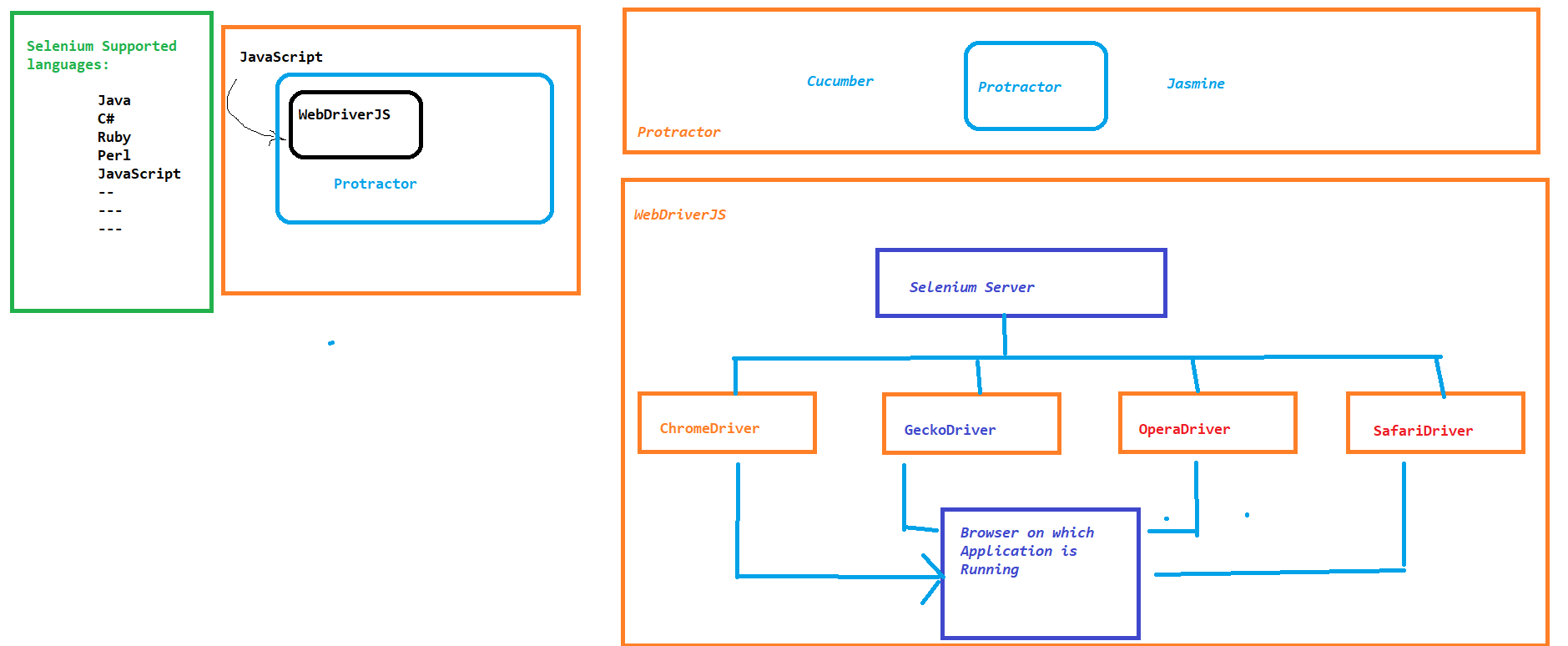
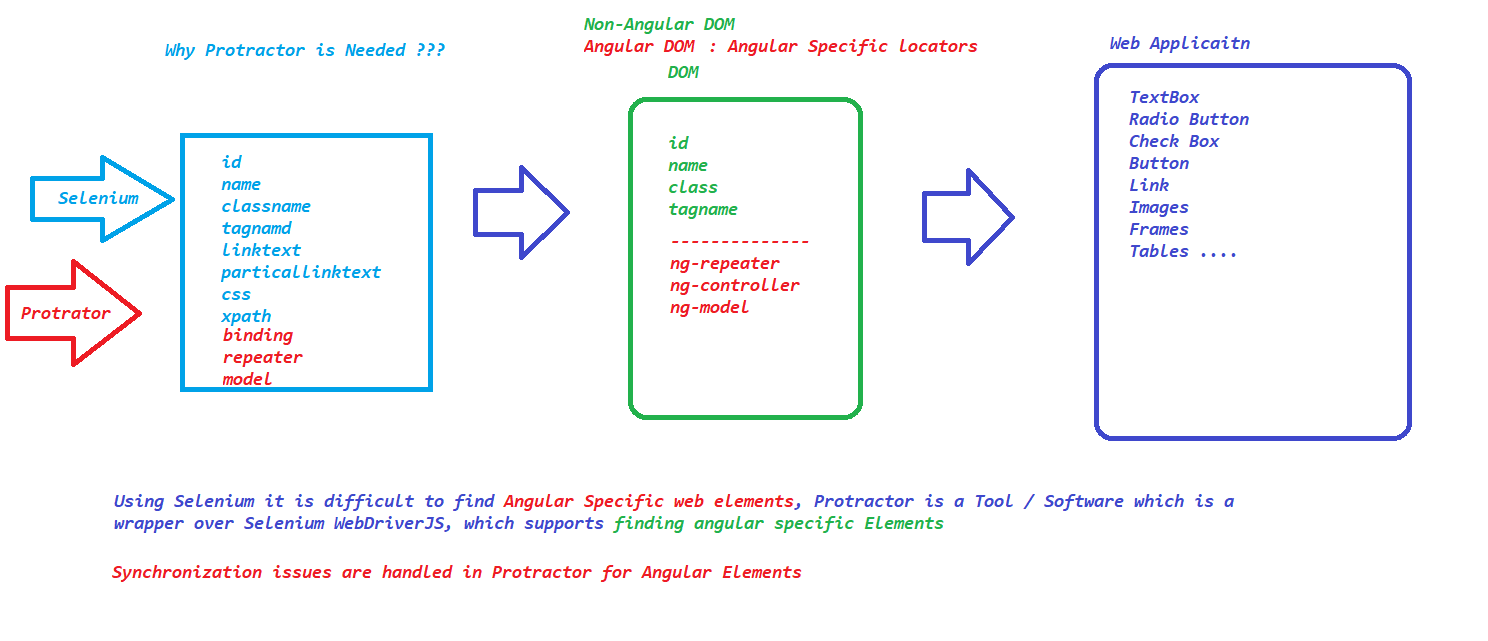
Java Script - Protractor

* Protractor is a end to end automation tool developed for Angular and Non-Angular Applications
* Protractor is a node.js Program which is built on WebDriverJS

