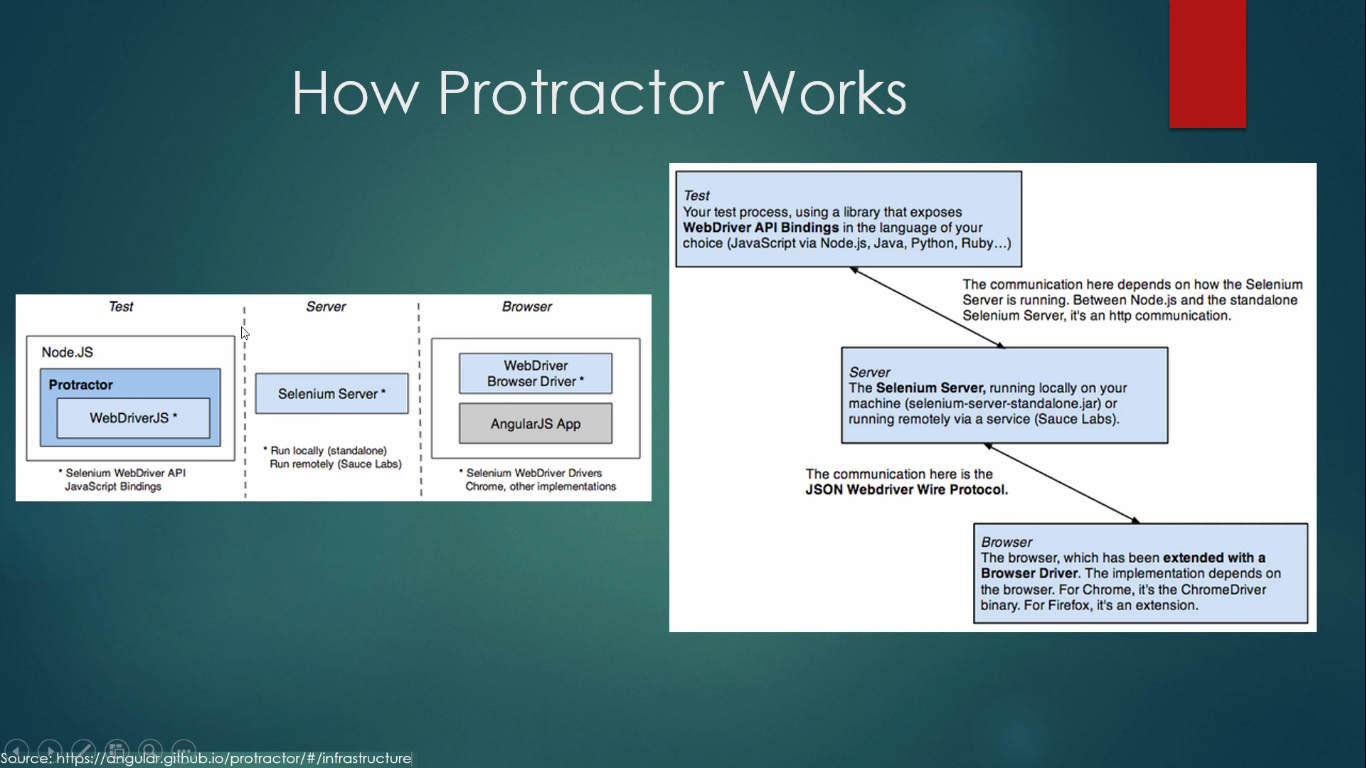
**PROTRACTOR**

**Introduction:**

* **The Protractor is an automation testing tool for web applications testing; combining powerful technologies such as Jasmine, Selenium Webdriver, Node.js etc.**
* **The Protractor testing tool is an end to end behavior-driven testing framework designed for Angular JS applications.**
* **It works with both Angular and non-Angular JS applications equally well.**
* **It’s a wrapper built on top of Selenium Webdriver and thus, provides all the capabilities of Selenium along with many useful additions. It offers:**

