



BATCH : Batch 44
LESSON : SELENIUM 01
DATE : 15.01.2022
SUBJECT : Selenium giriş

-  techproeducation
-  techproeducation
-  techproeducation
-  techproeducation
-  techproedu



Selenium giriş

SELENIUM GIRIS
OTOMASYON TEST NEDİR?
SELENIUM PROJESİ OLUSTURMA



SELENIUM NEDİR

- Selenium, web uygulamalarını farklı tarayıcılarda ve platformlarda test etmek için ücretsiz (açık kaynaklı) bir araçtır.
- Selenium yalnızca web tabanlı uygulamaları otomasyon yapmaya odaklanır. Mobil ve Windows testi yapmak için eklentiler selenium'a eklenebilir.
- Selenium jar dosyaları ile kurulabilir. Kurulum sırasında jar dosyalarını gördünüz.

[Documentation](#)

The Selenium Browser Automation Project

Selenium is an umbrella project for a range of tools and libraries that enable and support the automation of web browsers.

About Selenium

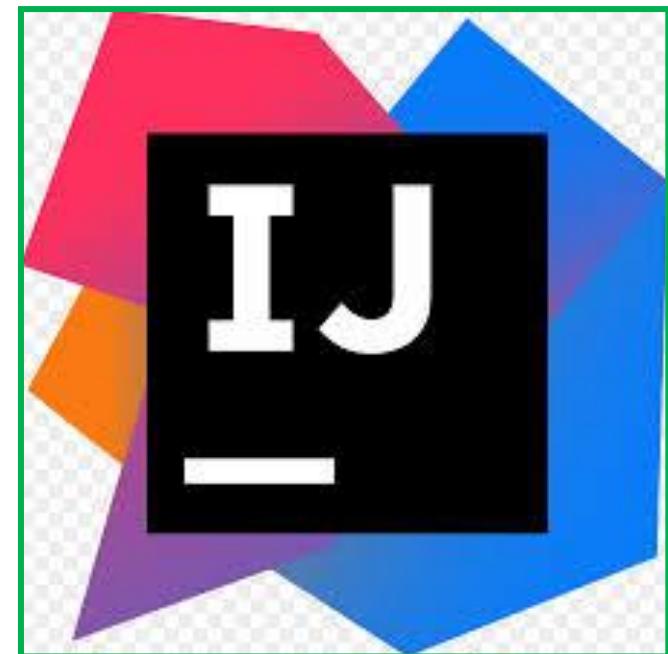
Selenium is a suite of tools for automating web browsers.

- Selenium, otomasyon yapmak için kendi sınıflarına ve yöntemlerine sahip bir kütüphanedir..
- 2021'de piyasaya sürülen Selenium 4'ü öğreneceğiz.
- Selenium'u çeşitli programlama dilleri ile yazabilirsiniz java, Python, ruby, .Net vb.



IntelliJ IDEA NEDIR

- IntelliJ IDEA 2000 yılında kurulmuş olan JetBrains isimli yazılım firmasına ait olan, popüler bir Java editörüdür.. (ide)
- IntelliJ IDEA'nın her yönü, geliştirici üretkenliğini en üst düzeye çıkarmak için tasarlanmıştır.
- Akıllı kodlama yardımcı ve ergonomik tasarım ile, kod yazınızı yalnızca verimli değil, aynı zamanda keyifli hale getirir.
- Banyak framework ve plugin ile çalışma imkanı verir
- Akıllı tamamlama özelliği ile kod yazınızı oldukça kolaylaştırır
- IDE ihtiyaçlarınızı tahmin eder ve sıkıcı ve tekrarlayan geliştirme görevlerini otomatikleştirir, böylece büyük resme odaklanabilirsiniz.





SOFTWARE TESTING NEDİR ?

EXPECTED RESULT (beklenen sonucun), **ACTUAL RESULT** (gerçek sonuca) esit olup olmadigini kontrol etme işlemidir.

- Eger Expected result = Actual result, ise status PASS (test basarili)
- Eger Expected result !=Actual result, ise status FAIL (test basarisiz)

Sonuç olarak, olması gereken şeylerin olmadığını veya olmaması gereken şeylerin olduğunu kontrol etmek ve ortaya çıkartmak yazılım testinin amacı olmalıdır.



- Her User Story icin Positive ve Negative Test(ler) yapılmalıdır
- Test, musteri/isletme ihtiyaçlarını karşılamak için yapılır.
- Bir uygulamayı test etmek için onceden belirlenmis user storyler (kullanıcı hikayeleri) ve tanımlanmış acceptance criterias (kabul kriterleri) dikkate alınır.

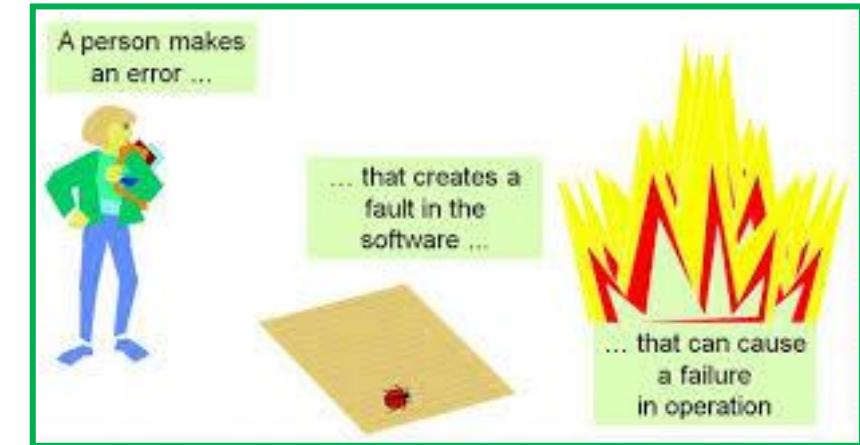


SOFTWARE TESTING NEDEN ONEMLIDIR ?

- İnsanlar hata yaparlar, bu hatalar kodda, yazılımda, sistemde ya da dokümanda defect (Kusur) oluşturur.
- Defect olan kod çalıştırıldığında sistem beklenen fonksiyonları gerçekleştiremez ve başarısız olur.

Bu sebeplerden dolayı;

- Müşteriye sunulmadan önce ürün kalitesinden emin olmak,
- Yeniden çalışma (düzeltme) ve geliştirme masraflarını azaltmak,
- Geliştirme işleminin erken aşamalarında yanlışları saptayarak ileri aşamalara yayılmasını önlemek, böylece zaman ve maliyetten tasarruf sağlamak amacıyla ürün müşteriye sunulmadan önce test edilmesi gerekmektedir.



KISACA : Testing saves money and lives .

Testing para kayiplarini onler ve hayat kurtarir. (Ucak kazalari vs..)



MANUAL(FUNCTIONAL) TESTING NEDİR?

- Manuel test, uygulamayı herhangi bir otomasyon aracı olmadan manuel olarak test etmektir.
- Manuel test kullanıcıları dokümantasyon için sınırlı teknoloji (Excel vb.) kullanır, ancak otomasyon araçları veya dili kullanmazlar.
- Manuel testte insan hatası olabilir.
- Tüm Otomasyon Tester'lar, herhangi bir otomasyon yapmadan önce uygulamayı anlamak için mutlaka manuel test yapmalıdır.
- İyi bir otomasyon tester aynı zamanda iyi bir manuel testerdir.





TEST OTOMATION NEDİR ?



- Bir sistemi bir otomasyon aracı (tool) yardımıyla test etmeye 'Test Otomasyonu' denir.
- Otomasyon test yazılımı test verilerini Test Edilen Sistem'e girebilir, beklenen ve gerçek sonuçları karşılaştırabilir ve ayrıntılı test raporları oluşturabilir.
- Bir test otomasyon tool'u kullanarak, calistirilan test paketini kaydetmek ve gerekiğinde yeniden calistirmak mümkündür. Test paketi otomatik hale getirildikten sonra hiçbir insan müdahalesi gerekmesizin programlandigi zamanda calisabilir.
- Giderek daha popüler hale gelmektedir.



MANUAL TESTING vs AUTOMATION TESTING

Asagida yazili metin sizce neyi ifade etmektedir ?

- A- Test Case
- B- Manuel tester icin test adimlari
- C- Otomasyon ile test yapan kodlar

Feature:US1011 kullanici dogru bilgilerle sayfaya girebilmeli

@CH1

Scenario: TC17 positive login testi

```
When kullanici "CHQAUrl" sayfasina gider
Then CH login linkine tiklar
And CH username kutusuna "manager" yazar
And CH password kutusuna "Manager1!" yazar
And CH login butonuna basar
Then CH basarili giris yapildigini test eder
```



MANUAL TESTING vs AUTOMATION TESTING

MANUAL TESTING

VS

AUTOMATION TESTING

EXECUTION TIME



PEOPLE



INFRESTRUCTURE



TOOLS



TURNAROUND TIME

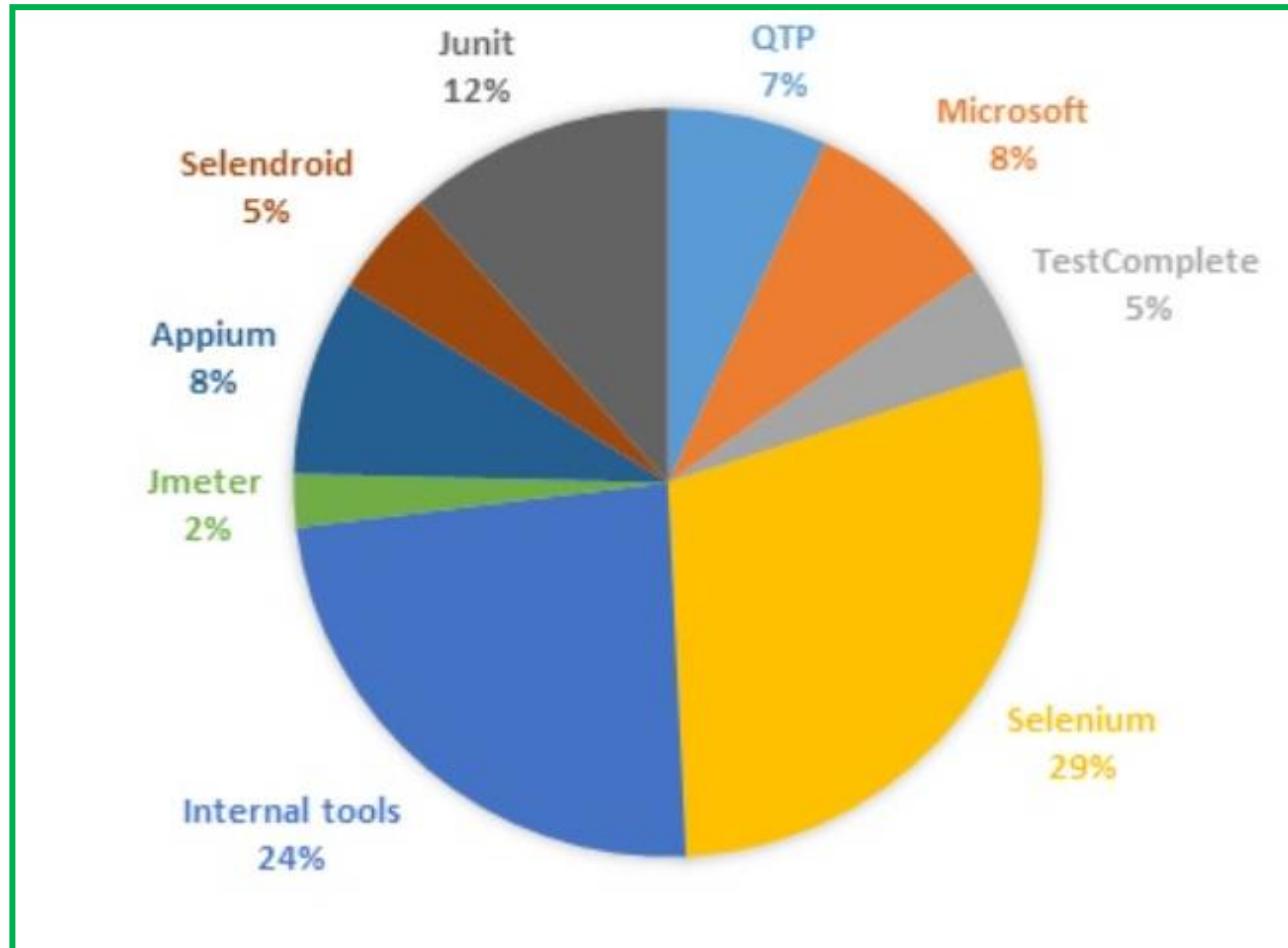


TRAINING





EN COK KULLANILAN TEST TOOL'LARI





Basarılı Bir Otomasyon Testi için Yapılması Gerekenler

- Doğru otomasyon araçlarını(tools) seçin. Genellikle şirketlerin kullandıkları tool'lari vardır, bazi şirketler de sizin isteginize gore tool secebilirler.
- Uygulamanız hakkında iyi bilgi sahibi olun.
- Test senaryolarınızı (test cases) kısa ve bağımsız tutun.
- Test otomasyonlarınızı önem derecesine göre sıralayın.
- Otomasyondan önce test verilerini hazırlayın(id,url,environment)
- Gerekirse, test caselerinizi yönetin(manage) ve bakımını yapın(maintain).Yeni test caselerinizin eski otomasyon test scriptlerini bozmadiginden emin olun.
- Testlerinizde hata olabilecegini goz onunde bulundurun. Bir BUG'i rapor etmeden once mutlaka testinizi gozden gecirin
- Her zaman test ekibinizle, özellikle team liderleri ile iyi iletisim kurun.





SELENIUM NEDİR ?

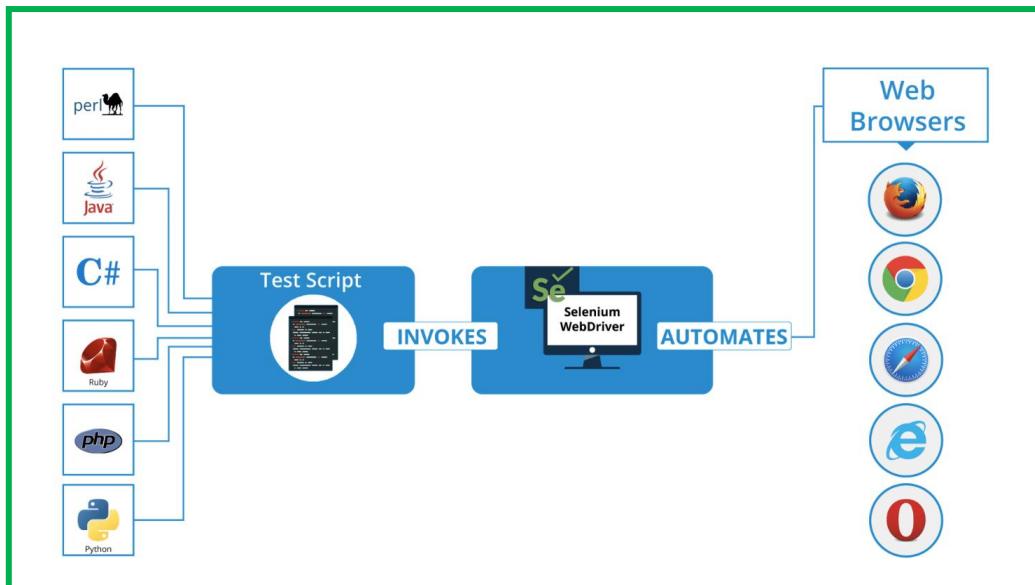


Selenium'un dört bileşeni vardır;

- Selenium Integrated Development Environment (IDE)
(Selenyum Entegre Geliştirme Ortamı (IDE))
- Selenium Remote Control (RC)(Selenyum Uzaktan Kumanda (RC))
- WebDriver (Biz Selenium WebDriver kullanacağız)
- Selenium Grid (paralel test için kullanılıyor)



SELENIUM NASIL CALISIR?



- 1) Driver selenium'dan komutları alır.
- 2) Bu komutları tarayıcısının API'nda dönüştürür
- 3) Browser'lardan donen sonuçları alır ve Selenium'a geri gönderir



SELENIUM'UN AVANTAJLARI NELERDIR?

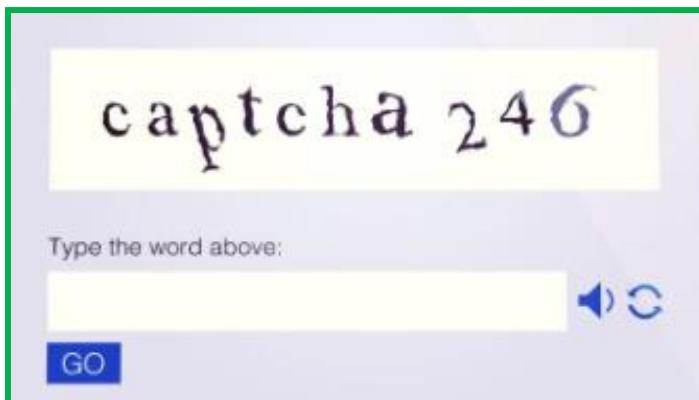
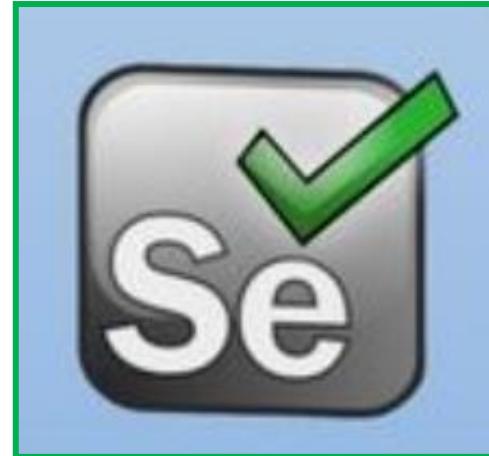


- 1) Ücretsiz ve acik kaynaklidir.
(Open source)
- 2) Bir çok programlama dilini destekler
(Java, Python, PHP, C#, Ruby vs.)
- 3) Çoklu işletim sistemleriyle çalışır.
Multiple operating systems
(Windows, MacOS, Linux)
- 4) Birden çok tarayıcı ile çalışır.
Multiple browsers
(IE, Safari, Chrome, Firefox vs.)



SELENIUM'UN DEZAVANTAJLARI NELERDIR?

- Programlama bilgisi gerektirir
(Biz Java biliyoruz)
- Yalnızca web tabanlı uygulamaları test eder
- Profesyonel desteği sahip değil
(Ama geniş bir kullanıcı kitlesi var)



SELENIUM'UN YAPAMADIKLARI

- performans testi
- handle captcha
(diğer tüm otomasyon araçları gibi)



TEST AUTOMATION FRAMEWORK NEDIR?

- Test Otomasyon Framework (TAF), yazılım projelerindeki Test Case'lerin oluşturulma ve gözden geçirme süresini azaltmak ve QA ekibimizin isini kolaylaştırmak için tasarlanmış yapılardır.

(İşinizi rahat yapabilmeniz için hazırlanmış çalışma ortamınızdır)

- Kullanılan Tool'lar

Java,
Eclipse/IntelliJ,
Selenium,
JUnit/
TestNG,
Cucumber

- Test Yapılan Layer'lar
UI , API , DataBase

The screenshot shows the IntelliJ IDEA interface with a Java project named "mymavenproject". The project structure is displayed in the left panel, showing a package named "com.techproed" containing several test classes like "CheckboxAndRadioBut", "DropDownAmazon", etc. The code editor on the right contains the following Java code:

```
package com.techproed;
public class TestNGAnnotationsExample { }
```



SELENIUM ve CHROME DRIVER KURULUMU

- 1) <https://www.selenium.dev/downloads/> adresine gidelim
- 2) Selenium Client & WebDriver Language Bindings altında Java driver'ini download edin
- 3) Browsers altında Chrome documentation linkini tiklayalim

Chrome'un kendi sayfasina gidip Current stable release'i tiklayip size uygun olani download edin
***** buradaki surum ile bilgisayarinizdaki Chrome surumunun ayni oldugundan emin olun**
- 4) src altında resources director'si olusturun
- 5) Bu klasor altında drivers ve libraries klasorleri olusturun
- 6) Indirdigimiz chromedriver'i drivers klasorune, selenium-java dosyasini ise libraries klasorune cikartalim
- 7) intelliJ 'de yeni project / package / class olusturalim ve class icinde main method olusturalim
- 8) File/Project Structure/Modules/Dependencies kismindan jar dosyalarini yukleyelim



WebDriver Olusturulmasi

1. Yeni bir class oluşturun: class name ⇒ C01_Get
2. main method oluşturun
3. Java'dan System.setProperty("", "") method'unu kullanarak webdriver'in turunu ve path'ini belirleyelim.

```
System.setProperty("webdriver.chrome.driver", "src/driver/chromedriver"); /MAC
```

```
System.setProperty("webdriver.chrome.driver", "src/driver/chromedriver.exe"); \\WINDOWS
```

4. Chrome driver oluşturun

```
WebDriver driver = new ChromeDriver();
```

5. Olusturdugumuz driver objesi ile WebDriver class'indan static method'lari kullanin



BATCH : Batch 44
LESSON : SELENIUM 02
DATE : 17.01.2022
SUBJECT : driver method'lari

- techproeducation
- techproeducation
- techproeducation
- techproeducation
- techproedu



driver method'lari

`driver.get();` Method'lari

`driver.navigate();` Method'lari

`driver.manage().window();` Method'lari



Onceki Dersten Aklımızda Kalanlar

1. Selenium : Web sitelerini otomasyon ile test edebilmemize imkan veren bir suit of tool'dur.
2. Artı yanları
 - ücretsiz , açık kaynaktır
 - birçok kodlama dili ile çalışır (biz java'yi selenium için öğrendik)
 - geniş bir kullanıcı kitlesi var, yaşadığımız sorunlara hızlı cevap bulabileceğimiz sadık bir topluluk var
 - suan piyasada en çok kullanılan otomasyon tool'udur
- 3- Eksi yanları
 - sadece web uygulamaları için kullanılabilir, mobil uygulamalar için Appium kullanılarak test yapılabilir.
 - captcha yi asamaz
- 4- WebDriver : bir interface'dir, dolayısıyla obje üretmemiz. Browser'ların driver oluşturabilmesi için bir kalıp gibi çalışır. Biz hangi browser'ı kullanmak istersek driver oluştururken o browser'a ait constructor'ı kullanırız.
- 5- driver, insan olarak yaptığımız veya gordugumuz tüm işlemleri otomasyon yapmamıza yardımcı olur.



driver.get(); Method'lari

`driver.get(String Url);` String olarak girilen Url'e gider

`driver.getTitle();` Icinde olunan sayfanin basligini String olarak getirir

`driver.getCurrentUrl();` Icinde olunan sayfanin Url'ini String olarak getirir

`driver.getPageSource();` Icinde olunan sayfanin kaynak kodlarini String olarak getirir

`driver.getWindowHandle();` Icinde olunan sayfa ve/veya tab'larin
`driver.getWindowHandles();` handle degerlerini getirir



driver.get(); Method'lari

Class Work

1. Yeni bir package olusturalim : day01
2. Yeni bir class olusturalim : C03_GetMethods
3. Amazon sayfasina gidelim. <https://www.amazon.com/>
4. Sayfa basligini(title) yazdirin
5. Sayfa basliginin "Amazon" icerdigini test edin
6. Sayfa adresini(url) yazdirin
7. Sayfa url'inin "amazon" icerdigini test edin.
8. Sayfa handle degerini yazdirin
9. Sayfa HTML kodlarinda "alisveris" kelimesi gectigini test edin
10. Sayfayı kapatın.



driver.navigate(); Method'lari

driver.navigate().to(String Url); String olarak girilen Url'e gider

driver.navigate().back (); Icinde olunan sayfadan, geldigi onceki sayfaya dondurur

driver.navigate().forward (); Back ile donulen bir sayfadan tekrar ileri gider

driver.navigate().refresh (); Icinde olunan sayfayı yeniler.



Selenium Navigation Methods

Class Work

1. Yeni bir Class olusturalim.C05_NavigationMethods
2. Youtube ana sayfasina gidelim . <https://www.youtube.com/>
3. Amazon soyfasina gidelim. <https://www.amazon.com/>
4. Tekrar YouTube'sayfasina donelim
5. Yeniden Amazon sayfasina gidelim
6. Sayfayı Refresh(yenile) yapalim
7. Sayfayı kapatalim / Tüm sayfaları kapatalim



driver.manage().window(); Method'lari

driver.manage().window().maximize(); Browser'i maximize yapar

driver.manage().window().minimize(); Browser'i minimize yapar

driver.manage().window().fullscreen(); Browser'i full screen yapar

driver.manage().window().getPosition(); Browser'in koordinatlarini verir

driver.manage().window().getSize(); Browser'in olculerini verir



Selenium Navigation Methods

Class Work

1. Yeni bir Class olusturalim.C06_ManageWindow
2. Amazon soyfasina gidelim. <https://www.amazon.com/>
3. Sayfanin konumunu ve boyutlarini yazdirin
4. Sayfayı simge durumuna getirin
5. simge durumunda 3 saniye bekleyip sayfayı maximize yapın
6. Sayfanin konumunu ve boyutlarini maximize durumunda yazdirin
7. Sayfayı fullscreen yapın
8. Sayfanin konumunu ve boyutlarini fullscreen durumunda yazdirin
9. Sayfayı kapatın



driver.manage(). Method'lari

```
driver.manage().window().setPosition(new Point ( x:80 , y:0 ) );
```

Browser'i istenen koordinata tasir

```
driver.manage().window().setSize(new Dimension( 800 , 600 ));
```

Browser'i istenen olculere getirir.

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

Istenen sayfa acilincaya kadar maximum bekleme suresini belirtir

```
driver.close(); sadece calisilan browser'i kapatir
```

```
driver.quit(); acik olan tum browser'lari kapatir
```



Selenium Navigation Methods

Class Work

1. Yeni bir Class olusturalim.C07_ManageWindowSet
2. Amazon soyfasina gidelim. <https://www.amazon.com/>
3. Sayfanin konumunu ve boyutlarini yazdirin
4. Sayfanin konumunu ve boyutunu istediginiz sekilde ayarlayın
5. Sayfanin sizin istediginiz konum ve boyuta geldigini test edin
8. Sayfayı kapatın



Homework

- 1.Yeni bir class olusturalim (Homework)
- 2.ChromeDriver kullanarak, facebook sayfasina gidin ve sayfa basliginin (title) "facebook" oldugunu dogrulayin (verify), degilse dogru basligi yazdirin.
- 3.Sayfa URL'inin "facebook" kelimesi icerdigini dogrulayin, icermiyorsa "actual" URL'i yazdirin.
- 4.<https://www.walmart.com/> sayfasina gidin.
5. Sayfa basliginin "Walmart.com" icerdigini dogrulayin.
6. Tekrar "facebook" sayfasina donun
7. Sayfayı yenileyin
8. Sayfayı tam sayfa (maximize) yapın
9. Browser'i kapatın



BATCH : Batch 44

LESSON : SELENIUM 03

DATE : 18.01.2022

SUBJECT : WEBELEMENTS
LOCATORS
(YER BULUCU-KONUM BELİRLEYICI)

- techproeducation
- techproeducation
- techproeducation
- techproeducation
- techproedu



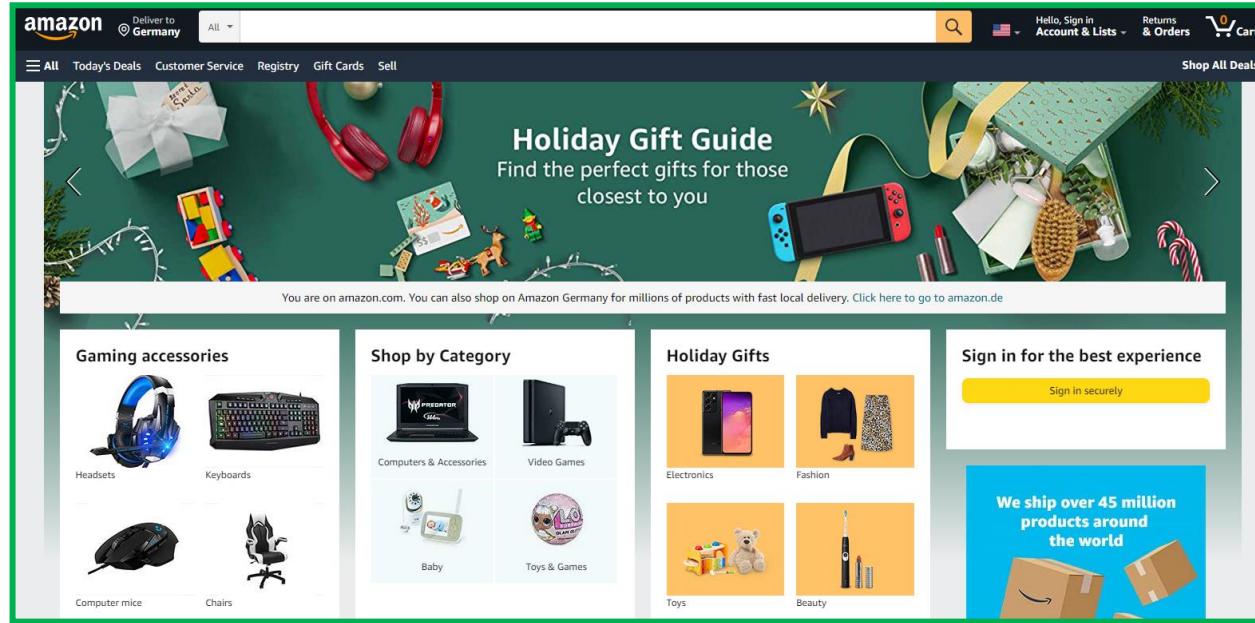
Tekrar Testi

1. Yeni bir class olusturun (TekrarTesti)
2. Youtube web sayfasına gidin ve sayfa başlığının "youtube" olup olmadığını doğrulayın (verify), eğer değilse doğru başlığı(Actual Title) konsolda yazdırın.
3. Sayfa URL'sinin "youtube" içerip içermediğini (contains) doğrulayın, içermiyorsa doğru URL'yi yazdırın.
4. Daha sonra Amazon sayfasına gidin <https://www.amazon.com/>
5. Youtube sayfasına geri donun
6. Sayfayı yenileyin
7. Amazon sayfasına donun
8. Sayfayı tamsayfa yapın
9. Ardından sayfa başlığının "Amazon" içerip içermediğini (contains) doğrulayın, Yoksa doğru başlığı(Actual Title) yazdırın.
- 10.Sayfa URL'sinin <https://www.amazon.com/> olup olmadığını doğrulayın, degilse doğru URL'yi yazdırın
- 11.Sayfayı kapatın



WEBELEMENTS

- Web sayfasında kullanılan etkilesimli olan veya olmayan herseye webelement denir



- Button,
- Search box(arama kutusu),
- Text box(metin kutusu),
- Headers(başlıklar),
- Tables(tablolar) vb...

