Latest Selenium TestNG Interview Questions – 2017.

### Q-1: Why Should We Prefer Using WebDriver Instead Of Selenium IDE?

First of all, Selenium IDE is a known Record and Playback tool and also very easy to use. But at times, it could be quite deceptive as well. Also, we can’t scale up automation using the record/playback technique while testing a large enterprise application. Next, web applications go through frequent changes which make the IDE more difficult for the testers to adopt. So, it’s not an ideal solution to automate in a production testing environment.

Let’s consider the following example. Say, we recorded a test and found an element with a dynamic ID. Then, we imported it into Eclipse but think what would we do if the test starts failing after some time. It could be because the ID is no more on available on the web page. Might the developer would have changed it while adding a new feature. So it’s better being Agile and use something like Webdriver to create a test suite adaptive to such changes.

Also, to your note, the idea of IDE was to work as a quick automation tool and not an alternative to as the functional test suite.

### Q-2: What Is The Easiest Way To Get The Value From A Text Field In Selenium?

We can use the **getText()** method to fetch the value of a text field in Selenium. For illustration, let’s consider the below example.

<p id="title">Hello world!</p>

When we use the **getText()** method, it’ll return the text as **“Hello world!”**.

For retrieving the text from a **<p>** tag, we’ll call **getText()** in the following manner.

findElement(By.id("title")).getAttribute("value");

If we’ve to get the text from a **Combobox**, then something like the below code would work.

Select comboBox = new Select(findElement(By.id("title")));

comboBox.getFirstSelectedOption().getText();

### Q-3: What Is The Best Approach For Reading A JavaScript Variable From Selenium WebDriver?

We’ll create a JavaScript file and add a variable to read using the Selenium Webdriver code. Then, we’ll define a **setter/getter** function in the JavaScript. Probably you can give the following code example to make the interviewer understand your answer.

First code is the JavaScript file which contains a global variable and the **get/set** functions.

var selVar;

$(document).ready(function() {

...

)};

function setVar(new) {

selVar = new;

}

function getVar() {

return selVar;

}

Next, we can use the below Selenium Webdriver code in Java to read the JavaScript variable.

// Do all the necessary imports.

public class JSReaderTest {

WebDriver driver = new FirefoxDriver();

JavascriptExecutor js = (JavascriptExecutor) driver;

@Test

public void testSelVar() {

String selVar = (String) js.executeScript("alert(getVar());");

Assert.assertEquals("new value for selVar is ", selVar);

}

}

So we can call the **getVar()** method from the Selenium test to read the Javascript variable and set asserts to verify it.

### Q-4: How Will You Get The HTML Source Of A <WebElement> In Selenium WebDriver Using Python/Java/C#?

We can call the API to extract the **<innerHTML>** attribute of a web element. Similarly, we can use the **<outerHTML>** attribute to get the source of the selected element.

**Example-1:** Get HTML source in Java.

webElement.getAttribute("innerHTML");

**Example-2:** Get HTML source in Python.

webElement.get\_attribute("innerHTML")

**Example-3:** Get HTML source in C#.

webElement.GetAttribute("innerHTML");

### Q-5: How To Initialize A List Of Web Elements In Selenium Webdriver?

Since it’s a common use case, so we had a chance to handle it in our project. We used the following approach in our test suite.

private List<WebElement> items;

public List<WebElement> getSelects() {

items = getDriver().findElements(By.xpath("..."));

return items;

}

### Q-6: What Is The Best Way To Click Screenshots When A Test Fails In Selenium 2.0?

Probably, for taking screenshots, the best we can do in Selenium 2.0 is as follows.

@AfterMethod public void takeScreenShotOnFailure(ITestResult testResult) throws IOException {

if (testResult.getStatus() == ITestResult.FAILURE) {

System.out.println(testResult.getStatus());

File errorImage = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

FileUtils.copyFile(errorImage, new File("C:\\errorImage.jpg"));

}

}

### Q-7: How To Fix The Below Chrome Driver Error In Selenium?

“The path to the driver executable must be set by the webdriver.chrome.driver system property.“

We can use the **<selenium-server-standalone-\*.jar>** and pass the **<webdriver.chrome.driver>** property while launching it.

java -jar selenium-server-standalone-2.53.1.jar -Dwebdriver.chrome.driver="C:\test\chromedriver.exe"

So, the above command would fix the error. Java command-line option **<-Dproperty=value>**sets a system property value as expected.

### Q-8: What Is The Right Way To Run Selenium WebDriver Test Cases In Chrome?

First of all, we’ve to download the chrome driver from the link: **<https://sites.google.com/a/chromium.org/chromedriver/downloads>**.

Finally, we can use the Java code given below to run Selenium webdriver tests in Chrome.

System.setProperty("webdriver.chrome.driver", "C:\\path\\to\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

### Q-9: What To Do For Setting Up The <InternetExplorerDriver> For Selenium 2.0?

First of all, we’ll download the IE driver module on our system. We can download it from the link: **<http://www.seleniumhq.org/download/>**. Then, unpack it and place on the local drive.

Finally, copy it to some **<C:\\Selenium\\iexploredriver.exe>** directory. Next, we need to set it up in the system. Therefore, we’ll use the following Java code to run Selenium tests in IE.

