# BestAutomationPractices - Guide for best automation frame work as per SELENIUM 4

https://www.lambdatest.com/blog/selenium-best-practices-for-web-testing/

\*\*1.)Avoid Blocking using sleep calls\*\* - We should abstain from using Thread.sleep within the test cases. Loading of web elements on any website depends on many factors like load on the server, netwrok speed, device (machine) capabilities, access location, browser (some time performing tests on out-dated browsers) and any more. So keeping this in view we can never sure how much exact time will the element will be going to take. Use of sleep blocking calls blocks the test thread for pre-specified time interval. For example we are putting Thread.sleep(5000) which 5 seconds however but when we are running the test framework using some other environments it is taking 10 seconds and ultimately our test will fail with No such Element exception. If we take more time in the Thread.sleep(9000) then it can increase the time of test pack execution i.e. if the test getting executed within 6 seconds it will still keep the thread on hold for another 3 seconds. Lets say if we have 1000 test cases then extra time will be e.g. 1000\*3 seconds .

Alternatives to deal with \*\*NO SUCH ELEMENT\*\* Exception-- Selenium provides inbuilt wait methods and we should alway prefer to use these as below :

\*\*a.) Implicit Wait -\*\* If element takes less than 5 seconds to load selenium moves on to next line of code and do not hold the process for all 5 seoconds.

SYNTAX - driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

\*\*b.) Explicit Wait -\*\* Explicit wait can be provided with respect to WebElement. So it will hold the execution untill the particular element loads upto a predefined time. If elements loads earlier then the execution moves to next line of code.

SYNTAX - WebDriverWait wait = new WebDriverWait(driver,Duration.ofSeconds(10));

wait.until(ExpectedConditions.visibilityOfElementLocated(By.cssSelector(".classlocator")));

\*\*c.) Fluent Waits -\*\* Fluent wait instance allow to wait for some specific period of time along with freqeuncy duration in which it will check the condition and if it matches it moves on to next line of code without waiting for full pre-specified time.

SYNTAX - Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)

.withTimeout(Duration.ofSeconds(30)) // Predfined total time limit it gonna be wait

.pollingEvery(Duration.ofSeconds(5)) // Frequency - after every 5 seconds it will check the condition if element found it can stop at 5 or 10 or 15 secs

.ignoring(NoSuchElementException.class);

LINK to know more about WebDriver waits - https://www.selenium.dev/documentation/webdriver/waits/

\*\*2.) Always Name the Test Cases and Test Suites appropriately -\*\*