- Farklı türde WebElement tag'ları(etiketleri) vardır.

<html>, <body>, <form>, <label>, <input>, <a> vb.

- Otomasyon için unique(tek) web öğelerini(element) tanımlamak üzere HTML kodunu inceleyeceğiz(inspect).
- Web elementleri birlikte kullanıcı arayüzünde (UI) bir web sayfası oluştururlar.



WEBELEMENTS

```
▼<tbody>
  ►<tr>...
  ▼<tr> == $0
    ▼<td>
      <input type="email" class="inputtext login_form_input_box" name="email" id="email" data-testid=
        "royal_email">
    </td>
    ▼<td>
      <input type="password" class="inputtext login_form_input_box" name="pass" id="pass" data-testid=
        "royal_pass">
    </td>
    ▼<td>
      ►<label class="login_form_login_button uiButton uiButtonConfirm" id="loginbutton" for="u_0_b">...
    </label>
    </td>
  </tr>
  ►<tr>...
</tbody>
```



Web Sayfalarını İnceleme (Inspect)

The screenshot shows a web browser window displaying the Amazon homepage. A right-click context menu is open over a search bar element, with the 'Inspect' option highlighted by a red oval. The developer tools panel at the bottom left shows the DOM tree for the page, with the search bar element selected.

```
<div class="nav-search">
  <div id="nav-bar-left"></div>
  <form accept-charset="utf-8" action="/s/ref=nb_sb_noss" class="nav-searchbar" method="GET" name="site-search" role="search">
    <div class="nav-left"></div>
    <div class="nav-right"></div>
  </div>
```

Elements Console Sources Network Performance Memory Application

Emoji & Symbols

- Undo
- Redo
- Cut
- Copy
- Paste
- Paste and Match Style
- Select All
- Language Settings
- Writing Direction
- Inspect**
- Speech

Shop deals for Alexa on the go
buds

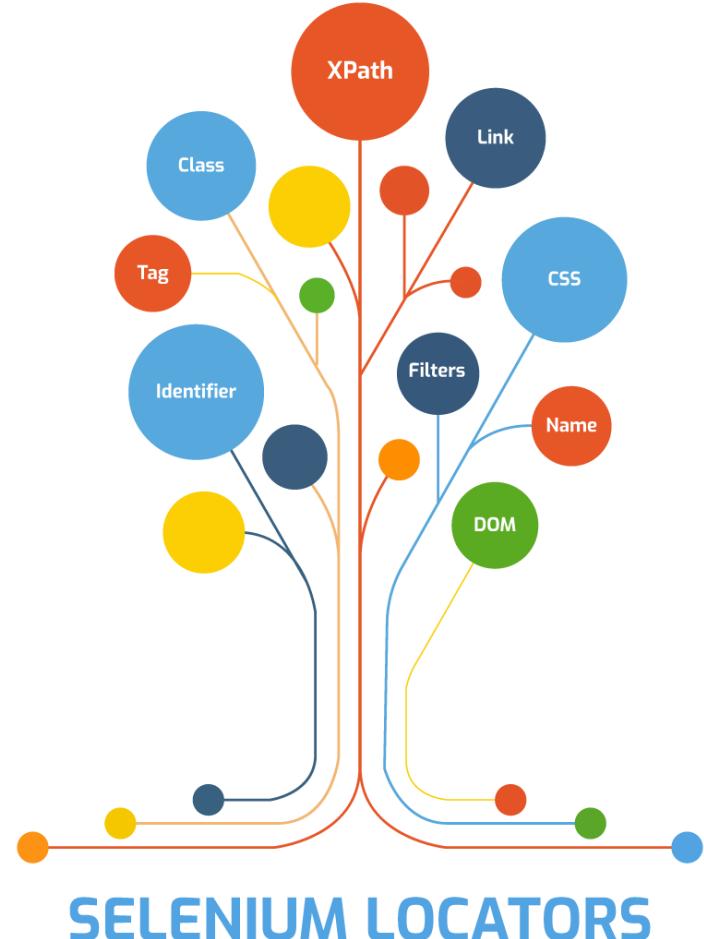
Search

field-keywords



LOCATORS (YER BULUCU-KONUM BELIRLEYICI)

- ❖ Selenium LOCATORS, web sayfasındaki web öğelerini tanımlamak için kullanılır.
- ❖ Selenium'da; metin kutuları, onay kutuları, linkler, radyo butonları, liste kutuları ve **diğer tüm web öğeler** üzerinde eylemler gerçekleştirmek için LOCATORS'a ihtiyacımız vardır.
- ❖ Konum belirleyiciler bize nesneleri tanımlamada yardımcı olur.
- ❖ Web Elementlerine ulaşmak için tag veya bazı attribute'ler kullanılır, bunlarla ulaşamayan webelementleri için özel olarak tanımlanan **Xpath** ve css locator'lari kullanılır.





LOCATORS (YER BULUCU-KONUM BELIRLEYICI)

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords"
    autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute"
    dir="auto" tabindex="0" aria-label="Search">
```

Bir web elementini tanımlamak için 8 tane selenium locator vardır.

1. id
2. name
3. className
4. tagName
5. linkText
6. partialLinkText
7. **xpath** => xpath yazmanın birden fazla yolu vardır
8. **cssSelector** => css yazmanın birden fazla yolu vardır



LOCATORS BULMAK ICIN KULLANILAN METHOD'LAR

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords"
    autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute"
    dir="auto" tabindex="0" aria-label="Search">
```

- 1) Tanimladigimiz web elementin yerini driver'in bulabilmesi icin findElement (Locator) method'unu kullaniriz.

```
driver.findElement(Locator);
```

- 2) findElement (Locator) method'unun icine parameter olarak yazacagimiz Locator'in 8 locator'dan hangisi oldugunu belirtmek icin de By.LocatorTuru("LocateBilgisi"); kullanilir

```
driver.findElement(By.id("twotabsearchtextbox"));
```

- 3) Locate ettigimiz Web Elementini kullanabilmek icin bir variable'a atama yapariz

```
WebElement aramaKutusu = driver.findElement(By.id("twotabsearchtextbox));
```



LOCATORS

1) By.id();

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords"
       autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute"
       dir="auto" tabindex="0" aria-label="Search">
```

```
WebElement aramaKutusu = driver.findElement(By.id ("twotabsearchtextbox"));
```

- Web ögesini tanımlamanın en popüler yolu id kullanmaktadır.
- id en güvenli ve en hızlı locator seçeneği olarak kabul edilir ve her zaman birden çok locator arasında ilk öncelik olmalıdır.
- Eğer yanlış id locate edilirse; **NoSuchElementException** hatası oluşur.

*** **NoSuchElementException** gordugumuzde hata veren satirdaki locator gozden gecirilmelidir



LOCATORS

2) By.name();

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords"  
autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute"  
dir="auto" tabindex="0" aria-label="Search">
```

```
WebElement passwordTextBox =driver.findElement(By.name("field-keywords"));
```

- ❖ Name ve value unique ise bu metodu da kullanabilirsiniz.



LOCATORS

3) By.className();

```
<input class="form-control" placeholder="Password" data-test="password"
type="password" name="session[password]" id="session_password">
```

```
WebElement passwordTextBox =driver.findElement(By.className("form-control"));
```

- ❖ Class attribute'u olduğunda kullanılabilir.
- ❖ Class ve value unique ise, bu metodu da kullanabilirsiniz, ancak genelde class attribute aynı islevi yapan bir grup Web Elementi için kullanılır
- ❖ Class attribute'nun değeri boşluk içeriyorsa genelde By.className() ile yapılan locator'lar başarılı çalışmaz



LOCATORS

4) By.linkText();

```
<a class="nav-item nav-link" data-test="addresses" href="/addresses">Addresses</a>
```

```
WebElement passwordTextBox = driver.findElement(By.linkText("Addresses"));
```

- ❖ Bu yalnızca HTML bağlantılarını(link) tanımlamak için kullanılabilir.
- ❖ HTML link elementleri, bir web sayfasında bağlantı etiketi(tag) kısaltması olan <a> etiketi(tag) kullanılarak temsil edilir.
- ❖ Kullanıcı arayüzündeki(UI) hyperlinkleri kolayca tanıyalabilir ve sonra bu yöntemi kullanabilirsiniz
- ❖ Büyük / küçük harfe duyarlıdır (case sensitive) ve bağlantı(link) metniyle eşleşmelidir



LOCATORS

5) By.partialLinkText();

```
<a class="nav-item nav-link" data-test="addresses" href="/addresses">Addresses</a>
```

```
WebElement passwordTextBox = driver.findElement(By.partialLinkText("dresses"));
```

- ❖ linkText () yöntemine benzer.
- ❖ Tek fark, tam metin vermek zorunda kalmamanızdır.
- ❖ Metnin yalnızca belirli bir bölümünü verebilirsiniz.
- ❖ Metnin tamamını verdığınızda de kabul eder.



LOCATORS

6) By.tagName();

```
<input class="form-control" placeholder="Password" data-test="password"
type="password" name="session[password]" id="session_password">
```

```
WebElement passwordTextBox =driver.findElement(By.tagName("input"));
```

- ❖ Bu, diğer konum belirleyicilerden biraz farklıdır.
- ❖ <div>, <a>, <input>, ... gibi belirli bir etiketi ilettiğinizde, birden fazla aynı ad etiketine sahip olabileceğiniz için birden çok öğeyi döndürür.
Çoğunlukla öğelerin bir listesini almak için kullanılır. Bu nedenle findElements() yöntemiyle kullanılması önerilir.
- Örneğin, kullandığımız bir sayfadaki tüm linkleri döndürmek için **By.tagName("a")** kullanılabilir



findElement() Method

```
WebElement elementName=driver.findElement(By.LocatorStrategy("LocatorValue"));
```

- ❖ Driver'in bir elementi bulması için findElement() yöntemini kullanırız.
- ❖ Bu, tek bir web elementini döndürür. Aynı locator ile ulaşılabilen birden fazla web element varsa ilkini dondurur
- ❖ Driver elementi bulamazsa, runtime exception verir : **NoSuchElementException**.
- ❖ **NoSuchElementException**'ı gördüğünüzde, locatorı tekrar kontrol etmelisiniz.



findElements() Method

```
List<WebElement> elementName=driver.findElements(By.LocatorStrategy("LocatorValue"));
```

- ❖ Locator degerine uygun Web elementlerinin listesini döndürür
- ❖ Locator stratejisiyle eşleşen web elementi yoksa boş bir liste döndürür.
- ❖ **NoSuchElementException** hatası vermez.
- ❖ Listedeki her Web elementi, 0'dan başlayan bir indeks alır.



findElements() ile findElement() arasindaki farklar

	findElement()	findElements()
websayfasinda birden fazla Web Element Locator ile uyuşursa	Ilk elemani dondurur	Tum elemanlari ondurus
websayfasinda hicbir Web Element Locator ile uyuşmazsa	NoSuchElementException fırlatir	Exception fırlatmaz, bos bir liste dondurur
Return Type	WebElement	List<WebElement>
Elemana erisim	Direk ulasilabilir	Liste'den index veya iterator ile ulasilabilir



WebElement Method'lari

Bir WebElement üzerinde eylemler gerçekleştirmek otomasyon tester'ları için çok önemlidir.

`webElement.click();` Web Element'e click yapar

`webElement.sendKeys("Metin");` Parametre olarak yazılan metni
Web Elemente gönderir

`webElement.submit();` Web element ile işlem yaparken
Enter tusuna basma görevi yapar

`webElement.sendKeys("Metin" + Keys.ENTER);` İstedigimiz metni yollayıp, sonra ENTER'a basar.



WebElement Method'lari

Class Work: Login Test

1. Bir class oluşturun: LoginTest
2. Main method oluşturun ve aşağıdaki görevi tamamlayın.
 - a. <http://a.testaddressbook.com> adresine gidiniz.
 - b. Sign in butonuna basın
 - c. email textbox,password textbox, and signin button elementlerini locate ediniz..
 - d. Kullanıcı adını ve şifreyi aşağıya girin ve oturum aç (sign in)buttonunu tıklayın:
 - i. Username : **testtechproed@gmail.com**
 - ii. Password : **Test1234!**
 - e. Expected user id nin **testtechproed@gmail.com** olduğunu doğrulayın(verify).
 - f. “Addresses” ve “Sign Out” textlerinin görüntülendiğini(displayed) doğrulayın(verify).
3. Sayfada kaç tane link olduğunu bulun.



WebElement Get Method'lari

`webElement.getText();` Web Element üzerindeki yazıyı getirir

`webElement.getAttribute("Att.ismi");` İsmi girilen attribute'un
değerini getirir

`webElement.getTagName();` Web elementin tag ismini
getirir

Not : Web element ile ilgili bu method'ların dışında size, istenen CSS özelliği, Location, Rect.geometrik özellikler, DOM değeri vb.. Bir çok method vardır ama Automation Test için kullanmıyoruz



WebElement Is Method'lari

webElement.isEnabled(); Web Element erisilebilir ise true
yoksa false doner

webElement.isDisplayed(); Web Element gorunur ise true
yoksa false doner

webElement.isSelected(); Web Element secili ise true
yoksa false doner

Not : Web element ile ilgili bu method'larin disinda size, istenen CSS ozelligi, Location, Rect.geometrik ozellikler, DOM degeri vb.. Bir cok method vardir ama Automation Test icin kullanmiyoruz



BATCH : Batch 44

LESSON : **SELENIUM 04**

DATE : 19.01.2022

SUBJECT : **XPath**
CssSelector
Relative Locators

- techproeducation
- techproeducation
- techproeducation
- techproeducation
- techproedu



WebElement Method'lari

Tekrar Testi

1. Bir class oluşturun : AmazonSearchTest
2. Main method oluşturun ve aşağıdaki görevi tamamlayın.
 - a.google web sayfasına gidin. <https://www.amazon.com/>
 - b. Search(ara) “city bike”
 - c. Amazon'da görüntülenen ilgili sonuçların sayısını yazdırın
 - d. “Shopping” e tıklayın.
 - e. Sonra karşınıza çıkan ilk sonucun resmine tıklayın.



HTML Parent-Child-Sibling Terimleri Nedir?

```
<div class="navFooterLine navFooterLinkLine navFooterDescLine" role="navigation" aria-label="More on Amazon.com">
  <table class="navFooterMoreOnAmazon" cellspacing="0">
    <tbody>
      <tr>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">
          <a href="https://advertising.amazon.com/?ref=footer_advtsing_amzn_com" class="nav_a">
            "Amazon Advertising"
            <br>
            <span class="navFooterDescText">...</span>
          </a>
        </td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
```

Tags: < > şeklinde görülen komutlara etiket (tag) adı verilir.

parent-child-sibling relationship(Ebeveyn-çocuk-kardeş ilişkisi) hakkında konuştuğumuzda, yalnızca tag adları önemlidir

<table>, <div> in çocuğu(dur(child)) ve <tbody>, <tr> in ebeveyndir (parent)

<td> tagları ise siblings (kardes)'dır



LOCATORS XPath

7) By.xpath();

Bir WebElement'i locate etmek icin kullanabilecegimiz en etkin method'dur.

```
WebElement passwordTextBox =driver.findElement(By.xpath("xpath"));
```

Method'un yazimi acisindan diger 6 yontem ile ayni olmakla beraber xpath'i digerlerinden ayiran
cok onemli bir fark vardir



Diger 6 method HTML kod'una baglidir. Web element'in
kodunda id yoksa By.id() method'unu, web element link
degilse By.linkText() method'unu kullanamazsiniz.

Xpath ise dinamiktir. Her turlu web element icin mutlaka
bir xpath yazilabilir

2 cesit Xpath yazilabilir

- 1.**Absolute** xpath (mutlak)
- 2.**Relative** xpath (bagil)



LOCATORS XPath

1. Absolute Xpath()

```
<div class="navFooterLine navFooterLinkLine navFooterDescLine" role="navigation" aria-label="More on Amazon.com">
  <table class="navFooterMoreOnAmazon" cellspacing="0">
    <tbody> ← // div/ table/ tbody
      <tr>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">
          <a href="https://advertising.amazon.com/?ref=footer_advtsing_amzn_com" class="nav_a">
            "Amazon Advertising"
            <br>
            <span class="navFooterDescText">...</span> ← // tbody / tr / td[3] // span
          </a>
        </td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
        <td class="navFooterDescItem">...</td>
        <td class="navFooterDescSpacer" style="width: 4%"></td>
```

Absolute xpath yazmak için en basa // sonraki her adımda / yazarak hedef web element'e kadar tüm tag'lar yazılır.

Eğer aynı path'e sahip birden fazla element varsa index kullanılabilir. [2] gibi

Eğer bir parent'in grand child'lari içinde unique bir tag varsa parent // grand child yazılabilir



LOCATORS XPath

2. Relative Xpath()

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords" autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute" dir="auto" tabindex="0" aria-label="Search">
```

Bir web element'te temel olarak 3 bileşen vardır.

- 1) tag name input
- 2) attribute type, id , value, name, autocomplete, placeholder, class
- 3) attribute value type → 'text', id→ 'twotabsearchtextbox' , value → "

Bu 3 bileşeni birlikte kullanarak her bir webelement için unique sonuc veren bir çok xpath yazılabilir

Relative xpath yazmak için bu 3 bileşen aşağıdaki gibi bir araya getirilir, unique sonuc veren her relative xpath kullanılabilir.

//tagName[@attributelsmi='attributeValue']



LOCATORS XPath

2. Relative Xpath()

```
<input type="text" id="twotabsearchtextbox" value="" name="field-keywords" autocomplete="off" placeholder="" class="nav-input nav-progressive-attribute" dir="auto" tabindex="0" aria-label="Search">
```

- ❖ Genelde 3 bilesen de kullanılır, Ancak bazen daha az bilesen yazmak da yeterli olabilir.

Sadece tag name kullanarak xpath yazmak için

```
driver.findElement(By.xpath("//input"));
```

Tag name farketmeksiz attribute ismi ve attribute value kullanarak xpath yazmak için

```
driver.findElement(By.xpath("// * [@type='text']"));
```

Attribute name farketmeksiz tag name ve attribute value kullanarak xpath yazmak için

```
driver.findElement(By.xpath("//input[@ *= 'text']"));
```

Attribute value farketmeksiz tag name ve attribute ismi kullanarak xpath yazmak için

```
driver.findElement(By.xpath("//input[@type]"));
```



LOCATORS XPath

Class Work: Add Remove Element

- 1- https://the-internet.herokuapp.com/add_remove_elements/ adresine gidin
- 2- Add Element butonuna basin
- 3- Delete butonu'nun gorunur oldugunu test edin
- 4- Delete tusuna basin
- 5- "Add/Remove Elements" yazisinin gorunur oldugunu test edin



LOCATORS XPath

2. Relative Xpath()

- ❖ Bazen de attribute'e bagli olmadan sadece web element icinde bulunan text kullanilabilir.

Exact Text(Belirli bir text) ile element bulma:

```
driver.findElement(By.xpath("//tagname[.= 'text name'] "));
```

```
driver.findElement(By.xpath("//*[.= 'text name'] "));
```

```
driver.findElement(By.xpath("//*[text()= 'exact text with extra space and all'] "));
```

Belirli bir metni içeren bir öğeyi bulmak için şunları kullanabiliriz:

```
driver.findElement(By.xpath("//*[contains(text(),'piece of text')] "));
```

- ❖ Tek attribute ile unique bir sonuca ulasamazsak birden fazla attribute yazabiliriz

```
driver.findElement(By.xpath("//div[@id='logo' or class='flex-col logo'] "));
```

```
driver.findElement(By.xpath("//div[@id='logo' and class='flex-col logo'] "));
```



LOCATORS XPath

Class Work: Add Remove Element

Asagidaki testi text'leri kullanarak locate edin

- 1- https://the-internet.herokuapp.com/add_remove_elements/ adresine gidin
- 2- Add Element butonuna basin
- 3- Delete butonu'nun gorunur oldugunu test edin
- 4- Delete tusuna basin
- 5- "Add/Remove Elements" yazisinin gorunur oldugunu test edin



LOCATORS CssSelector

8- By.cssSelector() Method

```
<input class="form-control" placeholder="Password" data-test="password"
type="password" name="session[password]" id="session_password">
```

Css selector xpath'e benzer. Üç ana tip kullanılır

1) css = tagName[attribute name= 'value'];

```
driver.findElement(By.cssSelector("input[name='session[password]']"));
```

2) css="tagName#idValue" veya sadece css="#idValue" =>yalnızca id value ile çalışır

```
driver.findElement(By.cssSelector("input#session_password"));
```

3) css="tagName.className" veya sadece css=".className"=>yalnızca class value ile çalışır

```
driver.findElement(By.cssSelector(".form-control"));
```



LOCATORS CssSelector

Home Work: Log in Test Using Css

- 1) Bir class oluşturun : Locators_css
- 2) Main method oluşturun ve aşağıdaki görevi tamamlayın.
 - a. Verilen web sayfasına gidin. http://a.testaddressbook.com/sign_in
 - b. Locate email textbox
 - c. Locate password textbox ve
 - d. Locate signin button
 - e. Asagidaki kullanıcı adını ve şifreyi girin ve sign in düğmesini tıklayın
 - i. Username : testtechproed@gmail.com
 - ii. Password : Test1234!

NOT: cssSelector kullanarak elementleri locate ediniz.



BATCH : Batch 44

LESSON : **SELENIUM 05**

DATE : 20.01.2022

SUBJECT : **Relative Locators**
Maven Projesi Oluşturma

- techproeducation
- techproeducation
- techproeducation
- techproeducation
- techproedu



LOCATORS XPath

Tekrar Testi

- 1-C01_TekrarTesti isimli bir class olusturun
- 2- <https://www.amazon.com/> adresine gidin
- 3- Browseri tam sayfa yapin
- 4-Sayfayı "refresh" yapın
- 5- Sayfa basliginin "Spend less" ifadesi içerdigini test edin
- 6- Gift Cards sekmesine basin
- 7- Birthday butonuna basin
- 8- Best Seller bolumunden ilk urunu tiklayin
- 9- Gift card details'den 25 \$'i secin
- 10-Urun ucretinin 25\$ oldugunu test edin
- 10-Sayfayı kapatın



LOCATORS Relative Locators

Relative Locators nedir ?

- ❖ Selenium 4 ile gelen yeniliklerden biri de bagil locator'lardir.
- ❖ Bir web elementi direk locate edemedigimiz durumlarda gunluk hayatimizda kullandigimiz sekilde o web elementi etrafındaki web elementlerin referansi ile tarif edebiliriz.
- ❖ Ornegin yandaki resimde Berlin icin bir cok relative locator tanimlayabiliriz.
 - Boston'in saginda , Sailor'in ustunde
 - NYC'nin altinda, Bay Area'nin solunda
 - Boston yakinlarinda Bay Areanin solunda ve Toronto'nun saginda vb..



<https://www.diemol.com/selenium-4-demo/relative-locators-demo.html>



LOCATORS Relative Locators

Class Work: Relative Locators

```
driver.get("https://www.diemol.com/selenium-4-demo/relative-locators-demo.html#");

WebElement boston=driver.findElement(By.id("boston"));
WebElement sailor = driver.findElement(By.id("sailor"));

WebElement berlin = driver.findElement(with(By.tagName("li")).above(sailor).toRightOf(boston));

WebElement mountie=driver.findElement(with(By.className("ui-li-has-thumb")).below(boston));
```





LOCATORS Relative Locators

Class Work: Relative Locators

- 1) <https://www.diemol.com/selenium-4-demo/relative-locators-demo.html> adresine gidin
- 2) Berlin'i 3 farkli relative locator ile locate edin
- 3) Relative locator'larin dogru calistigini test edin





Maven™

Bir projeyi geliştirmek kadar onu anlasılır kılmak, tekrar eden kodlardan kaçınmak, update'leri tek merkezden kolayca yapabilmek ve projenin surdurulebilirliğini sağlamak (maintenance) da önemlidir.

- ❖ Apache Maven bir yazılım proje yönetimi ve anlama aracıdır.
- ❖ Maven, en iyi uygulamaların geliştirilmesi için güncel ilkeleri bir araya getirmeyi ve bir projeye bu yönde rehberlik etmeyi kolaylaştırmayı amaçlamaktadır.
- ❖ Maven geliştiricileri birçok ayrıntıdan korur.
- ❖ Proje nesne modeli (POM Project Object Model) konseptine dayalıdır.

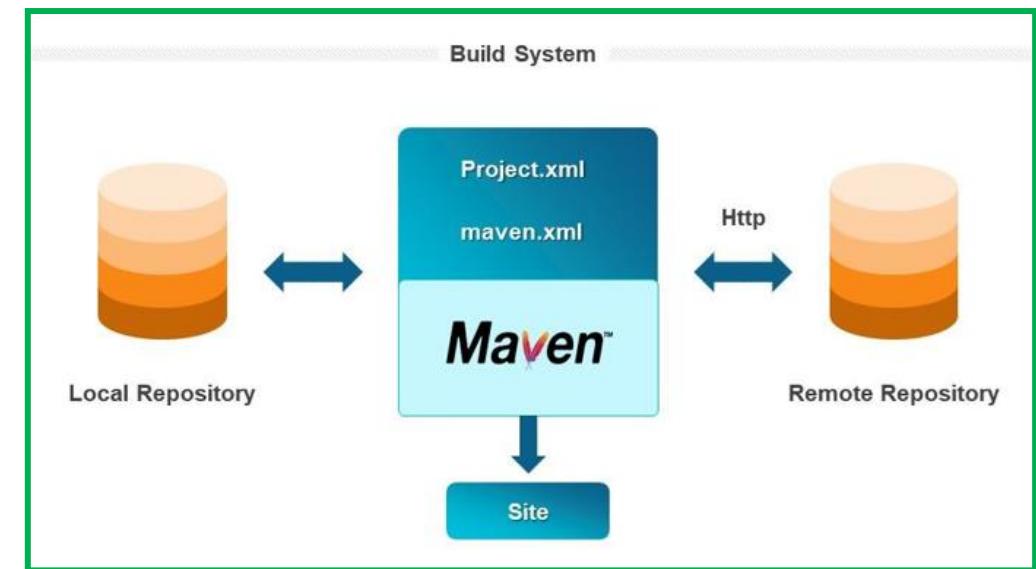
Apache
Maven™

<https://maven.apache.org/>



Maven™

- ❖ Maven bir Java derleme aracıdır (build tool). Maven proje otomasyon ve yönetim aracıdır (automation and management tool).
- ❖ Maven, konfigurasyon için pom.xml dosyasını kullanır. Bu dosya projenin insası, raporlaması ve dokümantasyonu için gerekli bütün bilgileri içerir (dependencies, plugins)
- ❖ Bir Maven projesine aşina olduğunuzda, tüm Maven projelerinin nasıl inşa edildiğini bilirsiniz. Bu, birçok projede gezinirken zaman kazandırır.





Maven™

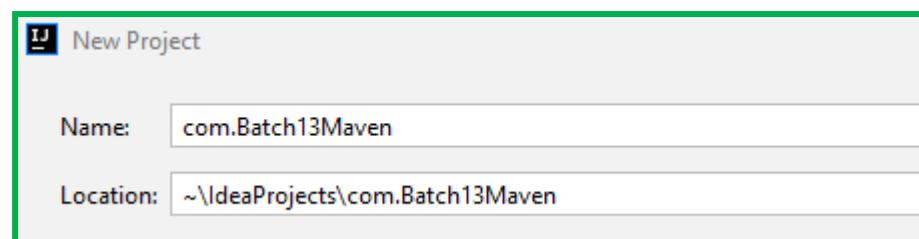
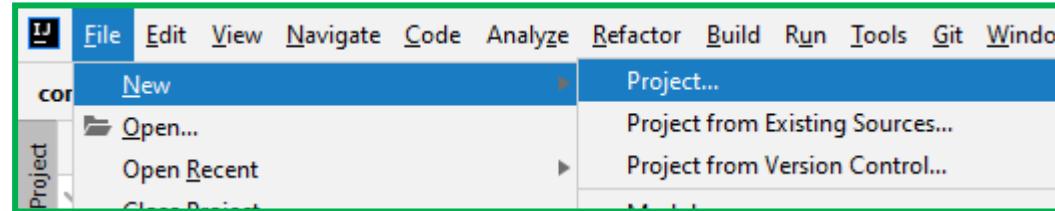
Neden Maven?

- ❖ Tekrarlanabilir derlemeler / yürütütmeleri yapmak kolaylaştırır.(Repeatable builds/executions.)
- ❖ Proje yönetimini kolaylaştırır.
- ❖ Maven, dependencylerle ilgili mevcut jar dosyalarını otomatik olarak indirir.
- ❖ Birden fazla IDE (intelliJ, eclipse, vb.) ve araçlarla(tools) çalışır.
- ❖ Open source
- ❖ Geniş kullanıcı tabanı



Maven™

Maven PROJESINI OLUSTURMA



1. Create Project: File -> New -> Project

2. Select Maven -> click next

Java versiyonunun bilgisayarinizdaki version ile
ayni oldugunu control edin

3. Name: com.Batch44Maven -> finish -> click on
(EnableAutoImport)

4. Package olusturun, name : day04 -> Right click on java create the package

5. Class olusturun, name : FirstMavenClass



POM XML

✓ Pom.xml dosyası nedir?

Maven projesindeki en onemli dosyadır. Dependencies'i POM dosyasına ekler ve POM dosyasından yönetiriz.

✓ Maven ve pom arasındaki fark nedir?

Maven bir araç (tool), pom bir xml dosyasıdır. Pom.xml maven'in bir parçasıdır ve pom dependencies'i yönetir.

✓ Projenizde pom'ı nasıl kullanırız? Neden kullanırız?

Dependencies'i yönetmek için . Tüm projeyi yönetmemeye yardımcı oldu. Pom ayrıca artifact id, group id, ve version gibi proje bilgilerine de sahiptir.

✓ Dependency nedir? Nasıl kullanıyorsunuz? Neden kullanıyorsunuz?

Sürücüler (drivers) kurmak, sürücülerini oluşturmak için dependencies gereklidir. Gerekli araçları içe aktarır(import).

✓ Pom dosyanızı ve dependencies'i nasıl güncellersiniz ?

Mvnrepository.com adresinden gerekli dosyalari bulur ve pom dosyamiza ekleriz



Maven™

POM XML DOSYASINA DEPENDENCIES EKLEME

- 1) Mvnrepository.com adresine gidelim
- 2) WebDriverManager aratalim (En guncel ve en cok kullanılan optimum versiyonu alalim)
- 3) Asagida Maven yazan bolumdeki kodlari kopyalayalim
- 4) pom XML dosyasina gidelim
- 5) <properties> satirinin altina <dependencies> </dependencies> tag'i olusturalim
- 6) Kopyaladigimiz kodlari <dependencies> bolumune yapistiralim
- 7) Ayni islemleri Selenium Java dependency icin de yapalim
- 8) Pom XML dosyasinin sag tarafinda **Maven** yazan bolumu tiklayalim
- 9) Yenile butonuna tiklayip asagida Dependencies bolumunun olustugunu ve altında ekledigimiz kutuphanelerin gorundugunden emin olun



Maven™

POM XML DOSYASINA DEPENDENCIES EKLEME

The screenshot shows the Maven tool window and the pom.xml file. The pom.xml file contains the following code:

```
<groupId>org.example</groupId>
<artifactId>com.mavenSe4</artifactId>
<version>1.0-SNAPSHOT</version>

<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
</properties>

<dependencies>
    <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
    <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>5.0.3</version>
    </dependency>

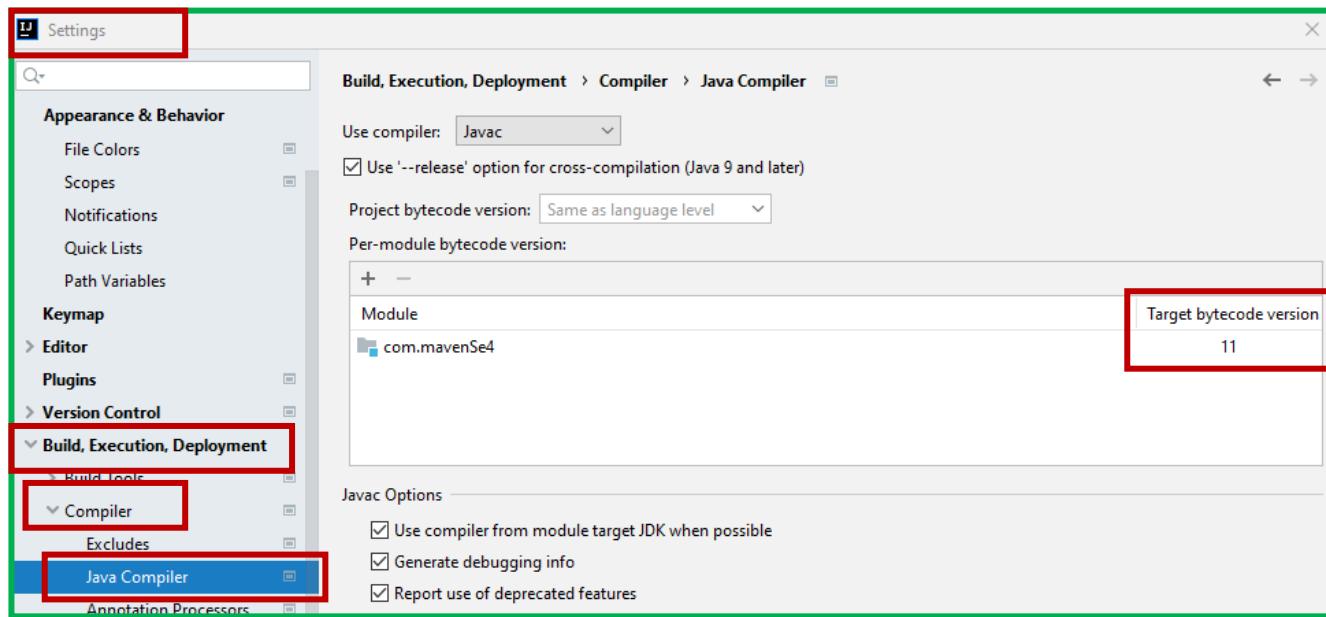
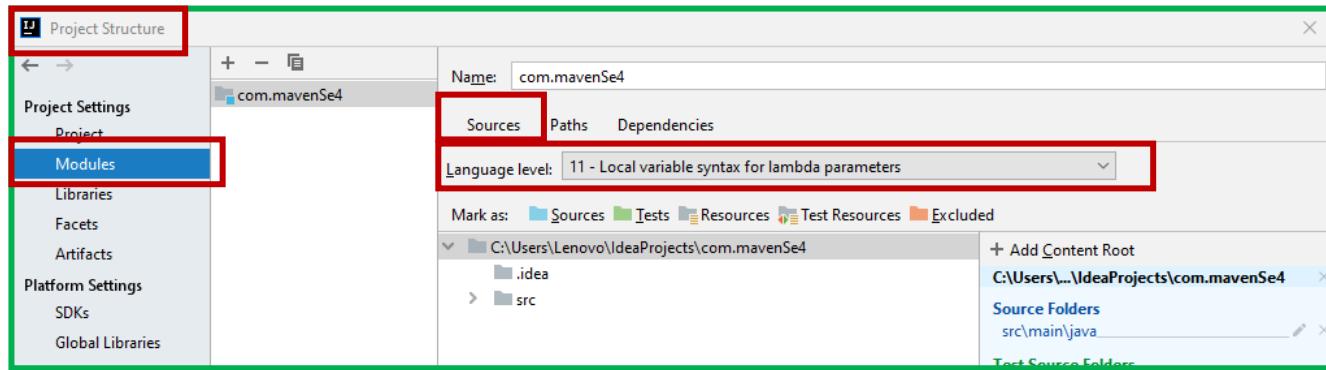
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.1.0</version>
    </dependency>
</dependencies>
</project>
```

The dependencies section is highlighted with a red box. In the Maven tool window, the 'Dependencies' section is also highlighted with a red box, showing the added dependencies: io.github.bonigarcia.webdrivermanager:5.0.3 and org.seleniumhq.selenium:selenium-java:4.1.0.



Maven™

CLASS AYARLARI



File
Project Structure
Modules
Sources
Language level : min 8 yapalim, bilgisayarinizda kurulu java 8 veya üstü ise java versiyonu aynı olmalı

File
Settings
Build,Execution,Deployment
Compiler
Java Compiler
Target bytecode version : min 8 yapalim, bilgisayarinizda kurulu java 8 veya üstü ise java versiyonu aynı olmalı



Maven™

CLASS WebDriver AYARLARI

```
WebDriverManager.chromedriver().setup();
WebDriver driver = new ChromeDriver();
```

Class Work Amazon Search Test

- 1- <https://www.amazon.com/> sayfasina gidelim
- 2- arama kutusunu locate edelim
- 3- "Samsung headphones" ile arama yapalim
- 4- Bulunan sonuc sayisini yazdiralim
- 5- Ilk urunu tiklayalim
- 6- Sayfadaki tum basliklari yazdiralim



Soru 1

1. <http://zero.webappsecurity.com> sayfasina gidin
2. Signin buttonuna tiklayin
3. Login alanine "username" yazdirin
4. Password alanine "password" yazdirin
5. Sign in buttonuna tiklayin
6. Pay Bills sayfasina gidin
7. amount kismina yatirmak istediginiz herhangi bir miktari yazin
8. tarih kismina "2020-09-10" yazdirin
9. Pay buttonuna tiklayin
10. "The payment was successfully submitted." mesajinin ciktigini control edin



BATCH : Batch 44
LESSON : SELENIUM 06
DATE : 21.01.2022
SUBJECT : Junit
Annotations

-  techproeducation
-  techproeducation
-  techproeducation
-  techproeducation
-  techproedu



Maven™

Tekrar Testi

- 1-C01_TekrarTesti isimli bir class olusturun
- 2- <https://www.google.com/> adresine gidin
- 3- cookies uyarisini kabul ederek kapatın
- 4-Sayfa basliginin “Google” ifadesi icerdigini test edin
- 5- Arama cubuguna “Nutella” yazip aratin
- 6-Bulunan sonuc sayisini yazdirin
- 7- sonuc sayisinin 10 milyon’dan fazla oldugunu test edin
- 8-Sayfayı kapatın



Soru 3

1. "https://www.saucedemo.com" Adresine gidin
2. Username kutusuna "standard_user" yazdirin
3. Password kutusuna "secret_sauce" yazdirin
4. Login tusuna basin
5. Ilk urunun ismini kaydedin ve bu urunun sayfasina gidin
6. Add to Cart butonuna basin
7. Alisveris sepetine tiklayin
8. Sectiginiz urunun basarili olarak sepete eklendigini control edin
9. Sayfayı kapatın



JUnit

- ❖ Java ile en temel framework JUnit ile oluşturulabilir.
- ❖ Developerlar da unit testleri calistirmak icin kullanirlar.
- ❖ Biz testlerimizi yapmak icin JUnit'in ileri sürümü olduğundan TestNG framework oluşturup kullanacagiz.
- ❖ Junit maven projesi üzerinde calistigindan <https://mvnrepository.com/> sitesinden dependency'leri projemize ekleriz.
- ❖ Test'lerimizi yaparken main method, if-else gibi java kod bloklarini kullanmak yerine Junit annotationlarini ve method'larine kullanabiliriz.
- ❖ Son framework olarak kuracagimiz Cucumber framework'da da Junit kullanacagiz





Annotations

- ❖ Selenium'da kodlarımızı yazarken "@" işaretini ile başlayan notasyonlar kullanırız.
- ❖ Java Annotationlar ile derleyiciye (Compiler) talimatlar verebiliriz.
- ❖ Annotation, bir veri hakkında bilgi barındıran veriyi sağlayan basit bir yapıdır. Bu sağladığı bilgiye de "metadata" denir.
- ❖ Notasyonlar(Annotation) genellikle Java'da konfigürasyon amacıyla kullanılır. Kullanıldığı bileşene ek özellikle katar. Bu bileşenler sınıf, metod, değişkenler, paket ya da parametreler olabilir. Bunların hepsinde notasyonları kullanabiliriz.

En çok kullanılan Junit annotation'ları

`@Test`

`@BeforeClass @AfterClass`

`@Before , @After`

`@Ignore`



@Test ve @Ignore

- ❖ Junit ile Main Method kullanma dönemini bitiriyoruz.
- ❖ Junit Framework kullandığımızda yazdığımız test metodunun çalışması için başına @Test notasyonu eklememiz yeterlidir.
- ❖ @Test notasyonu eklemedigimiz metot test sırasında çalıştırılmaz. Ancak çağrılsa çalışır.
- ❖ Yazdığımız bazı test metotları henüz tamamlanmamış veya değişiklikleri ugrayabileceğinden dolayı test sırasında çalıştırılmasını istemiyorsak @Ignore notasyonu eklememiz yeterlidir.
- ❖ @Ignore notasyonun tanımlı olduğu metotlar test sırasında çalıştırılmayacaktır. Ayrıca istenilirse @Ignore("açıklama") şeklinde yazılarak metodun neden test edilmesini istemediğimizde yazabiliriz.



Annotations

@Before ve @After

- ❖ Before notasyonu, her test method'undan önce çalışır. Örneğin bir sayfa ile test yapıyorsak ve her testten önce o sayfaya gitmemiz gerekiyorsa @before kullanabiliriz.

@before notasyanunun kullanıldığı method'a genelde setup() ismi verilir.

- ❖ After notasyonu, her test method'undan sonra çalışmaktadır. Örneğin test sırasında kullandığımız sayfanın kapatılması gibi.

@after notasyanunun kullanıldığı metoda genelde teardown() ismi verilir

Not : Test method'u ve Test farklı yapılardır.

Test dedigimizde tek bir method veya içinde birçok class ve package barındıran bir yapı olabilir. Regression test, smoke test vb..

Test method'u ise @Test notasyonu kullanılarak oluşturulan ve bağımsız olarak çalıştırabileceğimiz en küçük test yapısıdır



Annotations

@BeforeClass ve @AfterClass

- ❖ BeforeClass notasyonu, bir class'daki tüm testlerden önce yapılması gereken bir işlem işlem varsa kullanılır (precondition).
Örneğin test metodlarımız çalışmadan driver olusturup tüm methodlarda kullanabilirim.

- ❖ AfterClass notasyonu da, bir class'daki tüm testler tamamlandıktan sonra yapılması gereken işlemlerde kullanılır.
Örneğin actigimiz sayfayı kapatmak veya elde ettigimiz test sonuclarını raporlamak gibi.

- ❖ @BeforeClass ve @AfterClass notasyonları test sürecinde bir kere çalışırken, @Before ve @After notasyonları her test method'unun başında ve sonunda çalışmaktadır.



BATCH : Batch 44
LESSON : SELENIUM 07
DATE : 22.01.2022
SUBJECT : CheckBox

Radio Button
Assertions

- techproeducation
- techproeducation
- techproeducation
- techproeducation
- techproedu



JUnit

Annotations

@BeforeClass - @AfterClass ve @Before - @After notasyonlarının karşılaştırılması

1) Tüm Class çalıştırılırsa

```
Before class method'u calisti → @BeforeClass
Before method'u calisti
Test 1 calisti } @Before - @After
After method'u calisti
Before method'u calisti } @Before - @After
Test 2 calisti }
After method'u calisti } @Before - @After
Before method'u calisti } @Before - @After
Test 3 calisti }
After method'u calisti
After class method'u calisti → @AfterClass
```

2) Tek bir test method'u çalıştırılırsa

```
Before class method'u calisti → @BeforeClass
Before method'u calisti
Test 3 calisti } @Before - @After
After method'u calisti
After class method'u calisti → @AfterClass
```

```
public class C03 {
    @BeforeClass
    public static void beforeClassM(){
        System.out.println("Before class method'u calisti");
    }
    @Before
    public void beforeM() {
        System.out.println("Before method'u calisti");
    }
    @Test
    public void test1(){
        System.out.println("Test 1 calisti");
    }
    @Test
    public void test2(){
        System.out.println("Test 2 calisti");
    }
    @Test
    public void test3(){
        System.out.println("Test 3 calisti");
    }
    @AfterClass
    public static void afterClassM(){
        System.out.println("After class method'u calisti");
    }
    @After
    public void afterM() {
        System.out.println("After method'u calisti");
    }
}
```



CheckBox

1. Bir class oluşturun : CheckBoxTest
2. Gerekli yapıyi olusturun ve aşağıdaki görevi tamamlayın.
 - a. Verilen web sayfasına gidin.
<https://the-internet.herokuapp.com/checkboxes>
 - b. Checkbox1 ve checkbox2 elementlerini locate edin.
 - c. Checkbox1 seçili değilse onay kutusunu tıklayın
 - d. Checkbox2 seçili değilse onay kutusunu tıklayın



RadioButton

1. Bir class oluşturun : RadioButtonTest

2. Gerekli yapıyi olusturun ve aşağıdaki görevi tamamlayın.

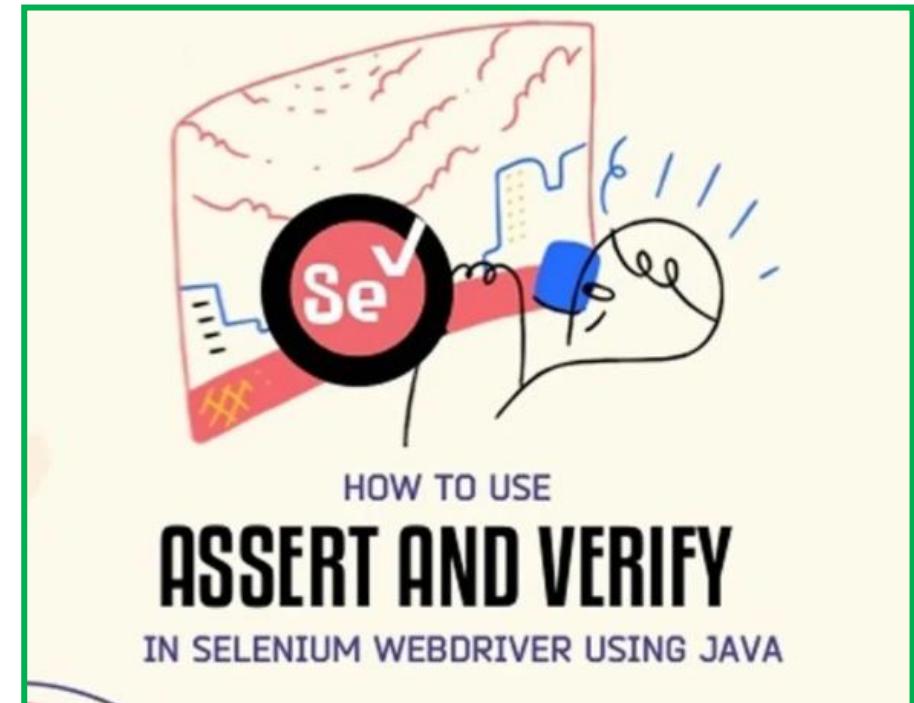
- <https://www.facebook.com> adresine gidin
- Cookies'i kabul edin
- "Create an Account" button'una basin
- "radio buttons" elementlerini locate edin
- Secili degilse cinsiyet butonundan size uygun olani secin



JUnit

Assertions: (Verification)

- ❖ Test icin uygun bir sekilde hazırlanmış bir framework'da expected değerlerin actual değerlere eşit olup olmadığını if-else statement ile bakılmaz.
- ❖ JUnit framework'ünde test senaryomuzu çalıştırıldıktan sonra beklediğimiz sonuçların olup olmadığını tespit etmek için assert class'ından method'lar kullanırız.
- ❖ Junit'de, belirli koşulları test etmek için isimleri assert ile başlayan statik method'lari kullanırız.
- ❖ Bu method'lar ile excepted ve actual değerleri karşılaştırarak testimizi sonuçlandırırız.





Assertions: (Verification)

- ❖ Assert method'unu secmeden once kiyaslamada bekledigimiz sonucun true veya false olmasina karar vermeliyiz.

```
String exceptedName= "Ali Can";
String actualName= "ALI CAN";

exceptedName.equals(actualName); → false

exceptedName.equalsIgnoreCase(actualName); → true
```

```
// Emeklilik yasi 65

int yas1=63;

int yas2=68;

emekliOlabilirMi(yas1); → false

emekliOlabilirMi(yas1); → true
```

- ❖ **ONEMLI OLAN** kiyaslama sonucunun true veya false olması değil, bekledigimiz sonucun olup olmamasıdır. Assertion başarısız olursa AssertionException ile hata mesajı verilir.

`Assert.assertEquals(actualName,expectedName)` → FAILED

Assert.assertTrue(yas2>65) → PASS

Assert.assertFalse(yas1>65) → PASS

Assert.assertTrue(yas2<65) → FAILED

Assert.assertFalse(yas1<65) → FAILED



JUnit

Assertions: (Verification)

A=20 B=30 C=40 D=20

Degerleri icin asagidaki assertion'larin sonuclarini PASS veya FAILED olarak yaziniz

Assert.assertEquals(A,B) → FAILED

false

Assert.assertEquals(A,D) → PASS

true

Assert.assertTrue(C>D) → PASS

true

Assert.assertTrue(C>B) → PASS

true

Assert.assertFalse(B>D) → FAILED

true

Assert.assertFalse(B>A) → FAILED

true



Assertions: (Verification)

- 1) Bir class oluşturun: BestBuyAssertions
- 2) <https://www.bestbuy.com/> Adresine gidin farklı test method'ları oluşturarak aşağıdaki testleri yapın
 - Sayfa URL'inin <https://www.bestbuy.com/> 'a esit olduğunu test edin
 - titleTest => Sayfa başlığının "Rest" içermediğini(contains) test edin
 - logoTest => BestBuy logosunun görüntülediğini test edin
 - FrancaisLinkTest => Fransızca Linkin görüntüldiğini test edin



Assertions: (Verification)

- 1) Bir class oluşturun: `YoutubeAssertions`
- 2) <https://www.youtube.com> adresine gidin
- 3) Aşağıdaki adları kullanarak 3 test metodu oluşturun ve gerekli testleri yapın
 - `titleTest` => Sayfa başlığının “YouTube” olduğunu test edin
 - `imageTest` => YouTube resminin görüntüülendiğini (`isDisplayed()`) test edin
 - `Search Box` 'in erisilebilir olduğunu test edin (`isEnabled()`)
 - `wrongTitleTest` => Sayfa basliginin “youtube” olmadığını doğrulayın



Assertions: (Verification)

1. Bir Class olusturalim YanlisEmailTesti
2. <http://automationpractice.com/index.php> sayfasina gidelim
3. Sign in butonuna basalim
4. Email kutusuna @isareti olmayan bir mail yazip enter'a bastigimizda "Invalid email address" uyarisi ciktigini test edelim



JUnit ASSERT ÖZET

Assert, bir test senaryosunun PASS veya FAILED durumunu belirlemeye kullanılan yararlı bir yöntemdir. Seçilen metod ve yazılan **boolean koşul'a** göre test sonucu belirlenir.

Assert yöntemleri, Java.lang.Object Class'ına bağlı org.junit.Assert Class'ı tarafından sağlanır.

En çok kullandığımız 3 Assert metodu;

1) Assert.assertTrue(**koşul**)

Yazılan koşul'un sonucu True ise test PASS, yoksa test FAILED olur

Assert.assertTrue(**20 > 15**) → Test PASS

True

Assert.assertTrue(**10 > 30**) → Test FAILED

False



JUnit

JUnit ASSERT OZET

2) Assert.assertFalse(**koşul**)

Yazılan koşul'un sonucu False ise test PASS, yoksa test FAILED olur

Assert.assertFalse(40 > 50) → Test PASS
False

Assert. assertFalse(30 > 20) → Test FAILED
True

3) Assert.assertEquals(**expected, actual**)

Yazılan expected ile actual eşit ise test PASS, yoksa test FAILED olur

Assert.assertEquals("Ali" , "Ali") → Test PASS
True

Assert. assertEquals(30 , 20) → Test FAILED
False



Test NG

- JUnit'in gelismis versiyonudur.
- İsmi **Next Generation** Test kelimelerinden türetilmiş , Cédric Beust tarafından oluşturulmuştur.
- Açık Kaynak kodludur.
- TestNG bir test kütüphanesidir.
- TestNG sadece JAVA ile calisir ve JDK 7 ve daha ust versiyonlari gereklidir
- TestNG ile ilgili dokumanlara asagidaki adresden ulasilabilir

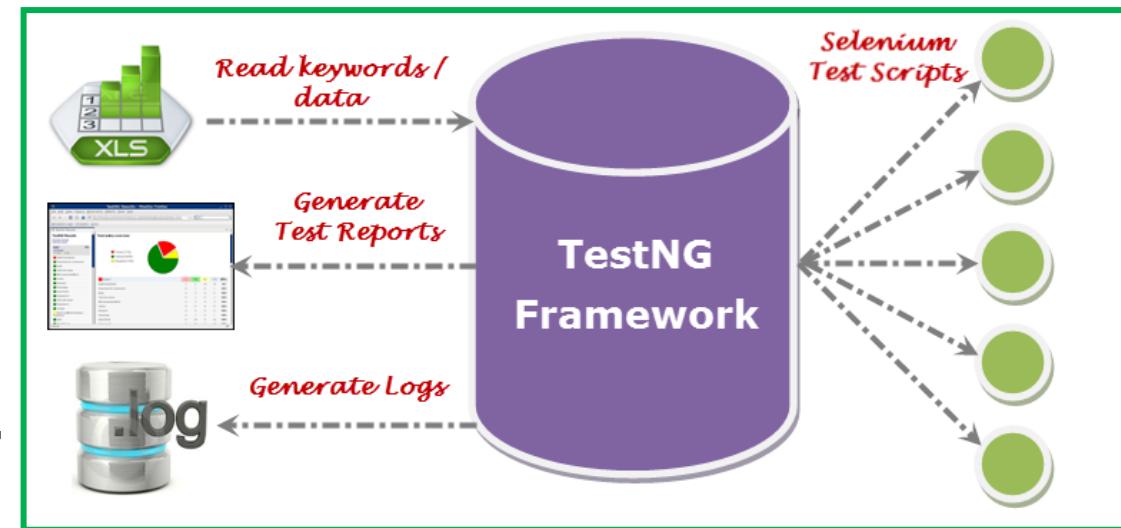
<https://testng.org/doc/documentation-main.html>





Test NG

- TestNG tester'lara daha fazla kontrol imkani verir ve testleri daha etkili yapmamizi saglar.
- Tester'lar TestNG'yi etkili bir framework tasarlamak ve test case'leri TestNG annotation'ları ile organize etmek için kullanırlar.
- Test caseleri siralama ozelligi (priority) ve test caselerin birbirine bagimliliği (dependsOnMethod) bize testleri organize etmekte yardım eder.
- Yazılan test case'lerin paralel olarak çalıştırılmasına, birden fazla browser kullanılmamasına imkan tanır.
- Error mesajlarının daha detaylı bir şekilde gösterilmesini sağlar.
- Paralel ve Cross-Browser Test yapmamiza imkan tanır
- Kullanisli HTML veya xml raporları olusturmaya yarar





Test NG

- Eclipse veya intelliJ için TestNG eklemenin farklı yolları vardır. Maven kullandığımız için dependency biçimini olarak ekleyeceğiz.
- <https://mvnrepository.com/>, adresine gidin
- TestNG'yi aratın ve dependency'i pom.xml'e ekleyin.

Nasıl Yüklenir?

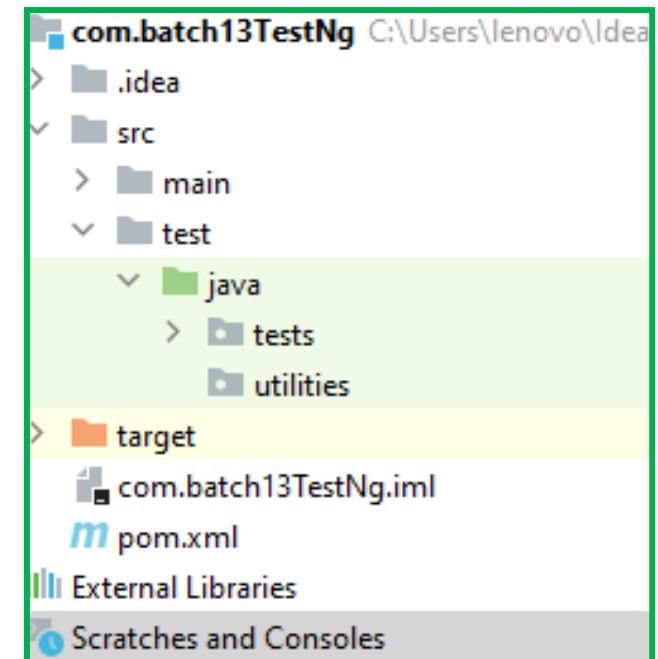
```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.1.0</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
    <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>5.0.3</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.testng/testng -->
    <dependency>
        <groupId>org.testng</groupId>
        <artifactId>testng</artifactId>
        <version>7.4.0</version>
        <scope>test</scope>
    </dependency>
```



Proje Olusturma

- ❖ TestNG ile olusturacagimiz framework gunumuzde de kullanilabilecek bir framework olacaktir
- ❖ Olusturacagimiz framework'de testlerimizi yazacagimiz tests package'i disinda bazi package ve class'lar da olusturacagiz

1. Create Project: File -> New -> Project-> Select maven -> click next
2. Isim: com.batch44TestNG -> finish
cikarsa EnableAutoimport'u tiklayın
3. src altinda yeni package olusturun, isim : tests
4. src altinda yeni bir package daha olusturun, isim : utilities
5. tests package altina yeni package olusturun isim: day07
6. day07 package altina yeni class olusturun isim: C01





Test NG

@Test annotation

- En çok kullanılan TestNG notasyonudur.
- Method'u test case olarak işaretler ve calistirir.
Ayrıca bir Main Method'a ihtiyac duymaz.