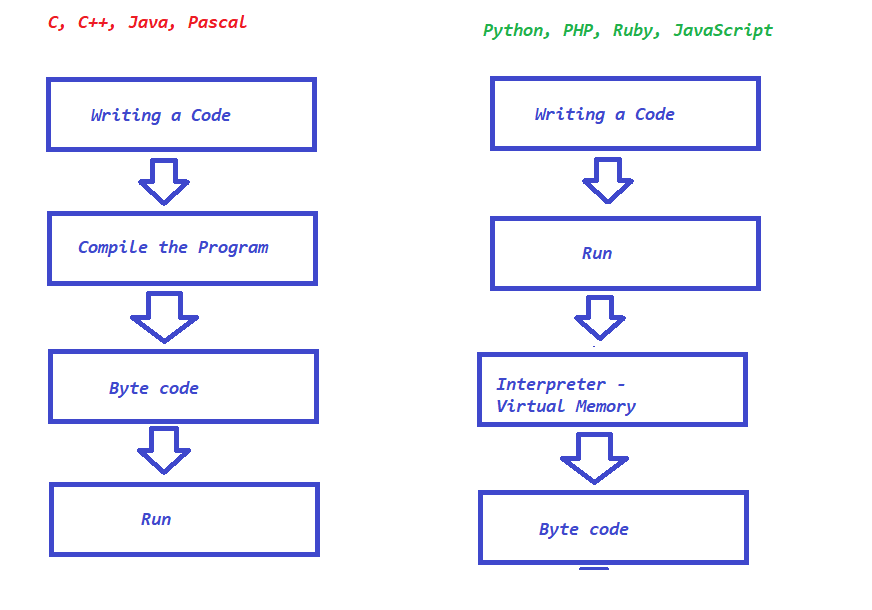


Why Protractor:



JavaScript :

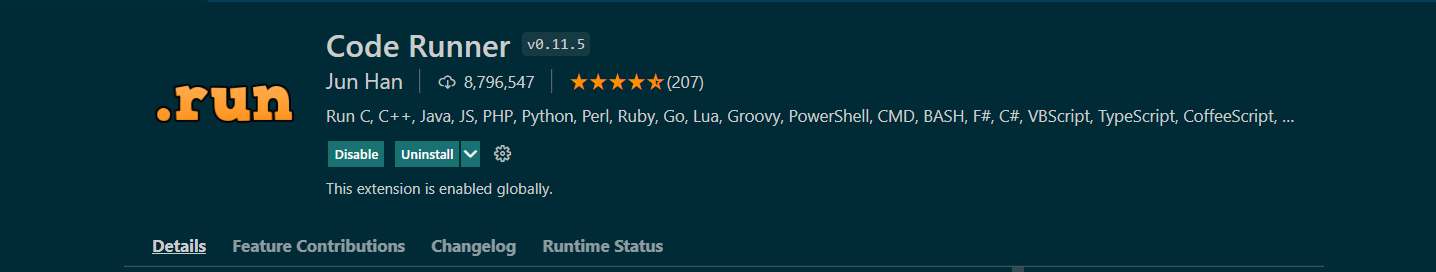
JavaScript is a High-level and interpreted programming language

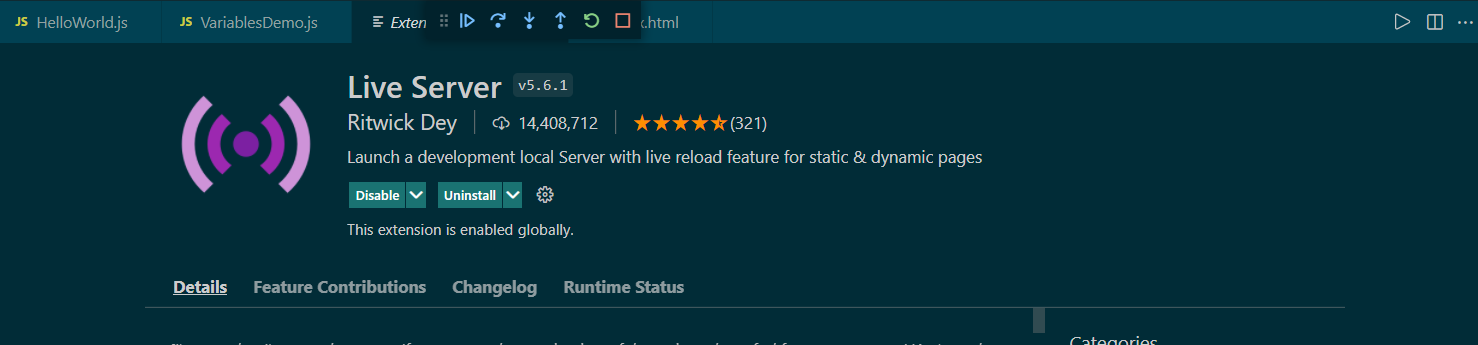


Editor Installation:

1. Download Visual Studio Code
   1. <https://code.visualstudio.com/download>
2. Install Required Plugins

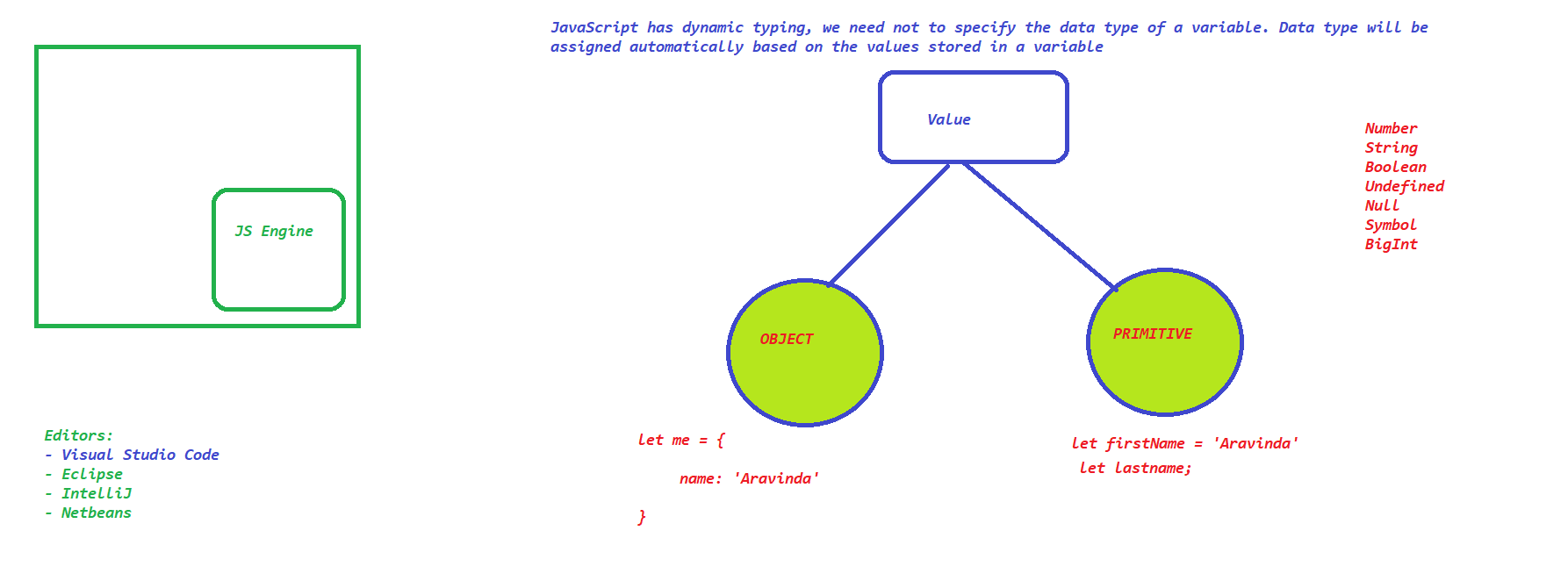






1. install node software on your machine
   1. <https://nodejs.org/en/download/>
   2. Once installation is complete open a terminal and execute
   3. node –v
   4. npm -v

Data types in JS:



We can have a variable with let, var or const in JavaScript

let and const are introduced after ES6 whereas var is the old way of declaring variables.

let – we can change the value of a variable

const – immutable i.e, value can never be changed.

