**SYNCHRONIZATION IN SELENIUM WEBDRIVER**

1. **SYNCHRONIZATION**: It is a mechanism which involves more than one components to work parallel with Each other.

in Test Automation, we have two components  
**1. Application Under Test**  
**2. Test Automation Tool**

### Both these components will have their own speed. We should write our scripts in such a way that both the components should move with same and desired speed, so that we will not encounter "Element Not Found" errors which will consume time again in debugging.

### Synchronization can be classified into two categories:

### 1. Unconditional  2. Conditional Synchronization

### Unconditional : In this we just specify timeout value only. We will make the tool to wait until certain amount of time and then proceed further.

### *Examples:*[*Thread.Sleep();*](https://docs.oracle.com/javase/tutorial/essential/concurrency/sleep.html)

### The main disadvantage for the above statements are, there is a chance of unnecessary waiting time even though the application is ready.

### The advantages are like in a situation where we interact for third party systems like interfaces, it is not possible to write a condition or check for a condition. Here in this situations, we have to make the application to wait for certain amount of time by specifying the timeout value.

### Conditional Synchronization :

### The biggest challenge in automating a web application is, the loading of a web page is always in the mercy of certain conditions,

### Load on the server

### Network speed

### Performance of AUT

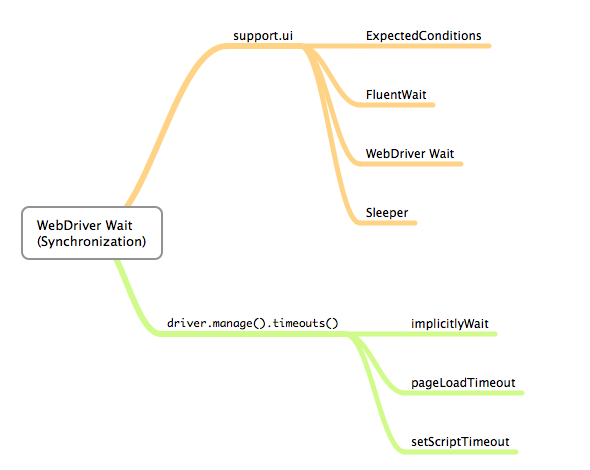
### Ajax call to load an element

### So the major task in automating such application is to wait for the HTML element to appear in the page before your automation test code start performing action on them. So you need to make sure that the Web Element is present before the code start working on it.

### To deal such kind of situation use synchronized wait.

### Synchronized wait means, wait till the condition satisfied once the condition satisfied, come out from the sleep and continue the test execution.

**In Selenium we have implicit Wait and Explicit Wait conditional statements.**



* 1. **IMPLICIT WAIT:**

### The Implicit Wait will tell the Web Driver to poll the DOM for a certain duration when trying to find the element, this will be useful when certain elements on the webpage will not be available immediately and needs some time to load.

### By default it ill take the value to 0, for the life of the Web Driver object instance through out the test script.

**WHY IMPLICIT WAIT IS REQUIRED IN WEB DRIVER:**

As we knows sometimes, some elements takes some time to appear on software web application page when browser is loading the page. In this case, sometime your web driver test will fail if you have not applied Implicit wait in your test case. If implicit wait is applied in your test case then web driver will wait for specified amount of time if targeted element not appears on page.

**If you write implicit wait statement in you web driver script then it will be applied automatically to all elements of your test case**.

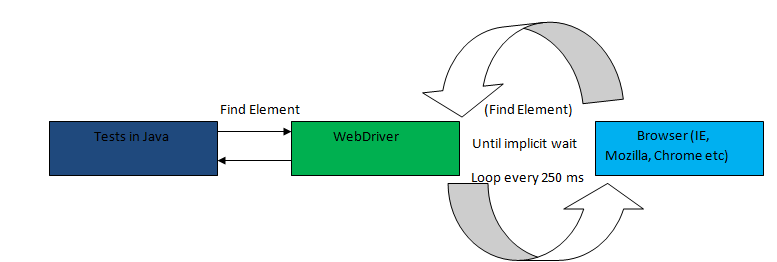
I am suggesting you to use Implicit wait in your all test script of software web application with 10 to 15 seconds.

