Synchronization in Selenium Webdriver

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

Generally 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: Wait() and Thread.Sleep();

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:

We specify a condition along with timeout value, so that tool waits to check for the condition and then come out if nothing happens.

It is very important to set the timeout value in conditional synchronization, because the tool should proceed further instead of making the tool to wait for a particular condition to satisfy.

In Selenium we have implicit Wait and Explicit Wait conditional statements. Check here for Examples on how to use Webdriver Waits

1. Implicit Wait.

An implicit wait is to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available.

The default setting is 0. Once when we define the implicit wait, it will set for the life of the WebDriver object instance.

It is a mechanism which will be written once and applied for entire session automatically. It should be applied immediately once we initiate the Webdriver.

Implicit wait will not work all the commands/statements in the application. It will work only for "FindElement" and "FindElements" statements.

If we set implicit wait, find element will not throw an exception if the element is not found in first instance, instead it will poll for the element until the timeout and then proceeds further. We should always remember to add the below syntax immediately below the Webdriver statement.

Syntax: driver.manage.TimeOuts.implicitwait(6,Timeunit.SECONDS);

Example:
WebDriver driver = new FirefoxDriver();
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
driver.get("www.google.com");

Explicit Wait:

We need to define a wait statement for certain condition to be satisfied until the specified timeout period. If the Webdriver finds the element within the timeout period the code will get executed.

Explicit wait is mostly used when we need to Wait for a specific content/attribute change after performing any action, like when application gives AJAX call to system and get dynamic data and render on UI.

Example: Like there are drop-downs Country and State, based on the country value selected, the values in the state drop-down will change, which will take few seconds of time to get the data based on user selection.

Example:

```
/*Explicit wait for state dropdown field*/
    WebDriverWait wait = new WebDriverWait(driver, 10);
    wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("statedropdow
n")));
```

Fluent Wait:

Using FluentWait we can define the maximum amount of time to wait for a condition, as well as the frequency with which to check for the condition.

And also the user can configure to ignore specific types of exceptions such as "NoSuchElementExceptions" when searching for an element.

Syntax:

```
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
                                        //Wait for the condition
                                       .withTimeout(30, TimeUnit.SECONDS)
                                         // which to check for the condition with int
erval of 5 seconds.
                                       .pollingEvery(5, TimeUnit.SECONDS)
                                      //Which will ignore the NoSuchElementException
                                       .ignoring(NoSuchElementException.class);
```

‹ How to Install TestNG step by step How to generate PDF report using iText java API ›

Tags:

Waits

Selenium Tutorials:

Selenium Tutorials

Add new comment

Comments

helpful post

Permalink Submitted by TtG on Wed, 09/10/2014 - 21:56

helpful post

reply

Selenium Easy

Permalink Submitted by Raghu on Tue, 10/14/2014 - 03:51

Really very helpful site , Thanks for the post

reply

Helpful Link

Permalink Submitted by depak on Fri, 01/23/2015 - 12:43

Helpful for me,thank you.

reply

Many thanks

Permalink Submitted by Piyush Jhawar on Tue, 01/27/2015 - 23:38

Thanks very much

reply

very helpful

Permalink Submitted by Naveen on Wed, 03/04/2015 - 23:40

i found lot of information rather to search in google

reply

very informative and helpful.

Permalink Submitted by Saurabh Agnihotri on Sun, 03/15/2015 - 10:36

very informative and helpful.

