**1. Introduction to Selenium Waits**

**Why Do We Need Waits?**

* When automating web applications, elements may take time to load due to various reasons such as network latency, dynamic content, or slow server responses.
* **Waits** help synchronize your test scripts with the web page to avoid issues like **ElementNotVisibleException** or **ElementNotInteractableException**.

**Types of Waits:**

* **Implicit Wait**
* **Explicit Wait**
* **Fluent Wait**

**2. Implicit Wait**

An **implicit wait** makes WebDriver poll the DOM for a specified amount of time before throwing an exception if the element is not found.

**Syntax:**

java

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

**When to Use:**

* When you want to apply a default wait time for all elements in your script.

**Example:**

java

WebDriver driver = new ChromeDriver();

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

driver.get("https://www.example.com");

WebElement element = driver.findElement(By.id("elementID"));

**3. Explicit Wait**

An **explicit wait** is used to wait for a specific condition to occur before proceeding with the next step in the script. It's more flexible than implicit waits.

**Syntax:**

java

WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(20));

WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("elementID")));

**Common Conditions in Explicit Wait:**

* visibilityOfElementLocated
* elementToBeClickable
* alertIsPresent

**Example:**

java

WebDriver driver = new ChromeDriver();

driver.get("https://www.example.com");

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

WebElement element = wait.until(ExpectedConditions.elementToBeClickable(By.id("elementID")));

element.click();

**When to Use:**

* When you want to wait for a specific element or condition, such as waiting for a button to be clickable or an alert to be present.

**4. Fluent Wait**

A **fluent wait** allows you to define the maximum wait time and the polling frequency. It also provides options to ignore specific types of exceptions.

**Syntax:**

java

Wait<WebDriver> fluentWait = new FluentWait<>(driver)

.withTimeout(Duration.ofSeconds(30))

.pollingEvery(Duration.ofSeconds(5))

.ignoring(NoSuchElementException.class);

WebElement element = fluentWait.until(ExpectedConditions.visibilityOfElementLocated(By.id("elementID")));

**Example:**

java

WebDriver driver = new ChromeDriver();

driver.get("https://www.example.com");

Wait<WebDriver> fluentWait = new FluentWait<>(driver)

.withTimeout(Duration.ofSeconds(30))

.pollingEvery(Duration.ofSeconds(5))

.ignoring(NoSuchElementException.class);

WebElement element = fluentWait.until(ExpectedConditions.elementToBeClickable(By.id("elementID")));

element.click();

**When to Use:**

* When you need more control over polling frequency and exception handling.

**5. Comparison Between Wait Types**

| **Wait Type** | **Timeout** | **Polling Frequency** | **Exception Handling** | **Scope** |
| --- | --- | --- | --- | --- |
| **Implicit Wait** | Fixed | Default (500 ms) | Automatically handled | Applies to all elements |
| **Explicit Wait** | Fixed | Default (500 ms) | Need to specify | Applies to specific elements or conditions |
| **Fluent Wait** | Flexible | Customizable | Customizable | Applies to specific elements or conditions |

**6. Practice Exercise: Implement Waits in a Script**

**Problem Statement:**

Write a script that:

1. Navigates to a webpage with dynamic content.
2. Waits for an element to be visible before interacting with it.