****

**Why Do We Need Protractor Framework**

**JavaScript is used in only almost all web applications. As the applications grows,** [**JavaScript**](http://www.guru99.com/interactive-javascript-tutorials.html) **also increases in size and complexity. In such case, it becomes difficult task for Testers to test the web application for various scenarios.**

**Sometimes it is difficult to capture the web elements in AngularJS applications using** [**Junit**](http://www.guru99.com/junit-tutorial.html) **or** [**Selenium**](http://www.guru99.com/selenium-tutorial.html) **WebDriver.**

**Protractor is a NodeJS program which is written in JavaScript and runs with Node to identinfy the web elements in AngularJS applications, and it also uses WebDriver to control the browser with user actions.**

**Angular JS Application:**

**AngularJS applications are Web Applications which uses extended HTML's syntax to express web application components. It is mainly used for dynamic web applications. These applications use less and flexible code compared with normal Web Applications.**

**Why can't we find Angular JS web elements using Normal Selenium Web driver?**

**Angular JS applications have some extra HTML attributes like ng-repeater, ng-controller, ng-model.., etc. which are not included in Selenium locators. Selenium is not able to identify those web elements using Selenium code. So, Protractor on the top of Selenium can handle and controls those attributes in Web Applications.**

**The protractor is an end to end testing framework for Angular JS based applications. While most frameworks focus on conducting unit tests for Angular JS applications, Protractor focuses on testing the actual functionality of an application.**

**What is Node.js?**

* **Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine).**
* **Node.js is a platform built on** [**Chrome's JavaScript runtime**](https://code.google.com/p/v8/) **for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.**
* **Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.**
* **Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.**

[**https://www.tutorialspoint.com/nodejs/nodejs\_introduction.htm**](https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm)

**FEATURES:**

**1) WaitForAngular**

**What this means is that there is no need to manually add waits to your script and Protractor will automatically wait for the web elements to load and only then executes the next steps.**

* **Simple syntax to write test cases**
* **The ability to run multiple browsers at once using Selenium Grid**
* **Angular-specific locators**
* **Support for Behavior-driven development such as Jasmine/Mocha**
* **No need to add sleeps/waits**
* **Supported integration with Jenkins/Browser Stack/Grunt etc.**
* **Get rid of dealing with synchronization issue in Angular JS websites**
* **Multiple browser support (Firefox, Chrome, Safari, Internet explorer)**
* **Ability to run the same scripts in mobile browsers also without the need to change the code**

**Which Framework to use?**

**It supports two Behavior-driven development (BDD) test frameworks right out of the box:**

* **Jasmine:** [**Jasmine**](https://jasmine.github.io/) **is the default test framework when Protractor is installed.**
* **Mocha:** [**Mocha**](https://mochajs.org/) **is a JavaScript test framework which runs on Node.js. If you wish to use Mocha as your test framework, you will have to set it up with your Protractor and also will need to use Behavior Driven Development interface and Chai Assertions with Chai.**

**How to Download and Setup Protractor**

**As it is a node.js program, you need to install node.js to get it running. Node.js can be downloaded from this URL by selecting the operating system:** [**Node.js download**](https://nodejs.org/en/download/)

**With node.js, you also get *Protractor npm* package, which can now be used to install Protractor.**

**Now that node.js is installed to your machine, open the command prompt(cmd) and hit the below command to install it globally:**

|  |  |
| --- | --- |
| **1** | **npm install -g protractor** |

***‘-g’* is used to install this tool globally. Remove it if you do not wish to install it globally. This command also installs the Protractor API along with a default selenium server, which means that you don’t necessarily have to start a standalone server.**

**Now, we need to install the selenium server and ChromeDriver. Use the following command (also comes with Protractor) in cmd:**

|  |  |
| --- | --- |
| **1** | **webdriver-manager update** |

**This is all we need to start with our first test case writing using this framework. If you wish, you can install any IDE/editor to write your code. Eclipse IDE is the popular one, but there are also some more cool editors to consider. Personally, I prefer ‘Atom’ editor for my Protractor code writing.**

