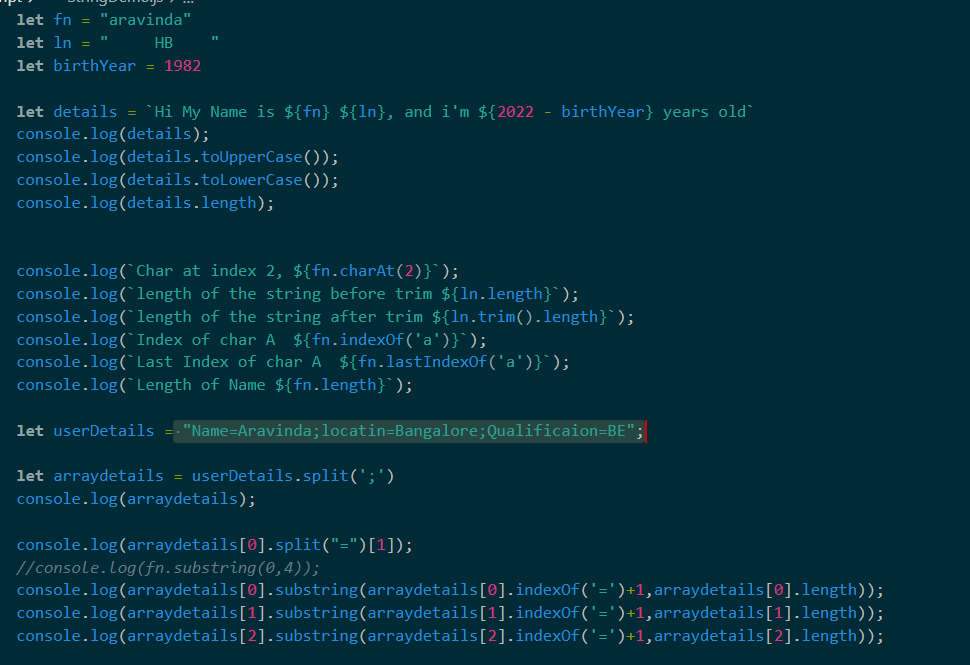
### Logical NOT ! !(x = = 1)

# Type of Operators

### typeOF – returns the type of operator

### instanceOF – returns true if object belongs to a class

Strings



Conditional statements

1. if
2. if else
3. if else if.. else
4. switch

Looping Statements

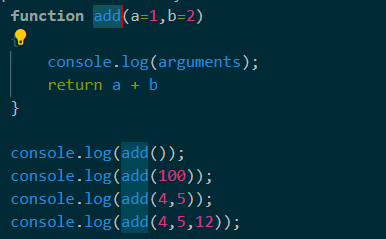
1. for
2. for each -
3. while
4. do while
5. for in – each iteration returns index
6. for of - each iteration returns element

Arrays

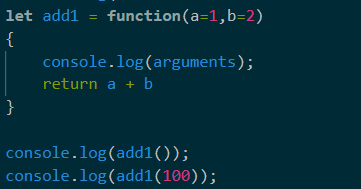
* push
* pop
* unshift
* shift
* delete
* splice
* slice
* concat
* join

Functions

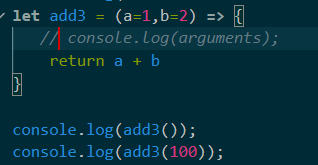
* Function with definition



* Function with Expression (Anonymous Functions)



