**Wait**

When a page loads on a browser, the various web elements that someone wants to interact with may load at various time intervals. This obviously creates difficulty in identifying any element. To eliminate this difficulty, we have the below approaches.

Synchronization (Wait) can be classified into:

* Implicit
* Explicit
  + WebDriverWait - Conditional
  + Fluent wait - Conditional

**Implicit Wait**

1. **ImplicicityWait()**

* Implicitly wait is applied globally, which means it is always available for all the web elements throughout the driver instance.
* Implicit waits are used to provide a default waiting time between each consecutive test step/command across the entire test script.

**Syntax:** driver. manage (). timeouts(). implicitlyWait(4,TimeUnit.SECONDS)

The implicit wait will accept 2 parameters, the first parameter will accept the time as an integer value and the second parameter will accept the time measurement in terms of SECONDS, MINUTES, MILLISECOND, MICROSECONDS, NANOSECONDS, DAYS, HOURS, etc.

With the use of this Syntax, for each and every locator in the program, the driver will wait for a maximum of 4 seconds on the visibility of webelement. if the locator is available before 4 seconds then the line of code will be executed immediately.

1. **pageLoadTimeout()**

pageLoadTimeout command waits for the page to load completely for a specified number of seconds.

Syntax: driver.manage().timeouts().pageLoadTimeout(30, TimeUnit.SECONDS);

**Explicit Wait**

* Explicit waits are used to halt the execution until the time a particular condition is met or the maximum time has elapsed.
* It can be implemented by the WebDriverWait class
* Below are the most common expected conditions for explicit Wait to implement
* alert is present
* element exists
* element is visible
* title contains
* title is
* element staleness
* visible text

**Implementation Package**

org.openqa.selenium.support.ui.WebDriverWait;

org.openqa.selenium.support.ui.ExpectedConditions;

**Implementation Class:**

WebDriverWait

**Declaration Syntax**: WebDriverWait explicitWait = new WebDriverWait(driver, Duration.ofSeconds(5))

explicitWait -> Object Created for WebDriverWait

driver -> WebDriver

Duration.ofSeconds(5) -> Duration of Timeout with Integer Value in Time Units

WebDriverWait is the Class that is used to implement Explicit wait.

The Explicit wait Class will accept 2 parameters, the first parameter will accept the WebDriver and the second parameter will accept the time in Integer and measurement in terms of SECONDS, MINUTES, MILLISECOND, MICROSECONDS, NANOSECONDS, DAYS, HOURS, etc.

**Executable Syntax** explicitWait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath(“xpath")));

Until is a method used with Expected Conditions for the locator to met.

explicitWait.until(ExpectedConditions -> Standard Syntax

visibilityOfElementLocated -> Standard Conditions Available as a part of ExpectionConditions Class

(By.xpath(“xpath") -> Locator

**Fluent Wait** - The Fluent Wait command defines the maximum amount of time for Selenium WebDriver to wait for a certain condition to appear. It also defines the **Frequency of** Setting up a repeat cycle with the time frame to verify/check the condition at the regular interval of time

**Implementation Package**

org.openqa.selenium.support.ui.FluentWait;

**Implementation Class**

FluentWait

**Implementation Interface**

Wait

**Declaration Syntax :**

Wait<WebDriver> wait = new FluentWait<WebDriver>(driver).withTimeout(Duration.ofSeconds(40)).pollingEvery(Duration.ofSeconds(8)).ignoring(Exception.class);

Wait -> Interface

<WebDriver> wait -> Objection Declaration

New FluentWait<WebDriver>(driver) -> Class

withTimeout(Duration.ofSeconds(40) -> Duration of Timeout with Integer Value in Time Units

pollingEvery(Duration.ofSeconds(8) -> Expected frequency of visit for the locator in DOM

ignoring(Exception.class) -> to Ignore any specified Expections reported during polling.

Some common exception classes are

* NoSuchWindowException.
* NoSuchFrameException.
* NoSuchElementException.
* NoAlertPresentException.
* InvalidSelectorException.
* TimeoutException.
* ElementNotVisibleException.
* ElementNotSelectableException.

**Executable Syntax**

WebElement clickseleniumlink = wait.until(new Function<WebDriver, WebElement>(){

public WebElement apply(WebDriver driver ) {

return driver.findElement(By.xpath(Locator));

}

}

WebElement clickseleniumlink -> Value of the Return WebElement stored

wait.until(new Function<WebDriver, WebElement>(){

public WebElement apply(WebDriver driver ) -> Standard Syntax

driver.findElement(By.xpath(Locator) -> WebElement that need to be Monitored