```
@Test  
public void test01(){  
    WebDriverManager.chromedriver().setup();  
    WebDriver driver = new ChromeDriver();  
  
    driver.get("https://amazon.com");  
}
```



@Before @After Annotations

@BeforeSuite: The annotated method will be run before all tests in this suite have run.

@AfterSuite: The annotated method will be run after all tests in this suite have run.

@BeforeTest: The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.

@AfterTest: The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.

@BeforeGroups: The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked.

@AfterGroups: The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked.

@BeforeClass: The annotated method will be run before the first test method in the current class is invoked.

@AfterClass: The annotated method will be run after all the test methods in the current class have been run.

NOT :

Bu notasyonlar kullanilinca her method icin ayri ayri calismaz, soylendigi sekilde sadece bir kere calisir.

@BeforeMethod: The annotated method will be run before each test method.

@AfterMethod: The annotated method will be run after each test method.

NOT : Her methoddan once veya sonra calisir. (JUnit deki @Before method'u gibi)



Test NG

- TestNG (default) olarak @Test methodlarını alfabetik sıraya göre run eder. (Yukardan asagi degil!)
- priority annotation Testlere öncelik vermek için kullanılır. Kucuk olan Numara daha once calisir
- priority yazmayan Test method'u varsa priority= 0 kabul edilir, siralama buna gore yapilar

Priority

```
public class C01 {  
    @Test  
    public void youtubeTest(){  
        System.out.println("youtubeTest calisti");  
    }  
    @Test  
    public void amazonTest(){  
        System.out.println("amazonTest calisti");  
    }  
    @Test  
    public void bestBuyTest(){  
        System.out.println("bestBuyTest calisti");  
    }  
}
```

```
amazonTest calisti  
bestBuyTest calisti  
youtubeTest calisti
```

```
public class C01 {  
    @Test (priority = -3)  
    public void youtubeTest(){  
        System.out.println("youtubeTest calisti");  
    }  
    @Test  
    public void amazonTest(){  
        System.out.println("amazonTest calisti");  
    }  
    @Test (priority = 20)  
    public void bestBuyTest(){  
        System.out.println("bestBuyTest calisti");  
    }  
}
```

```
youtubeTest calisti  
amazonTest calisti  
bestBuyTest calisti
```



Priority

- 1) Bir class oluşturun: YoutubeAssertions
- 2) <https://www.youtube.com> adresine gidin
- 3) Aşağıdaki adları kullanarak 4 test metodu oluşturun ve gerekli testleri yapın
 - titleTest => Sayfa başlığının “YouTube” olduğunu test edin
 - imageTest => YouTube resminin görüntüülendiğini (isDisplayed()) test edin
 - Search Box 'in erisilebilir olduğunu test edin (isEnabled())
 - wrongTitleTest => Sayfa basliginin “youtube” olmadığını doğrulayın



Handle Dropdown

- **Adım1:** Dropdown menuyu herhangi bir locator ile locate edin.

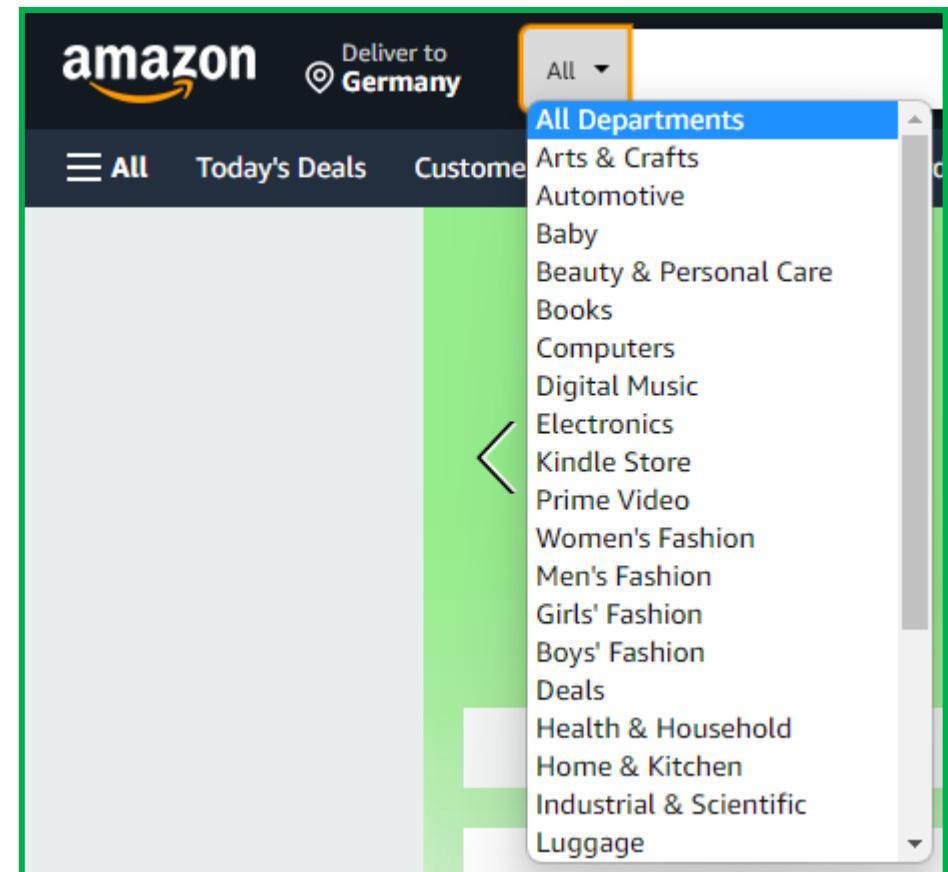
```
WebElement ddm=driver.findElement(By.id("value of id"));
```

- **Adım 2:** Yeni bir "Select" objesi olusturun ve daha once locate ettigimiz WebElement'i **parametre** olarak yeni objeye ekleyin

```
Select options=new Select(ddm);
```

- **Adım 3:** Select class'indan kullanabileceginiz 3 yontemden biriyle dropdown menusundeki elemanlardan istediginizi secin

```
options.selectByIndex(1);
```



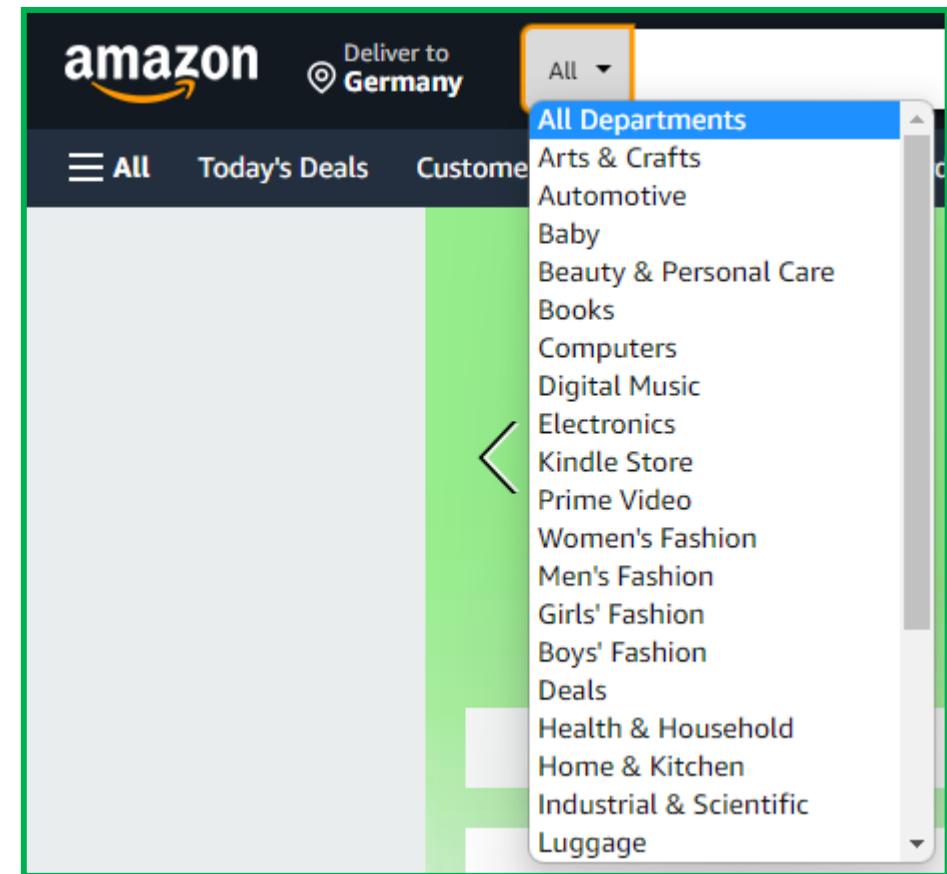


Handle Dropdown

Dropdown menusundeki elementleri Select Class'indan kullanacagimiz yontemlerle 3 sekilde secebiliriz

1. Index kullanarak **selectByIndex()**;
2. Deger kullanarak **selectByValue()**;
3. Gorunen degerini kullanarak **selectByVisibleText()**;

Istenirse **getOptions()**; methodu kullanilarak DropDown'daki tum elementler webelementlerden olusan bir listeye konabilir. List<WebElement>





Handle Dropdown

- Bir class oluşturun: DropDown
- <https://the-internet.herokuapp.com/dropdown> adresine gidin.
 - 1.Index kullanarak Seçenek 1'i (Option 1) seçin ve yazdırın
 - 2.Value kullanarak Seçenek 2'yi (Option 2) seçin ve yazdırın
 - 3.Visible Text(Görünen metin) kullanarak Seçenek 1'i (Option 1) seçin ve yazdırın
 - 4.Tüm dropdown değerleri(value) yazdırın
 5. Dropdown'un boyutunu bulun, Dropdown'da 4 öğe varsa konsolda True , degilse False yazdırın.



Handle Dropdown

- Bir class oluşturun: C3_DropDownAmazon
- <https://www.amazon.com/> adresine gidin.
 - Test 1

Arama kutusunun yanindaki kategori menusundeki kategori sayisinin 45 oldugunu test edin

-Test 2

1. Kategori menusunden Books secenegini secin
2. Arama kutusuna Java yazin ve aratin
3. Bulunan sonuc sayisini yazdirin
4. Sonucun Java kelimesini icerdigini test edin



Handle Dropdown

1. <http://zero.webappsecurity.com/> Adresine gidin
2. Sign in butonuna basin
3. Login kutusuna "username" yazın
4. Password kutusuna "password." yazın
5. Sign in tusuna basin
6. Pay Bills sayfasına gidin
7. "Purchase Foreign Currency" tusuna basin
8. "Currency" drop down menusünden Eurozone'u secin
9. "amount" kutusuna bir sayı girin
10. "US Dollars" in secildigini test edin
11. "Selected currency" butonunu secin
12. "Calculate Costs" butonuna basin sonra "purchase" butonuna basin
13. "Foreign currency cash was successfully purchased." yazısının çıktığını kontrol edin.



dependsOnMethods

- Bu yontem, bir metodun diğer bir metoda bağlı olmasını sağlamak için kullanılır.
- Yandaki ornekte, amazonAnasayfaTest metod'u amazonTest method'una bağlıdır. Yani, amazonTest başarılı olursa amazonAnasayfaTest de çalışacaktır.
- Diger durumda yani, amazonTest başarısız olursa amazonAnasayfaTest **ignore** edilecek, hic çalışmayacaktır.
- Yalnızca amazonAnasayfaTest method'unu çalıştırırsak bile, TestNG önce amazonTest metodunu çalıştırır. amazonTest başarılı olursa searchTest yürütülür
- Ustteki madde sadece 2 method icin gecerlidir. 3 method'u birbirine baglayip 3.method'u calistirirsaniz, 1.method'a kadar gitmez.

```
@Test  
public void amazonTest(){  
    WebDriverManager.chromedriver().setup();  
    WebDriver driver=new ChromeDriver();  
    driver.get("https://www.amazon.com");  
    Assert.assertTrue(driver.getTitle().contains("amazon"));  
}  
  
@Test (dependsOnMethods = "amazonTest")  
public void amazonAnasayfaTest(){  
    // amazon anasayfada yapacagim testler  
}
```



dependsOnMethods Class Work

- Bir class oluşturun: DependsOnTest
- <https://www.amazon.com/> adresine gidin.
 1. Test : Amazon ana sayfaya gittiginizi test edin
 2. Test : 1.Test basarili ise search Box'i kullanarak “Nutella” icin arama yapin ve aramanizin gerceklestigini Test edin
 3. Test : “Nutella” icin arama yapildiysa ilk urunu tiklayin ve fiyatinin \$16.83 oldugunu test edin



Test NG

Test Otomasyonun temel noktalarından biri Assertions'dır.

➤ Her bir test case için bir Assertion veya Verification kullanmalıyız.

➤ TestNG ile iki çeşit Assertion yapmak mümkünudur.

1) Junit'te kullandığımız şekilde Assert Class'ından method'lar kullanarak yapmak.

2) Junit'te olmayan, TestNG'ye özel olarak kullanabileceğimiz SoftAssert Class'ından method'lar kullanarak yapmak.

```
public class C01 {

    WebDriver driver;

    @Test
    public void amazonTest(){
        WebDriverManager.chromedriver().setup();
        driver=new ChromeDriver();
        driver.get("https://www.amazon.com");
        Assert.assertTrue(driver.getTitle().contains("amazon"));
    }

    @Test (dependsOnMethods = "amazonTest")
    public void amazonAnasayfaTest(){

        SoftAssert softAssert =new SoftAssert();
        WebElement aramaKutusu=driver.findElement(By.id("twotabsearchtextbox"));
        aramaKutusu.sendKeys( ...keysToSend: "java"+ Keys.ENTER);
        WebElement ilkUrun=driver.findElement(By.xpath("//span[@class='a-size-base-plus a-color-base a-text-normal'][1]"));
        softAssert.assertTrue(ilkUrun.getText().contains("java"), message: "ilk ürün java içermiyor");
        softAssert.assertAll();
    }
}
```



1. HARD ASSERT

JUnit'te Öğrendiğimiz Assertion ile aynıdır. TestNG'de soft assertion da olduğundan, ayristirmak için bu isim kullanılmıştır.

JUnit'ten bildigimiz üzere kullanabileceğimiz 3 çeşit hard assertion turu vardır

- i. Assert.assertEquals()
- ii. Assert.assertTrue()
- iii. Assert.assertFalse()

Hard assertion herhangi bir assertion FAILED olursa, test method'nun çalışmasını durdurur ve kalan kodları yürütmez (stops execution).

Test case'in nerede FAILED olduğunu hemen anlamak ve kod'a direkt müdahale etmek istenirse hard assertion tercih edilebilir.



2. SOFT ASSERT (VERIFICATION)

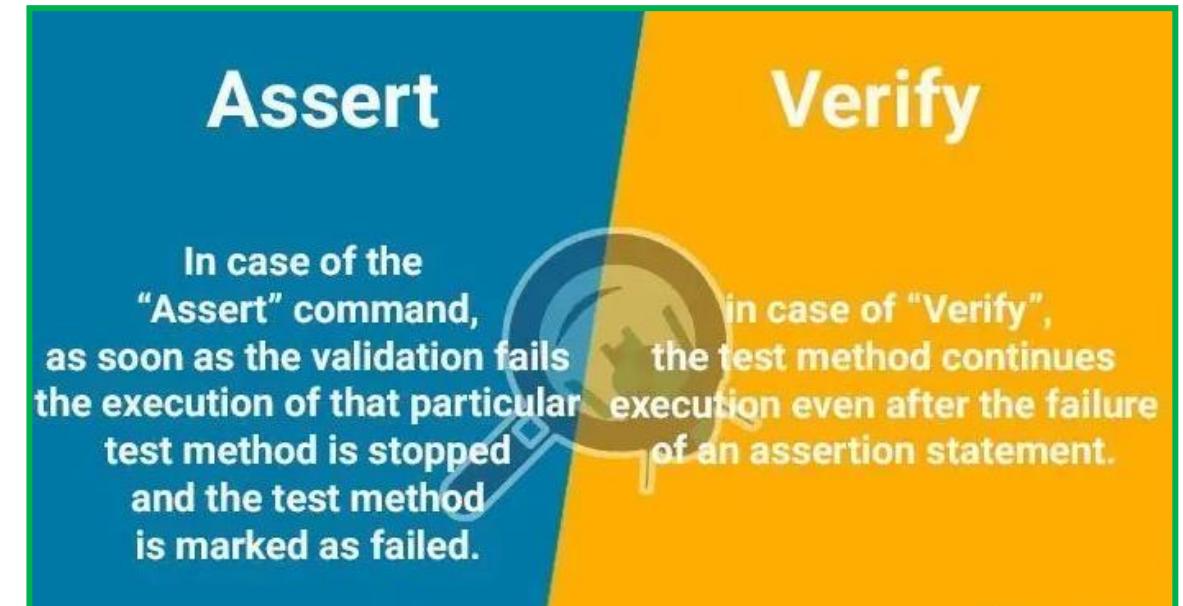
SoftAssert doğrulama (verification) olarak da bilinir.

softAssert kullandığımızda, assert FAILED olsa bile test method'unun istediginiz kismini durdurmaz ve yürütmeye devam eder. If else statement'da olduğu gibi.

Test method'unun istedigimiz bolumde yapılan tüm testleri raporlar

Eger assertion'lardan FAILED olan varsa raporlama yapılan satirdan sonrasini calistirmaz.

SoftAssert class'indaki method'lari kullanmak icin kullanabilmemiz için object olusturmamız gereklidir.





Test NG

Soft Assert

1

- 1.Adım: SoftAssert objesi olusturalim

```
SoftAssert softAssert = new SoftAssert();
```

2

- 2.Adım: Istedigimiz sayida verification'lari yapalim

```
softAssert.assertTrue();
```

```
softAssert.assertFalse(); ....
```

3

- 3.Adım: SoftAssert'in durumu raporlamasini isteyelim

```
softAssert.assertAll();
```



Soft Assert vs Hard Assert

- Ortak ozellikleri

SoftAssert ve HardAssert method'ları TestNG'den gelmektedir.

Kullanma amaçları farklı olsa da method'lar aynıdır.

- Farkları

- Hard Assertion fail olursa test method'larının execute etmesi durur. Ve FAILED olarak tanımlanır.
- Eğer softAssert FAIL olursa test method'ları execute etmeye devam eder. assertAll dedigimizde FAILED olan assertion varsa execution durur.



Yeni bir Class Olusturun : C03_SoftAssert

1. "http://zero.webappsecurity.com/" Adresine gidin
2. Sign in butonuna basin
3. Login kutusuna "username" yazın
4. Password kutusuna "password" yazın
5. Sign in tusuna basin
6. Pay Bills sayfasına gidin
7. "Purchase Foreign Currency" tusuna basin
8. "Currency" drop down menusünden Eurozone'u secin
9. soft assert kullanarak "Eurozone (Euro)" secildigini test edin
10. soft assert kullanarak DropDownList listesinin su seçenekleri oldugunu test edin
"Select One", "Australia (dollar)", "Canada (dollar)", "Switzerland (franc)", "China (yuan)", "Denmark (krone)", "Eurozone (euro)", "Great Britain (pound)", "Hong Kong (dollar)", "Japan (yen)", "Mexico (peso)", "Norway (krone)", "New Zealand (dollar)", "Sweden (krona)", "Singapore (dollar)", "Thailand (baht)"



Test NG

Soft Assert Class Work

Yeni bir Class Olusturun : D11_SoftAssert1

1. "<https://www.hepsiburada.com/>" Adresine gidin
2. Basliginin "Turkiye'nin En Buyuk Alisveris Sitesi" icerdigini dogrulayin
3. search kutusuna araba yazip arattirin
4. bulunan sonuc sayisini yazdirin
5. sonuc yazisinin "araba" icerdigini dogrulayin
6. Sonuc yazisinin "oto" kelimesi icermeydigini dogrulayin



Alert Nedir?

Alert kullanıcıya bir tür bilgi vermek veya belirli bir işlemi gerçekleştirmeye izni istemek için ekran bildirimleri görüntüleyen küçük bir mesaj kutusudur. Uyarı amacıyla da kullanılabilir.

HTML Alerts

Bir alert çıktığında sağ click ile inspect yapabiliyorsak html alert'dir ve extra bir işleme gerek yoktur.

Js Alerts

Js alerts inspect yapılamaz, ekstra işleme ihtiyaç vardır.

Click for JS Alert

1. Simple Alert : Bu basit alert ekranda bazı bilgiler veya uyarılar görüntüler. Ok denilerek kapatılır

Click for JS Confirm

2. Confirmation Alert : Bu onay uyarısı bir tür işlem yapma izni ister. Alert onaylanıyorsa OK, onaylanmıyorsa Cancel butonuna basılır.

Click for JS Prompt

3. Prompt Alert : Bu Prompt Uyarısı kullanıcıdan bazı girdilerin girilmesini ister ve selenium webdriver metni sendkeys ("input....") kullanarak girebilir.



Handle Alert Methods

- `accept()` => Bir uyarıda(alert) OK'ı tıklamakla aynı.

```
driver.switchTo().alert().accept();
```

- `dismiss()` => Bir uyarıda(alert) Cancel'ı tıklamakla aynı.

```
driver.switchTo().alert().dismiss();
```

- `getText()` => Uyarıdaki(alert) mesajı almak için.

```
driver.switchTo().alert().getText();
```

- `sendKeys("Text")` => Uyarı(alert) text kutusuna text göndermek için

```
driver.switchTo().alert().sendKeys("Text");
```



Handle Alert Class Work

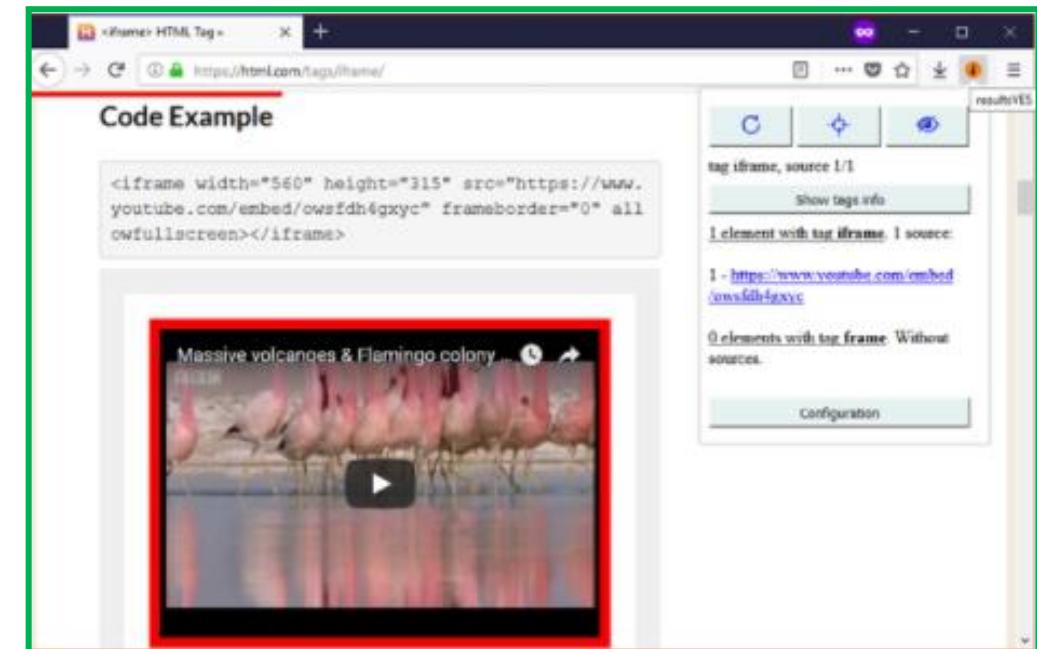
- Bir class olusturun: **Alerts**
- https://the-internet.herokuapp.com/javascript_alerts adresine gidin.
- Bir metod olusturun: **acceptAlert**
 - 1. butona tıklayın, uyarıdaki OK butonuna tıklayın ve result mesajının "You successfully clicked an alert" olduğunu test edin.
- Bir metod olusturun: **dismissAlert**
 - 2. butona tıklayın, uyarıdaki Cancel butonuna tıklayın ve result mesajının "successfully" icermedigini test edin.
- Bir metod olusturun: **sendKeysAlert**
 - 3. butona tıklayın, uyarıdaki metin kutusuna isminizi yazın, OK butonuna tıklayın ve result mesajında isminizin görüntüülendiğini doğrulayın.



Iframe nedir?

- IFrame, bir web sayfasına içine yerleştirilmiş başka bir web sayfasıdır veya bir HTML dokumanının içine yerleştirilmiş başka bir HTML dokumanıdır.
- IFrame genellikle bir Web sayfasına dokuman, video veya interaktif media gibi başka bir kaynaktan içerik eklemek için kullanılır. <iframe> tag'ı bir inline frame belirtir.

<https://html.com/tags/iframe/>





- Bir sayfada iframe varsa, Selenium bir iframe içindeki elementleri doğrudan göremez
- `switchTo()` metod'u ile iframe'e geçmenin 3 yolu vardır;
 - 1) **index ile :**

```
driver.switchTo().frame(index of the iframe); //index 0'dan baslar
```

- 2) **id veya name value ile**

```
driver.switchTo().frame("id of the iframe");
```

- 3) **WebElement ile**

```
driver.switchTo().frame(WebElement of the iframe);
```



Iframe'den cikmak icin 2 komut vardir

`driver.switchTo().parentFrame();` 1 ust seviyedeki frame'e cikartir

`driver.switchTo().defaultContent();` En ustteki frame'e cikmak icin kullanilir

Birden fazla iframe varsa gecislerde dikkatli olmak lazim.

Gecisler her zaman basit olamayabilir

<https://html.com/tags/iframe/>



Test NG

Handle Iframe Class Work

- Bir class olusturun: IframeTest
- <https://the-internet.herokuapp.com/iframe> adresine gidin.
- Bir metod olusturun: iframeTest
 - “An IFrame containing....” textinin erisilebilir oldugunu test edin ve konsolda yazdirin.
 - Text Box'a “Merhaba Dunya!” yazin.
 - TextBox'in altinda bulunan “Elemental Selenium” linkini textinin gorunur oldugunu dogrulayin ve konsolda yazdirin.



- Bir class olusturun: IframeTest02
- 1) <http://demo.guru99.com/test/guru99home/> sitesine gidiniz
 - 2) sayfadaki iframe sayısını bulunuz.
 - 3) ilk iframe'deki (Youtube) play butonuna tıklayınız.
 - 4) ilk iframe'den çıkışp ana sayfaya dönünüz
 - 5) ikinci iframe'deki (Jmeter Made Easy) linke (<https://www.guru99.com/live-selenium-project.html>) tıklayınız



Test NG

TestBase Class

- TestBase, testlerden önce ve sonra tekrar tekrar kullandığımız kodları içermektedir.
- İçerisindeki metodları kullanabilmemiz için extends yapıyoruz. Bu sayede test class'ımızda sadece test case'ler bulunmaktadır.
- Utilities package'da TestBase'i oluşturuyoruz.
 - setUp method
 - reports (Raporlar)
 - tearDown method
- TestBase class'i abstract yapabiliriz (Olmasada olabilir) , extends yaparak kullanabiliriz. Ve bu class'da object create edemeyiz.
- olusturdugumuz driver'in farklı package'lardan kullanılmak için PROTECTED yaparız

```
import java.util.concurrent.TimeUnit;

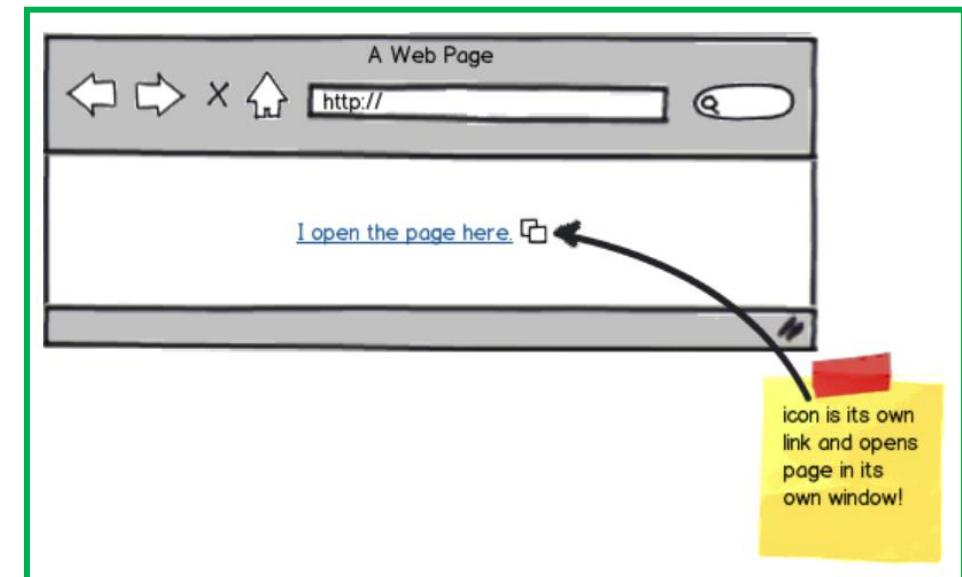
public abstract class TestBase {
    protected WebDriver driver;
    @BeforeMethod
    public void setUp() {
        WebDriverManager.chromedriver().setup();
        driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().pageLoadTimeout(10, TimeUnit.SECONDS);
        driver.manage().implicitlyWait(10, TimeUnit.SECONDS);
    }

    @AfterMethod
    public void tearDown(){
        driver.quit();
    }
}
```



Handle Windows

- Bazen bir butona tıkladığımızda, başka bir sekmede(tab) yeni bir pencere açılır.
- Birden fazla pencereyle çalışırken driver'a pencereler arasında geçis yapmamız gereklidir.
- Pencereler arasında geçis yapmak için **window handle** değerini kullanırız.
- **window handle** : Selenium WebDriver'in, WebDriver objesi başlatıldığında her pencereye verdiği unique alfanumerik kimlik değeridir.





Handle Windows Method'ları

- 1) İçinde olduğumuz sayfanın window handle değerini alma

```
driver.getWindowHandle();
```

- 2) Pencereler arasında geçiş yapma(switch)

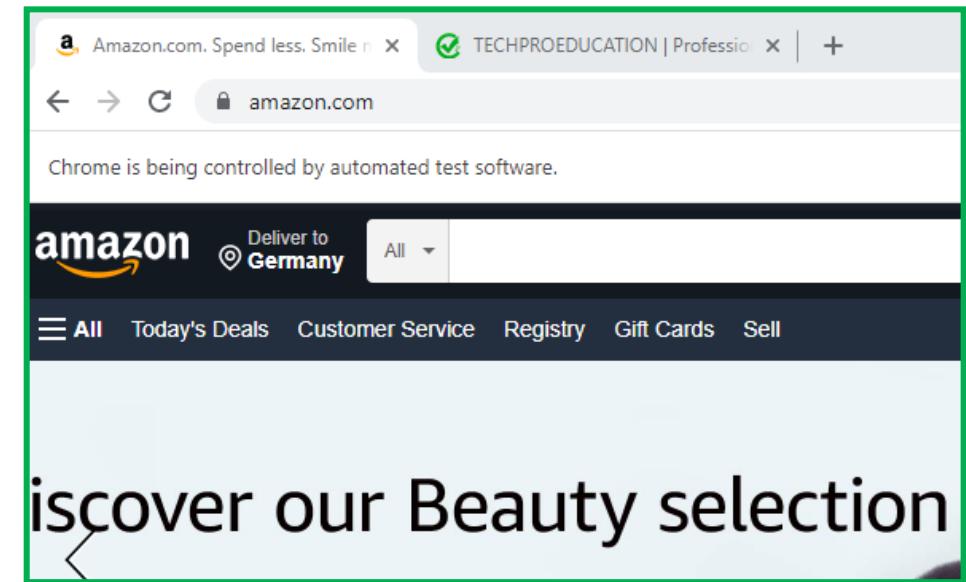
```
driver.switchTo().window(sayfa1HandleDegeri);
```

- 3) Yeni tab oluşturup geçiş yapma(switch)

```
driver.switchTo().newWindow(WindowType.TAB);
```

- 4) Yeni window oluşturup geçiş yapma(switch)

```
driver.switchTo().newWindow(WindowType.WINDOW);
```





- Yeni bir class olusturun: WindowHandle
- Amazon anasayfa adresine gidin.
- Sayfa'nin window handle degerini String bir degiskene atayin
- Sayfa title'nin "Amazon" icerdigini test edin
- Yeni bir tab olusturup, acilan tab'da techproeducation.com adresine gidin
- Sayfa title'nin "TECHPROEDUCATION" icerdigini test edin
- Yeni bir window olusturup, acilan sayfada walmart.com adresine gidin
- Sayfa title'nin "Walmart" icerdigini test edin
- Ilk acilan sayfaya donun ve amazon sayfasina dondugunu test edin



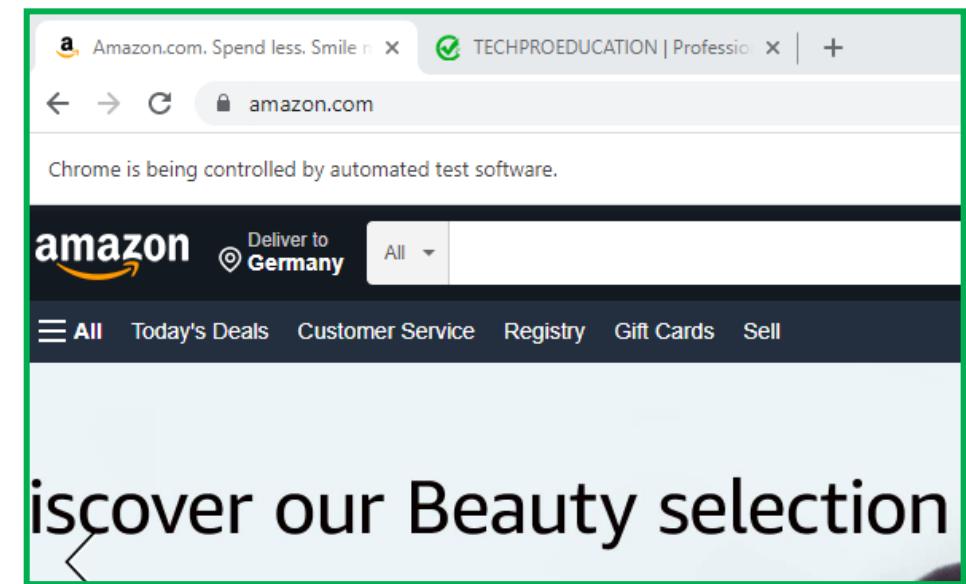
5) Acik olan pencelerin window handle degerlerini alma

```
driver.getWindowHandles();
```

Method'un sonundaki "s" den anlasilacagi uzere birden fazla window handle return eder

Birden fazla window handle degeri oldugu icin collection return etmelidir, method Set return edecek sekilde yazilmistir.

Set'de tum window handle degerleri olacaktir. Biz icinde oldugumuz ilk sayfanin handle degerini kaydedersek, set'te olan 2 window handle degerinden, ilk sayfa handle degerine esit olmayan ikinci handle degeridir.





Handle Windows Class Work

- Tests package'ında yeni bir class olusturun: WindowHandle2
- <https://the-internet.herokuapp.com/windows> adresine gidin.
- Sayfadaki textin “Opening a new window” olduğunu doğrulayın.
- Sayfa başlığının(title) “The Internet” olduğunu doğrulayın.
- Click Here butonuna basın.
- Acilan yeni pencerenin sayfa başlığının (title) “New Window” olduğunu doğrulayın.
- Sayfadaki textin “New Window” olduğunu doğrulayın.
- Bir önceki pencereye geri döndükten sonra sayfa başlığının “The Internet” olduğunu doğrulayın.



Actions Class

- TestNg'de Actions class'ini kullanarak mouse ve klavye ile yapabilecegimiz tum islevleri gerceklestirebiliriz.

- Actions Class birçok kullanışlı mouse ve klavye işlevine sahiptir.

- Çift tıklama (double click), sürükleme ve bırakma(drag and drop), mouse'u hareket ettirme (mouse actions) gibi karmaşık mouse eylemleri icin veya Keyword ile yapabilecegimiz pageUp, pageDown, shift, arrowDown gibi islemleri Actions classından object ureterek driver ile yapabiliriz.





Actions Class

- 1.Adım: Actions class'ta bir object oluşturulur.