reply

Selenium Tutorials

- Introduction to selenium
- Basic HTML concepts required for Selenium
- Locators for Selenium
- How to Run Webdriver in chrome browser?
- How to run webdriver in IE browser?
- Selenium webdriver using Eclipse
- Synchronization in Selenium Webdriver
- XPath tutorial for Selenium
- CSS selectors for Selenium with example
- Firefox Not Connected Exception in Selenium
- Handle windows popups using Selenium Webdriver
- Working with Frames in Selenium Webdriver
- How to handle javascript alerts, confirmation and prompts?
- Defining Firefox Profile preferences using Selenium Webdriver
- Why Automation Testing is required?
- Importance of Automation Testing
- Mouse Hover Actions in Selenium Webdriver
- StaleElementReferenceExceptions in Selenium Webdriver
- Difference between Webdriver get() and Webdriver navigate()
- Navigation Methods in Webdriver with Examples
- Handling Cookies in Webdriver
- How to addcookie with Selenium Webdriver
- How to delete Cookies in Selenium Webdriver
- WebDriver Methods
- Automation Test Frameworks
- Taking Screenshot using Webdriver
- Webdriver SELECT Methods to work with Dropdowns
- Webdriver Select with Multiple Attribute

- Handling keyboard events and mouse hover events using Webdriver
- Set browser width and height in Selenium Webdriver
- Selenium Automation Framework Example
- UI Map or Object Repository using Properties File
- Test Data in automation framework
- Reporting Structure in Selenium Automation Framework
- Test Configuration File / Executable File
- Uploading a file with Selenium Webdriver
- Upload file using AutoIT
- Uploading a file with sendKeys method
- Read data from Properties file using Java Selenium
- Scrolling web page with Selenium Webdriver using java
- Keyword Driven Framework Example
- Install AutoIt in Windows
- Examples with xpath and Css (ID, Name, Text and Links)
- Testing in multiple browsers using selenium and testng
- Working with Checkbox using Resuable Methods
- Working with Select examples
- File upload using Robots
- Download file using selenium webdriver
- Perform operations on new window
- Introduction to Page Object Model Framework
- Mouse hover using javaScript Executor
- Working with Multiple Checkboxes with Safecheck
- Click element using JavaScriptExecutor
- Drag and Drop using Webdriver Action Class
- WebDriver Waits Examples
- Simple Page Object Model example
- Double Click on element using Webdriver
- Right Click Context Menu example
- Working with AutoComplete Text box
- Working with Date Picker Calendar
- Working with Ajax controls using Webdriver
- Get Attribute values using Webdriver
- Get Css values using Webdriver
- Resizing a web element using movebyoffset
- Validate Downloaded file after clicking on downloaded button/ link
- How to verify entries in Exported CSV file

- Element is not clickable at point SeleniumWebdriverException
- Extract PDF text And Verify Text Present in PDF using WebDriver
- Handling Authentication Window with WebDriver (In Firefox, Chrome and IE)

Recent Post

- Handling Authentication Window with WebDriver (In Firefox, Chrome and IE)
- Selenium Webdriver Tutorials
- Locators for Selenium
- Optional annotation in TestNG
- Parameterization in TestNG using testng.xml

Automation Framework

- Framework Structure
- Object Repository
- Test Data
- Test Scripts
- Reports
- Configuration and Execution

Tags Cloud

Asserts (1) AutoComplete (1) AutoIt (2) Automation (3) Build.xml (1) Checkboxes (2) Cookies (3) Css (2) Css Examples (2) DataProvider (1) Date Picker (1) Drag and Drop (1) Exceptions (4) File Download (3) Firefox Profile (2) Framework Frameworks (5) iFrames (1) Interview Questions (1) JavascriptExecutor (3) JUnit (2) Keyword Driven (1) Listeners (3) Manual Testing (1) Maven (5) Mouse Hover (2) Page Object Model (2) Page Scroll (1) Parallel Test (3) properties file (1)

- TestNG Tutorial
- JUnit Tutorial
- JXL Tutorial
- Apache POI
- ANT Tutorial
- Maven Tutorial

Selenium Tutorials

- Basic Webdriver Tutorials
- Advanced Webdriver Tutorials
- Selenium Framework Examples
- JavascriptExecutor Examples

Interview Questions

- Testing Concepts
- Quality Standards
- General Questions
- Roles & Responsibilities
- Technology & Environment

We welcome your questions and requests