NOTE : var and let or similar, only the scope is different.

Operators in JS :

* Arithmetic Operators
  + Addition +
  + Subtraction –
  + Multiplication \*
  + Division /
  + Modulus %
  + 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
  + Modulus and Assign %= x%=y x = x%y
  + Exponential and Assign \*\*= x\*\*=y x=x\*\*y
* Comparison Operators
  + equal to ==
  + equal values and equal type ===
  + not equal to !=
  + not equal values and type !==
  + greater than >
  + less than <
  + greater than or equal >=
  + less than or equal <=
  + ternary operator ?:
* Logical Operators
  + Logical AND && True if both conditions are true
  + Logical OR || True if any one condition is true
  + Logical NOT ! Will return true if the condition is false and vice versa
* Type Operators
* Bitwise Operators

Strings in Java Script:

Strings are immutable in JS, once created it can not be altered. if we try to change the content it will always create a new string.

console.log("String Demo");

**const** fn = "Aravinda "

**const** ln = "HB"

**const** birthYear = 1983

**const** year = 2021

**const** info = "i'm " + fn + "," + ln + " and i'm " + (year - birthYear) + " years old"

console.log(info);

*//templet literal*

**const** infoUsingTemplet = `i'm ${fn} ${ln} and i'm ${year - birthYear} Years old`

console.log(infoUsingTemplet);

console.log(`Char at index 2 ${fn.charAt(2)}`);

console.log(`Index of A ${fn.indexOf("a")}`);

console.log(`Last Index of A ${fn.lastIndexOf("a")}`);

console.log(`Length of the String ${fn.length}`);

console.log(`With out trim  ${fn}`);

console.log(`Trim ${fn.trim()}`);

console.log(`To Upper ${fn.toUpperCase()}`);

console.log(`To lower case ${fn.toLocaleLowerCase()}`);

Conditional Statements in Java Script :

* if

**if(condition) {**

**-----\-------**

**}**

* if..else

**if(condition) {**

**-----\-------**

**}**

**else {**

**$$$$$$$$$$$**

**}**

* **if..else..if**

if(condition) {

%%%%%%%%%%%%%%%

}

else if(condition) {

$$$$$$$$$$$$$$$$

}

else {

@@@@@@@@@@@

}

* **switch**

switch(value) {

case 1 :

&&&&&&&&&&&

break;

case 2 :

############

break;

case 3 :

$$$$$$$$$$$$

break;

default:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

Looping Statements in JS:

* for
* for in
* for of
* for each
* while
* do,while

Arrays and Functions in Array

* length
* push
* pop
* shift
* unshift
* delete
* splice
* slice
* concate
* join

Functions

* function with definition
* function with expression (Anonymous Function)
* Arrow function

Set and Map in JS

**const** resto = new Map();

resto.set('name' , 'Nandana Palace')

resto.set(1,'Bangalore india')

resto

.set('categories',['Andra Style', 'Chines', 'North indian', 'South Indian'])

.set('open' , 11)

.set('close', 23)

.set(true, 'We are Open')

.set(false, 'We are Closed')

*//FUN*

time = 15

console.log(resto.get((time >= resto.get('open') && time <= resto.get('close'))));

console.log(`Convert Maps To Array`);

console.log(resto);

console.log(...resto);

console.log(resto.get('name'))

if(resto.has(10))

{

    resto.delete(1)

}

console.log(resto);

SET

**const** grade = new Set();

grade

.add('A')

.add('B')

.add('C')

.add('A')

.add('B')

.add('C')

console.log(grade.size);

console.log(grade.values());

grade.forEach(element **=>** {

    console.log(element);

})

console.log("Using function Expression ");

grade.forEach(**function**(element) {

    console.log(element);

})

OOPS:

* Class
* Objects
  + function
  + constructor function
  + class
  + prototype
* Abstraction
* Encapsulation
* polymorphism
* Inheritance

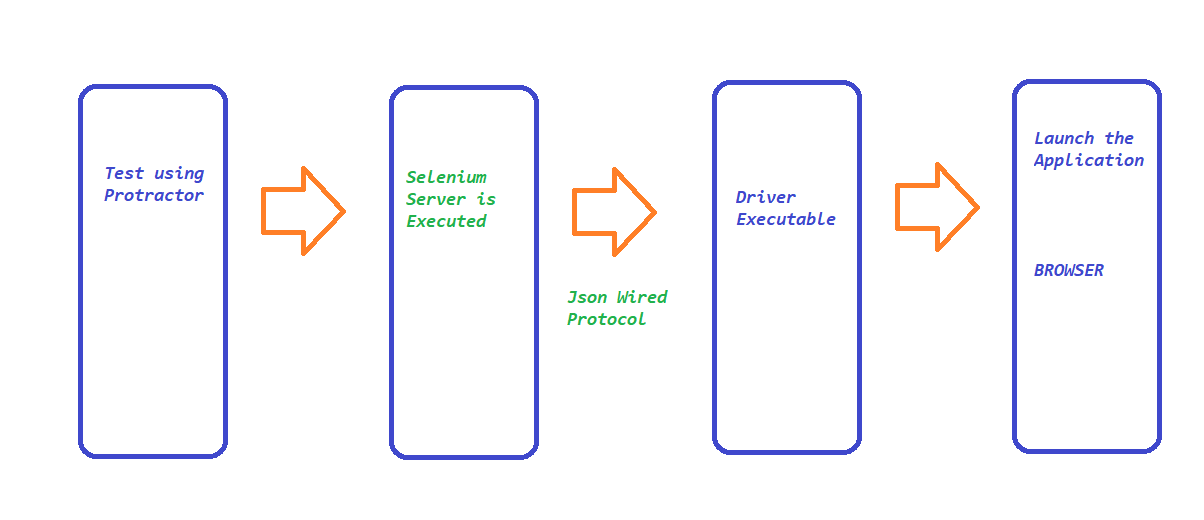
#########################################################################################################################################################################################################################################################################################################################################################################################################################################

Protractor:

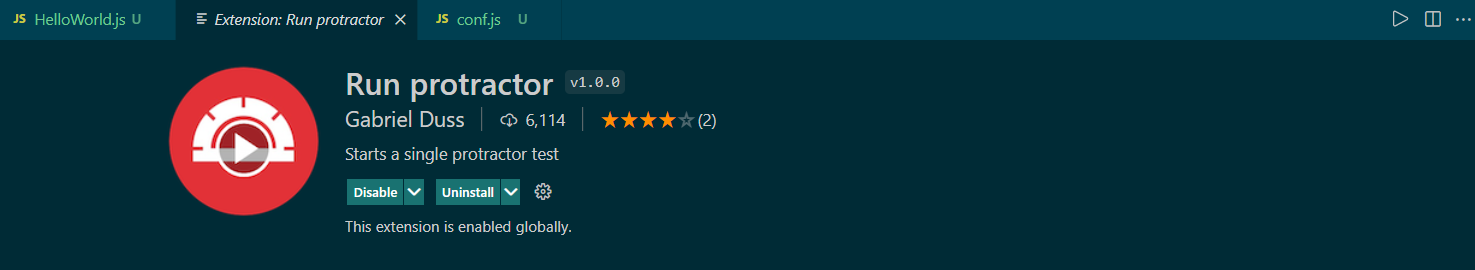
Installation:

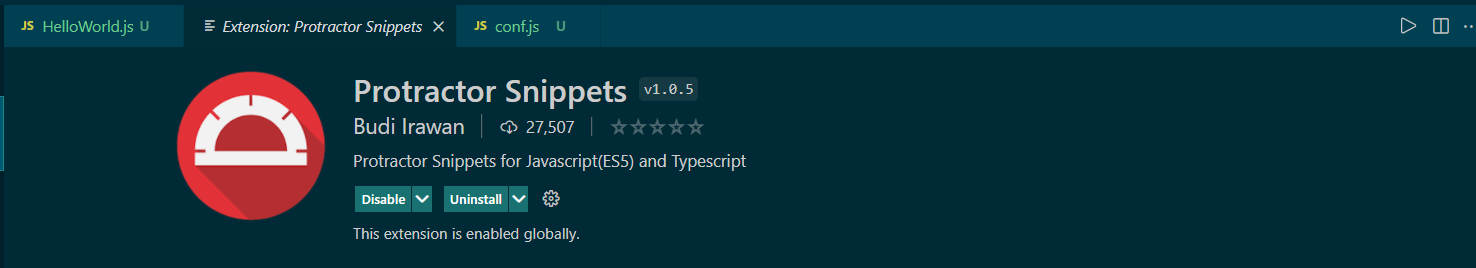
1. Download and install node.js
   1. node –v
   2. npm -v
2. Install Protractor
   1. npm install -g protractor
   2. protractor --version
3. Update the Webdriver-manager
   1. webdriver-manager update
   2. verify the directory for the latest driver executables C:\Users\**<<USERDIRECTORY>>**\AppData\Roaming\npm\node\_modules\protractor\node\_modules\webdriver-manager\selenium
   3. **<<USERDIRECTORY>> - REPLACE WITH YOUR DIRECTORY**
4. Write the Test
   1. Copy the sample code from example\_spec.js and conf.js from C:\Users\**<<USERDIRECTORY>>** \AppData\Roaming\npm\node\_modules\protractor\example
   2. **<<USERDIRECTORY>> - REPLACE WITH YOUR DIRECTORY**
   3. **NOTE : IF ANY ERROR THROWN DURING EXECUTION THEN FORM THE TERMINAL EXECUTE THE BELOW COMMAND**
5. Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

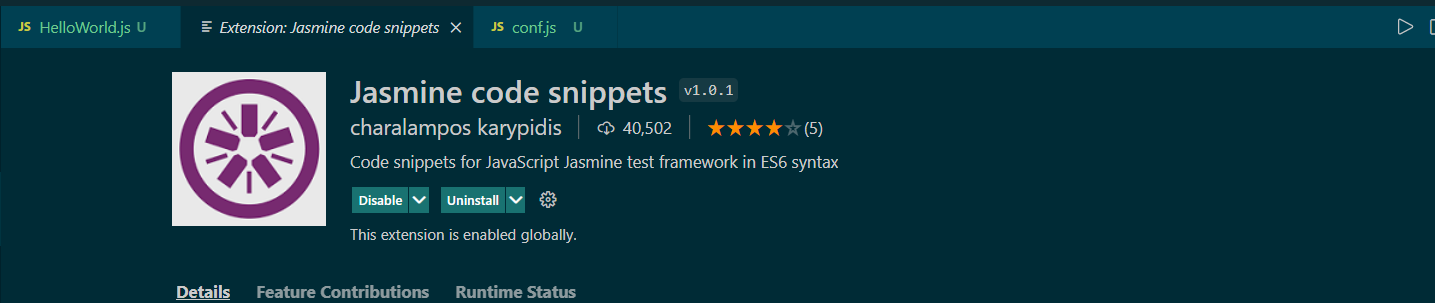
How Protractor Works:



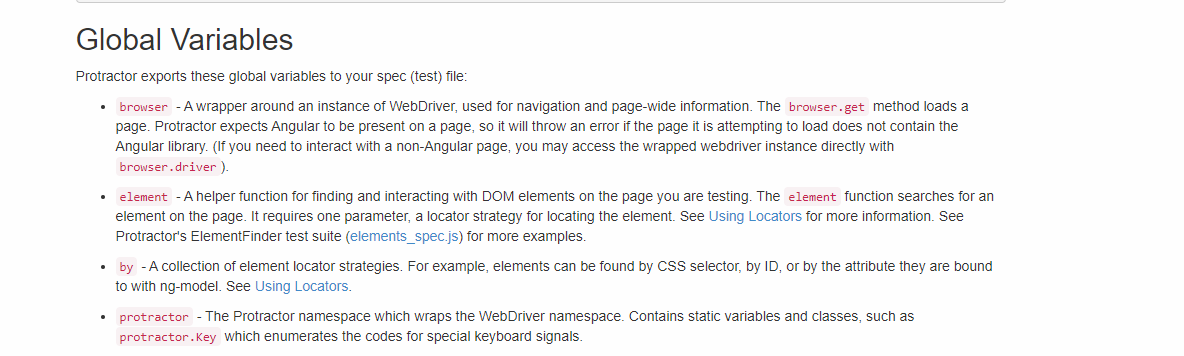
Plugins :



