Waits in Selenium

- Most of the web elements are written using ***Java script*** or ***AJAX.***
- They may take certain amount of time to load on page.
- In this situation chances of getting '***NoSuchElementException'*** are more.
- To avoid such exceptions, Selenium has provided two types of Waits - **Implicit Wait** & **Explicit Wait.**

1. Implicit Wait:

- This wait is set for the lifetime of WebDriver instance.
- Wherever we use WebDriver instance before that line wait will be applied.
- This wait, polls for 500milliseconds (*This time depends on the browsers implementation class of webdriver. For firefox it is 500milliseconds, it depends on type and version of browser*).
- If element becomes available in first 500milliseconds then element will be returned else it will wait for next 500 milliseconds. Our script will wait for max timeout.
- We can specify default time out.
- If element is not available in timeout value then '***NoSuchElementException'*** will be thrown.
- We can specify timeouts in terms of: NanoSeconds, MicroSeconds, MilliSeconds, Seconds, Minutes, Hours and Day.
- E.g

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

- In above example timeout value is 10 Seconds.
- Now our script will wait for min 500 milliseconds and maximum 10 Second before each and every WebElement that we wish to find using '*driver*' instance.
- Eg. -

```
public class ImplicitWaitDemo {
   public static void main(String[] args) {
     WebDriver driver;
     driver=new FirefoxDriver();
     driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
     driver.findElement(By.xpath("Element1")).click();
     driver.findElement(By.xpath("Element2")).click();
     driver.findElement(By.xpath("Element3")).click();
  }
}
```

In above example, our script will wait before clicking all three elements (*Element1, Element2 and Element3*).

- Minimum wait time is 500milliseconds and maximum wait time is 10 seconds

2. Explicit Wait:

- This wait is more customized than implicit wait.
- We can apply this wait for particular web element.
- We can even customize the polling time in explicit wait.
- We can also ignore the exceptions that we don't want while locating an element.
- Same as implicit wait, we can set timeout for the web element.
- We can also make our script wait until certain condition occurs.

We can achieve explicit wait using two classes of Selenium: ***FluentWait*** and ***WebDriverWait.***

1. Using FluentWait:

- FluentWait class implements ***Wait*** interface.
- We can set our own ***timeout*** and ***polling time*** using FluentWait.
- We can also ***ignore exceptions*** while waiting for particular WebElement.
- We can ***wait till certain condition*** of a web element.
- This wait can be ***applied*** *to* ***particular WebElement*** unlike 'Implicit Wait' which applied for entire life span of WebDriver instance.

Methods of FluentWait class:

1. ***withTimeout(long duration, TimeUnit unit)***

- This method is used to set maximum time that script will wait.
- This method takes two parameters: ***long duration*** and ***TimeUnit*** in terms of milliseconds, microseconds, seconds, hours, days etc.
- E.g.

```
public class FluentWaitDemo {
   public static void main(String[] args) {
      WebDriver driver;
      driver=new FirefoxDriver();
      FluentWait wait=new FluentWait(driver);
      wait.withTimeout(10, TimeUnit.SECONDS);
   }
}
```

- In above example, wait is configured for maximum 10 seconds. If element is not available in 10 seconds, then ***NoSuchElmentException*** will be thrown.

2. ***pollingEvery(long duration, TimeUnit time);***

- This method is used to set ***polling time***.
- Polling time is the frequency with which our script will check for web element.
- This method also takes two arguments: duration and time

E.g.

```
public class FluentWaitDemo {
   public static void main(String[] args) {
      WebDriver driver;
      driver=new FirefoxDriver();
      FluentWait wait=new FluentWait(driver);
      wait.pollingEvery(2, TimeUnit.SECONDS);
  }
}
```

- In above example, wait is configured for 2 seconds as polling time.
- This script will search web element after every 2 seconds till maximum timeout.

3. ***ignoring(exceptionType):***

- This method will ***ignore specific*** types of ***exceptions*** while waiting for a condition.
  E.g.

wait.ignoring(NoSuchElementException.class);

- Above script will ignore ***NoSuchElementException*** while searching a web element.

4. ***until(ExpectedConditions):***

- After configuring wait using ***until()*** method, script will keep on waiting until one of the following encounters:
  - Until Expected condition is encountered.
  - Until the timeout expires
  - Until the current thread is interrupted

E.g.

wait.until(ExpectedConditions.alertIsPresent());

- Above script will wait till alert is present on web page.

2. Using WebDriverWait:

- It is a ***specialized version*** of FluentWait.
- All its constructors take WebDriver instance as parameter.
- It inherits almost all methods of FluentWait class.
- So it is recommended to use FluentWait class when explicit wait is required.
- But when we are dealing with WebDriver instance, it is recommended to use WebDriverWait class.
- It has three constructors.

WebDriverWait(WebDriver driver, long timeOutInSeconds)

WebDriverWait (WebDriver driver, long timeOutInSeconds, long sleepInMillis)

WebDriverWait (WebDriver driver, Clock clock, Sleeper sleeper, long timeOutInSeconds, long sleepTimeOut)

Interview Questions -

1. **What are the types of waits available in Selenium WebDriver?**

In Selenium we could see three types of waits such as Implicit Waits, Explicit Waits and Fluent Waits.

