21-Feb-2020

-----------

(a) Implicit Wait:

-------------

Using this option we can pause execution for specified amount of time (i.e 30 sec)

When any webelement is not able to identify in a page during runtime

Syntax: driver.manage().timeouts().implicitlyWait(time, TimeUnit parameter);

Ex: driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

Ex-24: Write script to select "BRAHMADEVI-NLR" in "FROM" dropdown after enter "NLR" using Implicit wait option

Note: Implicit wait will work only on driver.findElement "scripts" and not on other transaction scripts

using Implicit wait option

Script:

// to Initialize browser

System.setProperty("webdriver.gecko.driver", "./Drivers//geckodriver.exe");

WebDriver driver= new FirefoxDriver();

driver.get("https://www.apsrtconline.in/oprs-web/");

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

//Implicitwait

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

//to enter "NLR" in From dropdown

driver.findElement(By.name("source")).sendKeys("NLR");

//to pause execution

//Thread.sleep(9000);

//to select "BRAHMADEVI-NLR"

driver.findElement(By.linkText("BRAHMADEVI-NLR")).click();

=======================================================================================================================

(b) Explicit Wait:

-------------

To pause execution at particular position based on presence of particular web element in DOM (i.e. to wait when webelement code is not available in HTML page

source code)

Syntax: Wait<WebDriver>obj=new WebDriverWait(driver,30);

obj.until(ExpectedConditions.presenceOfElementLocated(By.LocatorName("value"));

Ex-25: Write script to select "BRAHMADEVI-NLR" in "FROM" dropdown after enter "NLR" using Explicit wait option

using Explicit wait option

// to Initialize browser

System.setProperty("webdriver.gecko.driver", "./Drivers//geckodriver.exe");

WebDriver driver= new FirefoxDriver();

driver.get("https://www.apsrtconline.in/oprs-web/");

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

//to enter "NLR" in From dropdown

driver.findElement(By.name("source")).sendKeys("NLR");

//to pause execution using Explicit wait

Wait<WebDriver> wt= new WebDriverWait(driver, 30);

wt.until(ExpectedConditions.presenceOfElementLocated(By.linkText("BRAHMADEVI-NLR")));

//to select "BRAHMADEVI-NLR"

driver.findElement(By.linkText("BRAHMADEVI-NLR")).click();

===================================================================================================================================

Ex-26: Develop Automation test script to validate login functionality in Gmail application using valid data

using valid data

// initialize browser and open gmail

System.setProperty("webdriver.chrome.driver", "D:\\SelResource\\chromedriver.exe");

WebDriver driver= new ChromeDriver();

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

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

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

//to enter Email

driver.findElement(By.name("identifier")).sendKeys("dfsdfdfdf");

driver.findElement(By.xpath("//span[text()='Next']")).click();

Thread.sleep(3000);

//to Enter pwd

driver.findElement(By.name("password")).sendKeys("xxxxxx");

driver.findElement(By.xpath("//span[text()='Next']")).click();

Thread.sleep(9000);

//to read page title

String pgTitle=driver.getTitle();

if(pgTitle.contains("seleniumuser15")) {

System.out.println("Successful login operation");

}

else {

System.out.println("Unsuccessful login operation");

}

================================================================================================================================

" FLUENT WAIT"

-----------

Note: In WD they have also provided "Fluent Wait" option to synchronize test execution based on specific webelement availibility

--> Using Fluentwait option we can set polling time to check webelement availibility

--> \*\* Where as Fluentwait is not preferred present versions of WD

\_\_> In fluent wait you can specify and overwrite the polling interval but in implicit wait there is no way to modify time interval, in explicit wait by default the time interval will be 500 milliseconds

Syntax

Wait wait = new FluentWait(WebDriver reference). with Timeout(timeout, SECONDS). pollingEvery(timeout, SECONDS)