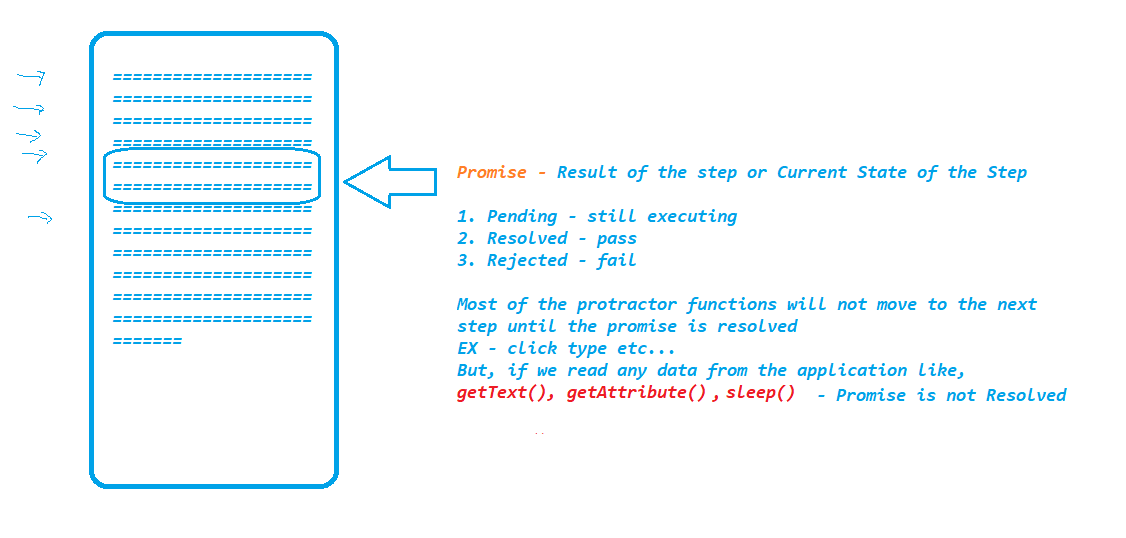




Global Variables which are Exposed in protractor



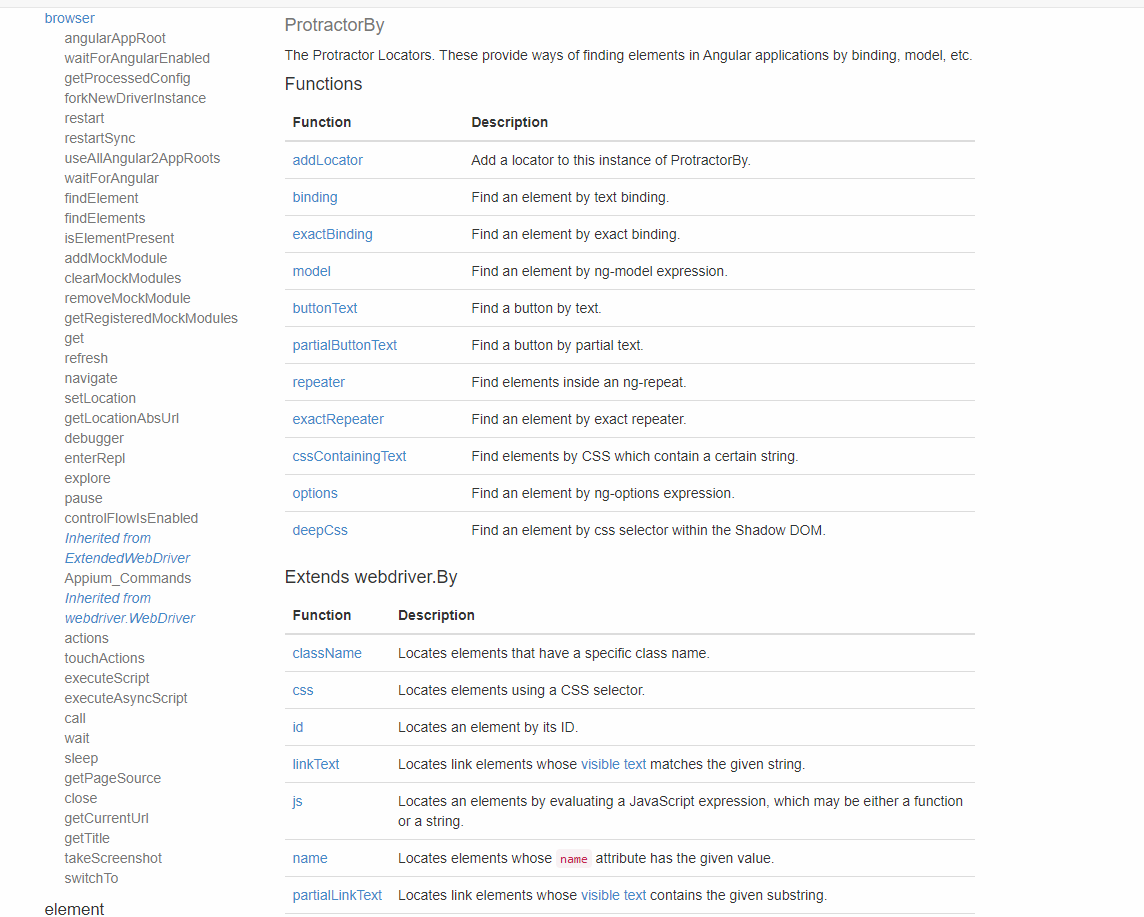
ASynchronisation Nature of JS



Validations in protractor :

We use **expect** statements from Jasmine framework to perform Validations in Protractor.

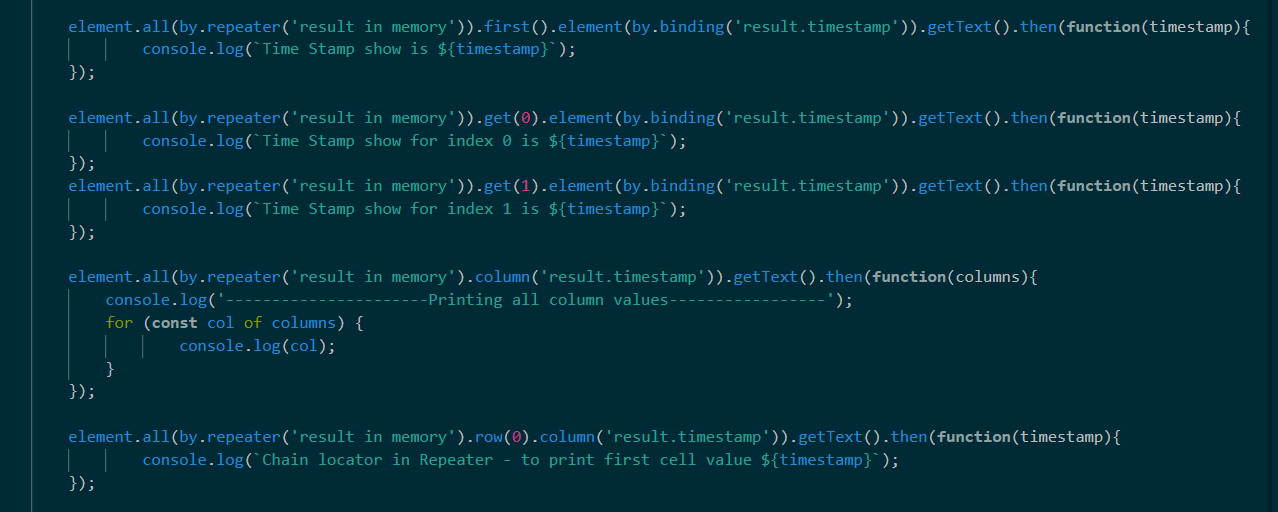
Different ways of finding elements :

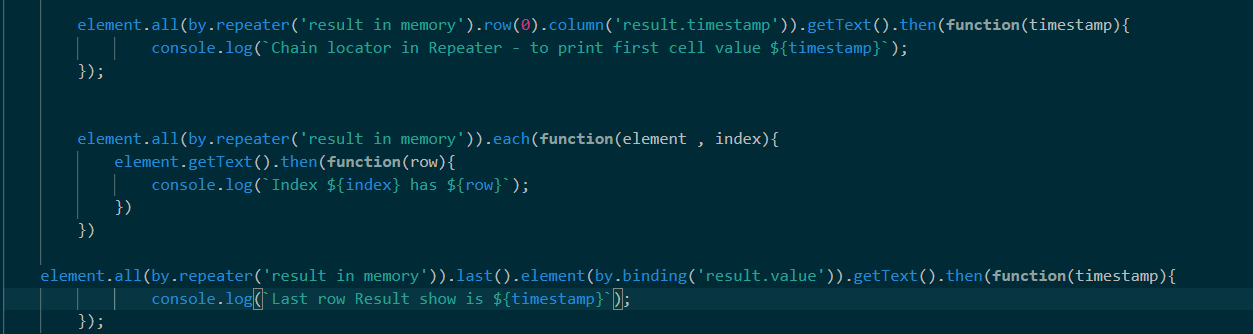


Playing with Multiple Elements

if we want to retrieve multiple elements form the application then we have to use element.all(By)