- **Implicit Wait:** Implicit waits are used to provide a default waiting time (say 30 seconds) between each consecutive test step/command across the entire test script. Thus, subsequent test step would only execute when the 30 seconds have elapsed after executing the previous test step/command.
- **Explicit Wait:** Explicit waits are used to halt the execution till the time a particular condition is met or the maximum time has elapsed. Unlike Implicit waits, explicit waits are applied for a particular instance only.

2. **What is Implicit Wait in Selenium WebDriver?**

Implicit waits tell to the WebDriver to wait for a certain amount of time before it throws an exception. Once we set the time, WebDriver will wait for the element based on the time we set before it throws an exception. The default setting is 0 (zero). We need to set some wait time to make WebDriver to wait for the required time.

3. **What is WebDriver Wait in Selenium WebDriver?**

WebDriverWait is applied on a certain element with defined *expected condition* and *time*. This wait is only applied to the specified element. This wait can also throw an exception when an element is not found.

4. **Write a code to wait for a particular element to be visible on a page. Write a code to wait for an alert to appear.**

We can write a code such that we specify the XPath of the web element that needs to be visible on the page and then ask the WebDriver to wait for a specified time. Look at the sample piece of code below:

WebDriverWait wait=new WebDriverWait(driver, 20);
Element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath ("<xpath>")));

Similarly, we can write another piece of code asking the WebDriver to wait until an error appears like this:

WebDriverWait wait=new WebDriverWait(driver, 20);
Element = wait.until(ExpectedConditions.alertIsPresent());

5. **What is Fluent Wait In Selenium WebDriver?**

FluentWait can define the maximum amount of time to wait for a specific condition and frequency with which to check the condition before throwing an "*ElementNotVisibleException*" exception.

6. **Difference b/w implicitly Wait and Explicit wait.**

No	Implicit Wait	Explicit Wait
1.	Valids for life time of webdriver instance hence once used applicable to all elements	Can apply to particular web element
2	Achieved by using **implicitlyWait**	Achieved by using **fluentWait** and **WebDriverWait**
3.	Polling time is browser & version specific and fixed and **timeouts** is **customizable**.	**Polling time** and **timeouts** is **customizable**.
4.	this wait doesn't provide a feature to wait till a certain condition occur	this wait provides a feature to wait till a certain condition occur
5.	We **can't ignore** any **exception** using implicit Wait	We **can ignore exception** using explicit Wait
6.	**Syntax** : driver.manage().timeouts.implicitlyWait(XX, TimeUnit.*SECONDS*);	**Syntax : 1) using FluentWait** FluentWait **w** = new FluentWait(driver); **w**.withTimeout(10,TimeUnit.*SECONDS*); **Syntax : 1) using WebDriverWait** WebDriverWait **w** = new WebDriverWait(driver, 20); w.until(ExpectedConditions.elementToBeClickable(By.id(" -------- "))); **Syntax : 1) using WebDriverWait**
7.	throws **NoSuchElementException** after timeout.	throws **NoSuchElementException** after timeout if not ignored

7. **What are the different types of WAIT statements in Selenium WebDriver?** ***Or the question can be framed like this:*** **How do you achieve synchronization in WebDriver?**

There are basically two types of wait statements: **Implicit Wait** and **Explicit Wait**.

Implicit wait instructs the WebDriver to wait for some time by polling the DOM. Once you have declared implicit wait, it will be available for the entire life of the WebDriver instance. By default, the value will be 0. If you set a longer default, then the behavior will poll the DOM on a periodic basis depending on the browser/ driver implementation. Explicit wait instructs the execution to wait for some time until some condition is achieved. Some of those conditions to be attained are:

- elementToBeClickable
- elementToBeSelected
- presenceOfElementLocated

8. What are the Expected Conditions that can be used in Explicit Wait ?

- The following are the Expected Conditions that can be used in Explicit Wait

1. alertIsPresent()
2. elementSelectionStateToBe()
3. elementToBeClickable()
4. elementToBeSelected()
5. frameToBeAvaliableAndSwitchToIt()
6. invisibilityOfTheElementLocated()
7. invisibilityOfElementWithText()
8. presenceOfAllElementsLocatedBy()
9. presenceOfElementLocated()
10. textToBePresentInElement()
11. textToBePresentInElementLocated()
12. textToBePresentInElementValue()
13. titleIs()
14. titleContains()
15. visibilityOf()
16. visibilityOfAllElements()
17. visibilityOfAllElementsLocatedBy()
18. visibilityOfElementLocated()

9. Can you write the syntax of Fluent Wait ?

FluentWait **w** = new FluentWait(driver);
w.withTimeout(10,TimeUnit.*SECONDS*);

10. Can you write syntax of Explicit Wait ?

WebDriverWait **w** = new WebDriverWait(driver, 20);
w.until(ExpectedConditions.elementToBeClickable(By.id(“ -------- ”)));

11. How to speed execution time of Test Script using waits ?

By using explicit wait we can wait for specified time period or till the condition to met.

If condition met before the specified time period then script has not to wait for complete specified time period. In this way speed of execution will improve by using explicit wait.