* Fluent wait uses two parameters only – timeout value and polling frequency

Example

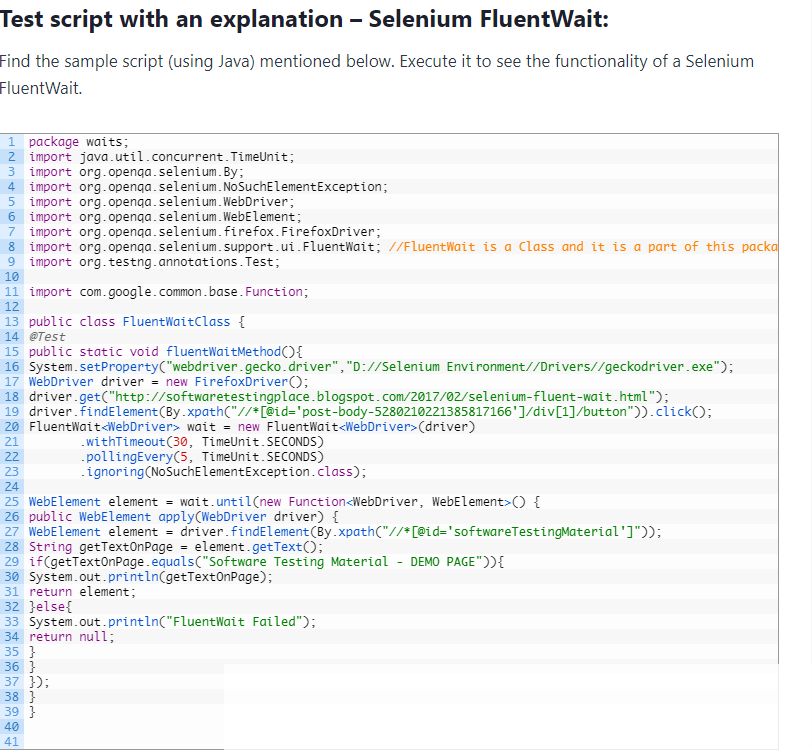
Wait wait = new FluentWait<WebDriver>(driver)

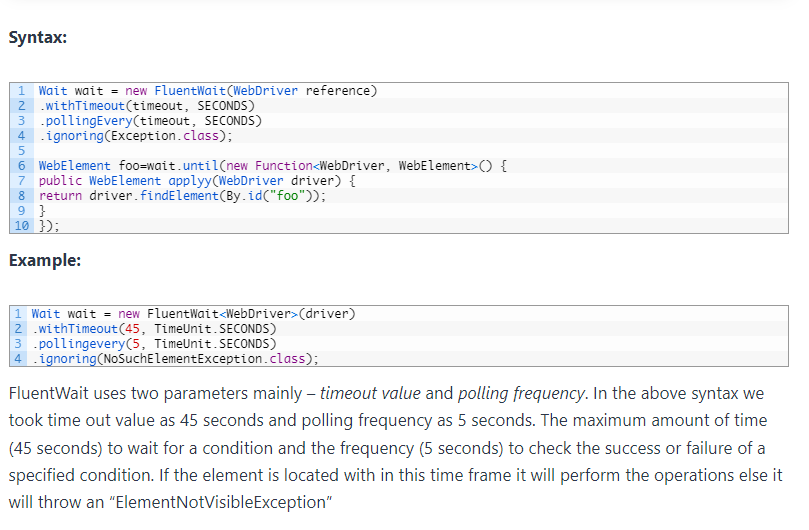
.withTimeout(45, TimeUnit. SECONDS)

.pollingevery(5, TimeUnit. SECONDS)

.ignoring(NoSuchElementException.class)

FluentWait uses two parameters mainly – timeout value and polling frequency. In the above syntax we took time out value as 45 seconds and polling frequency as 5 seconds. The maximum amount of time (45 seconds) to wait for a condition and the frequency (5 seconds) to check the success or failure of a specified condition. If the element is located with in this time frame it will perform the operations else it will throw an “ElementNotVisibleException”





Implicit wait - Implicit wait commands Selenium to wait for a certain amount of time before throwing a “No such element” exception.