```
Actions actions= new Actions(driver);
```

- 2. Adım: Üzerinde çalışmak istediğiniz WebElement öğesini bulunur.

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

- 3.Adım : Ardından bu webelement üzerinde action gerçekleştirilir. Örneğin tıklamak için.

```
actions.click(element).perform();
```

```
driver.get("https://www.amazon.com");
Actions actions = new Actions(driver);

WebElement gununFirsatiElementi=driver.findElement(By.linkText("Today's Deals"));
actions.
    click(gununFirsatiElementi).
    perform();

Thread.sleep( millis: 3000);
WebElement amazonDevicesElement= driver.findElement(By.xpath("//span[text()='Amazon Devices'][2]"));

actions.
    sendKeys(Keys.PAGE_DOWN).
    perform();

Thread.sleep( millis: 3000);
actions.
    click(amazonDevicesElement).
    perform();
```

NOT : Action Class'ını her kullanmak istedigimizde yeniden obje olusturmamız gerekmekz.

action objesi'ni bir kere olusturunca, istediginiz kadar action. ile baslayan komut yazar ve calismasi icin sonuna **perform()** yazariz.

action objesi kullanilarak baslayan her komut, calismak icin **perform()** bekler.



Mouse Base Actions Mouse Aksiyonları

- **doubleClick ()**: WebElement'e çift tıklama yapar
- **clickAndHold ()**: WebElement üzerinde click yapılmış olarak bizden komut bekler.
- **dragAndDrop ()**: WebElement'i bir noktadan diğerine sürüklüyor ve bırakır
- **moveToElement ()**: Mouse'u istedigimiz WebElement'in üzerinde tutar
- **contextClick ()**: Mouse ile istedigimiz WebElement'e sağ tıklama yapar.





Test NG

Mouse Base Actions Class Work

- 1- Yeni bir class olusturalim: MouseActions1
- 2- https://the-internet.herokuapp.com/context_menu sitesine gidelim
- 3- Cizili alan üzerinde sağ click yapalim
- 4- Alert'te çıkan yazinin "You selected a context menu" oldugunu test edelim.
- 5- Tamam diyerek alert'i kapatalim
- 6- Elemental Selenium linkine tiklayalim
- 7- Acilan sayfada h1 taginda "Elemental Selenium" yazdigini test edelim



Mouse Base Actions Class Work

Yeni bir class olusturalim: MouseActions2

- 1- <https://demoqa.com/droppable> adresine gidelim
- 2- “Drag me” butonunu tutup “Drop here” kutusunun ustune birakalim
- 3- “Drop here” yazisi yerine “Dropped!” oldugunu test edin



Mouse Base Actions Class Work

Yeni bir class olusturalim: MouseActions3

- 1- <https://www.amazon.com/> adresine gidelim
- 2- Sag ust bolumde bulunan “Account & Lists” menusunun acilmasi icin mouse'u bu menunun ustune getirelim
- 3- “Create a list” butonuna basalim
- 4- Acilan sayfada “Your Lists” yazisi oldugunu test edelim



Mouse Base Actions Class Work

Yeni bir class olusturalim: D15_MouseActions4

- 1- <https://www.facebook.com> adresine gidelim
- 2- Yeni hesap olustur butonuna basalim
- 3- Ad, soyad, mail ve sifre kutularina deger yazalim ve kaydol tusuna basalim
- 4- Kaydol tusuna basalim



Keyboard Base Actions Klavye Aksiyonları

- Action Class'ından kullanacağımız method'lar ile klavyedeki tusları kontrol edebiliriz.
- Bunun için 3 method kullanırız.
 - **sendKeys ()**: Öğeye bir dizi anahtar gönderir
 - **keyDown ()**: Klavyede tuşa basma işlemi gerçekleştirir
 - **keyUp ()**: Klavyede tuşu serbest bırakma işlemi gerçekleştirir





Test NG

Keyboard Base Actions Class Work

- 1- Bir Class olusturalim KeyboardActions1
- 2- <https://www.amazon.com> sayfasina gidelim
- 3- Arama kutusuna actions method'larine kullanarak samsung A71 yazdirin ve Enter'a basarak arama yaptirin
- 4- aramanin gerceklestigini test edin





Test NG

Keyboard Base Actions Class Work

- 1- Bir Class olusturalim KeyboardActions2
- 2- <https://html.com/tags/iframe/> sayfasina gidelim
- 3- video'yu gorecek kadar asagi inin
- 4- videoyu izlemek icin Play tusuna basin
- 5- videoyu calistirdiginizi test edin



Keyboard Base Actions Homework

Yeni Class olusturun ActionsClassHomeWork

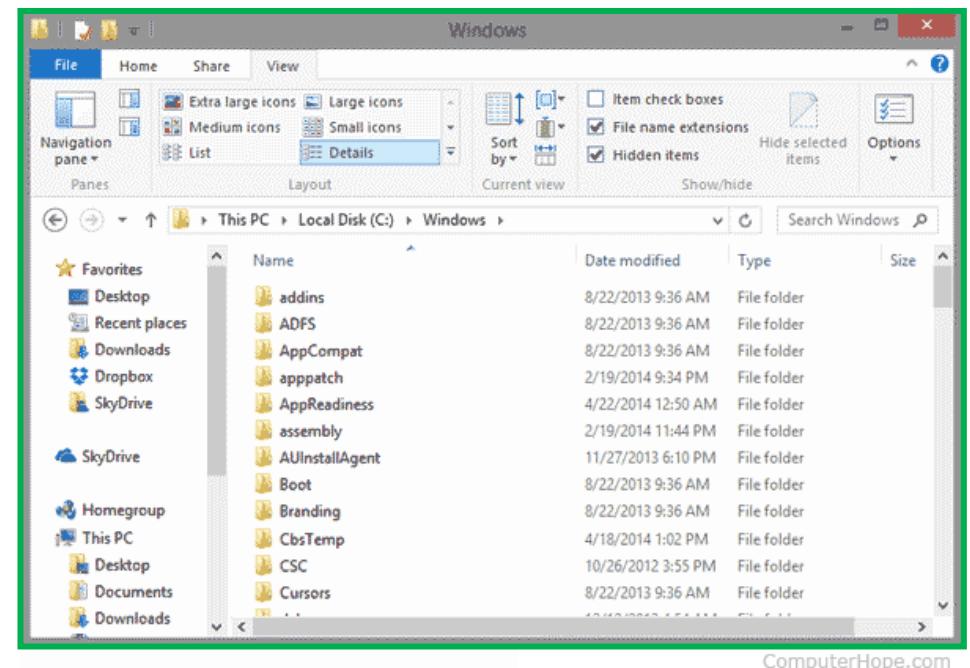
- 1- "http://webdriveruniversity.com/Actions" sayfasina gidin
- 2- Hover over Me First" kutusunun ustune gelin
- 3- Link 1" e tiklayin
- 4- Popup mesajini yazdirin
- 5- Popup'i tamam diyerek kapatin
- 6- "Click and hold" kutusuna basili tutun
- 7- "Click and hold" kutusunda cikan yaziyi yazdirin
- 8- "Double click me" butonunu cift tiklayin



Test NG

File Exist

- Selenium ile windows uygulamalarını test edemiyoruz. Ama test etmek için JAVA kullanabiliriz.
- Bilgisayarımızda bir dosya olup olmadığını(exist) kontrol etmek için Java'yı kullanabiliriz
 - `System.getProperty ("user.dir");` içinde bulunulan klasörün yolunu (Path) verir
 - `System.getProperty ("user.home");` bilgisayarımızda bulunan user klasörünü verir
 - `Files.exists (Paths.get (filePath));` Bilgisayarınızda dosyanın olup olmadığını kontrol eder
- İndirilen bir dosyanın indirme klasörümüzde olup olmadığını kontrol etmek için bu Java konseptini kullanabiliriz





Test NG

File Download/Exist Class Work

1. Tests packagenin altına bir class oluşturalım : C04_FileDownload
2. İki tane metod oluşturun : `isExist()` ve `downloadTest()`
3. `downloadTest ()` metodunun içinde aşağıdaki testi yapalım:
 - <https://the-internet.herokuapp.com/download> adresine gidelim.
 - `code.txt` dosyasını indirelim
4. Ardından `isExist()` methodunda dosyanın başarıyla indirilip indirilmediğini test edelim



Test NG

File Upload (Dosya Yukleme)

Dosya yükleme işlemini anlamak için önce manuel olarak test yapılmalı.

Daha sonra dosya, dosyanın bulunduğu yer (path) kullanılarak yüklenebilir.

<https://the-internet.herokuapp.com/upload>

File Uploader

Choose a file on your system and then click upload.

Dosya seçilmedi



Test NG

File Upload Class Work

1. Tests packagenin altina bir class olusturun : C05_UploadFile
2. <https://the-internet.herokuapp.com/upload> adresine gidelim
3. chooseFile butonuna basalim
4. Yuklemek istediginiz dosyayı secelim.
5. Upload butonuna basalim.
6. “File Uploaded!” textinin goruntulendigini test edelim.



Synchronization - Selenium Waits

- Synchronization(Senkronizasyon), UI (kullanıcı arayüzü) otomasyon testi için çok çok önemlidir.
- Bir sayfanın uygulama sunucusu veya web sunucusu çok yavaşsa veya internet ağı çok yavaşsa, web sayfasındaki öğe (webelement) çok yavaş yüklenir.
- Bu durumda, komut dosyanız (test script) öğeyi bulmaya çalıştığında, öğeler yüklenmez.
- Bu yüzden test komut dosyası(test script) öğeyi bulamaz ve başarısız olur ve **NoSuchElementException** alırız.





Synchronization - Selenium Waits

- Driver ile cihaz veya internet arasında yasanan senkronizasyon sorunlarını çözmek için sürücümüz(driver) belli şartlar ile bekletmemiz(wait) gereklidir.

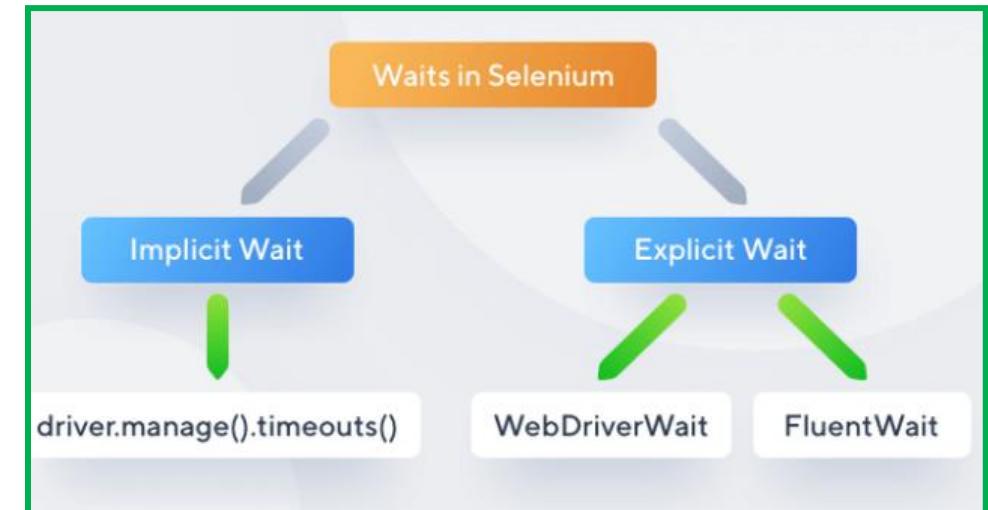
1) Java tabanlı wait

Thread.sleep : Javadan gelir ve kodları yazılan sure kadar bekletir. Sure dolduktan sonra alt satırdan işleme devam eder

2) Selenium tabanlı wait'ler

Implicitly Wait: Sayfadaki tüm öğeler için global bir zaman aşımıdır(timeout).

Explicitly Wait: Çoğunlukla belirli öğeler için belirli bir koşul(expected condition) için kullanılır.





IMPLICITLY WAIT

Bir sayfanın yüklenmesi veya sayfadaki her bir öğenin locate edilebilmesi için driver'i bekletir.

Selenium tabanlı wait'lerde verilen sure max. bekleme süresidir, işlem daha önce biterse surenin bitmesi beklenmez, kod çalışmaya devam eder.

Genellikle otomasyon frameworklerinde olası senkronizasyon problemleri için default olarak implicitly wait ile kullanılır.

Implicitly wait'i TestBase class'ımızda kullanıyoruz.

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

Bu kod, driver'in sayfadaki herhangi bir weblement için maximum 10 saniye beklemesini istediğimiz anlamına gelir.



Weblement 10 saniyeden kısa sürede yüklenirse driver bulur ve devam eder. Örneğin, Weblement 3 saniye içinde yüklenirse, driver sadece 3 saniye bekleyecek ve bir sonraki satırı geçecektir.

Weblement 10 saniye içinde yüklenmezse, test case başarısız olur ve **NoSuchElementException** uyarısı verir.



Test NG

EXPLICIT WAIT

Beklenen bir durum(expected condition) olduğunda explicit wait kullanılır.

Implicitly wait ile cozulebilecek durumlar için explicitly wait kullanımına ihtiyaç yoktur.

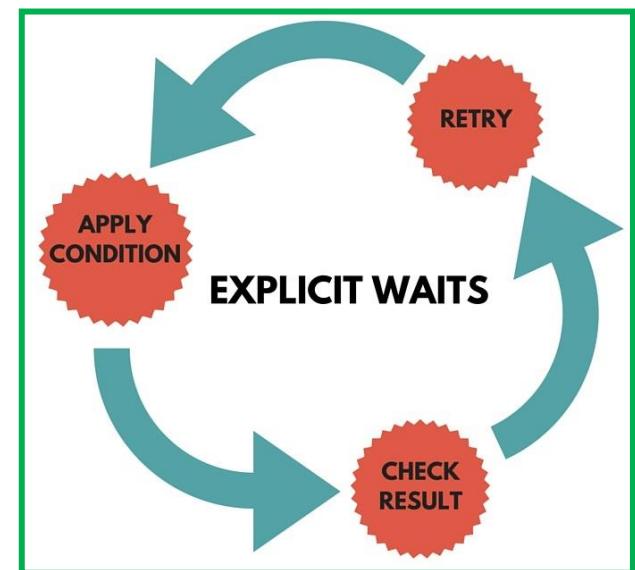
Nadiren karsımıza çıkan ve daha fazla bekleme süresi gerektiren belirli öğeler(web element) için explicitly wait kullanılır.

İlk olarak belirli miktarda bekleme süresi ile wait object create edilir.

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

Oluşturduğumuz wait objesi sayesinde WebDriverWait class'ından until() method'u ile birlikte ExpectedConditions Class'ını kullanabiliriz. Örneğin WebElement'i locate etmek için wait object'i kullanma

```
WebElement element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath("...")));
```





EXPLICIT WAIT Expected Conditions

- 1.alertIsPresent()
- 2.elementSelectionStateToBe()
- 3.elementToBeClickable()
- 4.elementToBeSelected()
- 5.frameToBeAvailableAndSwitchToIt()
- 6.invisibilityOfTheElementLocated()
- 7.invisibilityOfElementWithText()
- 8.presenceOfAllElementsLocatedBy()
- 9.presenceOfElementLocated()
- 10.textToBePresentInElement()
- 11.textToBePresentInElementLocated()
- 12.textToBePresentInElementValue()
- 13.titleIs()
- 14.titleContains()
- 15.visibilityOf()
- 16.visibilityOfAllElements()
- 17.visibilityOfAllElementsLocatedBy()
- 18.visibilityOfElementLocated()



Explicit Wait Class Work

1. Bir class olusturun : WaitTest
2. Iki tane metod olusturun : implicitWait() , explicitWait()

Iki metod icin de asagidaki adimlari test edin.

3. https://the-internet.herokuapp.com/dynamic_controls adresine gidin.
4. Remove butonuna basin.
5. “It's gone!” mesajinin goruntulendigini dogrulayin.
6. Add buttonuna basin
7. It's back mesajinin gorundugunu test edin



Explicit Wait Class Work

1. Bir class olusturun : EnableTest
2. Bir metod olusturun : isEnabled()
3. https://the-internet.herokuapp.com/dynamic_controls adresine gidin.
4. Textbox'in etkin olmadigini(enabled) doğrulayın
5. Enable butonuna tıklayın ve textbox etkin oluncaya kadar bekleyin
6. "It's enabled!" mesajinin goruntulendigini doğrulayın.
7. Textbox'in etkin oldugunu(enabled) doğrulayın.



Test NG

Faker Kutuphanesi

Faker class'i testlerimizi yaparken ihtiyac duyduğumuz isim, soyisim, adres vb bilgiler için fake değerler üretmemize imkan tanır.

Faker değerler üretmek için Faker class'ından bir obje üretir ve var olan method'lari kullanırız.

1. "https://facebook.com" Adresine gidin
2. "create new account" butonuna basin
3. "firstName" giriş kutusuna bir isim yazın
4. "surname" giriş kutusuna bir soyisim yazın
5. "email" giriş kutusuna bir email yazın
6. "email" onay kutusuna emaili tekrar yazın
7. Bir şifre girin
8. Tarih için gün seçin
9. Tarih için ay seçin
10. Tarih için yıl seçin
11. Cinsiyeti seçin
12. İsaretlediğiniz cinsiyetin seçili, diğer cinsiyet kutusunun seçili olmadığını test edin.
13. Sayfayı kapatın



1. "http://webdriveruniversity.com/Actions" sayfasina gidin
2. "Hover over Me First" kutusunun ustune gelin
3. "Link 1" e tiklayin
4. Popup mesajini yazdirin
5. Popup'i tamam diyerek kapatin
6. "Click and hold" kutusuna basili tutun
7. "Click and hold" kutusunda cikan yaziyi yazdirin
8. "Double click me" butonunu cift tiklayin



Test NG

Iframe Home Work

1. "http://webdriveruniversity.com/IFrame/index.html" sayfasina gidin
2. "Our Products" butonuna basin
3. "Cameras product"i tiklayin
4. Popup mesajini yazdirin
5. "close" butonuna basin
6. "WebdriverUniversity.com (IFrame)" linkini tiklayin
7. "http://webdriveruniversity.com/index.html" adresine gittigini test edin



Window Handle Home Work

- 1."<http://webdriveruniversity.com/>" adresine gidin
- 2."Login Portal" a kadar asagi inin
- 3."Login Portal" a tiklayin
- 4.Diger window'a gecin
- 5."username" ve "password" kutularina deger yazdirin
- 6."login" butonuna basin
- 7.Popup'ta cikan yazinin "validation failed" oldugunu test edin
- 8.Ok diyerek Popup'i kapatin
- 9.Ilk sayfaya geri donun
- 10.Ilk sayfaya donuldugunu test edin



Test NG

Java'dan Hatırlamamız Gerekenler

Baska bir Class'dan variable veya method kullanmak istersek 3 yontem kullanabiliriz

A- inheritance (Miras)

kullandigimiz Class'i extends anahtar kelimesi (key word) ile istedigimiz Class'in child'i yapabiliriz.

Bu durumda object olusturmaya gerek kalmadan Parent class'a ulasabilir ve oradaki variable ve methodlari kullanabiliriz. (Test Base gibi)

Inheritance ile variable ve method kullanirken **static** keyword'e dikkat etmek gerekir. Static olarak tanimlanmis bir variable veya method static olmayan method icinden kullanilamaz.

```
public class Okul {  
    String okulIsmi="Yildiz Koleji";  
    static int ogrenciSayisi=120;  
  
    public void okulMethod(){  
        System.out.println("Yildiz Koleji");  
    }  
}
```

```
public class Ogrenci extends Okul {  
  
    public void ogrenciMethod(){  
        System.out.println(okulIsmi);  
        okulMethod();  
  
        System.out.println(ogrenciSayisi);  
    }  
}
```

extends



Test NG

Java'dan Hatırlamamız Gerekenler

B- Object olusturarak

Bir class'dan obje olusturarak istedigimiz class'a ulasabilir ve o class'daki variable ve methodlari object'imizi aracılığıyla kullanabiliriz

ornek: Servis class'indan Okul class'ina ulasmak istiyorsak

- Okul class'indan bir obje olustururuz
- obje uzerinden variable veya method'lara ulasabiliriz

C- Static Class Uyeleri : Eger kullanacagimiz variable veya method static ise object olusturmadan direkt class ismi ile variable veya method'a ulasabiliriz.

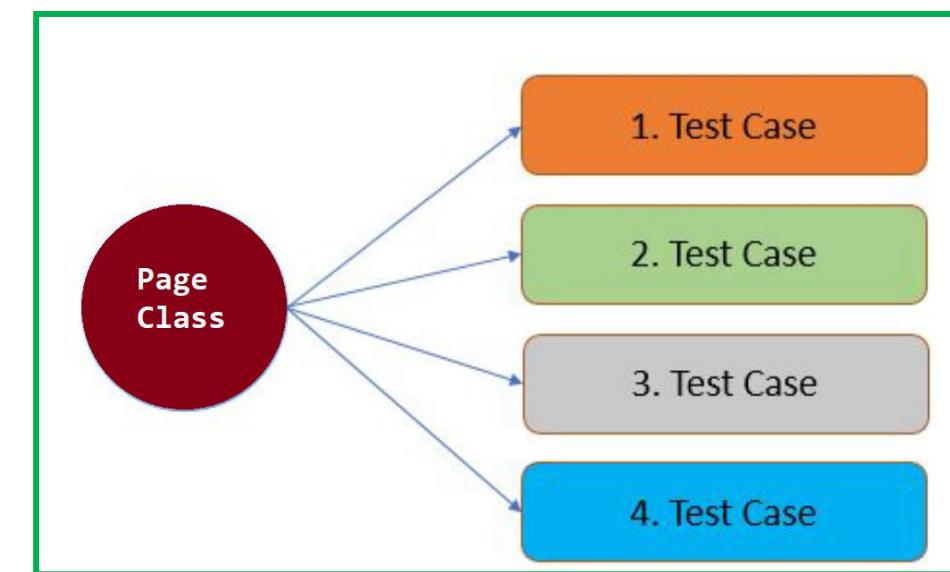
```
public class Okul {  
    String okulIsmi="Yildiz Koleji";  
    static int ogrenciSayisi=120;  
  
    public void okulMethod(){  
        System.out.println("Yildiz Koleji");  
    }  
}
```

```
public class Servis {  
    public static void main(String[] args) {  
        Okul okulObj = new Okul();  
        System.out.println(okulObj.okulIsmi);  
        okulObj.okulMethod();  
  