* Arrow Function



Collections

* Set
* Map

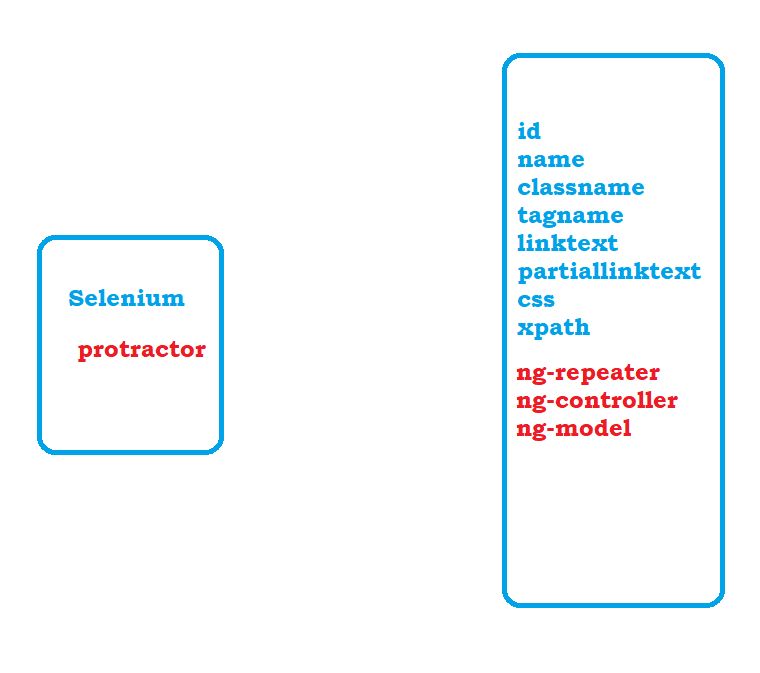
OOPS

* Objects
* class
* abstraction
* encapsulation
* polymorphism
* inheritance

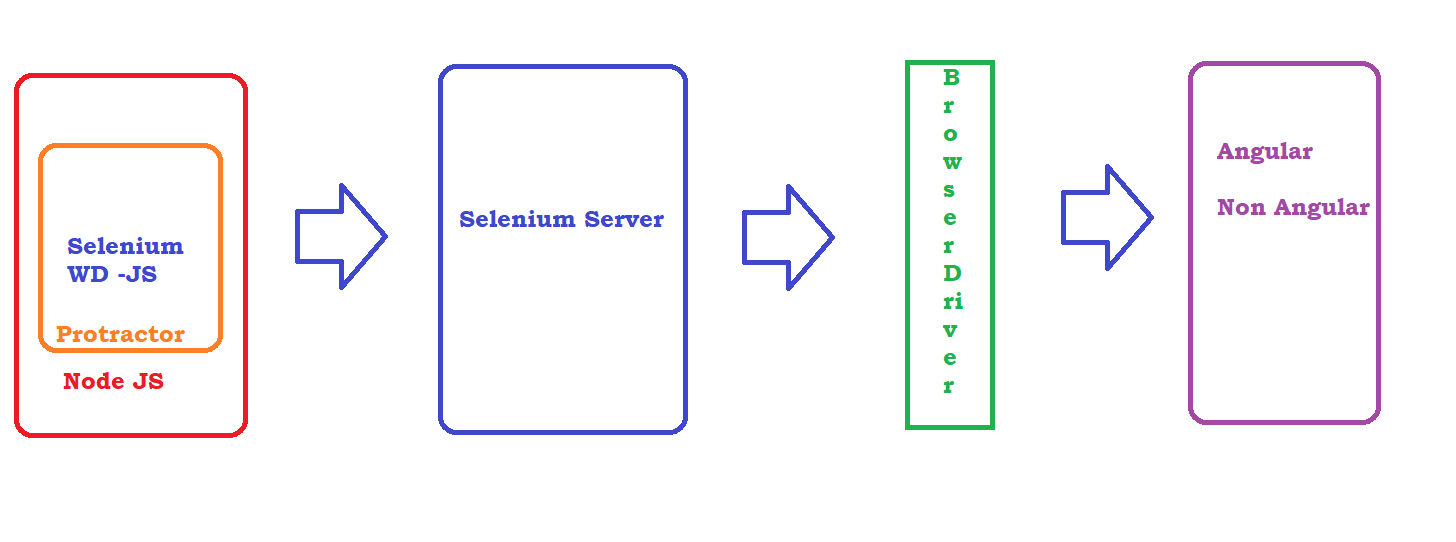
Protractor

#### Protractor is a end to end automation tool which can used to automate angular or angularJS applications. we can also use protractor for non-angular applications.

