**Selenium interview questions and answers**

1. **What is Selenium?**

Selenium is a software program which automates web applications and browsers, not desktop applications. It supports various OS platforms, including Linux, Mac, and Windows. Its tools are used basically for Regression and Functional Testing.

1. **What are the Components of Selenium?**

* Selenium Integrated Development Environment (IDE)
* Selenium Remote Control (RC)
* Selenium WebDriver
* Selenium Grid

1. **What is Selenium IDE?**

Selenium IDE comes as a Firefox plugin. It is an integrated development platform to develop and deploy Selenium Test series. It is used to record, playback, edit, and debug the test cases.

1. **What is Selenium RC?**

* Selenium RC (Remote Control) is a testing tool to write automated UI tests of web applications in any language for any HTTP website through the JavaScript-based browser.

1. **What is Selenium WebDriver?**

Selenium WebDriver creates Test cases to run on several browsers, through WebDriver API commands and element locators. With WebDriver, we can use Testing Framework Annotations and Programming features to enhance Test cases.

It is built to provide more concise, easier programming UI along with facing some issues in Selenium-RC API. This tool is developed to support dynamic pages where elements may change without even reloading the page.

**6.What is Selenium Grid?**

With Selenium Grid, we can run Tests on various environments against several browsers. In other words, it supports distributed execution of Tests. We can run multiple tests against various machines at the same time in different operating systems and browsers.

1. **What is Apache Maven? How is it used in Selenium?**

Maven is a Java build management and project management component. With Maven, we can easily manage Java project builds. As a Java build management component, Maven enables us to manage the Selenium build with ease. It manages the documentation, build compilation, and other project related tasks of Selenium test project. With it, we can also create the accurate project structure, manage and add jar files in build path, etc.

1. **What is TestNG? What are its Usage in Selenium?**

Like NUnit or JUnit in Group Test cases TestNG is also a Testing Framework used to run parameterized Tests, Testbatches, and to provide in-depth HTML test reports.

1. **What are the Pros of Selenium?**

* Supports several programming languages to create Test scripts
* Open Source program
* Multi-browser support to conduct Test Scripts/Test Cases.
* Supports several OS platforms to initiate Testing.
* Enables Parallel Test execution.

**10.What are the Cons of Selenium?**

* No built-in support for Reporting Results
* No Tech Support
* Only for Browser/Web-based applications
* No support for new features

**11.how to open browser in selenium?**

**Syntax:**syntax to open browser using selenium is

WebDriver d=new ChromeDriver();//any browser

**12.how to navigate to a specific website?**

d.get(“link”);

13.how to maximize the opened website?

d.manage().window().maximize();

**14.how to find an element in selenium?**

Syntax:syntax to find a element is

d.findElement(“xpath”);

**15.how to enter text in the textbox?**

We can enter text in the textbox using sendkeys method

d.findElement(“xpath”).sendKeys(“data”);

16.what is thread.sleep()? in selenium

* As per the official definition from the [Oracle Java Documentation](https://docs.oracle.com/javase/tutorial/essential/concurrency/sleep.html), **Thread.sleep()** causes the current thread to suspend execution for a specified period.
* **Thread.sleep()** is not a Selenium wait, it is provided by Java.
* **Thread.sleep()** does not have a return type, and it returns void.

**Thread.sleep** methods throws **InterruptedException** when any other thread interrupts the current thread and should be handled by the throws method or try catch block.

Example:

public void threadTest() throws InterruptedException {

Thread.sleep(2000);

}

**17.what are the Limitations of Thread.sleep()**

Using **Thread.sleep()** frequently in an automation framework is not a good practice. If the applied sleep is of 5 secs and the web element is displayed in 2 secs only, the extra 3 secs will increase the execution time. And if you use it more often in the framework, the execution time would increase drastically.

You always have to guess and apply **Thread.sleep()** seconds in advance, as there is no guarantee that the web element would be discoverable under that specified time.

# 18. **What are the different types of wait available in Selenium?**

Implicit wait:

* Implicit [wait](https://www.selenium.dev/documentation/en/webdriver/waits/) tells the**web driver to wait** for a certain amount of time before throwing  an exception. In implicit wait, we give wait time **globally** and it will remain applicable to entire test script. WebDriver will wait for the element to load on the basis of time provided in wait condition. However, if web driver is unable to find an element in a given time, it will throw “**ElementNotVisibleException**“.

:

Syntax:

driver.manage().timeouts().implicitlyWait(12, TimeUnit.SECONDS);

example:

**public** **static** **void** main(String[] args) {

WebDriver d=**new** ChromeDriver();

d.manage().timeouts().implicitlyWait(Duration.*ofSeconds*(6));

d.manage().window().maximize();

d.get("https://www.hyrtutorials.com/p/waits-demo.html");

d.findElement(By.*id*("btn1")).click();

d.findElement(By.*id*("txt1")).sendKeys("hyr");

}

explicit wait:

Definition:

Description:

Syntax:

WebDriverWait w = new WebDriverWait(driver,);

w.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("<<xpath expression>>")))

example:

**public** **static** **void** main(String[] args)

{

WebDriver d=**new** ChromeDriver();

d.get("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login");

d.manage().window().maximize();

WebDriverWait w=**new** WebDriverWait(d,Duration.*ofSeconds*(30));

WebElement username=w.until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//input[@name='username']")));

username.sendKeys("Admin");

}

fluent wait:

Definition:

Description:

Syntax:

Wait<WebDriver> w = new

FluentWait<WebDriver>(driver).withTimeout(Duration.ofSeconds(30))

.pollingEvery(Duration.ofSeconds(3)).ignoring(NoSuchElementException.class);

Example:

Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)

.withTimeout(Duration.ofSeconds(30))

.pollingEvery(Duration.ofSeconds(5))

.ignoring(NoSuchElementException.class);

WebElement foo = wait.until(driver -> {

return driver.findElement(By.id("foo"));

});