        System.out.println(Okul.ogrenciSayisi);  
    }  
}
```



Page Object Model Framework design

- Bir sirkette test framework'u olusturdugumuzda kullanici adi, sifresi, gidilecek web adresi gibi test datalari tum testler icin gecerlidir. Ayrice surekli kullanmamiz gereken variable ve method'lar olacaktir.
- Daha kullanisli bir Framework olusturmak icin temel hedefimiz, tekrar tekrar yaptigimiz islemleri ve testlerimizde kullandigimiz Test Data'larini onceden hazırladigimiz dosyalarda tutmaktir.
- Bu sekilde testlerimizde ihtiyac duyduğumuzda bu verilere kolayca ulasabilir veya test datalari ile ilgili bir degisiklik yapmamiz gerektiginde sadece kaynak dosyadan bir degeri degistirerek tum test case'leri guncellemis oluruz.





Test NG

Page Object Model Framework design

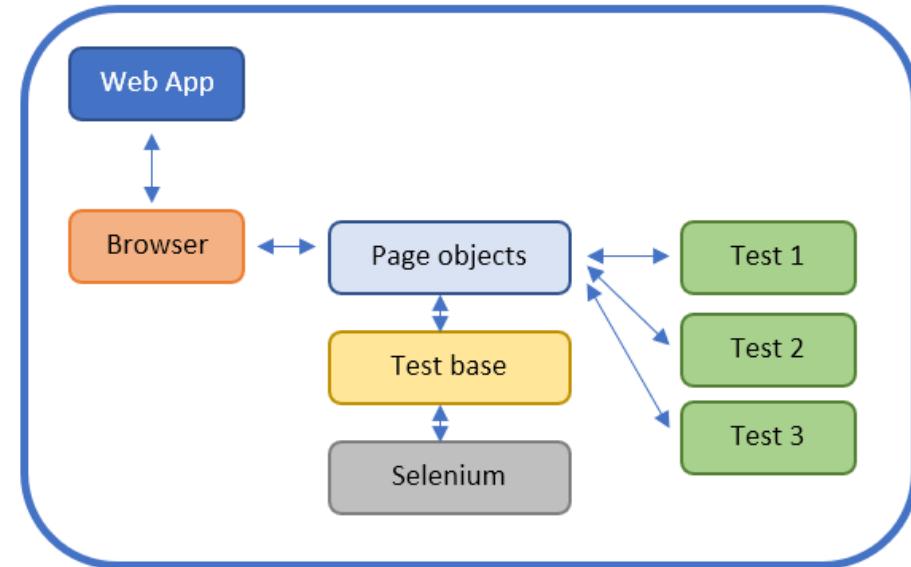
- POM çok popüler bir Framework Design Pattern 'dir.
- Test suitlerimizde çok fazla testimiz olduğunda, test caseleri ve kodları korumak daha karmaşık hale gelir.

Bu nedenle,

sürdürülebilir(maintainable),
yeniden kullanılabilir(reusable),
daha hızlı(faster),
anlaşılabilir(understandable)

daha iyi bir framework dizaynına ihtiyacımız vardır.

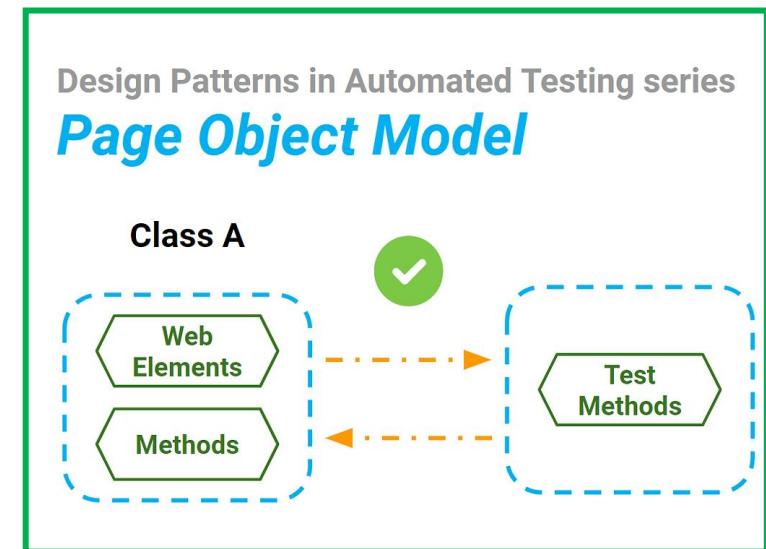
- Page object model ile, sayfaya özgü elementleri veya methodları page class içinde tutar, ve bunları gerçek test classlarından uzak tutarız.
- POM ile ihtiyacımız olan class üyelerini sadece bir kez create edip birden çok kez kullanabiliriz.

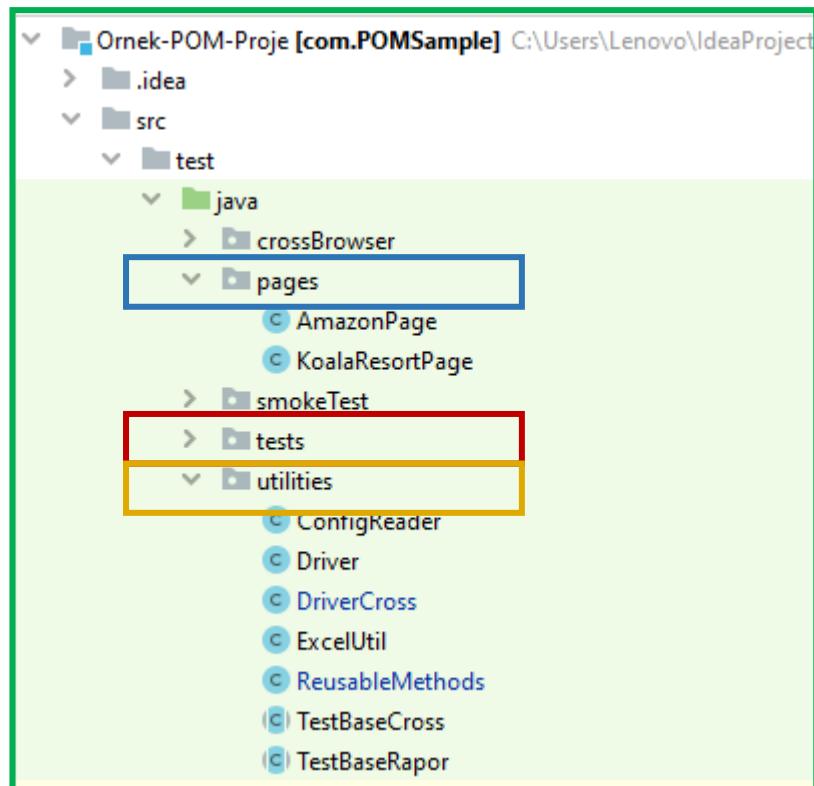




Page Object Model Framework design

- Framework'un verimliliğini artırmak için core Java ve Selenium konseptini kullanarak temel olarak page classları ve test classları oluşturacağız.
- Tüm şirketler page object model dizaynını kullanmaz, ancak herkes bunu bilir ve daha da popüler hale gelmektedir.
- Daha iyi bir tasarım, testin yürütme süresini daha hızlı hale getirir.
- Bir uygulamanın(application) işlevselligi değiştiğinde, kodu düzeltmek için framework kontrol edilmesini ve gerekli düzeltmelerin yapılmasını kolaylaştırır.
- Page Object Design daha bağımsız test senaryoları oluşturmamıza yardımcı olur, böylece test komut dosyalarında(script) hata ayıklamak daha kolay olacaktır.



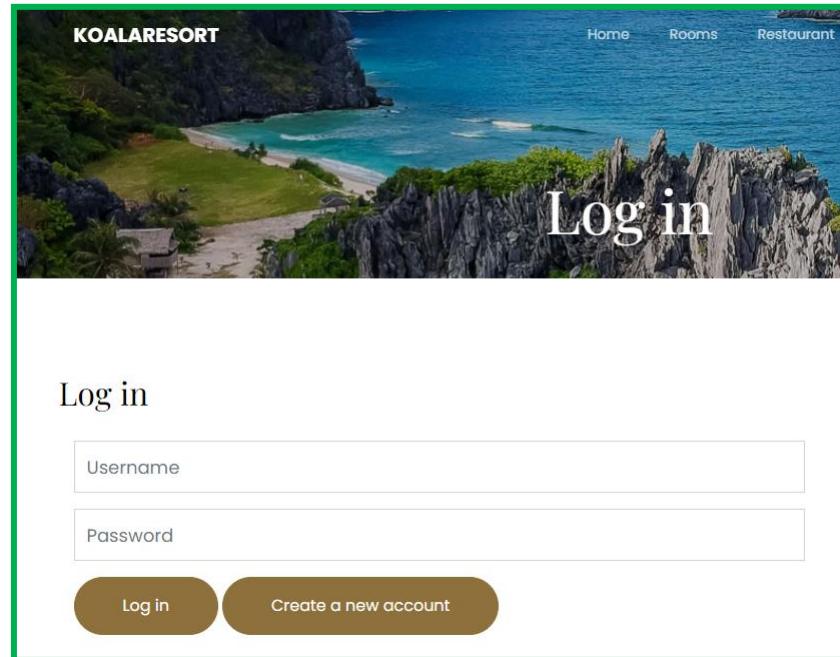


Page Object Model

Framework design

Page Object Model temelde 3 package icerir

- **Tests** : Sadece testleri execute etmek icin gerekli adimlari yazacagimiz class'lar icerir. Hicbir data girişi yapmayacagiz
- **Pages** : Test yapacagimiz sayfalardaki Web Elementlerini locate etmek ve temel methodlari olusturmak icin kullanilir.
- **Utilities** : Driver,TestBase ve ConfigurationReader class'larini icerir



Page Object Model

Framework design

Yandaki otel rezervasyon sitesinde calistiginizi dusunelim,
Yaptiginiz tum testlerde oncelikle register olmaniz gerekecektir.

```
Driver.get("https://www.koalaresorthotels.com/")
userNameTextBox.sendKeys("mehmet")
passwordTextBox.sendKeys("12345")
logInButonu.click()
```

Her test'n basinda yeniden locate'leri yapmaniz, her test icin yeniden test datalarini girmeniz gerekir.

Kullanici adi ve sifresi degistiginde ise, var olan yuzlerce test method'unu gozden gecirmeniz
kullanici adi ve sifresini duzeltmeniz gerekir



Test NG

PAGE CLASS

```
public class KoalaResortPage {  
  
    public KoalaResortPage(){  
  
        PageFactory.initElements(Driver.getDriver(), page: this);  
    }  
  
    @FindBy(linkText = "Log in")  
    public WebElement ilkLoginLinki;  
  
    @FindBy(id = "UserName")  
    public WebElement kullaniciAdiTextBox;  
  
    @FindBy(id = "Password")  
    public WebElement passwordTextBox;  
  
    @FindBy(id = "btnSubmit")  
    public WebElement loginButton;  
  
    @FindBy(xpath = "//*[text()='Try again please ']")  
    public WebElement girisYapilamadiElementi;  
}
```

Page Object Model Framework design

TEST CLASS

```
@Test  
public void positiveLoginTest() {  
    // http://qa-environment.koalaresorthotels.com adresine git  
    Driver.getDriver().get(ConfigReader.getProperty("kr_url"));  
  
    //login butonuna bas  
    KoalaResortPage koalaResortPage = new KoalaResortPage();  
    koalaResortPage.ilkLoginLinki.click();  
  
    //test data username: manager ,  
    koalaResortPage.kullaniciAdiTextBox.sendKeys(ConfigReader.getProperty("kr_valid_username"));  
  
    //test data password : Manager1!  
    koalaResortPage.passwordTextBox.sendKeys(ConfigReader.getProperty("kr_valid_password"));  
  
    //Degerleri girildiginde sayfaya basarili sekilde girilebildigini test et  
    koalaResortPage.loginButton.click();  
    Assert.assertEquals(Driver.getDriver().getCurrentUrl(), ConfigReader.getProperty("kr_basarili_giris_url"));  
  
    Driver.closeDriver();  
}
```



Page Object Model Page Factory

Bir page class'i olusturdugumuzda ilk isimiz bir constructor olusturup, pageFactory class'indan initElements() method'unu kullanmak olmalidir.

1. **PageFactory**, page object dizayni icin bir önemli classtır.
2. Page objelerini instantiate(ilk deger atama) için page classlarında **PageFactory** kullanıyoruz.
3. PageFactory.initElements(driver,this);

this => page instance

driver => bizim gonderecegimiz driver
4. Aslinda PageFactory class'ina, elementlere ilk degeri atayan **initElements()** metodunu kullanmak icin ihtiyacimiz var

```
public class ConcorHotelPage {  
  
    WebDriver driver;  
    public ConcorHotelPage(WebDriver driver){  
        this.driver=driver;  
        PageFactory.initElements(driver,this);  
    }  
  
    @FindBy(linkText = "Log in")  
    public WebElement ilkLoginLinki;  
  
    @FindBy(id="UserName")  
    public WebElement usernameKutusu;  
  
    @FindBy(id="Password")  
    public WebElement passwordKutusu;  
  
    @FindBy(id="btnSubmit")  
    public WebElement loginButonu;
```



Page Object Model @FindBy Annotation

Page class'larinda webElementleri locate etmek icin simdiye kadar kullandigimiz driver.findElement() method'unu kullanmayacagiz.

1. **@FindBy** notasyonu test class'larinda kullanacagimiz Web Elementlerini Page sayfasinda locate etmek icin kullanilir
2. Bunun icin kullanacagimiz Web Elementini public olarak olusturmali, sonra da **@FindBy** notasyonu ile locate etmeliyiz
3. Bu islemi yaptiktan sonra hangi test methodumuzda bu web elemente ihtiyac duyarsak page class'indan uretecegimiz obje uzerinden rahatlikla kullanabiliriz

```
public class ConcordeHotelPage {  
  
    WebDriver driver;  
    public ConcordeHotelPage(WebDriver driver){  
        this.driver=driver;  
        PageFactory.initElements(driver,this);  
    }  
  
    @FindBy(linkText = "Log in")  
    public WebElement ilkLoginLinki;  
  
    @FindBy(id="UserName")  
    public WebElement usernameKutusu;  
  
    @FindBy(id="Password")  
    public WebElement passwordKutusu;  
  
    @FindBy(id="btnSubmit")  
    public WebElement loginButonu;
```



- 1 - <https://www.facebook.com/> adresine gidin
- 2- POM'a uygun olarak email, sifre kutularini ve giris yap butonunu locate edin
- 3- Faker class'ini kullanarak email ve sifre degerlerini yazdirip, giris butonuna basin
- 4- Basarili giris yapılamadigini test edin

E-posta veya Telefon Numarası

Şifre

Giriş Yap

Şifreni mi Unuttun?

Yeni Hesap Oluştur



PositiveTest

- 1) Bir Class olustur : PositiveTest
- 2) Bir test method olustur positiveLoginTest()

<https://www.concorthotel.com/> adresine git

login butonuna bas

test data username: **manager** ,

test data password : **Manager1!**

Degerleri girildiginde sayfaya basarili sekilde girilebildigini test et



NegativeTest

- 1) Bir Class olustur : NegativeTest
- 2) Bir test method olustur NegativeLoginTest()

<https://www.concorthotel.com/> adresine git

login butonuna bas

test data username: **manager1** ,

test data password : **manager1!**

Degerleri girildiginde sayfaya girilemediğini test et



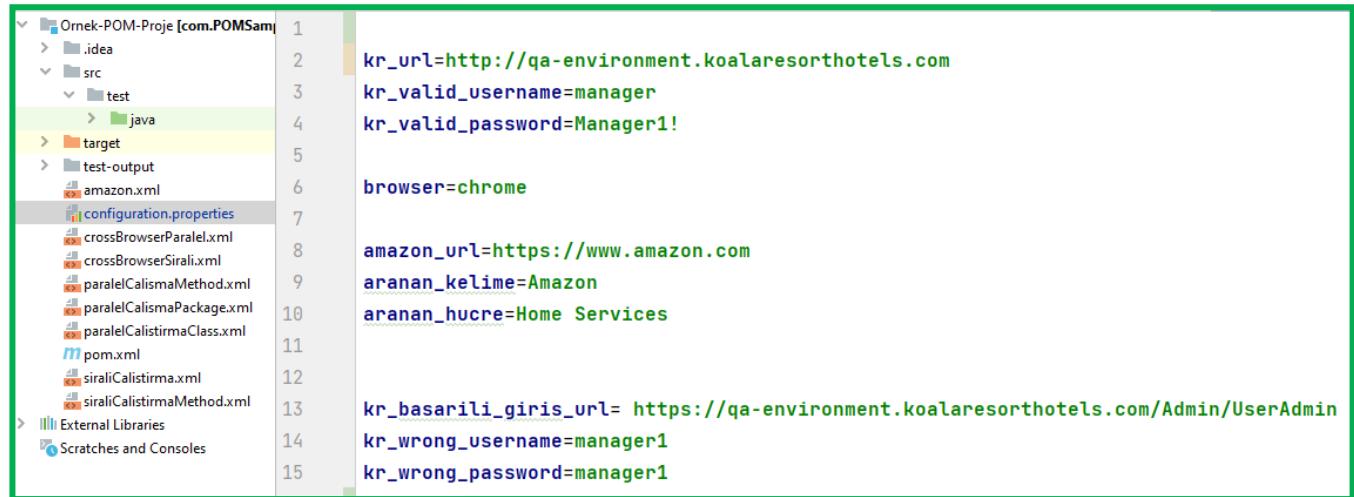
Test NG

Page Object Model Properties File

- configuration.properties Test datalarini tuttugumuz **.properties** uzantılı bir dosyadır.
- Bu dosya , projeyi temiz ve dinamik hale getirir. Test datalarini dinamik hale getirebiliriz.

Örneğin :

`driver.get("https://www.google.com")`
yazmak yerine configuration
dosyamiza **url**'i tanımlayıp test
classında sadece
`driver.get(url)` kullanırız.



The screenshot shows a file explorer window with the following structure:

- Ornek-POM-Proje [com.POMSam]
- .idea
- src
- test
 - java
 - target
 - test-output
 - amazon.xml
 - configuration.properties
 - crossBrowserParallel.xml
 - crossBrowserSiralixml
 - parallelCalismaMethod.xml
 - parallelCalismaPackage.xml
 - parallelCalistirmaClass.xml
 - pom.xml
 - siralCalistirma.xml
 - siralCalistirmaMethod.xml
- External Libraries
- Scratches and Consoles

The configuration.properties file content is as follows:

```
kr_url=http://qa-environment.koalaresorthotels.com
kr_valid_username=manager
kr_valid_password=Manager1!
browser=chrome
amazon_url=https://www.amazon.com
aranan_kelime=Amazon
aranan_hucre=Home Services
kr_basarili_giris_url= https://qa-environment.koalaresorthotels.com/Admin/UserAdmin
kr_wrong_username=manager1
kr_wrong_password=manager1
```

- Temel olarak key(anahtar) ve value(değer) çiftlerini kullanırız ve ihtiyaç duyduğumuzda key kullanarak value cagırırız(url,credentials,browser,environments,...)
 - **url=https://www.fhctrip.com/**
 - **password=Manager2!**
 - **browser=chrome**
 - **name=Ali**
 - **username=manager**



Page Object Model Properties File

- **configuration.properties file** Olusturmak icin project'imize sag click yapin
New =>File => isim : **configuration.properties**
- Dosya olustururken bizim icin onemli olan ismi degil, uzantisidir (.properties)
- Bu uzanti sayesinde Properties Class'ini kullanabilir, olusturacagimiz obje yardimiyla **configuration.properties** dosyasindaki key-value ikililerine ulasabiliriz
- properties file test datalari saklar.
- Bu dosyayı kullanmanın amacı, kodu sabit(hard coded) değil, dinamik yapmaktır. Bu dosya sayesinde tum kullanicilar verilere kolayca ulasabilir ve update edebilir.

A screenshot of an IDE showing a file structure. The project folder is named 'mymavenproject'. Inside it are '.idea', 'src', 'target', '.gitignore', and 'configuration.properties'. To the right of the file list, there is a detailed view of the 'configuration.properties' file's contents:

1	url=https://www.google.com/
2	browser=chrome
3	user=John
4	address=45th freeway
5	test_script_path=US1234
6	environment=UITesting
7	
8	



Test NG

ConfigurationReader class test method'larimiz ile Configuration.properties arasında iletisimi saglar.

Bu class'da test class'larindan kolayca ulasmak icin **static** method bulunur.

Method **static** oldugundan method icerisinden cagiracagimiz variable da **static** olmalidir.

Kullanicagimiz static variable'a ilk degeri atamak icin(instantiate) de **static** block kullaniriz.

Page Object Model ConfigurationReader Class

The screenshot shows the file structure of a Java project named 'Ornek-POM-Proje [com.POMSample]'. The 'src/test/java' directory contains several packages: 'crossBrowser', 'pages', 'smokeTest', 'tests', and 'utilities'. Inside 'utilities', there is a file named 'ConfigReader.java'. The code for this file is as follows:

```
package utilities;
import ...;

public class ConfigReader {
    private static Properties properties;

    static {
        String path="configuration.properties";
        try {
            FileInputStream fileInputStream=new FileInputStream(path);
            properties =new Properties();
            properties.load(fileInputStream);

            fileInputStream.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static String getProperty(String key){
        return properties.getProperty(key);
    }
}
```



Test Class

```
public class TestClass {  
    @Test  
    public void test1(){  
        https://www.amazon.com  
        String url = ConfigReader.getProperty("amazon_url");  
    }  
}
```

- 1) Test method'undan “amazon_url” key'e ait value'yu kullanmak istedigimizde
- 2) ConfigReader Class'indan getProperty() method'unu kullanırız.
- 3) getProperty() method'u configuration.properties dosyasına gidip istedigimiz key'e ait value'yu bize dondurur.

Page Object Model Config Reader Data Flow

ConfigurationReader Class

```
public class ConfigReader {  
    private static Properties properties;  
  
    static {  
        String path="configuration.properties";  
        try {  
            FileInputStream fileInputStream=new FileInputStream(path);  
            properties =new Properties();  
            properties.load(fileInputStream);  
  
            fileInputStream.close();  
        } catch (FileNotFoundException e) {  
            e.printStackTrace();  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }  
  
    public static String getProperty(String key){  
        return properties.getProperty(key);  
    }  
}
```

Not: Properties dosyasında olmayan bir anahtar(key) alırsak, null değeri döndürür

. Örn: getProperty (“country”) dosyada ülke yok, bu nedenle null değerini döndürür

Configuration.properties file

```
amazon_url=https://www.amazon.com  
aranan_kelime=Amazon  
aranan_hucre=Home Services
```



Page Object Model Driver Class

Temel Hedefimiz: Test methodlarimizda kullanacagimiz WebDriver driver'i utilities altindaki Driver Class'inda olusturmak ve testlerimizde bu class ismi üzerinden driver'a ulasip olusturma, kullanma ve kapatma islemlerini yapmak.

Basic Driver Class

```
public class Driver {  
  
    private static WebDriver driver;  
  
    public static WebDriver getDriver() {  
  
        if (driver == null) {  
            WebDriverManager.chromedriver().setup();  
            driver = new ChromeDriver();  
        }  
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));  
        driver.manage().window().maximize();  
  
        return driver;  
    }  
}
```

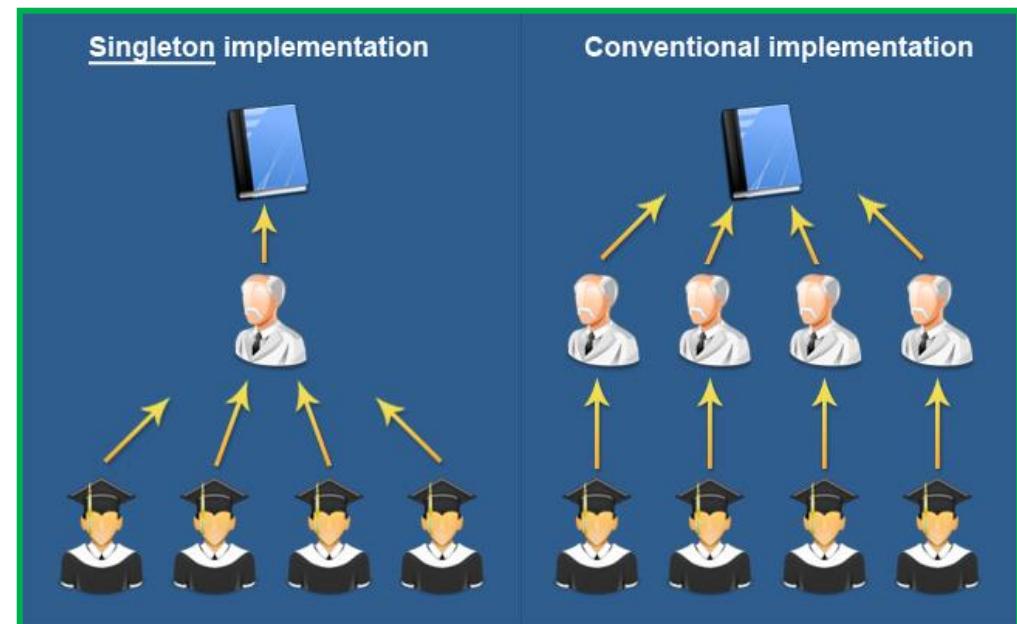
```
public class C02 {  
  
    @Test  
    public void test01(){  
  
        Driver.getDriver().get("https://www.amazon.com");  
    }  
}
```

Test Class(driver->Driver.getDriver())



Singleton Pattern (Tekli Kullanım)

- Herhangi bir Java classından obje kullanımını sınırlayabiliriz. Buna Singleton pattern(tekli kullanım) denir.
- Singleton pattern, class'ı tek bir instance ile kısıtlayan bir software dizayn kalıbidır.
- Proje genelinde farklı objeler oluşturulmadan tek bir instance gereksinimimiz olduğunda Singleton class kullanırız.
- Sadece bir obje oluşturmalı ve buna her ihtiyaç duyduğumuzda kullanmamız performans ve bellek kullanımı için de faydalıdır.





Page Object Model Driver Class

Tüm Browserlar icin Kullanım

- Driver Classımızı tüm tarayıcılar(browser) için geliştireceğiz.
- Driver objesi create etmeden önce farklı tarayıcı koşullarını kontrol etmek için switch statement kullanıyoruz.
- Driver Class'i singleton pattern'e uygun dizayn ederek tüm projede farklı driver üretilmesinin onune geceriz
- Sonra TestBasede yaptığımız gibi "wait" koyabiliriz.
- Ardından driver'i kapatmak için method oluşturuyoruz.
- Şu andan itibaren TestBase Classını değil Driver Classını kullanacağız.

```
public class Driver {  
    private Driver(){  
    }  
    private static WebDriver driver;  
    public static WebDriver getDriver() {  
        if (driver == null) {  
            switch (ConfigReader.getProperty("browser")) {  
                case "chrome":  
                    WebDriverManager.chromedriver().setup();  
                    driver = new ChromeDriver();  
                    break;  
                case "firefox":  
                    WebDriverManager.firefoxdriver().setup();  
                    driver = new FirefoxDriver();  
                    break;  
                case "safari":  
                    WebDriverManager.getInstance(SafariDriver.class);  
                    driver = new SafariDriver();  
                    break;  
                case "opera":  
                    OperaDriverManager.operadriver().setup();  
                    driver = new OperaDriver();  
                    break;  
            }  
        }  
        driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);  
        driver.manage().window().maximize();  
        return driver;  
    }  
    public static void closeDriver(){  
        if (driver !=null){  
            driver.close();  
            driver=null;  
        }  
    }  
}
```



Test NG

Smoke Test

Smoke Test: Kullandigimiz uygulamanin onemli/temel fonksiyonlarini test etmek icin yapilir. Genellikle sabah ilk ise baslama gorevimizdir. Login,logout,sepete ekle,odeme yap.. gibi temel islevleri test ederiz. Eger smoke test FAILED olursa zaman gecirmeksizin tum ekibi haberدار ederiz.

<http://qa-environment.koalaresorthotels.com>
sayfasinda genel testlerimizi yapacagiz

Sistemin saglikli calistigini anlamak icin 3 test yapacagiz
1- Pozitif Login Test
2- Negatif Login Test
3- E2E Testi (system testi)



End to End Test(E2E): Bir uygulamanin tum adimlarini bastan sona kadar test etmek demektir. Ornegin biz yonetici bir oda olusturabilir mi diye test yaptigimizda sisteme giristen oda olusturuldu yazisi gorulunceye kadar tum adimlari test etmis oluruz, dolayisiyla E2E testi yapmis oluruz. Bu testin diger adi da **Sistem Testi**'dir.



Smoke Test

- 1) com.techproed altinda bir package olustur : smoketest
- 2) Bir Class olustur : PositiveTest
- 3) Bir test method olustur positiveLoginTest()

<https://qa-environment.concorthotel.com/> adresine git

login butonuna bas

test data username: **manager** ,

test data password : **Manager1!**

Degerleri girildiginde sayfaya basarili sekilde girilebildigini test et



Smoke Test

1) smokeTest paketi altinda yeni bir Class olustur: NegativeTest

3 Farkli test Methodunda 3 senaryoyu test et

- yanlisSifre
- yanlisKullanici
- yanlisSifreKullanici

test data yanlis username: **manager1** , yanlis password : **manager1**

2) <https://qa-environment.concorthotel.com/> adresine git

3) Login butonuna bas

4) Verilen senaryolar ile giris yapilamadigini test et



1. Tests packagenin altına class olusturun: **D17_CreateHotel**
2. Bir metod olusturun: `createHotel`
3. <https://qa-environment.concorthotel.com/> adresine git.
4. Username textbox ve password metin kutularını locate edin ve asagidaki verileri girin.
 - a. Username : **manager**
 - b. Password : **Manager1!**
5. Login butonuna tıklayın.
6. Hotel Management/Hotel List menusunden ADD HOTEL butonuna tiklayın
7. Açılan sayfadaki tüm metin kutularına istediğiniz verileri girin.
8. Save butonuna tıklayın.
9. “Hotel was inserted successfully” textinin göründüğünü test edin.
10. OK butonuna tıklayın.



1- Page Object Model : Testlerimizi daha kolay ve düzenli olarak hazırlamamız ve çalıştırırmamız için oluşturulan bir modeldir.

Framework için üretilmiş benzer modeller olmakla birlikte en güncel olan ve çok kullanılan model olduğu için POM'i öğrendik

2- POM dosya yapısı :

-Pages : Test yapacağımız web page'ler için Pages package'in altında bir class oluşturuyoruz. Bu class'larda mutlaka yapmamız gereken şey driver'i oluşturduğumuz Driver clasından alıp PageFactory.initElements ile ilk değer ataması yapmaktadır. Sonrasında web sayfamızda kullanacağımız WebElementlerin tamamını public olarak oluşturmak ve **@FindBy** notasyonu ile locate etmektedir. Eğer istersek login gibi bazı adımları yapacak metodları da bu class'da oluşturabiliriz.

Test clasımızdan Page sayfasındaki variable ve method'lara obje oluşturup erişim sağlarız.



- **configuration.properties** : Bu dosyayı testlerimizde kullanacağımız url,test dataları gibi kullanıcının aldigımız dataları dinamik yapmak için kullanırız.

Tüm testlerimizi bu sayfadan alacağımız datalara göre dizayn ederiz. Böylece bu dosyada yapacağımız bir değer değişikliği ile tüm testCase'lerindeki test datalarını güncelleyebiliriz.

Bu sayfayı basit bir text dosyası gibi dizayn ederiz her test datasını key=value şeklinde key,value ile oluştururuz.

- **ConfigReader** : Bu class test clasımız ile configuration.properties dosyası arasında tercumanlık yapar. İçinde .properties uzantılı dosyaları okumak için gerekli bir static blok oluştururuz. Ayrıca Test classlarımızdan çağrılmak için getProperty() methodunu oluştururuz. Bu method test class'ından gönderdiğimiz key değerini static blok yardımı ile configuration.properties'de bulup karşısındaki value'yu bize dondurur.



-Driver : Test clasimizda ve page clasinda kullanacagimiz driver'i olusturdugumuz class'tir. Utilities Package'i altında olustururuz. Driver class'ini **Singleton** yapabiliriz.

Driver'i static olarak olusturur ve olusturdugumuz **getDriver() method** icinde driver'imiza deger atamasi yapariz.

Is hayatinda karsilasacagimiz farkli browser'lar (chrome,firefox,safari vb..) deger atama islemi yapmadan once kullanicinin tercihini aliriz.

Kullanici tercihini almak icin configuration.properties dosyasinda browser=chrome gibi bir key,value ikilisi olusturur buradaki tercihe gore driver'a deger atamak icin de switch case kullaniriz.

Ayrica her driver cagirdigimizda yeni driver olusturmaması icin once if ile driver'in atamasi yapılmış mı control ederiz, atama yapılmışsa aynı driver ile devam eder, atama yapılmamışsa yeni bir driver olusturur ve deger atayip test sayfasına doneriz.

Bu Class'ta ayrica window.manage ayarlarini da yapar, en sonda da closeDriver method ile driver'i kapatma islemine de yardimci oluruz



1. Tests packagenin altına class olusturun: D18_HotelRoomCreation
2. Bir metod olusturun: RoomCreateTest()
3. <https://qa-environment.concorthotel.com/> adresine gidin.
4. Username textbox ve password metin kutularını locate edin ve aşağıdaki verileri girin.
 - a. Username : **manager**
 - b. Password : **Manager1!**
5. Login butonuna tıklayın.
6. Hotel Management menusunden Add Hotelroom butonuna tıklayın.
7. Açılan sayfadaki tüm metin kutularına istediğiniz verileri girin.
8. Save butonuna basın.
9. "HotelRoom was inserted successfully" textinin göründüğünü test edin.
10. OK butonuna tıklayın.
11. Hotel rooms linkine tıklayın.
12. "LIST OF HOTELROOMS" textinin göründüğünü doğrulayın..



Test NG

Web Tables

Bir Web sayfasında webelementleri düzenlemek için HTML tablo oluşturabiliriz

HTML tablo farklı sekillerde oluşturulabilir ama genellikle `<table>`, `<thead>`, `<tbody>`, `<tr>`, `<th>` ve `<td>` attribute'leri kullanılır.

Amazon Music Stream millions of songs	Amazon Advertising Find, attract, and engage customers	Amazon Drive Cloud storage from Amazon	6pm Score deals on fashion brands
Sell on Amazon Start a Selling Account	Amazon Business Everything For Your Business	AmazonGlobal Ship Orders Internationally	Home Services Experienced Pros Happiness Guarantee
Audible Listen to Books & Original Audio Performances	Book Depository Books With Free Delivery Worldwide	Box Office Mojo Find Movie Box Office Data	ComiXology Thousands of Digital Comics
Goodreads Book reviews & recommendations	IMDb Movies, TV & Celebrities	IMDbPro Get Info Entertainment Professionals Need	Kindle Direct Publishing Indie Digital & Print Publishing Made Easy
Zappos Shoes & Clothing	Ring Smart Home Security Systems	eero WiFi Stream 4K Video in Every Room	Neighbors App Real-Time Crime & Safety Alerts