Protractor is a nods js application (node.js is must)



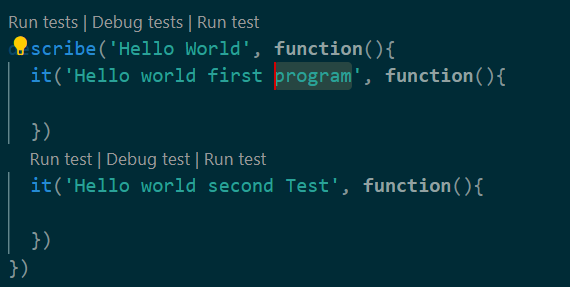
How protractor works



Protractor Setup

1. Download and install node.js
2. Install Protractor : <http://www.protractortest.org/#/tutorial>
   1. npm install -g protractor
3. Update webDriver manager
   1. webdriver-manager update
   2. OPTIONAL : webdriver-manager start
4. Write Test : copy sample conf.js and spec.js(TEST) from the above link or from the protractor installed directory to your project ( C:\Users\<<USERDIR>>\AppData\Roaming\npm\node\_modules\protractor\example)
   1. conf.js is the heart of protractor which contains many information like
      1. browser
      2. framework
      3. timeout
      4. plug-ins
      5. any configuration – before execution
      6. any reset – after execution
5. Executing protractor test
   1. protractor conf.js

First program



Async Nature of JS

#### Execution will not wait until the previous line completes the execution in case of any wait or in case of PROMISE is not resolved.

#### In case of PROMISE is resolved then JS will wait for the line of code to complete the execution before it goes to next step

What is promise???????

#### Promise is nothing but the current state of the step of the result of the step

Promise has 3 states

1. pending

2. resolved

3. rejected

In case of synchronous programming execution will go to next step only if the current step is either resolved or rejected. but in case of Async programming, execution will be just move to next step even in case of pending state.

90% of protractor functions will not go to next step until the promise is resolved

NOTE : any information we retrieve from the browser, we have to *resolve the promise* ex: gettext, getattribute

Resolving Promise :

- use then function and pass call back function to then block as mentioned below

 browser.sleep(10000).then(**function**(){})

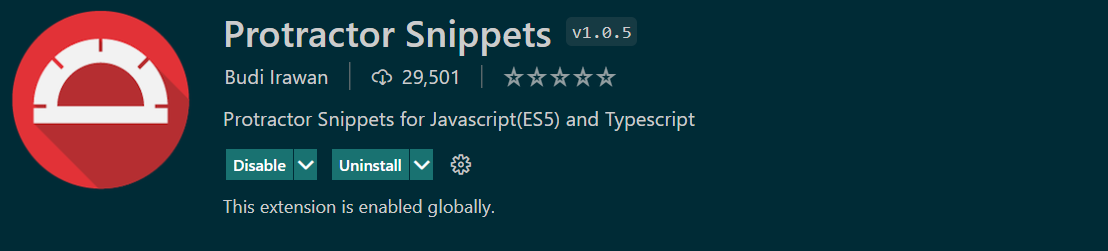
- write code inside call back function

 browser.sleep(10000).then(**function**(){

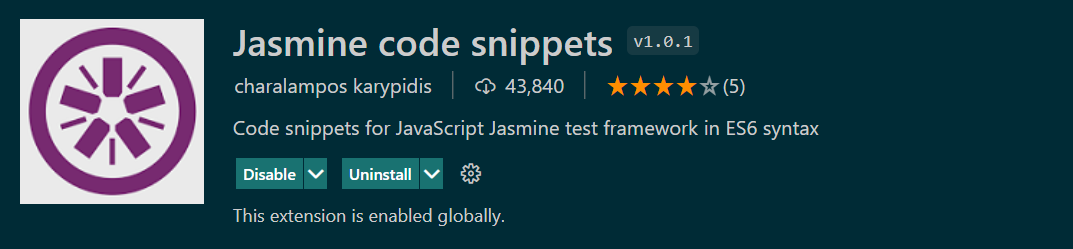
      console.log('Execution is complete');