**To create first test case:**

**It needs 2 files to run:**

* **Configuration file**
* **Spec file.**

**The *Configuration file* is the one that tells Protractor where to find the test files (specs)/which browser to pick/which framework to use (Jasmine/Mocha)/where to talk to your Selenium browser and other configurations. If any configuration is not defined in configuration files, it will use defaults.**

**Spec file is the one where we write our actual test code. All of our test functional flow/assertions will be in this specific file. There might be several spec files based on the number of test cases but just 1 spec file will be able to run the entire test suite of several specs.**

**#1) *Spec.js***

* **All of the browser level commands will be handled by ‘*browser*’, a global created by Protractor.**
* **As we are following Jasmine framework, ‘*describe’* and ‘*it*’ are the syntaxes of Jasmine. Describe will contain the entire end to end the flow of your test case, whereas ‘it’ might contain some of test steps/scenarios etc. You can have multiple ‘*it*’ blocks in your program if you wish so.**
* ***browser.get* is a simple Selenium syntax which tells Protractor to hit a specific URL in the browser.**
* **As the website we are trying to hit is a non-angular website, we set the *ignoreSynchronization* tag to ‘*true*’ as displayed at line# 4. If you do not make this tag true, your test will fail with error “Angular could not be found on the page”. The reason behind this is that Protractor expects to work with angular websites by default, and if we are using Protractor to validate the non-angular website, we need to tell this to Protractor explicitly. However, if you are working on angular websites, there is no need to use this statement as Protractor will by default consider the web page to be angular.**
* **‘expect’ is nothing but the assertion where we are comparing the web page title to equal some predefined data. We will discuss more assertions in detail.**

**#2) *conf.js***

* **As discussed earlier, the configuration file is the one which tells Protractor the core details. As displayed in the code, the framework is ‘Jasmine’.**
* **Inside the capabilities section, browser configurations are set. You can define the browser name such as Firefox/chrome etc. You can also set the maximum instances of the browsers so that at one time, you can run multiple test cases on different available browser windows.**
* **In the ‘*specs*’ section, we give the path of the spec file, i.e. exactly where the spec file is located with respect to the configuration file.**
* **There are also many other cool features you can attach to your configuration file such as reporting/onPrepare function/threshold timeout etc. We will cover few of these in this tutorial.**

**Code Explanation of spec.js:**

1. **The describe syntax is from the Jasmine framework. Here "describe" typically defines components of an application, which can be a class or function etc. In the code suite called as "Enter GURU99," it's just a string and not a code.**
2. **it('should add a Name as GURU99', function()**
3. **browser.get('https://angularjs.org')**
4. **As like in Selenium Webdriver browser.get will open a new browser instance with mentioned URL.**
5. **element(by.model('yourName')).sendKeys('GURU99')**
6. **Here we are finding the web element using the Model name as "yourName," which is the value of "ng-model" in the webpage.**
7. **varguru=element(by.xpath('html/body/div[2]/div[1]/div[2]/div[2]/div/h3'))**
8. **Here we are finding the web element using xpath and store its value in a variable "guru".**
9. **expect(guru.getText()).toEqual('Hello GURU99!')**
10. **Finally we are verifying the text which we have got from the webpage (using gettext() ) with expected text .**

**Logic of conf.js:**

**exports.config = {  
 seleniumAddress: 'http://localhost:4444/wd/hub',  
 specs: ['spec.js']  
};**

**Code Explanation of conf.js**

1. **seleniumAddress: 'http://localhost:4444/wd/hub'**
2. **The Configuration file tells Protractor the location of Selenium Address to talk with Selenium WebDriver.**
3. **specs: ['spec.js']**
4. **This line tells Protractor the location of test files spec.js**

[**http://www.guru99.com/protractor-testing.html**](http://www.guru99.com/protractor-testing.html)

**http://www.protractortest.org/#/api**