```
<table> ==>>>table
  <thead> ==>>>header
    <tr> ==>>>table rows
      <th> ==>>>table header dat
      </tr>
    </thead>
    <tbody> ==>>>table body
      <tr> ==>>>table row
        <td> </td> ==>>>table data-cell
        </tr>
      </tbody>
</table>
```



Test NG

Web Tables

<https://qa-environment.concorthotel.com>

ID	UserName	Email	Name Surname	BirthDate
1				<input type="button" value=""/>
2	manager	manager@st.com	manager	22.02.2020
3	customerservice	customerservice@st.com	customerservice	14.01.2020
154	123456	kristalhotel@gmail.com	cevdet oz	12.12.2020

table

thead

tbody

data

row

```
<table class="table table-striped table-bordered table">
  <thead>...</thead>
  <tbody>
    <tr role="row" class="odd"> ← row
      <td>2</td>
      <td class="sorting_1">manager</td> ← red box
      <td>manager@st.com</td>
      <td>manager</td>
      <td>22.02.2020</td>
      <td>(545) 345-3453</td>
      <td>Manager</td>
    </tr>
    <tr role="row" class="even"> ← row
      <td>3</td>
      <td class="sorting_1">customerservice</td> ← red box
      <td>customerservice@st.com</td>
      <td>customerservice</td>
      <td>14.01.2020</td>
      <td>(543) 545-4354</td>
      <td>CustomerService</td>
    </tr>
    <tr role="row" class="odd">...</tr> ← row
    <tr role="row" class="even">...</tr> ← row
    <tr role="row" class="odd">...</tr> ← row
```



Test NG

Web Tables Class Work

- Bir class oluşturun : C02_WebTables
- login() metodun oluşturun ve oturum açın.
- <https://qa-environment.concorthotel.com/admin/HotelRoomAdmin> adresine gidin
 - Username : manager
 - Password : Manager1!
- table() metodunu oluşturun
 - Tüm table body'sinin boyutunu(sutun sayısı) bulun. /tbody
 - Table'daki tüm body'l ve başlıklarını(headers) konsolda yazdırın.
- printRows() metodunu oluşturun //tr
 - table body'sinde bulunan toplam satır(row) sayısını bulun.
 - Table body'sinde bulunan satırları(rows) konsolda yazdırın.
 - 4.satırındaki(row) elementleri konsolda yazdırın.



Test NG

Web Tables Class Work

- Bir class oluşturun : D18_WebTables
- login() metodun oluşturun ve oturum açın.
- <https://qa-environment.concorthotel.com/admin/HotelRoomAdmin> adresine gidin
 - Username : manager ○ Password : Manager2!
- table() metodu oluşturun
 - Tüm table body'sinin boyutunu(sutun sayısı) bulun. /tbody
 - Table'daki tüm body'i ve başlıklarını(headers) konsolda yazdırın.
- printRows() metodu oluşturun //tr
 - table body'sinde bulunan toplam satır(row) sayısını bulun.
 - Table body'sinde bulunan satırları(rows) konsolda yazdırın.
 - 4.satırındaki(row) elementleri konsolda yazdırın.
- printCells() metodu oluşturun //td
 - table body'sinde bulunan toplam hücre(cell) sayısını bulun.
 - Table body'sinde bulunan hücreleri(cells) konsolda yazdırın.
- printColumns() metodu oluşturun
 - table body'sinde bulunan toplam sutun(column) sayısını bulun.
 - Table body'sinde bulunan sutunları(column) konsolda yazdırın.
 - 5.column daki elementleri konsolda yazdırın.



WebTables class'ini kullanın.

1. Bir metod oluşturun : `printData(int row, int column);`
 - a. Satır(row) ve sütun(column) numarasını girdiğinizde, `printData` metodu bu hücredeki(cell) veriyi yazdırmalıdır.
2. Baska bir Test metodu oluşturun: `printDataTest();`
 - a. Ve bu metodу `printData()` methodunu cagirmak icin kullanın.
 - b. Örnek: `printData (3,5); => 3. satır, 5. Sütundaki veriyi yazdırmalıdır`
 - c. yazdirilan datanin olmasi gereken dataya esit oldugunu test edin



Test NG

Web Tables Homework

Bir Class olusturun D19_WebtablesHomework

1. "https://demoqa.com/webtables" sayfasina gidin
2. Headers da bulunan department isimlerini yazdirin
3. sutunun basligini yazdirin
4. Tablodaki tum datalari yazdirin
5. Tabloda kac cell (data) oldugunu yazdirin
6. Tablodaki satir sayisini yazdirin
7. Tablodaki sutun sayisini yazdirin
8. Tablodaki 3.kolonu yazdirin
9. Tabloda "First Name" i Kierra olan kisinin Salary'sini yazdirin
10. Page sayfasinda bir method olusturun, Test sayfasindan satir ve sutun sayisini girdigimde bana datayi yazdirlsin



Test NG

Excel'in Yapısı

- Excel için daha önce incelediğimiz Web Table yapısına benzer bir yapı vardır.
- Excel'de bir hücredeki bilgiye ulaşmak için dosya/sayfa/satır/sütun sırasıyla ilerlemeliyiz
- Excel ile ilgili otomasyonda web table'da olduğu gibi sütun yapısı yoktur, ihtiyaç duyarsak kodla sütunu elde edebiliriz ancak bir dataya ulaşmak için sütun kullanamayız

Workbook excel dosyamız

Sheet Her açık excel sekmesi (Sheet1, Sheet2, etc)

Row(satır) Java, yalnızca içindeki veri varsa satırları sayar.
Default olarak, Java perspektifinden satır sayısı 0'dır

Cells (hücre) Java her satıra bakar ve yalnızca hücrede veri varsa hücre sayısını sayar.

	A	B	C	D
1	Ülke (İngilizce)	Başkent (İngilizce)	Ülke (Türkçe)	Başkent (Türkçe)
2	Afghanistan	Kabul	Afganistan	Kabil
3	Albania	Tirana	Arnavutluk	Tiran
4	Algeria	Aljiers	Cezayir	Cezayir
5	Andorra	Andorra la Vella	Andorra	Andorra la Vella
6	Angola	Luanda	Angola	Luanda
7	Antigua & Barbuda	Saint John's	Antigua ve Barbuda	Saint John's
8	Argentina	Buenos Aires	Arjantin	Buenos Aires
9	Armenia	Yerevan	Ermenistan	Erivan
10	Australia	Canberra	Australya	Canberra
11	Austria	Vienna	Avusturya	Viyana
12	Azerbaijan	Baku	Azerbaycan	Bakü
13	The Bahamas	Nassau	Bahamalar	Nassau
14	Bahrain	Manama	Bahreyn	Manama
15	Bangladesh	Dhaka	Bangladeş	Dakka
16	Barbados	Bridgetown	Barbados	Bridgetown
17	Belarus	Minsk	Belarus	Minsk
18	Belgium	Brussels	Belçika	Brüksel
19	Belize	Belmopan	Belize	Belmopan
20	Benin	Porto Novo	Benin	Porto Novo
21	Bhutan	Thimphu	Butan	Thimphu
22	Bolivia	La Paz	Bolivya	La Paz
23	Bosnia & Herzegovina	Sarajevo	Bosna-Hersek	Saraybosna
24	Botswana	Gaborone	Botswana	Gaboron
25	Brazil	Brasilia	Brezilya	Brasilia
26	Brunei	Bandar Seri Begawan	Brunei	Bandar Seri Begawan
27	Bulgaria	Sofia	Bulgaristan	Sofya



- Apache POI, microsoft ofis dokumanlarına erişmek için kullanılan Java API'idir.
- Poi.apache.com sayfasından official dokumanlar incelenebilir.
- Excel kullanmak için;

```
<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi</artifactId>
    <version>4.1.2</version>
</dependency>
```



Test NG

Read Excel Class Work

1. apache poi dependency'i pom file'a ekleyelim
2. Java klasoru altinda **resources** klasoru olusturalim
3. Excel dosyamizi resources klasorune ekleyelim
4. **excelAutomation** isminde bir package olusturalim
5. ReadExcel isminde bir class olusturalim
6. readExcel() method olusturalim
7. Dosya yolunu bir String degiskene atayalim
8. FileInputStream objesi olusturup,parametre olarak dosya yolunu girelim
9. Workbook objesi olusturalim,parameter olarak inputStream objesini girelim
10. **WorkbookFactory.create(inputStream)**
11. Worksheet objesi olusturun **workbook.getSheetAt(index)**
12. Row objesi olusturun **sheet.getRow(index)**
13. Cell objesi olusturun **row.getCell(index)**



Test NG

Read Excel Class Work

Yeni bir test method olusturalim readExcel2()

- 1.satirdaki 2.hucreye gidelim ve yazdiralim
- 1.satirdaki 2.hucreyi bir string degiskene atayalim ve yazdiralim
- 2.satir 4.cell'in afganistan'in baskenti oldugunu test edelim
- Satir sayisini bulalim
- Fiziki olarak kullanilan satir sayisini bulun
- Ingilizce Ulke isimleri ve baskentleri bir map olarak kaydedelim



Test NG

Write Excel Class Work

- 1) Yeni bir Class olusturalim WriteExcel
- 2) Yeni bir test method olusturalim writeExcelTest()
- 3) Adimlari takip ederek 1.satira kadar gidelim
- 4) 4.hucreye yeni bir cell olusturalim
- 5) Olusturdugumuz hucreye "Nufus" yazdiralim
- 6) 2.satir nufus kolonuna 1500000 yazdiralim
- 7) 10.satir nufus kolonuna 250000 yazdiralim
- 8) 15.satir nufus kolonuna 54000 yazdiralim
- 9) Dosyayı kaydedelim
- 10) Dosyayı kapatelim



XML File Oluşturma

XML, hem insanların hem de makinelerin okuyabileceği belgeleri kodlamak için bir sözdizimi tanımlamak üzere World Wide Web Consortium (W3C) tarafından oluşturulan bir biçimlendirme dilidir.

Veri saklamak ve farklı işletim sistemleri arasında veri transfer etmek için kullanılan metin işaretleme dili XML ile hazırlanmış dökümanlar .xml formatına sahip dosyalarda saklanır.

Biz de framework'umuzdeki belirli testleri veya tüm testleri otomatik olarak çalıştırmak için xml dosyaları kullanırız

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >

<suite name="Suite1" verbose="1" >
  <test name="Nopackage" >
    <classes>
      <class name="NoPackageTest" />
    </classes>
  </test>

  <test name="Regression1">
    <classes>
      <class name="test.sample.ParameterSample"/>
      <class name="test.sample.ParameterTest"/>
    </classes>
  </test>
</suite>
```



Test NG

XML File Oluşturma

- Testng framework'de xml dosyasi kullanma nedenlerinden biri, belirli suitleri, testleri, package lari, classları veya method lari çalıştırmaktadır.
- Belirli testleri, package lari, classları veya method'lari dahil edebilir (**include**) veya hariç (**exclude**) tutabiliriz. Bu da bize esneklik (**flexiblity**) kazandırır.
- Sadece birkaç basit yapılandırma ile TestNG.xml dosyasını kullanarak belirli test senaryolarını yürütebiliriz.
- Daha fazla için: <https://testng.org/doc/documentation-main.html>

The screenshot shows a file browser window with the following structure:

- Ornek-POM-Proje [com.POMSample] C:\Users\L...
- .idea
- src
 - test
 - java
 - crossBrowser
 - pages
 - smokeTest
 - tests
 - D22_IlkClass
 - D22_ReusableMethodsWebList
 - D23_HomeworkAmazon
 - D23_HomeworkSatirSayisi
 - D29_DataProvider
 - utilities
 - ConfigReader
 - Driver
 - DriverCross
 - ExcelUtil
 - ReusableMethods
 - TestBaseCross
 - TestBaseRapor
 - target
 - test-output
 - amazon.xml
 - configuration.properties

Next to the file browser is the content of the amazon.xml file:

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="amazon">
  <test name="amazon butun testler">
    <classes>
      <class name="tests.D23_HomeworkAmazon">
        <methods>
          <include name="hucreTest"></include>
        </methods>
      </class>
      <class name="tests.D23_HomeworkSatirSayisi">
        <methods>
          <exclude name="sutunSayisi"></exclude>
        </methods>
      </class>
    </classes>
  </test>
</suite>
```

Two specific lines in the XML code are highlighted with red boxes:
 - The `<include name="hucreTest"></include>` line under the first class definition.
 - The `<exclude name="sutunSayisi"></exclude>` line under the second class definition.



Test NG

Belirli Test'ler Nasıl Çalıştırılır?

- XML dosya olustururken hiyerarsi (buyukten kucuge siralama) onemlidir. Her zaman suite ile baslayip hangi seviyede test calistirmak istersek o seviyeye kadar sirali olarak kademeleri yazmaliyiz

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="coklu class calistirma">
  <test name="coklu calistirma testi">
    <classes>
      <class name="tests.D23_HomeworkSatirSayisi"></class>
      <class name="tests.D23_HomeworkAmazon"></class>
    </classes>
  </test>
</suite>
```

- Eger calistiracagimiz class'lar farkli hiyerarsilere ait ise yine suite ile baslariz, sonra ayrisma kadamesinden itibaren farkli hiyerarsi kumeleri olustururuz.

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="sirali method calistirma">
  <test name="sirali method">
    <classes>
      <class name="tests.D23_HomeworkAmazon">
        <methods>
          <include name="AmazonYazisi"></include>
        </methods>
      </class>
      <class name="tests.D23_HomeworkSatirSayisi">
        <methods>
          <exclude name="sutunSayisi"></exclude>
        </methods>
      </class>
    </classes>
  </test>
</suite>
```



Test NG

XML File Class Work

Istenen Package'lari Calistirma

- Yeni bir xml dosyasi olusturalim :
belirliPackageCalistirma
- Smoke Test package'indaki tum testleri calistiralim

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="smoke package">
    <test name="smoke package">
        <packages>
            <package name="smokeTest"></package>
        </packages>
    </test>
</suite>
```



XML File Class Work

Istenen Class'lari Calistirma

- Yeni bir xml dosyasi olusturalim :
belirliClasslariCalistirma
- <package> attribute yerine <classes> ve sonra <class> attribute kullanarak istediginiz class'lari calistirin

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="coklu class calistirma">
    <test name="coklu calistirma testi">
        <classes>
            <class name="tests.D23_HomeworkSatirSayisi"></class>
            <class name="tests.D23_HomeworkAmazon"></class>
        </classes>
    </test>
</suite>
```



XML File Class Work

Istenen Method'lari Calistirma

- Yeni bir xml dosyasi olusturalim : belirliMethodlariCalistirma
- <classes> attribute altinda <class> ve <methods> attribute'lerini ve icinde <include>, <exclude> attribute'lerini kullanalim

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="sirali method calistirma">
    <test name="sirali method">
        <classes>
            <class name="tests.D23_HomeworkAmazon">
                <methods>
                    <include name="AmazonYazisi"></include>
                </methods>
            </class>
            <class name="tests.D23_HomeworkSatirSayisi">
                <methods>
                    <exclude name="sutunSayisi"></exclude>
                </methods>
            </class>
        </classes>
    </test>
</suite>
```



XML File Class Work

Istenen Gruplari Calistirma

- Yeni bir xml dosyasi olusturalim : belirliGruplariCalistirma
- Group calistirmak icin hem grup adini tanimlamak hem de nerede arayacagimizi belirtmek zorundayiz

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="minor regression" verbose="2">
<test name="Regression1">
  <groups>
    <run>
      <include name="Regression1" />
    </run>
  </groups>
  <packages>
    <package name="com.techproed.smokeTest"></package>
  </packages>
</test>
</suite>
```



Test NG

XML File Class Work

Tum Testleri Calistirma

- Yeni bir xml dosyasi olusturalim : tumTestleriCalistirma

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="minor regression" verbose="2">
    <test name="Regression1">
        <packages>
            <package name=".*"></package>
        </packages>
    </test>
</suite>
```



Test NG

Html Reports

TESTNG rapor hazırlamaz. Eğer testimiz ile ilgili rapor hazırlamak istersek, farklı kütüphanelerden yardım almamız gereklidir.

Pom.xml dosyamıza aventstack dependency'sini ekliyoruz.

```
<!-- https://mvnrepository.com/artifact/com.aventstack/extentreports -->
<dependency>
    <groupId>com.aventstack</groupId>
    <artifactId>extentreports</artifactId>
    <version>5.0.1</version>
</dependency>
```





Extent Reports :

HTML raporlama aracıdır. Bize Html raporları verir. Test adımlarını kaydetmemize yardımcı olur. Ayrıca ekran görüntüleri ekleyebiliriz.

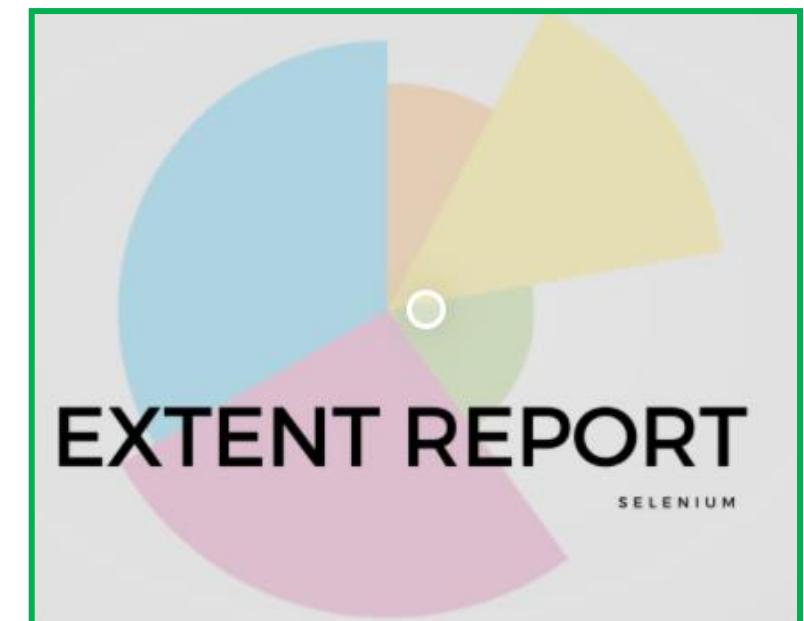
3 tane obje oluşturup kullanırız

1. ExtentReports extendReports; Raporlamayı başlatmak için ExtentReports'a ihtiyacımız var. flush() metodunu için ExtentReports kullanıyoruz.

2. ExtentHtmlReporter extentHtmlReporter; Bu, özel raporlara ve raporların yapılandırmasına sahip olmamıza yardımcı olur, html raporlarını oluşturur. Bunu raporun oluşturulacağı yolu ayarlamak için de kullanıyoruz.

3. ExtentTest extentTest; Bilgi eklemek için. Test adımlarını eklemek için (açıklama). Günlükleri(logs) ekliyoruz.

`extentTest.info ("URL'yi Açma");` bilgi sadece adım eklemek içindir





Html Reports

```
@BeforeTest      : burada rapor için nesne oluşturuyoruz, hazırlık yapıyoruz  
@Test           : raporu dolduruyoruz, içerisinde veriler ekliyoruz.  
@AfterMethod    : eğer @Test başarısızsa, rapora ekran görüntüsü ekliyoruz.  
@AfterTest       : raporlandırma işlemini sonlandırıyoruz.
```

Testimize rapor oluşturma adımları

1- Test class'ini extends ile TestBaseRapor Class'ına child yapalım

2- *extentTest=extentReports.createTest("Test ismi", "Tanim");* rapor oluşturalim

3- Gerekli/istedigimiz satirlara extentTest.info("Aciklama") ekleyelim

4- Assert olan satırda acıklamayı extentTest.pass ile yapabiliriz



Test NG

Html Reports

Testimiz bittikten sonra olusturulan raporu “open in browser” ile acabiliriz.

Eger test basarisiz ise Screenshots klasorunden resmine de ulasabiliriz

FHC Trip Automation Reports

Status ⚠ Dashboard ⌚ Search 🔍

Tests

2 test(s) passed
0 test(s) failed, 0 others

Manager Login Positive Test Pass
Feb 26, 2020 01:16:02 AM

Manager Login Positive Test Pass
Feb 26, 2020 01:16:02 AM Feb 26, 2020 01:16:07 AM 0h 0m 4s+193ms

Manager Login Positive Test Pass
Feb 26, 2020 01:16:07 AM

Status	Timestamp	Details
ⓘ	1:16:02 AM	Going to the url
ⓘ	1:16:03 AM	Logging in the application
ⓘ	1:16:07 AM	Verifying if the title is as expected
✓	1:16:07 AM	Completed the positive login Test

Elemental Selenium

A free, once-weekly e-mail on how to use Selenium like a Pro

Sponsored by [Sauce Labs](#)

Tip 4 - Working with Multiple Windows

The Problem

Occasionally you will run into a link or action on a site that opens a new window. In order to work with this new window, or in an ongoing one, you will need to switch between them.

On the face of it, this is pretty straightforward thing. But, lurking within it is a small pitfall to watch out for.

Let's say we have:

An Example

```
NOTE: You can find it in our example repository here
require selenium-webdriver
require 'selenium-examples'

def setup
  @driver = selenium-webdriver :browser => :firefox
end
```



Test NG

TestNG Ornek Proje Olusturma

1. File – New – Project e tikliyoruz
2. Maven'i seciyoruz
3. Name'e projemizin ismini yaziyoruz
4. Cikan Alert mesajinda New Window veya This Window secilebilir
5. Pom xml'imizi düzenliyoruz

a) <properties>
<maven.compiler.source>1.8</maven.compiler.source>
<maven.compiler.target>1.8</maven.compiler.target>

</properties>
*** bu kod Javanin sürümüyle alakali sorunları halletmeye yarıyor

b) <dependencies>
Kutuphanelerimizi bu tag'lar arasina yaziyoruz
</dependencies>



6) <https://mvnrepository.com/> a gidip kutuphanelerimizi tek tek aliyoruz

a) Selenium-Java Kutuphanesi

```
<!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.141.59</version>
</dependency>
```

b) WebDriverManager Kutuphanesi

```
<!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
<dependency>
    <groupId>io.github.bonigarcia</groupId>
    <artifactId>webdrivermanager</artifactId>
    <version>4.2.0</version>
</dependency>
```



Test NG

TestNG Ornek Proje Olusturma

c) Testng Kutuphanesi

```
<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.1.0</version>
    <scope>test</scope>
</dependency>
```

d) Java Faker Kutuphanesi

```
<!-- https://mvnrepository.com/artifact/com.github.javafaker/javafaker -->
<dependency>
    <groupId>com.github.javafaker</groupId>
    <artifactId>javafaker</artifactId>
    <version>1.0.2</version>
</dependency>
```



TestNG Ornek Proje Olusturma

- 7) Pom Dosyasini oluşturma işlemimiz bitti. Kutuphaneler kırmızı renkte olabilir. Sağ tarafta Maven yazan sekmede Reload oklarina tiklayip beklediğimiz de hata gitmiş oluyor
- 8) Kullanicinin gordugu arayuzde test ederiz. (UI)
Kullanacagimiz paketleri uygun isimlerde test-Java bolumun içerisinde oluşturuyoruz
- 9) Java ya sağ tiklariz new package - com (paketin ismi) yazariz
- 10) com package'ina sağ tiklariz - new package - techproed (paketin ismi) yazariz
- 11) artik projemiz com. techproed

Bu package'in altına frameworkumuzun package'larını yolosturuyoruz. Bunlar

- A-pages
- B-smokeTest
- C-tests
- D-utilities



TestNG Ornek Proje Olusturma

12) Resources paketi olusturma:

Java'ya sağ tıklıyoruz-new-package -->resources (yeni package)

Bu resources paketinin altına dokumanlarımızı copy-paste ederiz

13) configuration.properties dosyasi olusturma

En yukarıda Projemize sağ tıklıyoruz new- File ' a tıklıyoruz

İsmi önemli değil ama uzantısı MUTLAKA .properties olmalı

İsmi configuration.properties yazıyoruz

Bu dosyanın içine Data'larımızı key=value şeklinde yazıyoruz

kr_url=<http://qa-environment.koalaresorthotels.com>

kr_valid_username=manager kr_valid_password=Manager1!

14) ConfigReader Class'l olusturma

utilities package inin altında ConfigReader Classı oluşturuyoruz. Bu class configuration.properties deki dosyalarımızı okumak için bir aracı



TestNG Ornek Proje Olusturma

15) ConfigReader Classinda :

1-ilk yapacagimiz sey Instance olarak Properties objesi olusturmak. Bu objeyi static blok icinde kullanacagimdan static yapmam gerek

Bu objeyi sadece bu class ta kullanacagim icin private yapmamiz önerilir

2-Properties objesini kullanmak üzere bir static blok kurmalıyız. neden static? Cunku her zaman ilk static block calisir

```
public class ConfigReader {  
  
    private static Properties properties;  
  
    static {  
        String path="configuration.properties";  
        try {  
            FileInputStream fileInputStream=new FileInputStream(path);  
            properties=new Properties();  
            properties.load(fileInputStream);  
  
            fileInputStream.close();  
        } catch (FileNotFoundException e) {  
            e.printStackTrace();  
        } catch (IOException e) {  
            e.printStackTrace();  
        }  
    }  
    public static String getProperty(String key){  
  
        return properties.getProperty(key);  
    }  
}
```



Test NG

TestNG Ornek Proje Olusturma

16) Driver class'ini düzenleniyoruz

Singleton class : object olusturulmasi

kontrol altina alınan (genelde izin verilmeyen) classdir. Bunun icin baska classlarda Driver clasindan obje uretmemizi saglayan default constructor'i gorunur sekilde yazip access modifier'i private yapariz

Bu class'da test class'larimizda kullanacagimiz driver'i olusturacak ve kapatacak getDriver() ve closeDriver() methodlarini olusturuyoruz

Bu methodlari static yaparak obje olusturmadan Class adi ile cagirmak icin kullanisli hale getiriyoruz

```
public class Driver {  
    private Driver(){  
    }  
    static private WebDriver driver;  
    static public WebDriver getDriver(){  
  
        if(driver==null){  
            switch (ConfigReader.getProperty("browser")){  
                case "chrome":  
                    WebDriverManager.chromedriver().setup();  
                    driver=new ChromeDriver();  
                    break;  
                case "firefox":  
                    WebDriverManager.firefoxdriver().setup();  
                    driver=new FirefoxDriver();  
                    break;  
                case "safari":  
                    WebDriverManager.getInstance(SafariDriver.class);  
                    driver=new SafariDriver();  
                    break;  
                case "opera":  
                    OperaDriverManager.operadriver().setup();  
                    driver=new OperaDriver();  
                    break;  
            }  
            driver.manage().window().maximize();  
            driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
        }  
        return driver;  
    }  
  
    static public void closeDriver(){  
  
        if (driver != null){  
            driver.close();  
            driver=null;  
        }  
    }  
}
```



TestNG Ornek Proje Olusturma

18) pages package inin altinda kullanacagimiz her websayfasi icin bir page Class'l olustururuz

a) Bu class'da ilk yapmamiz gereken test class'larinda bu class'dan obje uretebilmemiz icin gerekli olan Constructor'i olusturmaktir.

```
public CkHotelsHomePage(){  
    PageFactory.initElements(Driver.getDriver(), page: this);  
}
```

b) Ardinda Locate islemlerimizin tamamini yaziyoruz bu sayfaya

```
@FindBy(linkText="Log in")  
public WebElement ilkLogIn;  
  
@FindBy(id="UserName")  
public WebElement userNameTextBox;  
  
@FindBy(id="Password")  
public WebElement passwordTextBox;  
  
@FindBy(id="btnSubmit")  
public WebElement loginButonu;
```



Soru 1 :

- Amazon anasayfaya gidebilecek sekilde bir page sayfasi olusturun : AmazonPage
- Amazon ana sayfasinda en alta bulunan Webtable'i inceleyebilmek icin AmazonPage clasinda en alta gitme isini yapacak bir method olusturun
- Tests paketi altında yeni bir class olusturun: D26_AmazonSatirSutunSayisi
- Bu class'in altında bir test method olusturun : satirSayisi() ve webtable'da 10 satir oldugunu test edin
- Yeni bir method olusturun : sutunSayisi() ve yazi olan sutun sayisinin 7oldugunu test edin



Soru 2 :

- AmazonPage sayfasında istedigim satır ve sutun sayısı ile çağrıdigimda bana hcredeki yazıyı getirecek bir method oluşturun
- Tests paketi altında yeni bir class oluşturun: D26_AmazonHucreTesti
- Bu class'in altında bir test method oluşturun : hucretesti() ve webtable'da 3. satır 2.sutundaki yazının "Home Services" yazısı içerdigini test edin
- Yeni bir method oluşturun : AmazonYazisi() ve tabloda 9 Hucrede "Amazon" yazısı bulundugunu test edin



Soru 3 :

- Amazon uzerine yapılan 4 testi otomatik olarak calistiracak xml kodunu yazin ve calistirin

- D26_AmazonSatirSutunSayisi class'indan satirSayisi() testini ve D26_AmazonHucreTesti class'indan hucretesti() testini calistiracak xml kodunu yazin ve calistirin



Soru 4 :

- D26_AmazonSatirSutunSayisi class'indan satirSayisi() testini ve D26_AmazonHucreTesti class'indan hucretesti() testini rapor alacak sekilde hazirlayin ve 3.sorudaki xml dosyasi ile calistirip raporu olusturun



Web Tables Homework

- Cucumber bizim son framework'umuz olacak.
- Cucumber **BDD** (behaviour driven development) (Davranış tabanlı geliştirme) yaklaşımı için kullanılmakta olan açık kaynak kodlu bir kütüphanedir.
- Cucumber bir iş ararken önemli bir rol alacaktır.
- TestNg hakkında her şeyi bilmemek sorun değil ama Cucumber hakkında her şeyi bilmeniz GEREKİR.
- Agile methodolojisinde, insanlar uygulamanın işlevsellliğini geliştirmek için birlikte çalışmak zorundadır. İnsanlar development sırasında birlikte hareket etmeli.

