Attend dans Selenium

- La plupart des éléments Web sont écrits en utilisant***Script Java*** ou ***AJAX.***
- Ils peuvent prendre un certain temps à charger sur la page.
- Dans cette situation, les chances d'obtenir '***NoSuchElementException'***sont plus.
- Pour éviter de telles exceptions, Selenium a fourni deux types d'attente -**Attente implicite**&**Attente explicite.**

1. Attente implicite :

- Cette attente est définie pour la durée de vie de l'instance WebDriver.
- Partout où nous utilisons l'instance WebDriver avant que cette attente de ligne ne soit appliquée.
- Cette attente, interroge pendant 500 millisecondes (*Ce temps dépend de la classe d'implémentation des navigateurs de webdriver. Pour firefox c'est 500 millisecondes, cela dépend du type et de la version du navigateur*).
- Si l'élément devient disponible dans les 500 premières millisecondes, l'élément sera renvoyé, sinon il attendra les 500 millisecondes suivantes. Notre script attendra le délai maximal.
- Nous pouvons spécifier le délai d'attente par défaut.
- Si l'élément n'est pas disponible dans la valeur du délai d'attente, alors '***NoSuchElementException'***sera jeté.
- Nous pouvons spécifier des délais d'attente en termes de : NanoSeconds, MicroSeconds, MilliSeconds, Seconds, Minutes, Hours et Day.
- Par exemple

```
driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
```

- Dans l'exemple ci-dessus, la valeur du délai d'attente est de 10 secondes.
- Maintenant, notre script attendra au moins 500 millisecondes et au maximum 10 secondes avant chaque WebElement que nous souhaitons trouver en utilisant '*conducteur*' exemple.
- Par exemple. -

```
classe publiqueImplicitWaitDemo {
    Vide public statiqueprincipal(chaîne[]arguments) {
        WebDriverconducteur;
        conducteur=nouveauFirefoxDriver();
        conducteur.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
        conducteur.findElement(By.xpath("Élément1")).Cliquez sur(); conducteur
        .findElement(By.xpath("Élément2")).Cliquez sur(); conducteur
        .findElement(By.xpath("Élément3")).Cliquez sur();
    }
}
```

Dans l'exemple ci-dessus, notre script attendra avant de cliquer sur les trois éléments (*Élément1, Élément2 et Élément3*).

-Le temps d'attente minimum est de 500 millisecondes et le temps d'attente maximum est de 10 secondes

2. Attente explicite :

- Cette attente est plus personnalisée que l'attente implicite. Nous pouvons
- appliquer cette attente pour un élément Web particulier. Nous pouvons même
- personnaliser le temps d'interrogation en attente explicite.
- Nous pouvons également ignorer les exceptions que nous ne voulons pas lors de la localisation d'un élément.
- Identique à l'attente implicite, nous pouvons définir un délai d'expiration pour l'élément Web.
- Nous pouvons également faire attendre notre script jusqu'à ce que certaines conditions se produisent.

Nous pouvons réaliser une attente explicite en utilisant deux classes de Selenium :***FluentWait***et***WebDriverWait.***

1. Utilisation de FluentWait :

- Implémentation de la classe FluentWait***Attendez***interface.
- Nous pouvons définir notre propre***temps libre***et***heure du scrutin***en utilisant FluentWait.
- Nous pouvons aussi***ignorer les exceptions***en attendant un WebElement particulier.
- Nous pouvons***attendre jusqu'à certaines conditions***d'un élément Web.
- Cette attente peut être***appliqué****pour****WebElement particulier***contrairement à "l'attente implicite" qui s'appliquait à toute la durée de vie de l'instance WebDriver.

Méthodes de la classe FluentWait :

1.withTimeout(longue durée, unité TimeUnit)

- Cette méthode est utilisée pour définir le temps d'attente maximal du script.
- Cette méthode prend deux paramètres :***longue durée***et***Unité de temps***en termes de millisecondes, microsecondes, secondes, heures, jours, etc.
- Par exemple

```
classe publiqueFluentWaitDemo {
    Vide public statiqueprincipal(chaîne[]arguments) {
        WebDriverconducteur;
        conducteur=nouveauFirefoxDriver(); FluentWait
        attendez=nouveauFluentWait(conducteur); attendez.
        avecTimeout(10, TimeUnit.SECONDS);
    }
}
```

- Dans l'exemple ci-dessus, l'attente est configurée pour un maximum de 10 secondes. Si l'élément n'est pas disponible dans 10 secondes, alors***NoSuchElmentExceptionNoSuchElmentException***sera jeté.