**Chain locators OR finding Element from Elements**

****

****

**Capturing ScreenShot on test Failure**

1. Install protractor jasmine2 screenshot reporter plugin
   1. <https://www.npmjs.com/package/protractor-jasmine2-screenshot-reporter>
2. Install the plugin by executing npm command from the terminal
   1. - npm i protractor-jasmine2-screenshot-reporter
3. Update conf.js as provided in the document,(NOTE: MAKE SURE TO ADD NEW CONTENT WITH “,” SEPERATED )
4. Execute the test and we will be able to see the html report in target folder along with screenshots for every test

**HTML Reports by Allure Reports**

1. Install jasmine allure reporter
   1. [**https://www.npmjs.com/package/jasmine-allure-reporter**](https://www.npmjs.com/package/jasmine-allure-reporter)
2. Install the plugin by executing below command on the terminal
   1. npm i jasmine-allure-reporter
3. Update conf.js as suggested in document (MAKE SURE TO ADD NEW CONTENT WITH “,” SEPERATED )
4. Run the test – result will be in xml format, to generate HTML output we have to install allure commandline
   1. [**https://www.npmjs.com/package/allure-commandline**](https://www.npmjs.com/package/allure-commandline)
5. Install allure command line by executing
   1. npm i allure-commandline
6. Execute allure serve “Location of allure-results” TO GET THE HTML OUTPUT

**HTML Reports using protractor html reporter**

1. Install protractor html reporter-2
   1. [**https://www.npmjs.com/package/protractor-html-reporter-2**](https://www.npmjs.com/package/protractor-html-reporter-2)
2. Install the plugin by executing below command on the terminal
   1. npm i protractor-html-reporter-2
3. Update conf.js as suggested in document (MAKE SURE TO ADD NEW CONTENT WITH “,” SEPERATED )
4. Run the test and verify ProtractorTestReport.html is generated

**Executing Test on Different Browsers :**

**NOTE : on chrome and firefox we can execute the tests just by changing the browser name in conf.js**

  'browserName': 'chrome'

if we want to execute our tests other than chrome or firefox then,

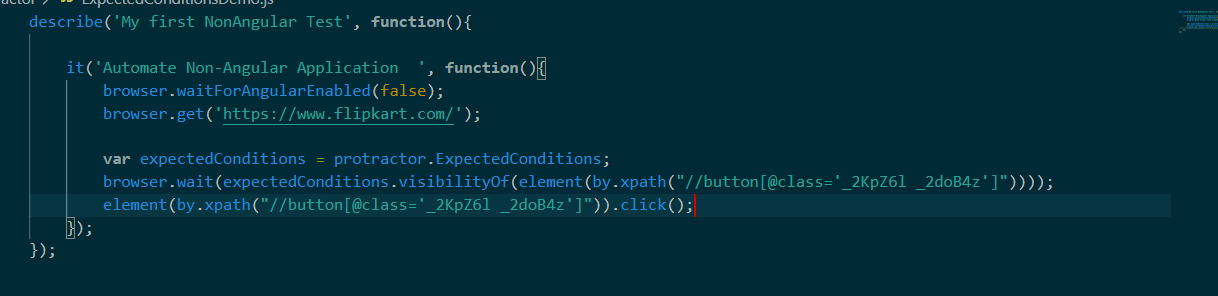
* Download the driver executable for the corresponding browser
  1. Webdriver-manager update --<<BROWSER NAME>>
     1. i.e., webdriver-manager update --ie
  2. Start the server manually
     1. Webdriver-manager start
  3. update conf.js
     1. ‘browserName’ : ‘ie’
     2. directConnect: true // comment this line
     3. seleniumAddress : ‘http://localhost:4444/wd/hub’

Automate Non-Angular Applications

to automate non-angular applications just add browser.waitForAngularEnabled(false), before launching the applications

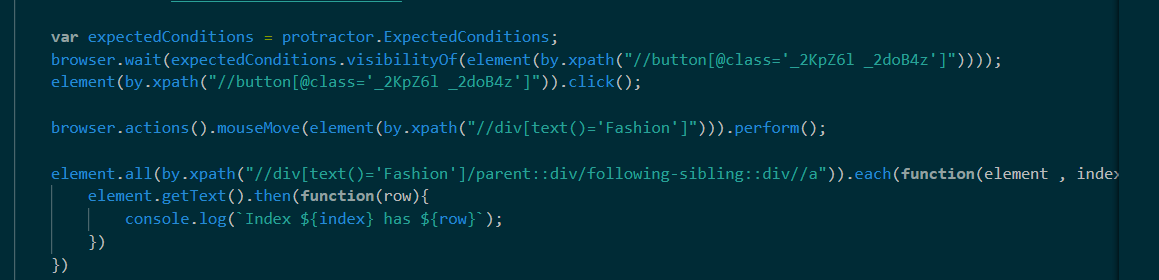
Dynamic Waits in protractor

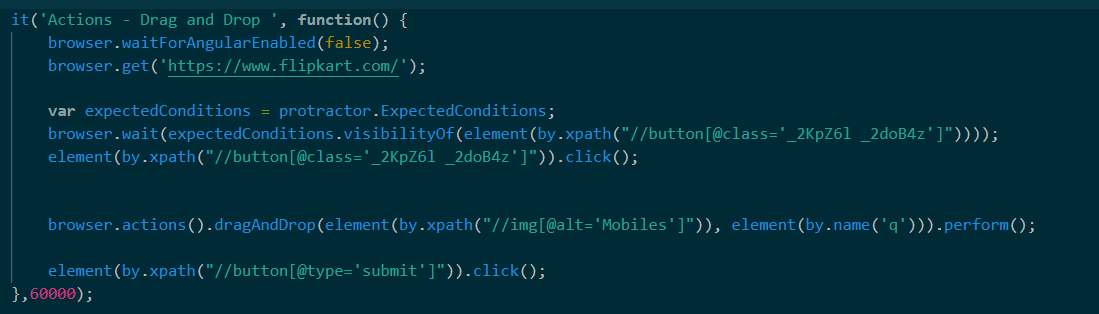
We can use ExpectedConditions from protractor obj to handle any sync issues or to handle dynamic elements in the webpage for non-angular applications



