What is protractor:

Protractor is node.js program written on top of webdrive.js to test mainly for Angular.js applications. So it is like wrapper on top of webdriver.js.

Applications written in angular has some extra elements which is difficult to selenium to locate these elements like ng:model, ng:repeater, ng:binding. ng:controller etc. In protractor we can find Angular elements directly like:

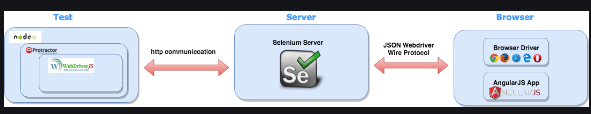
element(by.model(“”)) or element(by.repeater(“”))

We can test non-angular applications as well by protractor.

There are also sync issues while writing test in Selenium for Angular application, Protractor manages all sync related issues.

Angular application are asynchronous application and Protract manages all async call backs. In normal web application for any event entire page loads which is scnchronus behavior and selenium can manage this. But in Angular application there is no page load happens, only data is changed at that particular section at client side, Protractor manages all these call backs and there is no sync issue between elements. In selenium we face sync issues while doing automation for Angular applications and we need to put some wait conditions. Protractor automatically manage all this behavior.

Protractor workflow:



Test scripts send commands to the Selenium server.

Selenium Server interacts with browser driver.

Browser driver interact with application on the real browser.

What is Jasmine:

It is BDD framework to test any java script application, while we do automation testing with Protractor we use Jasmine as default BDD framework. Like Junit is default automation framework for Selenium, Jasmine is for Protractor. It is downloaded default with Protractor. It is open source.

It has generic methods which are used to write test case like in Cucumber we have when then and, Jasmine has describe, it and except keywords which are mostly used. There are other methods as well like afterall, aftereach, beforeall, beforeeach etc.

Every test case should have at least one describe and it method.

describe(“Testing Search Engine”,function()){

it(“Open browser and search”,function()){

browser.get(<http://google.com>);

element(by.name(“q”);

}

}

Working with Protractor in Eclipse:

Download Tern plugin for eclipse from marketplace.

Create new java project in eclipse and covert project to Tern project and select node.js, Jasmine and protractor and convert.

Tern plugin gives all intellisence while writing code.

Create one conf.js file which will have all information about browser, testfile, seleniumAddress and other capabilities.

exports.config = {

capabilities: {

'browserName': 'chrome',

'ignoreProtectedModeSettings': **true**

},

seleniumAddress: 'http://localhost:4444/wd/hub',

specs: ['../Day1/Test1.js']

};

Now create test.js where you will have all your tests like:

describe("Open Google", **function**() {

it("Search in Google", **function**() {

browser.get("http://www.google.com");

browser.sleep(3000);

element(by.name("q")).sendKeys("Protractor");

})

})

**Interview questions**

1. **What are the prerequisites required to run Protractor?**

To run Protractor, we need to have Selenium WebDriver and Node.js installed. We can download Protractor package using npm.

1. **How Protractor, Selenium Server, and Selenium WebDriver work together?**

**Defined above.**

**What are the test frameworks supported by Protractor?**

Protractor supports two behavior driven development (BDD) test frameworks such as Jasmine & Mocha.

**What is Typescript?**

TypeScript is a super set of JavaScript compiled to JavaScript. TypeScript is a strongly typed, object oriented, compiled language developed and maintained by Microsoft. TypeScript may be used to develop JavaScript applications for both client-side and server-side (Node.js) execution

**What is a conf file in Protractor?**

The configuration file tells Protractor how to set up the Selenium Server, which tests to run, how to set up the browsers, and which test framework to use. The configuration file can also include one or more global settings. The config file provides explanations for all of the Protractor configuration options. Default settings include the standalone Selenium Server, the Chrome browser, and the Jasmine test framework.

**11. What is a spec file in Protractor?**

Spec file is the one where we write actual test code. It contains the logic and locators to interact with an application.

**How to exclude a spec file in Protractor**?

You can exclude the spec’s by adding them in the exclude tag in your conf.js file. Say we want to exclude test name “myTest.js”  
exclude: [myTest.js]

**What is the use of directConnect in Protractor?**

Protractor can test directly against Chrome and Firefox without using a Selenium Server. To use this, in your config file set directConnect: true.  
directConnect: true – Your test script communicates directly Chrome Driver or Firefox Driver, bypassing any Selenium Server. If this is true, settings for seleniumAddress and seleniumServerJar will be ignored. If you attempt to use a browser other than Chrome or Firefox an error will be thrown.  
The advantage of directly connecting to browser drivers is that your test scripts may start up and run faster.

**What are the locators in Protractor?**

Protractor supports all the element location strategies given by Selenium and it also has unique set of locators particularly to identify elements based on AngularJS attributes.

Selenium locators:

by.className  
by.css  
by.id  
by.linkText  
by.name  
by.partialLinkText  
by.tagName  
by.xpath

Angular Specific Locators

by.binding  
by.exactBinding  
by.model  
by.repeater  
by.exactRepeater  
by.options

**16. How do you check the status of a webdriver manager and how do you update webdriver manager?**

To check the status of webdriver manager, run the following in your command prompt

**webdriver-manager status**

To update webdriver manager, run the following code in your command prompt

***webdriver-manager update***

**19. What is browser.refresh in Protractor?**

browser.refresh makes a full reload of page.

**What is the difference between “GET” and “NAVIGATE in Protractor?**

**Get:**  
Get method is used to navigate to the given destination.  
browser.get(“https://www.softwaretestingmaterial.com”);

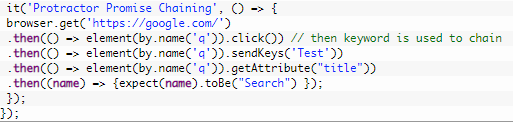
browser.navigate()

if we want to load a previous page or next page in a browser we can use back and forward.

browser.navigate().back();  
browser.navigate().forward();

What is promise in Protractor or use of then?

A promise in protractor confirms that next action will be performed only when last action is done, it manages all sync issues. For ex:



1. *browser.get(‘https://google.com/’) executes and then it waits for page to load*
2. *once the page is loaded it executes element(by.name(‘q’)).click() then it ensures action is performed*
3. *Then it executes the line element(by.name(‘q’)).sendKeys(‘Test’) once this action is done next line and continues…*
4. *.then(() => element(by.name(‘q’)).getAttribute(“title”)) .then((name) => { expect(name).toBe(“Search”) }); watch this line we are getting attribute once this operation is successful it returns value that is resolved now the name will hold the actual attribute value.*

So by using then, there is no need for any sleep or wait as it will manage all syncing issues.

Protractor automatically manage sync issues but still if you are facing any you can use then method.

**How do you verify tooltip text using Protractor?**

**We can read title attribute which is nothing but tooltip like:**