# Selenium Introduction

# Selenium Architecture

# Selenium components

# Supported Languages in Selenium

# DOM Introduction

# Selenium Webdriver – First code

# Locators

# Methods

# Action classes

# Waits

Selenium wait commands instruct the test execution to hold/pause for a certain length of time before executing the next step.

It will be useful when we might need WebDriver to check if one or more web elements are present/visible/enriched/clickable, etc.

**Types of Wait:**

* Implicit wait
* Explicit wait
* Fluentwait

**Implicit wait:**

Implicit Wait instructs the Selenium WebDriver to wait for a certain measure of time before throwing an exception. Once this time is set, WebDriver will wait for the element before the exception occurs. But not always. If a webelement is found, then it will proceed to perform the next steps.

**Syntax:**

driver.manage().timeouts().implicitlyWait(20, TimeUnit.*SECONDS*);

**Explicit wait:**

Unlike implicit wait, explicit wait in Selenium will wait for certain conditions to occur. The conditions could be waiting for the presence of the web element, waiting for the element to be clickable, waiting for the element to be visible, etc

Explicit wait in Selenium works only on the particular web element on which it is set, rather than all the elements on the page.

But it can be used when we need a synchronization i.e loading of the web page is complete and you are waiting for the any element to be visible.

Below are few options available as Explicit wait.

* alertIsPresent()
* elementSelectionStateToBe()
* elementToBeClickable()
* elementToBeSelected()
* frameToBeAvaliableAndSwitchToIt()
* invisibilityOfTheElementLocated()
* invisibilityOfElementWithText()
* presenceOfAllElementsLocatedBy()
* presenceOfElementLocated()
* textToBePresentInElement()
* textToBePresentInElementLocated()
* textToBePresentInElementValue()
* titleIs()
* titleContains()
* visibilityOf()
* visibilityOfAllElements()
* visibilityOfAllElementsLocatedBy()
* visibilityOfElementLocated()

Few examples of ExpectedConditions class:

new WebDriverWait(driver,10).until(ExpectedConditions.*presenceOfElementLocated*(By.*xpath*("//div[contains(@id,'block\_top\_menu')]/ul/li[3]"))));

new WebDriverWait(driver,10).until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//div[contains(@id,'block\_top\_menu')]/ul/li[3]")))

new WebDriverWait(driver,10).until(ExpectedConditions.*presenceOfElementLocated*(By.*xpath*("//div[contains(@id,'block\_top\_menu')]/ul/li[3]")))

**Fluent wait:**

Also sometimes called as Smart waits. It is also a type of Explicit wait.

We can customize the explicit wait to be working more smarter, using Fluent wait, by using its “polling” and “ignoring” capability.

**Syntax:**

// Waiting 30 seconds for an element to be present on the page, checking  
// for its presence once every 5 seconds.  
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)  
 .withTimeout(30, *SECONDS*)  
 .pollingEvery(3, *SECONDS*)  
 .ignoring(NoSuchElementException.class);  
  
WebElement foo = wait.until(new Function<WebDriver, WebElement>() {  
 public WebElement apply(WebDriver driver) {  
 return driver.findElement(By.*id*("Pragra"));  
 }  
});

Above code will try to find the element by id “Pragra”, for 30 seconds by polling the request for every 3 seconds.

Note: Default polling in Explicit wait is 500 milliseconds

**Implicit Wait Exceptions:**

Throws a NoSuchElementException when the element is not present in the DOM

Throws a ElementNotVisibleException when element is present in the DOM, however, it is hidden and cannot be interacted with.

**Explicit Wait Exceptions (Default):**

Throws a WebDriverTimeoutException, depending on our expected condition

# Javascript Executor

# Desired Capabilities

# Page Object Modeling

# Selenium Grid