2.pollingEvery(longue durée, temps TimeUnit);

- Cette méthode est utilisée pour définir***heure du scrutin***.
- Le temps d'interrogation est la fréquence à laquelle notre script vérifiera l'élément Web. Cette
- méthode prend également deux arguments : durée et heure

Par exemple

```
classe publiqueFluentWaitDemo {
    Vide public statiqueprincipal(chaîne[]arguments) {
        WebDriverconducteur;
        conducteur=nouveauFirefoxDriver(); FluentWait
        attendez=nouveauFluentWait(conducteur); attendez.
        pollingChaque(2, TimeUnit.SECONDS);
    }
}
```

- Dans l'exemple ci-dessus, l'attente est configurée pendant 2 secondes comme temps d'interrogation.
- Ce script recherchera l'élément Web toutes les 2 secondes jusqu'au délai d'expiration maximal.

3.ignorant (exceptionType):

- Cette méthode va***ignorer spécifique***types de***des exceptions***en attendant une condition. Par exemple

```
wait.ignoring(NoSuchElementException.class);
```

-Le script ci-dessus ignorera***NoSuchElementException***lors de la recherche d'un élément Web.

4.jusqu'à(Conditionsprévues):

- Après avoir configuré, attendez en utilisant***jusqu'à()***, le script attendra jusqu'à l'une des rencontres suivantes :
  - Jusqu'à ce que la condition attendue soit rencontrée.
  - Jusqu'à l'expiration du délai
  - Jusqu'à ce que le fil en cours soit interrompu

Par exemple

```
wait.until(ExpectedConditions.alertIsPresent());
```

-Le script ci-dessus attendra que l'alerte soit présente sur la page Web.

2. Utilisation de WebDriverWait :

- C'est un***version spécialisée***de FluentWait.
- Tous ses constructeurs prennent l'instance de WebDriver comme paramètre. Il
- hérite de presque toutes les méthodes de la classe FluentWait.
- Il est donc recommandé d'utiliser la classe FluentWait lorsqu'une attente explicite est requise.
- Mais lorsque nous traitons avec l'instance WebDriver, il est recommandé d'utiliser la classe WebDriverWait. Il a
- trois constructeurs.

```
WebDriverWait (pilote WebDriver, long timeOutInSeconds)
```

```
WebDriverWait (pilote WebDriver, long timeOutInSeconds, long sleepInMillis)
```

```
WebDriverWait (pilote WebDriver, horloge, dormeur dormant, long timeOutInSeconds, long sleepTimeOut )
```

Questions d'entretien -

1. Quels sont les types d'attentes disponibles dans Selenium WebDriver ?

Dans Selenium, nous avons pu voir trois types d'attentes telles que les attentes implicites, les attentes explicites et les attentes fluides.

- **Attente implicite :**Les attentes implicites sont utilisées pour fournir un temps d'attente par défaut (disons 30 secondes) entre chaque étape/commande de test consécutive sur l'ensemble du script de test. Ainsi, l'étape de test suivante ne s'exécutera que lorsque les 30 secondes se seront écoulées après l'exécution de l'étape/commande de test précédente.
- **Attente explicite :**Les attentes explicites sont utilisées pour arrêter l'exécution jusqu'à ce qu'une condition particulière soit remplie ou que le temps maximum se soit écoulé. Contrairement aux attentes implicites, les attentes explicites ne s'appliquent qu'à une instance particulière.

2. Qu'est-ce que l'attente implicite dans Selenium WebDriver ?

Les attentes implicites indiquent au WebDriver d'attendre un certain temps avant de lever une exception. Une fois que nous avons défini l'heure, WebDriver attendra l'élément en fonction de l'heure que nous avons définie avant de lever une exception. Le réglage par défaut est 0 (zéro). Nous devons définir un temps d'attente pour que WebDriver attende le temps requis.

3. Qu'est-ce que WebDriver Wait dans Selenium WebDriver ?

WebDriverWait est appliqué sur un certain élément avec défini*état attendu*et*temps*. Cette attente n'est appliquée qu'à l'élément spécifié. Cette attente peut également lever une exception lorsqu'un élément n'est pas trouvé.

4. Écrivez un code pour attendre qu'un élément particulier soit visible sur une page. Écrivez un code pour attendre qu'une alerte apparaisse.

Nous pouvons écrire un code tel que nous spécifions le XPath de l'élément Web qui doit être visible sur la page, puis demander au WebDriver d'attendre un temps spécifié. Regardez l'exemple de code ci-dessous :

```
WebDriverWait wait=new WebDriverWait(driver, 20);
Element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath ( "<xpath>")));
```

De même, nous pouvons écrire un autre morceau de code demandant au WebDriver d'attendre qu'une erreur apparaisse comme ceci :

```
WebDriverWait wait=new WebDriverWait(driver, 20); Element
= wait.until(ExpectedConditions.alertIsPresent());
```

5. Qu'est-ce que Fluent Wait In Selenium WebDriver ?

FluentWait peut définir la durée maximale d'attente d'une condition spécifique et la fréquence à laquelle vérifier la condition avant de lancer un "*ElementNotVisibleException*" exception.