    })

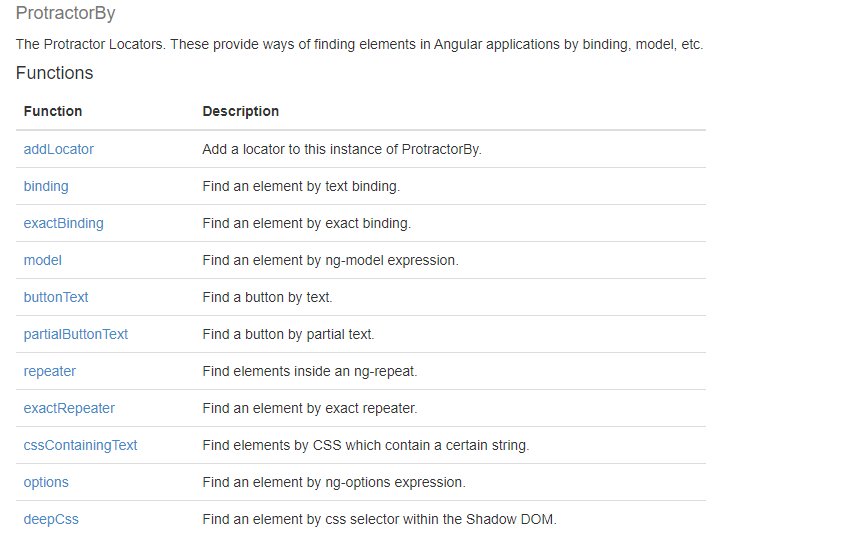
Plugins







Finding Elements in Protractor



Validations in Protractor

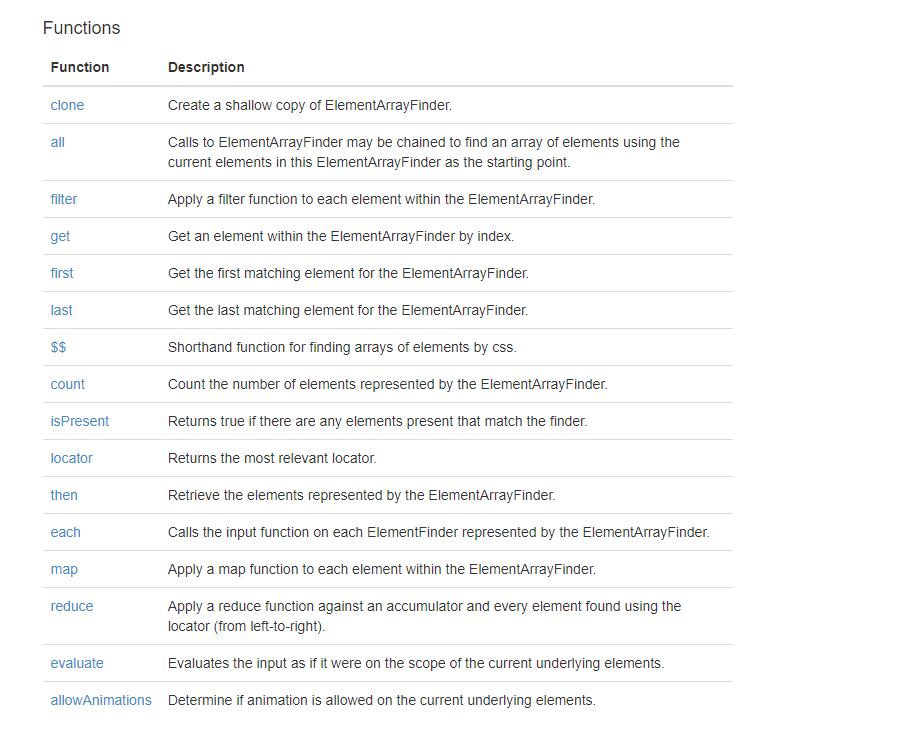
- <https://jasmine.github.io/>

- Validations can be performed by using functions available inside Jasmine framework