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

Explicit wait - Explicit wait is used to tell the Web Driver to wait for certain conditions before throwing an "ElementNotVisibleException" exception.

WebDriverWait wait = new WebDriverWait(WebDriver Reference, TimeOut);

Fluent wait - It is used to tell the web driver to wait for a condition, as well as the frequency with which we want to check the condition before throwing an "ElementNotVisibleException" exception.

Wait wait = new FluentWait(WebDriver reference).withTimeout(timeout, SECONDS).pollingEvery(timeout, SECONDS).ignoring(Exception.class);

------------------------------------------------------------------------------------------------------------------------------

--> Due to security issue will not automate Gmail option -- instead use Mercury Tours

Ex-27: Develop ATS to validate login functionality in Mercury Tours application using valid data

We need to write the script for this

--- Using login validation

--- getTitle()

=======================================================================================================================================

Handling popups/Alerts using WebDriver

--------------------------------------

There are two types of alerts:

1. Web based alert/ popups

2. Windows based alert/popups

----------------------------------------------------------------------------------------------------------------------------------------

1. Handling web based popups

-------------------------

There are 2 types of web based popups

a. HTML popups

b. Java popups

a. Working on HTML popups:

--> which are developed in HTML

--> where as to handle this type of popups we can write script directly to perform operations on that popup window webelements

---------------------------------------------------------------------------------------------------------------------------------------------

Ex-28: Write ATS to click on "OK" in html popup at www.spicejet.com application when we select "multicity" option

--> If we don't find element use

Thread.sleep (or) Implicit wait

// initialize browser and open gmail

System.setProperty("webdriver.gecko.driver", "./Drivers\\geckodriver.exe");

WebDriver driver= new FirefoxDriver();

driver.get("https://spicejet.com/");

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

//to select "Multicity" option

driver.findElement(By.xpath("//\*[text()='Multicity']")).click();

Thread.sleep(4000);

//to click on "OK"

driver.findElement(By.linkText("Ok")).click();

-------------------------------------------------------------------------------------------------------------------------------------------

b.\*\* Handling Java popups:

--------------------

If we right click on popup window webelements then it is not showing any html code related options then it is considered as Java popup

-->\*\*\* To handle Java popups we use "Alert" class

Syntax: { to create instance object for Alert class}

Alert obj= driver.switchTo().alert();

We can use following commands to work on Alerts

(i) obj.accept();

To accept alert (i.e. to click on "OK/Yes")

(ii) obj.getText();

To read alert message

(iii) obj.dismiss();

It will dismiss the alert (i.e. to click on "Cancel")

(iv) obj.sendKeys(value);

It will enter text into prompt/editbox in popup window

-------------------------------------------------------------------------------------------------------------------------------------------

Ex-29: Write program to handle java popup in http://aponline.gov.in application

Note: If we are not able to right click on the webpage-- we can copy the url in the different/new window where there DOM is already opened (or)

Go for icon of Inspect/Inspector

---> Alert altobj= driver.switchTo().alert(); // where driver is 'page' ; altobj is a variable ; Alert is a class for the ref obj 'altobj'

Steps:

open Firefox browser

navigate to aponline login page

enter user id

click on Login

read error message from popup

close the popup

enter pwd

// initialize browser and open gmail

System.setProperty("webdriver.gecko.driver", "D:\\SelResource\\geckodriver.exe");

WebDriver driver= new FirefoxDriver();

driver.get("https://aptonline.in/citizenportal/home.aspx");

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

//To enter username

driver.findElement(By.name("txtloginusername")).sendKeys("MadhukarQAIT");

//to click on "Login"

driver.findElement(By.name("btnlogin")).click();

Thread.sleep(4000);

Alert altObj=driver.switchTo().alert();

//to read Popup msg

String eMsg=altObj.getText();

System.out.println(eMsg);

//to click on "OK" in popup

altObj.accept();

//to enter pwd

driver.findElement(By.name("txtloginpassword")).sendKeys("Livetech");

===================================================== END OF CLASS =============================================================================