File file = new File("C:\\Selenium\\iexploredriver.exe");

System.setProperty("webdriver.ie.driver", file.getAbsolutePath());

WebDriver driver = new InternetExplorerDriver();

### Q-10: What Would You Do To Make The <WebDriver> Wait For A Page To Refresh Before Executing The Test?

It seems like the following Java code will make the **<WebDriver>** wait for a page to refresh. It’ll also ensure the tests should run later.

public void waitForPageToRefresh(WebDriver driver) {

ExpectedCondition < Boolean > expectation = new

ExpectedCondition < Boolean > () {

public Boolean apply(WebDriver driver) {

return ((JavascriptExecutor) driver).executeScript("return document.readyState").equals("complete");

}

};

Wait < WebDriver > wait = new WebDriverWait(driver, 30);

try {

wait.until(expectation);

} catch (Throwable error) {

assertFalse("Timeout occurs.", true);

}

}

### Q-11: How To Get The Selenium Wait Until The Document Is Ready?

In Webdriver, we can write something like the code given below.

driver.manage().timeouts().pageLoadTimeout(10, TimeUnit.SECONDS);

You can check that the code has set a 10-second timeout for page loading. If the page takes more time to load, then the Webdriver will throw a **<TimeoutException>**. We can catch the exception and take action accordingly.

Also, just for clarity, it’s an implicit wait that we set up above. So if you define this once, it’ll remain active till the lifetime of the Web Driver instance.

### Q-12: How To Get Selenium Wait Until The Element Is Present?

Since the **<WebDriver>** provides Fluent Wait concept, so we’ll use it in this case. We can apply Fluent Wait to ignore the **<NoSuchElementException>** exception. It’ll also make the **<WebDriver>** wait for the element.

Probably we can use the below Java code.

FluentWait fluentWait = new FluentWait<>(WebDriver) {

.withTimeout(30, TimeUnit.SECONDS)

.pollingEvery(200, TimeUnit.MILLISECONDS);

.ignoring(NoSuchElementException.class);

}

Next, we’ll check whether the element is present on the page or not. It seems like the below code would do the job for us.

WebElement item = (new WebDriverWait(driver, 10))

.until(ExpectedConditions.elementToBeClickable(By.id("button")));

### Q-13: What Is The Fundamental Difference Between A Wait() And The Sleep()?

The primary difference between a **wait()** and the **sleep()** is as follows.

**1-** First of all, we can end a wait from another thread by using the notify method. But it’s not at all possible to interrupt a sleep call.

**2-** Wait always occurs in a synchronized block whereas the sleep isn’t.

### Q-14: How Can We Run A Particular TestNG Test Group Using A Maven Command?

If we’ve several TestNG test groups like **<group1>, <group2>, <group3>** etc. Then, first of all, we’ll need to get them added to our project’s **<testng.xml>**. Also, if want all of them to run at once, then we need to execute the below **<mvn>** command.

mvn test

Consequently, we can run an another command to run a particular group.

mvn test -Dgroups=group3,group2

### Q-15: Which Command Will You Use To Run A Single Test Method In Maven?

To run a single test method in Maven, we can use the same approach as we did in the previous question. See the below command to run a single test method suing maven.

mvn -Dtest=TestClass#testMethod test

Just for a note, we can use wildcard characters with both method name and class name.

### Q-16: How Can We Run All The Tests Of A TestNG Class?

It is easy to achieve using maven. We can issue the below command to run tests of a class.

mvn test -Dtest=classname

### Q-17: What Will You Do To Run The Selenium Tests Without Using The <Testng.Xml>?

It’s one of the Selenium TestNG interview questions that interviewers love to ask. Please check out the answer below.

Yes, we can do it by using listeners in TestNG. Please refer the below Java code.

TestListenerAdapter listener = new TestListenerAdapter();

TestNG ng = new TestNG();

ng.setTestClasses(new Class[] { MyTestClass.class });

ng.addListener(listener);

ng.run();

### Q-18: How To Disable An Entire Selenium Test In TestNG?

TestNG provides a granular level control to manage test cases. So, it’s relatively easy to turn of a test or a full test suite in TestNG. Hence, please check out the below code.

@Test(enabled=false)

public class MyFirstTest {

//

}

### Q-19: How Can We Attach A Failure Screenshot To The TestNG Report?

Though, we can’t attach an error snapshot. But we can add a link to the screenshot in the test report.

There is a TestNG class as **<org.testng.Reporter.log>** which gives a method to add the link to the TestNG report.

Here are a few steps to do it.

**1-** First of all, create a Listener class and add it to the TestNG project.  
**2-** Write a method to override the **<ontestfailure()>** method. It’s the default error handler for failures in TestNG.  
**3-**Take a screenshot inside the above method, and save it to a file.  
**4-** Use the **Reporter.log()** method to put the hyperlink of the screenshot file in the report.

Finally, you’ll see the failure screenshot linked to the TestNG report.

### Q-20: What Would You Do To Configure The Parallel Execution Of Tests And Classes In <Testng.Xml>?

It’s simple to understand the solution by looking at the below **<testng.xml>** file. It’ll run both the tests and classes in parallel.

<suite name="My Test Suite" allow-return-values="true" verbose="1" parallel="tests" thread-count="2">

<test name="Login Test case 01" parallel="classes" thread-count="2">

<parameter name="Operating\_System" value="Windows 7"/>

<parameter name="Browser\_Name" value="Internet Explorer"/>

<parameter name="Browser\_Version" value="11"/>

<parameter name="Base\_URL" value="http://www.google.com"/>

<classes>

<class name="com.mytest.tool.testing\_01"/>

<class name="com.mytest.tool.testing\_02"/>

</classes>

</test>

<test name="Login Test case 02" parallel="classes" thread-count="2">

<parameter name="Operating\_System" value="Windows 7"/>

<parameter name="Browser\_Name" value="Mozilla Firefox"/>

<parameter name="Browser\_Version" value="47.0.1"/>

<parameter name="Base\_URL" value="http://www.google.com"/>

<classes>

<class name="com.mytest.tool.testing\_01"/>

<class name="com.mytest.tool.testing\_02"/>

</classes>

</test>

</suite>