NOTE : All functions of jasmine comes with promise resolved. so we need not to resolve the promise when we are using protractor functions

Playing with more than one element

we can use element.all function provided by protractor



Chain locators in protractor:

chain locator is a concept using we can find the element which is present inside the other element

Capturing Screenshot :

1. go to npm website : <https://www.npmjs.com/>
2. search for **protractor-jasmine2-screenshot-reporter : https://www.npmjs.com/package/protractor-jasmine2-screenshot-reporter**
3. make sure your in your terminal you are inside the project and Install the plugin : npm i protractor-jasmine2-screenshot-reporter
4. make sure node\_modules folder is created in your project and you can see the above package installed
5. update conf.js as per the suggestion in official document
6. Execute the test and verify the reports and screenshots

Allure Reports

1. go to npm website : <https://www.npmjs.com/>
2. search for jasmine allure report <https://www.npmjs.com/package/jasmine-allure-reporter>
3. install the plugin using npm i jasmine-allure-reporter
4. Update onprepare function as per the instruction
5. Execute the tests and verify allure-results folder created with xml inside it
6. To get HTML output – install allure command line tool : <https://www.npmjs.com/package/allure-commandline>
7. On terminal execute allure serve <<Location of xml files>>

HTML Reports using Protractor

1. Go to npm website : <https://www.npmjs.com/>
2. search for **protractor-html-reporter-2** [**https://www.npmjs.com/package/protractor-html-reporter-2**](https://www.npmjs.com/package/protractor-html-reporter-2)
3. Install plugin using npm i protractor-html-reporter-2
4. Update onprepare with

var jasmineReporters **=** require('jasmine-reporters');

jasmine.getEnv().addReporter(**new** jasmineReporters.JUnitXmlReporter({

    consolidateAll**:** true,

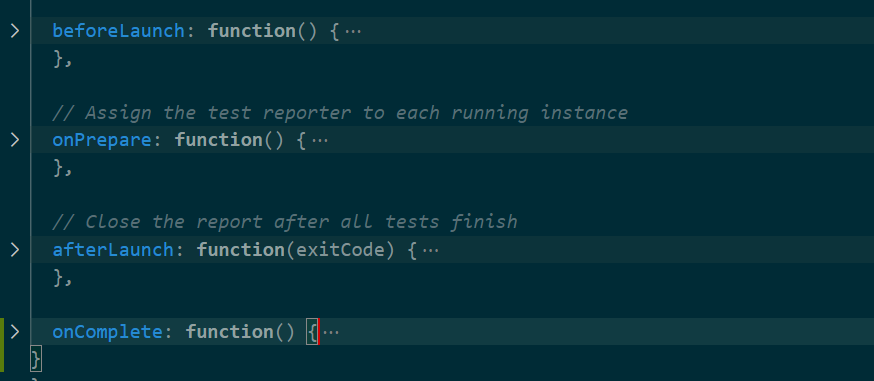
    savePath**:** './',

    filePrefix**:** 'xmlresults'

}));

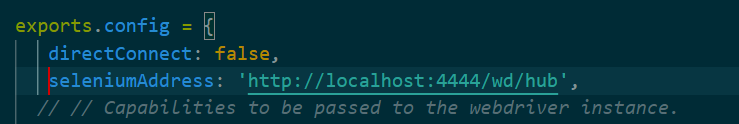
1. add given code in document to capture screenshot inside onPrepare function
2. add onComplete to conf.js as per the instruction
3. Execute the test and verify the reports