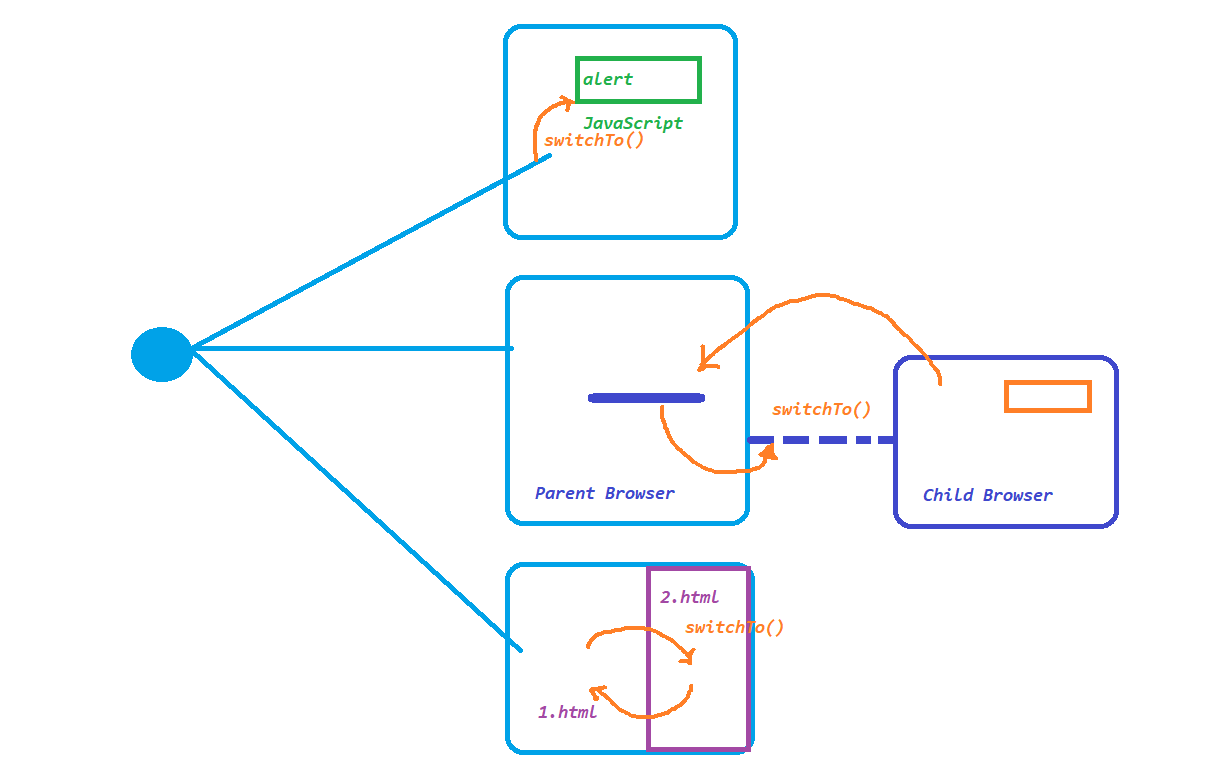
**Actions**

To perform any kb or mouse operations in protractor we can use actions() method present in browser object





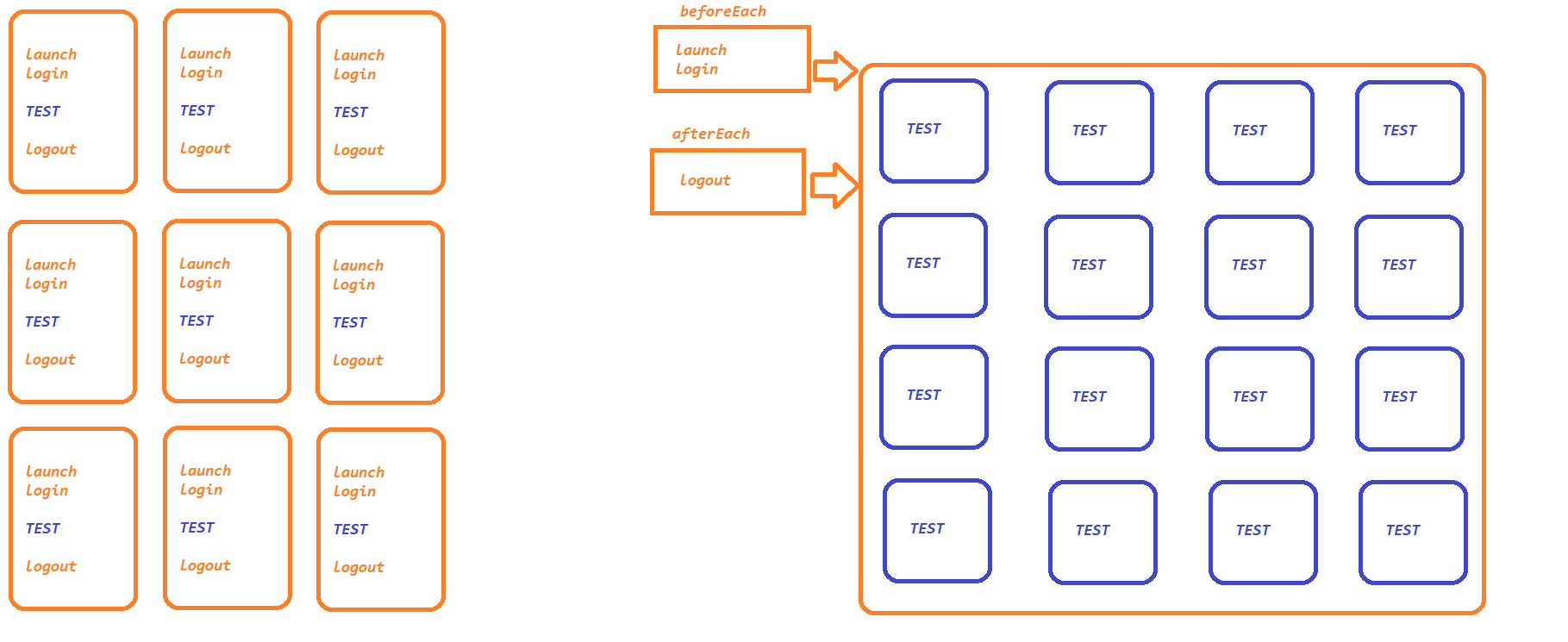
SwitchTo()



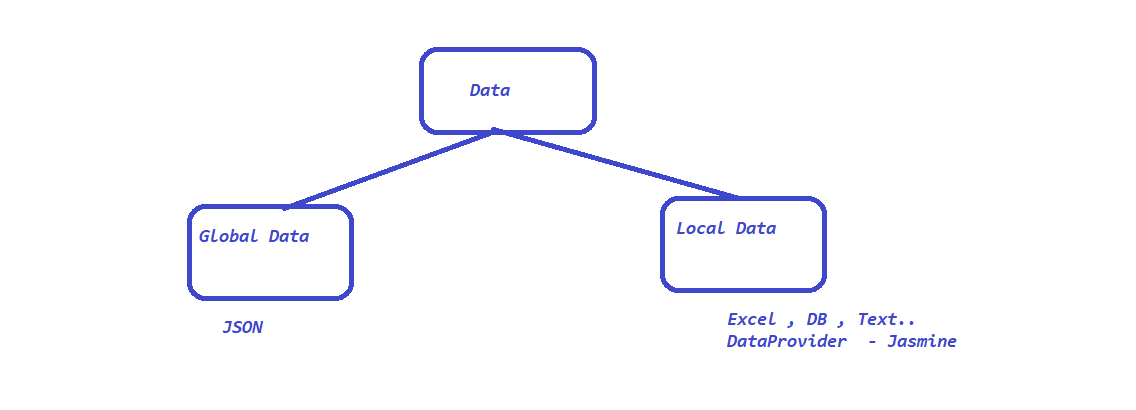
Browser Operations :