In web driver, Implicit wait statement is as bellow.

**driver.manage().timeouts().implicitlyWait(time period, TimeUnit*.*SECONDS);**

**How Implicit wait works**

1. Look in the DOM for a particular element. If found move out of the wait.  
2. If not found wait for 250 milli seconds.  
3. After 250 mili seconds poll the DOM again and check for the element. If element found move out of the wait.  
4. If element not found, go back to step number 2 and repeat the following steps



* 1. **TIMEOUTS:**

There are two timeouts supported: 

1. pageLoadTimeout
2. setScriptTimeout.

**pageLoadTimeout**:

It sets the amount of time to wait for a page load to complete before throwing an error. If the timeout is negative, page loads can be indefinite.   
  
          driver.manage().timeouts().pageLoadTimeout(100, SECONDS);

**setScriptTimeout:**

It sets the amount of time to wait for an asynchronous script to finish execution before throwing an error. If the timeout is negative, then the script will be allowed to run indefinitely.   
  
         driver.manage().timeouts().setScriptTimeout(100,SECONDS);

* 1. **Explicit Wait:**

Explicit wait is a programmatic approach to problem of waiting for specific elements. Contrary to Implicit Wait t Explicit Wait requires more coding but also gives much more flexibility.This provide you better option than implicit wait.

**Example :**

@Test

public void selectByValueExample() {

driver = new FirefoxDriver();

driver.get("http://facebook.com");

driver.manage().window().maximize();

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

driver.get("http://facebook.com");

// explicit wait for search field

WebDriverWait wait = new WebDriverWait(driver, 10);

**//Below is the syntax to check if the element is present on the DOM of a page and visible. Visibility means that the element is not just displayed but also should also has a height and width that is greater than 0.**

**wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("searchInput")));**

driver.findElement(By.id("searchInput")).clear();

driver.findElement(By.id("searchInput")).sendKeys("India");

driver.findElement(By.id("searchButton")).click();

}

}

**EXAMPLES FOR VARIOUS EXPLICIT WAITS:**

1. isElementClickable
2. isElementVisible
3. isElementInVisible
4. isElementEnabled
5. isElementDisplayed
6. waitforinvisibilityofelement
7. waitforinvisibilityofelementwithText
8. waitForElementToBeClickable
9. alertIsPresent
10. titleIs
11. frameToBeAvailableAndSwitchToIt

**How to Implementation :**

1. **elementToBeClickable**() – The expected condition waits for an element to be clickable i.e. it should be present/displayed/visible on the screen as well as enabled.  
  
wait.until(ExpectedConditions.elementToBeClickable(By.xpath(“//div[contains(text(),’COMPOSE’)]”)));  
  
2. **textToBePresentInElement**() – The expected condition waits for an element having a certain string pattern.  
  
wait.until(ExpectedConditions.textToBePresentInElement(By.xpath(“//div[@id= ‘forgotPass'”), “text to be found”));  
  
3. **alertIsPresent**()- The expected condition waits for an alert box to appear.  
  
wait.until(ExpectedConditions.alertIsPresent()) !=null);  
  
4. **titleIs**() – The expected condition waits for a page with a specific title.  
  
wait.until(ExpectedConditions.titleIs(“gmail”));  
  
5. **frameToBeAvailableAndSwitchToIt**() – The expected condition waits for a frame to be available and then as soon as the frame is available, the control switches to it automatically.  
  
wait.until(ExpectedConditions.frameToBeAvailableAndSwitchToIt(By.id(“newframe”)));

**Q. What is Selenium's default timeout for page loading?**

The implicit wait timeout is set to 0 by default. This means that if a command that finds elements does not find anything, it won't wait.

The page load timeout is set to -1 by default. This means that Selenium will wait indefinitely for the page to load.

**Q**: **As we know that if we apply implicit wait in our script once and this will be applied to all code statement of the script.  
But if we add explicit wait in addition to this wait for specific element, then does explicit wait actually overrides the implicit wait or not ?**

**ANS**: Explicit will never override the Implicit wait.It will wait for a specific condition only .Once set, the implicit wait is set for the life of the WebDriver object instance.