**Test configuration in conf.js :**

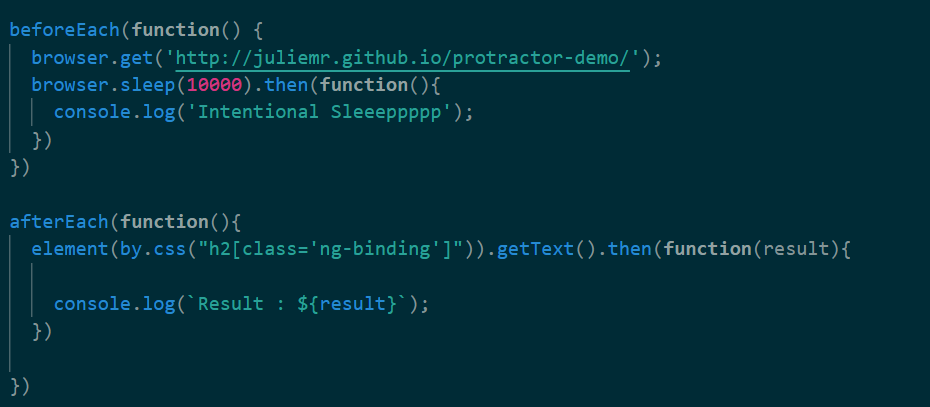


**Executing Test on different browser**

* When we execute **webdriver-manager update** by default it download and unzip both chrome and ff driver executable
* To execute on IE browser we have to download the driver executable
  + **webdriver-manager update –ie**
  + **webdriver-manager update --ie64 --versions.standalone=3.150.0 --versions.ie=3.150.0**
* start the selenium server : **webdriver-manager start**
* **Update conf.js as below**



beforeEach and afterEach

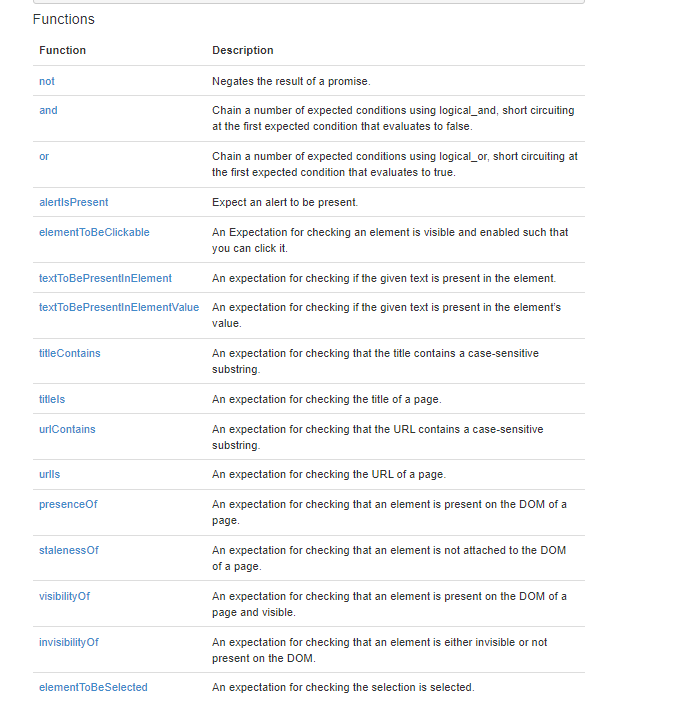


Non-Angular automation



Handle Sync Issues / Handling Elements with Dynamic Waits

var EC = protractor.ExpectedConditions;



Actions in Protractor

browser.actions().mouseDown(element1).mouseMove(element2).mouseUp().perform();

browser.actions().dragAndDrop(element1, element2).perform();

browser.actions().mouseMove(element).mouseMove({x: 50, y: 0}).doubleClick().perform();

Browser operations

browser.driver.manage().window().maximize();

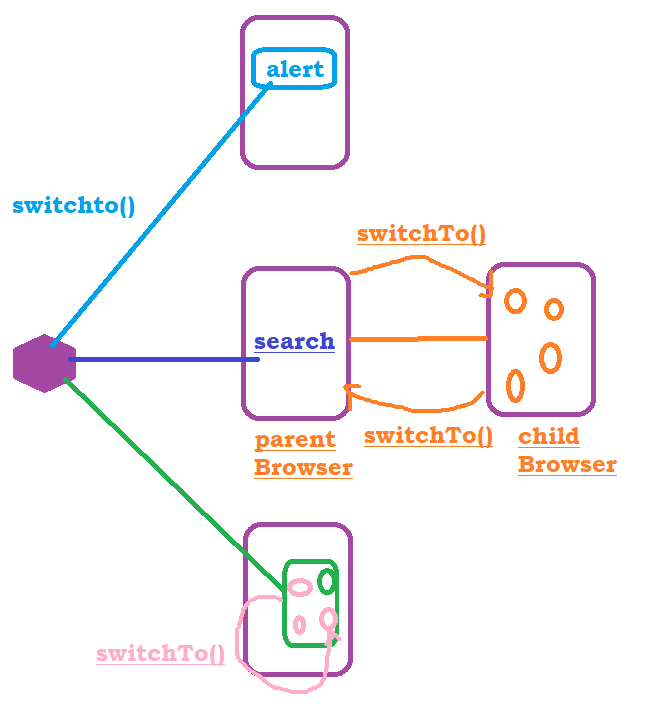
browser.driver.getCurrentUrl();