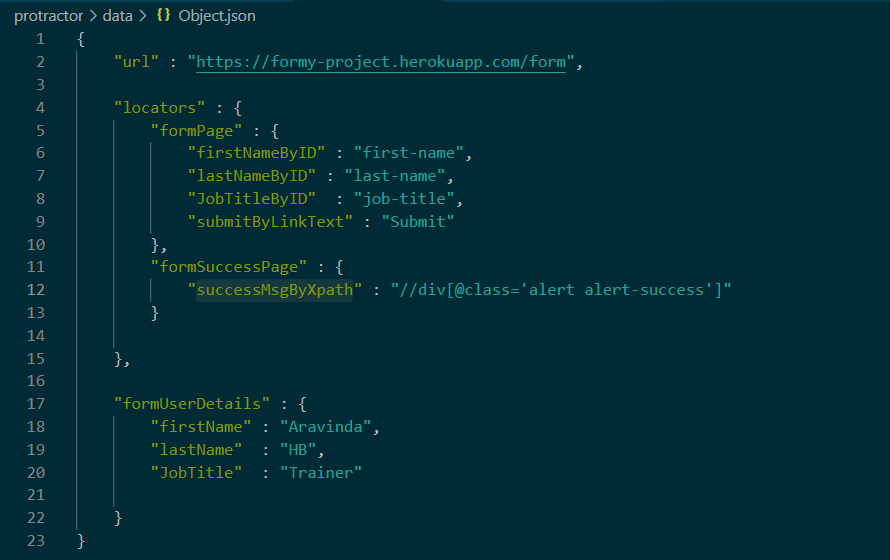
BeforeEach and AfterEach in protractor



Data driven Testing in Protractor

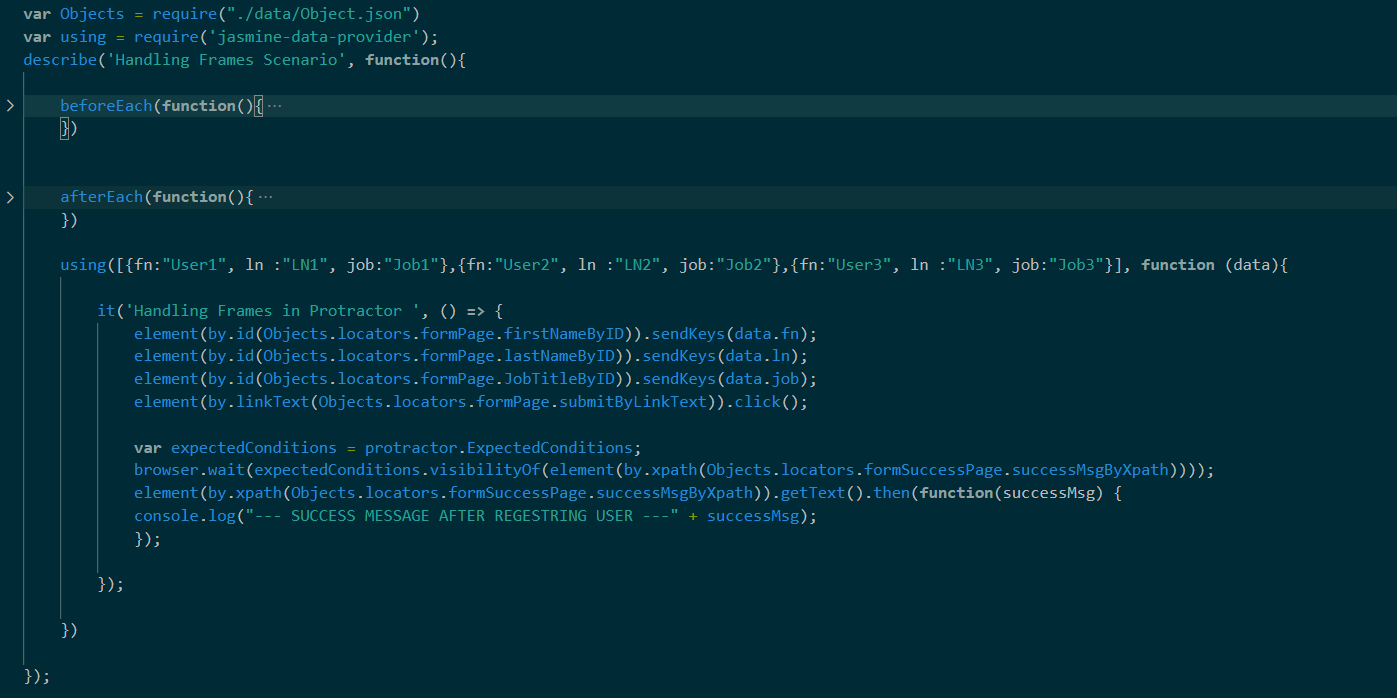


JSON:

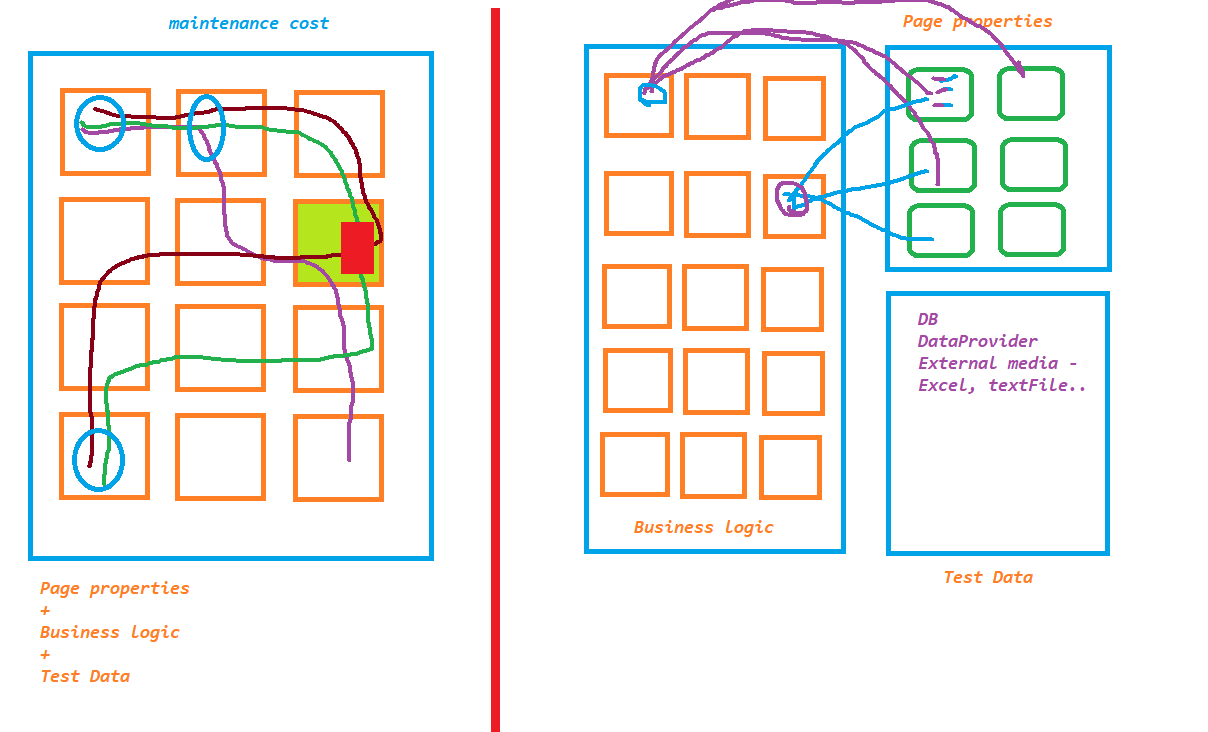


DataProvider:

===========

* Install Data provider plugin <https://www.npmjs.com/package/jasmine-data-provider>
* npm i jasmine-data-provider
* using takes 2 Arguments
  1. first argument is an array
  2. second argument is callback function
* 

POM



Executing Test Suite