6. Différence b/w implicitement Wait et Explicit wait.

Non	Attente implicite	Attente explicite
1.	Valable pour la durée de vie de l'instance webdriver, donc peut s'appliquer à un élément Web particulier une fois utilisé, applicable à tous les éléments	
2	Réalisé en utilisant**implicitementAttendre**	Réalisé en utilisant**courammentAttendre** et **WebDriverWait**
3.	L'heure d'interrogation est spécifique au navigateur et à la version et fixe et**délais d'attente**est**personnalisable**.	**Heure du scrutin**et**délais d'attente**est**personnalisable**.
4.	cette attente ne fournit pas de fonctionnalité pour attendre qu'une certaine condition se produise	cette attente fournit une fonctionnalité pour attendre qu'une certaine condition se produise
5.	Nous**ne peut pas ignorer**n'importe quel**exception**en utilisant implicitement We**peut ignorer l'exception**en utilisant une attente explicite	
6. Sy	**taxe**: driver.manage().timeouts.implicitlyWait(XX, TimeUnit.*SECONDES*);	**Syntaxe : 1) en utilisant FluentWait** FluentWait **w**=nouveau FluentWait (pilote); **w** .withTimeout(10,TimeUnit.*SECONDES*); **Syntaxe : 1) en utilisant WebDriverWait** WebDriverWait**w**=nouveau WebDriverWait(pilote, 20); w.until(ExpectedConditions.elementToBeClickable(By.id(" .....")));
7.	jette**NoSuchElementException**après temporisation.	jette**NoSuchElementException**après timeout si non ignoré

7. Quels sont les différents types d'instructions WAIT dans Selenium WebDriver ?*Ou la question peut être formulée comme ceci:*Comment réalisez-vous la synchronisation dans WebDriver ?

Il existe essentiellement deux types d'instructions d'attente :**Attente implicite**et**Attente explicite**.

L'attente implicite indique au WebDriver d'attendre un certain temps en interrogeant le DOM. Une fois que vous avez déclaré une attente implicite, elle sera disponible pendant toute la durée de vie de l'instance WebDriver. Par défaut, la valeur sera 0. Si vous définissez une valeur par défaut plus longue, le comportement interrogera le DOM de manière périodique en fonction de l'implémentation du navigateur/ pilote. L'attente explicite demande à l'exécution d'attendre un certain temps jusqu'à ce qu'une condition soit atteinte. Certaines de ces conditions à remplir sont les suivantes :

- elementToBeClickable
- elementToBeSelected
- presenceOfElementLocated

8. Quelles sont les conditions attendues pouvant être utilisées dans l'attente explicite ?

- Voici les conditions attendues qui peuvent être utilisées dans l'attente explicite

1. alertIsPresent()
2. elementSelectionStateToBe()
3. elementToBeClickable()
4. elementToBeSelected()
5. frameToBeAvaliableAndSwitchToIt()
6. invisibilityOfTheElementLocated()
7. invisibilityOfElementWithText()
8. presenceOfAllElementsLocatedBy()
9. presenceOfElementLocated()
10. textToBePresentInElement()
11. textToBePresentInElementLocated()
12. textToBePresentInElementValue()
13. titreEst()
14. titreContient()
15. visibilitéDe()
16. visibilitéDeTousLesÉléments()
17. visibilitéOfAllElementsLocatedBy()
18. visibilitéOfElementLocated()

9. Pouvez-vous écrire la syntaxe de Fluent Wait ?

```
FluentWait w=nouveau FluentWait (pilote); w
.withTimeout(10,TimeUnit.SECONDES);
```


10. Pouvez-vous écrire la syntaxe de Explicit Wait ?

```
WebDriverWait w=nouveau WebDriverWait(pilote, 20);
w.until(ExpectedConditions.elementToBeClickable(By.id("----------")));
```

11. Comment accélérer le temps d'exécution du script de test en utilisant les attentes ?

En utilisant l'attente explicite, nous pouvons attendre une période de temps spécifiée ou jusqu'à ce que la condition soit remplie.

Si la condition est remplie avant la période de temps spécifiée, le script n'a pas à attendre la période de temps spécifiée complète. De cette façon, la vitesse d'exécution s'améliorera en utilisant une attente explicite.