browser.driver.navigate().refresh();

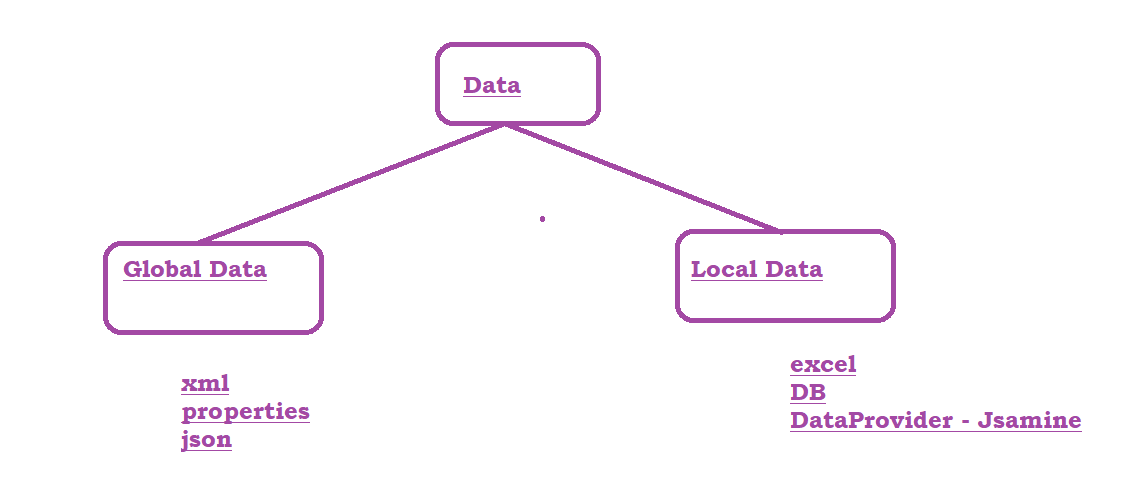
browser.driver.navigate().back();

browser.driver.navigate().forward();

switchTo in Protractor



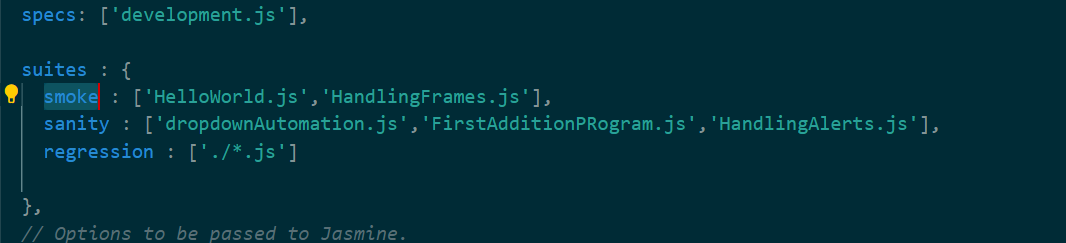
Data driven testing in protractor



# Using Data Provider in Protractor :

1. download the jsmine data provider from npm
2. <https://www.npmjs.com/package/jasmine-data-provider>
3. Install the plugin – npm i jasmine-data-provider
4. Update the code according to the instructions given

Executing Test Suite



provide suite parameter to the execution command : protractor conf.js --suite=smoke