Feature: US1001 Ck Hotels Login

@wip

Scenario: TC01 kullanıcı gecerli bilgilerle giriş yapar

Given kullanıcı Ck Hotels ana sayfasında

Then Log in yazısına tıklar

And gecerli username girer

And gecerli password girer

And Login butonuna basar

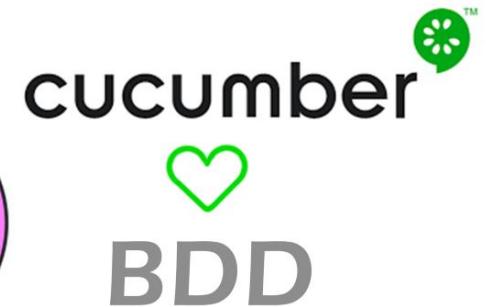
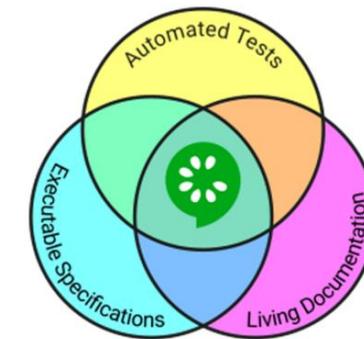
Then sayfaya giriş yaptığını kontrol eder

And kullanıcı sayfayı kapatır



Web Tables Homework

- BDD(behaviour driven development) (Davranış güdümlü geliştirme)- ilk olarak behavior(davranış) veya functionalitileri yazıyorsunuz (Epic=Feature, Story, AC, etc), daha sonra development and testing baslıyor.
- BDD'de behaviour'lar başarısız olduğunda kod başarısız olur..
- Anlasılabilir Gherkin Language nedeniyle BDD development için Cucumber harika bir uygulamadır.
- **Gherkin:** Projede her bir davranış için .feature uzantılı bir Gherkin dosyası oluşturulur. Bu feature dosyasına ilgili özelliğin farklı durumlardaki davranışları tanımlanır.



Given anahtar kelimesi ile ön koşul yani başlangıç durumu tanımlanır,

When, And anahtar kelimeleri ile olayı

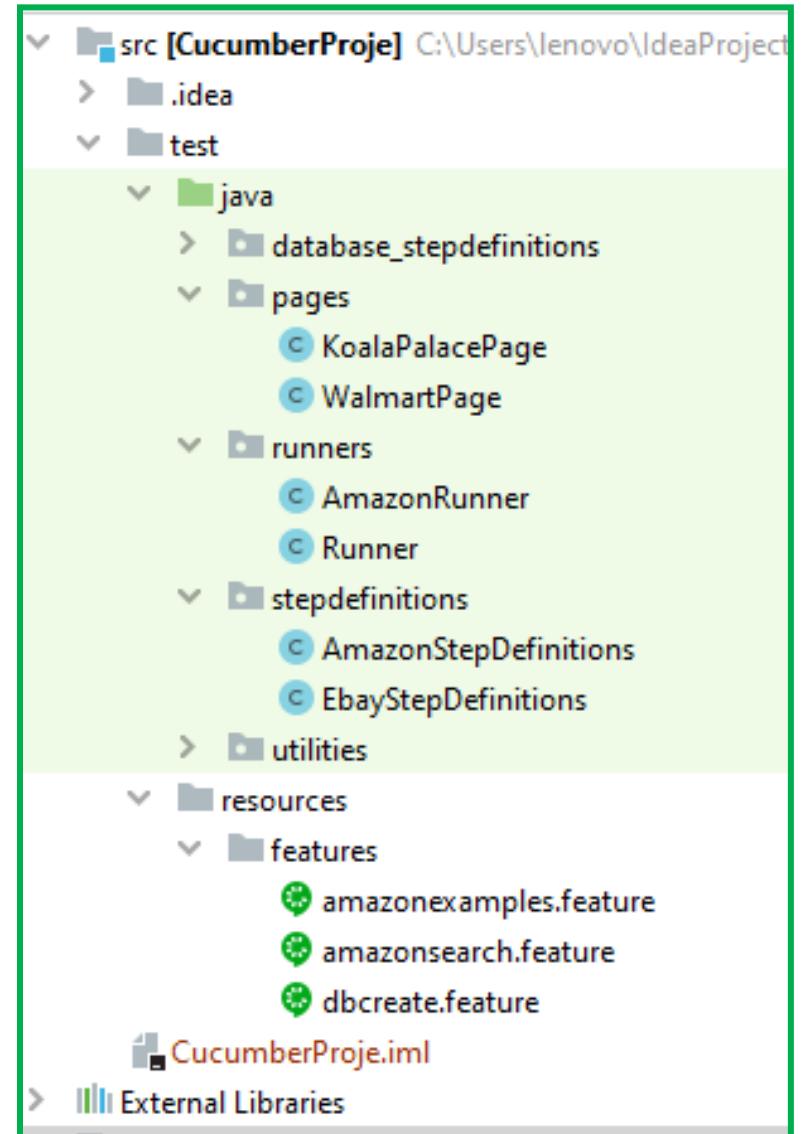
Then anahtar kelimesi ile de sonuç tanımlanır.

Given-When-And-Then adımları sadece okuyanlar için farkeder Java için hepsi birdir.



cucumber

- Cucumber (TDD)(Test Driven Development) test odaklı geliştirmeye izin verir, çünkü Cucumber ile Junit veya TestNG kullanabiliriz.
- Cucumber iş için önemlidir, çünkü **anlaşılabilir ve harika raporlara** sahiptir.
- Cucumber, teknik olmayan (none-technical) kişileri ve teknik kişileri birbirine bağlar.
- Developer veya team lead gibi teknik elemanlar da bazen testerların yaptığını anlayamayabilir. Gherkin onların da testlerimizi anlamalarını kolaylaştırır





Proje Oluşturma

1. **Create Project:** File -> New -> Project-> Select maven -> click next
2. Name: batch44Cucumber->finish
3. Add Dependencies =>Selenium-java, webdrivermanager, cucumber java, cucumber junit
4. Click Maven => click “Enable auto-reload after any changes” (Reload)
5. Javanın sürümüyle alakalı sorunları halletmek için compiler dependency yüklenebilir

<properties>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>1.8</maven.compiler.target>

</properties>



Proje Olusturma

6. Java'ya sag click yapip asagidaki paketleri olusturalim

- a. utilities
- b. pages
- c. runners (test case'leri calistirmak ve control etmek icin kullanacagiz)
- d. stepdefinitions (kodlarimizi burada olusturacagiz)

7- Utilities paketi altinda **Driver** ve **ConfigReader** Class'larini olusturalim

8- Projeye sag click yapip **configuration.properties** dosyasi olusturalim

9- test paketi altinda yeni bir klasor olusturalim : **resources**

10- resources klasoru altinda yeni bir klasor olusturalim : **features** (Java kodu icermeyen dosyalari buraya koyacagiz)

11- features'a sag clik yapip dosya olusturalim amazonsearch.feature

12- cucumber for Java plugin'i intelliJ'e ekleyelim (settings/Plugins)

MAC => IntelliJ Idea->Preference->Plugins->Marketplace->Type Cucumber for Java->Install->Restart



Class Work : First Cucumber Test Case

Yeni bir feature file olusturalim : amazonsearch.feature

Given kullanici amazon sayfasina gider

And iPhone icin arama yapar

Then sonuclarin Iphone icerdigini test eder

Given kullanici amazon sayfasina gider

And tea pot icin arama yapar

Then sonuclarin tea pot icerdigini test eder

Given kullanici amazon sayfasina gider

And flower icin arama yapar

Then sonuclarin flower icerdigini test eder



Class Work : First Cucumber Test Case

Feature File: Yeni bir feature file olusturalim : **amazonsearch.feature** , Test Case'i Gherken language kullanarak yazalim

stepdefinitions package: FirstFeatureFileStepDefinitions Class'ini olusturalim

runner package :

```
Runner class'i olusturalim (Runner class'i bir kez olusturuyoruz)
@RunWith(Cucumber.class)
@CucumberOptions(
    features="features folder path"
    glue="stepdefinitions folder path"
    tags="@istediginiz tag"
    dryRun=false)
```

Runner class'i calistirip step definitions'i olusturun ve iclerine kodlari yazin



Background

Feature: US1001 amazon arama

@amazon @nutella

Scenario: TC01 amazon nutella arama

```
When kullanici amazon sayfasina gider  
And nutella icin arama yapar  
Then sonucun nutella icerdigini test eder  
And sayfayı kapatır
```

@amazon @java

Scenario: TC02 amazon java arama

```
When kullanici amazon sayfasina gider  
And java icin arama yapar  
Then sonucun java icerdigini test eder  
And sayfayı kapatır
```

@amazon @ipad

Scenario: TC03 amazon ipad arama

```
When kullanici amazon sayfasina gider  
And ipad icin arama yapar  
Then sonucun ipad icerdigini test eder  
And sayfayı kapatır
```

Farklı senaryoların başında ortak adımlarımız varsa:

1. Feature file in basına Background oluşturun.
2. Bu ortak adımları Background altına yazın.
3. Background, Feature file'daki her Scenario'dan önce çalışır
4. Duplication olmadığından emin olun.
Background un altındaki adımı yazdıktan sonra senaryolardan silin.

Feature: US1002 amazon background ile arama

Background: amazon sayfasina gitme

When kullanici amazon sayfasina gider

Scenario: TC04 amazon nutella arama

```
And nutella icin arama yapar  
Then sonucun nutella icerdigini test eder  
And sayfayı kapatır
```

@wip

Scenario: TC05 amazon java arama

```
And java icin arama yapar  
Then sonucun java icerdigini test eder  
And sayfayı kapatır
```

Scenario: TC06 amazon ipad arama

```
And ipad icin arama yapar  
Then sonucun ipad icerdigini test eder  
And sayfayı kapatır
```

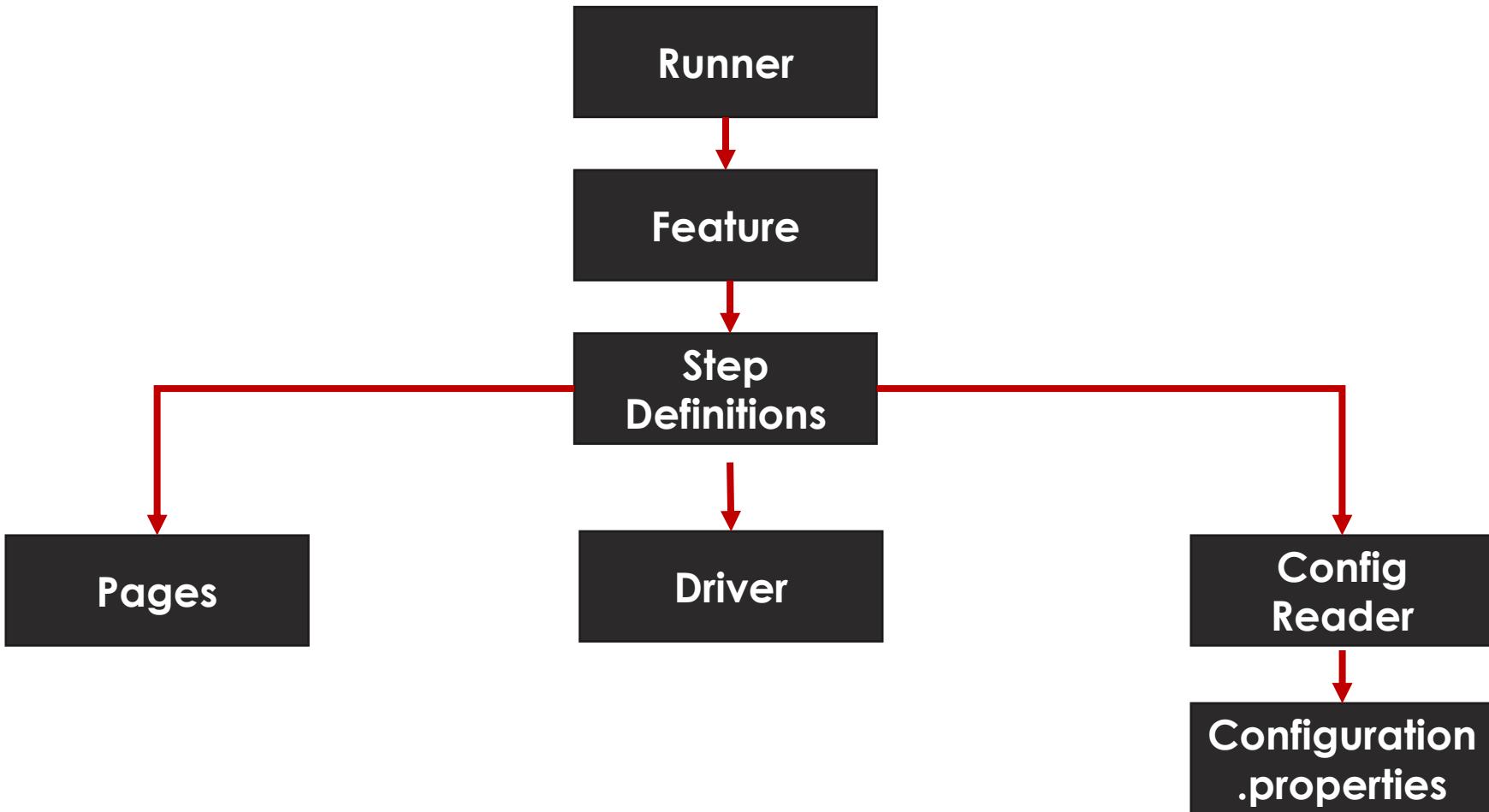


Background Class Work

US1001 feature file'daki tekrar eden aramalar yerine parameter kullanarak arama yapabilecegimiz sekilde US1002 feature file ve TC02 parameter ile arama Scenerio'su olusturalim



Cucumber Mini Tekrar





@tags

- Tag'ları onceden belirledigimiz senaryoları (scenario) çalıştırmak için kullanırız.
- Tag'ları senaryolarımızı grüplendirmek için de kullanabiliriz (smoke test, regression test, vs.)

```
@RunWith(Cucumber.class)
@CucumberOptions (
    plugin={"html:target\\cucumber-reports.html",
            "json:target/json-reports/cucumber.json",
            "junit:target/xml-report/cucumber.xml"},
    features="src/test/resources/features",
    glue="stepdefinitions",
    tags="@toplu" ,
    dryRun= false          // dryRun=true dedigimizde test
```

Feature: US1001 amazon page search

@amazon @search @apple

Scenario: TC01 amazon arama testi

```
Given kullanici amazon sayfasina gider
And "apple" icin arama yapar
Then sonuclarin "apple" icerdigini test eder
Then sayfayı kapatır
```



Feature File'i Parametre ile Kullanma

```
Feature: US1000 Amazon search test
```

```
Scenario: TC01 iphone aramasi testi
```

```
Given kullanici amazon sayfasina gider  
And iphone icin arama yapar  
Then sonuclarin iphone icerdigini test eder
```

```
Scenario: TC02 tea pot aramasi testi
```

```
Given kullanici amazon sayfasina gider  
And tea pot icin arama yapar  
Then sonuclarin tea pot icerdigini test eder
```

```
Scenario: TC03 flower aramasi testi
```

```
Given kullanici amazon sayfasina gider  
And flower icin arama yapar  
Then sonuclarin flower icerdigini test eder  
Then sayfayı kapatır
```

- Kodlarımızı parametreli ve dinamik hale getirmek için feature file da degisken olarak kullanacagımız kelimeyi çift tırnak " " icine alırız.

```
@rapor2
```

```
Scenario: TC07 istenen kelimenin oldugunu test etme
```

```
Given kullanici "amazonUrl" sayfasina gider  
And "iphone" icin arama yapar  
Then sonucun "iphone" icerdigini test eder  
And sayfayı kapatır
```

- Kodlarımızı parametreli olarak yazdıktan sonra sadece " " içindeki değeri değiştirerek test datalarını feature file dan kontrol edebiliriz.
- Kodlarımızı parametreli olarak yazmak framework'u daha dinamik hale getirir(kodumuz artık hard coded degildir diyebiliriz).



Feature File'i Parametre ile Kullanma Class Work

US1001 de kullandigimiz feature dosyasi altinda yeni bir Scenario olusturalim
TC03 ve orada yaptigimiz aramayı parametre kullanarak yapalim

```
Feature: US1000 Amazon search test

  Scenario: TC01 iphone araması testi
    Given kullanıcı amazon sayfasına gider
    And "iphone" için arama yapar
    Then sonuçların "iphone" içerdigini test eder
    Then sayfayı kapatır
```



Feature File'i Scenario Outline ile Kullanma

- **Scenario Outline:** ayni teste birden fazla datayi kullanmamizi saglar
- Bir liste kullanmak istedigimiz degeri “**<value>**” seklinde yazariz
- Daha sonra testin sonuna Examples: yazip ilk satir olarak **|value|** yazariz ve altina kullanmak istedigimiz degerleri ekleriz. (**elma|larmut|...** gibi)

Class Work : US1001 de kullandigimiz feature dosyasi altında yeni bir Scenario olusturalim TC04 ve orada yaptigimiz aramayı **Scenario Outline** kullanarak farkli urunler icin yapalim

```
@amazon @search @apple
Scenario Outline: TC01 amazon arama testi

Given kullanici amazon sayfasina gider
And "<kelime>" icin arama yapar
Then sonuclarin "<kelime>" icerdigini test eder
Then sayfayı kapatır

Examples:
|kelime|
|teapot|
|flower|
|avsfdfhvbj|
```



Class Work

Yeni bir feature file olusturalim:

US1004_Walmart_parameter_arama.feature

Yeni bir Scenario olusturalim:

TC07_aranan_kelime_title'da_olmali

Nutella, pencil, milk, tomatoes ve popcorn ile arama yapip sonuclari test edin

Scenario Outline: TC09 amazonda verilen bir listeyi

```
Given kullanici "amazonUrl" sayfasina gider
Then "<arananUrun>" icin arama yapar
And sonuc sayisini yazdirir
And sonucun "<arananUrun>" icerdigini test eder
Then sayfayı kapatır
```

Examples:

arananUrun
nutella
pencil
milk
tomatoes
popcorn



Background Class Work

Yeni bir feature file olusturalim:
US1006_Dinamik_url_test.feature

Yeni bir Scenario olusturalim:
TC08_yazilan_her_url'e_gitmeli

Configuration.properties dosyasinda tanimlanmis tum url'lerden key olarak yazdigimda ilgili sayfaya gidecek sekilde bir stepdefinition olusturun.

Bu stepdefinition'i amazon_url, bestbuy_url ve ebay_url ile test edin

Scenario Outline: TC01 amazon arama testi

```
Given kullanici "<sayfa_url>" sayfasina gider  
And url'in "<sayfa_url>" oldugunu test eder  
Then sayfayı kapatır
```

Examples:

```
|sayfa_url|  
|amazon_url|  
|bestbuy_url|  
|ebay_url|
```



Class Work

ConcordHotels Login negative test case'i asagidaki 5 kullanici ismi ve sifresi icin calisacak sekilde duzenleyin

Kullanici adi	Password	Feature: US1009 Ck Hotels Login
Manager5	Manager5!	Scenario: TC11 kullanici gecerli bilgilerle giris yapar Given kullanici ConcordHotels ana sayfasinda Then Log in yazisina tiklar
Manager6	Manager6!	And gecersiz username girer
Manager7	Manager7!	And gecersiz password girer
Manager8	Manager8!	And Login butonuna basar
Manager9	Manager9!	Then sayfaya giris yapilamadigini kontrol eder And kullanici sayfayi kapatir



Class Work

Yeni bir feature file olusturun: US1007_kullanici_data_ekleyebilmeli

DataTableStepDefinition dosyasi ve gerekli step definition'lari olusturun ve 5 farkli kayit ekleyin

When kullanici <https://editor.datatables.net/> adresine gider

Then new butonuna basar

And tum bilgileri girer

And Create tusuna basar

When kullanici ilk isim ile arama yapar

Then isim bolumunde isminin oldugunu dogrular



Html Rapor Ekleme

- Cucumber raporları, şirketlerin Cucumber kullanmasının ana nedenlerinden biridir.
- Html rapor almak için runner classına eklenen plugin eklememiz yeterlidir.

plugin={"html:target\\cucumber-reports.html"}

```
@RunWith(Cucumber.class)
@CucumberOptions(
    plugin={"html:target/cucumber_rapor.html"},
    features = "src/test/resources/features",
    glue="stepdefinitions",
    tags="@amazon",
    dryRun = false
)
```



Class Work

Feature: US1011 Concorp Hotels Login

Scenario: TC12 kullanici gecerli bilgilerle giris yapar

Given kullanici ConcorpHotels ana sayfasinda

Then Log in yazisina tiklar

And gecerli username girer

And gecerli password girer

And Login butonuna basar

Then sayfaya giris yaptigini kontrol eder

And kullanici sayfayı kapatir

Feature: US1001 Ck Hotels Login

| @wip

Scenario: TC01 kullanici gecerli bilgilerle giris yapar

Given kullanici Ck Hotels ana sayfasinda

Then Log in yazisina tiklar

And gecerli username girer

And gecerli password girer

And Login butonuna basar

Then sayfaya giris yaptigini kontrol eder

And kullanici sayfayı kapatir



Class Work

Koala Resort Hotels Login testinde kullandigimiz feature file ve step definitions'i kullanarak negative test case yazin

Feature: US1009 Ck Hotels Login

Scenario: TC11 kullanici gecerli bilgilerle giris yapar
Given kullanici Ck Hotels ana sayfasinda
Then Log in yazisina tiklar
And gecersiz username girer
And gecersiz password girer
And Login butonuna basar
Then sayfaya giris yapılamadığını kontrol eder
And kullanici sayfayı kapatır



Class Work

Yeni bir feature file olusturun: US1007_kullanici_data_ekleyebilmeli

DataTableStepDefinition dosyasi ve gerekli step definition'lari olusturun

When kullanici <https://editor.datatables.net/> adresine gider

Then new butonuna basar

And tum bilgileri girer

And Create tusuna basar

When kullanici ilk isim ile arama yapar

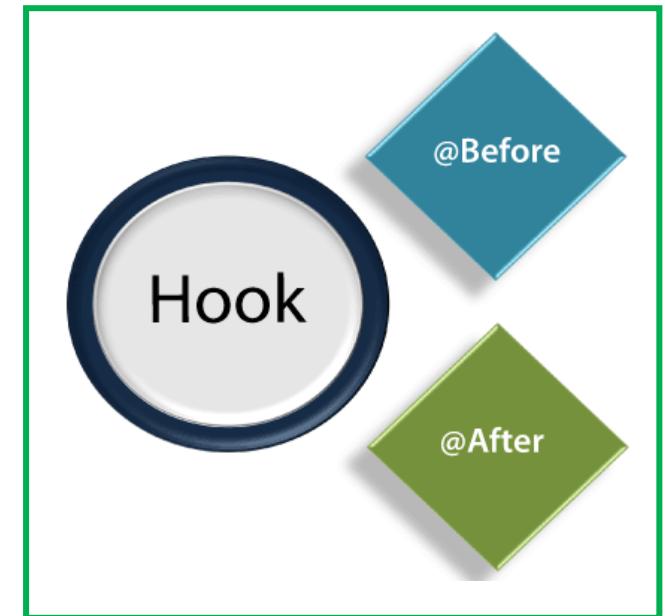
Then isim bolumunde isminin oldugunu dogrular

Hooks ve Screen Shot Ekleme

- Cucumber hooks, senaryolardan önce veya sonra çalışan kod bloklarına sahip olan bir classtır. (Daha once kullandigimiz TestBase gibi)
- @Before ve @After annotation'ları kullanılarak kodları projemizde ve step definitionlarda kullanabiliriz.
- ucumber hooks, kod çalışma akışını daha iyi yönetmemizi kolaylaştırır ve kod fazlalığını azaltmamıza yardımcı olur.

@After

```
public void tearDown(Scenario scenario){  
    final byte[] screenshot=((TakesScreenshot)  
Driver.getDriver()).getScreenshotAs(OutputType.BYTES);  
    if (scenario.isFailed()) {  
        scenario.attach(screenshot, "image/png","screenshots");  
    }  
    Driver.closeDriver();  
}
```





Paralel Testing

- Paralel testing: Birden fazla browser'in es zamanlı çalıştırılmasıdır.
- Cucumber ile parallel test çalıştırmak testing'ye göre daha zordur.
Ancak raporlama ihtiyacı varsa TestNG'deki karmaşık raporlama prosesleri yerine cucumber'da parallel çalıştırmanın zorluguna katlanmak tercih edilebilir.
- Paralel çalıştırılmak için birden fazla Runner Class'ına ihtiyacımız var
- Cucumber'da, testleri paralel olarak çalıştırılabilmek için bazı eklentilere(plugin) ve yapılandırmalara da ihtiyacımız vardır.
- Pom da yaptığımız ayarlamalardan sonra testleri Runner'dan değil Terminalden "**mvn clean verify**" kodunu yazarak çalıştıracağız



Paralel Testing

1. Birden fazla runner classı ekleyin. Aynı anda calistirmak istediginiz kadar Runner Class'ına sahip olmalisiniz. Class isimleri belirledigimiz ortak bir String icermelidir. Ornegin :TestRunner.
 - a. SmokeTestRunner
 - b. FirstTestRunner
 - c. SecondTestRunner
2. maven-failsafe-plugin eklentisi ekleyin.(Belirli testler başarısız olduktan sonra testleri çalışmaya devam için.)

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-failsafe-plugin</artifactId>
    <version>3.0.0-M1</version>
    <configuration>
        <testFailureIgnore>true</testFailureIgnore>
        <skipTests>false</skipTests>
        <includes>
            <include>**/runners/*TestRunner*.java</include>
        </includes>
    </configuration>
</plugin>
```



Paralel Testing

3. maven-surefire-plugin ekleyin ve yapılandırın. Paralel test için gerekli eklentidir

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <version>3.0.0-M1</version>
  <configuration>
    <parallel>classes</parallel>
    <forkMode>perthread</forkMode>
    <threadCount>4</threadCount>
    <reuseForks>false</reuseForks>
    <argLine>-Duser.language=en</argLine>
    <argLine>-Xmx1024m</argLine>
    <argLine>-XX:MaxPermSize=256m</argLine>
    <argLine>-Dfile.encoding=UTF-8</argLine>
    <useFile>false</useFile>
    <includes>
      <include>**/runners/*TestRunner*.java</include>
    </includes>
    <testFailureIgnore>true</testFailureIgnore>
  </configuration>
</plugin>
```



Paralel Testing

JDK sorunu yasayanlar icin opsiyonel plugin

```
<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.1</version>
  <configuration>
    <source>1.7</source>
    <target>1.7</target>
    <fork>true</fork>
    <executable>C:\Program
Files\Java\jdk1.8.0_251\bin\javac</executable>
  </configuration>
</plugin>
```



4. maven-cucumber-reporting plugin ekle. Gelişmiş rapor için gereklidir

```
<plugin>
  <groupId>net.masterthought</groupId>
  <artifactId>maven-cucumber-reporting</artifactId>
  <version>5.0.0</version>
  <executions>
    <execution>
      <id>execution</id>
      <phase>verify</phase>
      <goals>
        <goal>generate</goal>
      </goals>
      <configuration>
        <projectName>cucumber-jvm-example</projectName>
        <outputDirectory>${project.build.directory}</outputDirectory>
        <inputDirectory>${project.build.directory}</inputDirectory>
        <jsonFiles>
          <param>**/json-reports/*.json</param>
        </jsonFiles><classificationFiles>->
        <param>sample.properties</param>
        <param>other.properties</param>
      </classificationFiles>
      </configuration>
    </execution>
  </executions>
</plugin>
```

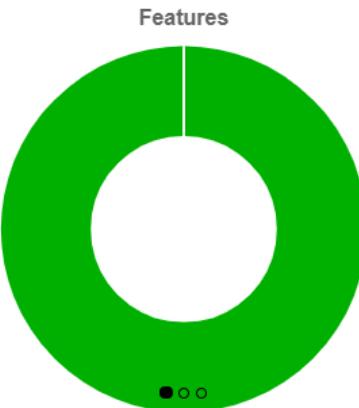


Paralel Testing

Project	Number	Date
cucumber-jvm-example	1	01 Jan 2021, 15:21

Features Statistics

The following graphs show passing and failing statistics for features



Feature	Steps						Scenarios			Features	
	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
US1001 amazon page search	10	0	0	0	0	10	1	0	1	30.902	Passed
US1002_amazon_search_background	12	0	0	0	0	12	3	0	3	41.928	Passed
US1004_amazon_search_scenario_outline	16	0	0	0	0	16	4	0	4	53.219	Passed
	7	0	0	0	0	7	1	0	1	28.033	Passed
US1009 Ck Hotels Log	7	0	0	0	0	7	1	0	1	15.471	Passed
	52	0	0	0	0	52	10	0	10	2:49.553	5
	100.00%	0.00%	0.00%	0.00%	0.00%		100.00%	0.00%			100.00%



Paralel Testing

Project	Number	Date
cucumber-jvm-example	1	01 Jan 2021, 15:21

Tags Statistics

The following graph shows passing and failing statistics for tags

Tag	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
@amazon	35	0	0	0	0	35	8	0	8	1:35.482	Passed
@amazonarama	3	0	0	0	0	3	1	0	1	3.187	Passed
@mehmet	3	0	0	0	0	3	1	0	1	3.865	Passed
@smoke	14	0	0	0	0	14	2	0	2	43.504	Passed
@smoketest	6	0	0	0	0	6	2	0	2	7.052	Passed
	61	0	0	0	0	61	14	0	14	2:33.090	5
	100.00%	0.00%	0.00%	0.00%	0.00%		100.00%	0.00%			100.00%



Paralel Testing

Project	Number	Date
cucumber-jvm-example	1	01 Jan 2021, 15:21

Steps Statistics

The following graph shows step statistics for this build. Below list is based on results. step does not provide information about result then is not listed below. Additionally @Before and @After are not counted because they are part of the scenarios, not steps.

Implementation	Occurrences	Average duration	Max duration	Total durations	Ratio
stepdefinitions.AmazonStepDefinitions.flower_icin_arama_yapar()	2	3.892	4.117	7.784	100.00%
stepdefinitions.AmazonStepDefinitions.icinAramaYapar(java.lang.String)	4	3.460	4.144	13.842	100.00%
stepdefinitions.AmazonStepDefinitions.iphone_icin_arama_yapar()	2	3.216	3.448	6.432	100.00%
stepdefinitions.AmazonStepDefinitions.kullaniciSayfayiKapatir()	10	0.088	0.119	0.884	100.00%
stepdefinitions.AmazonStepDefinitions.kullanici_amazon_anasayfaya_gider()	10	8.741	13.112	127.416	100.00%
stepdefinitions.AmazonStepDefinitions.sonucunIcerdiginiTestEder(java.lang.String)	4	0.118	0.177	0.474	100.00%
stepdefinitions.AmazonStepDefinitions.sonucun_flower_icerdigini_test_eder()	2	0.165	0.206	0.331	100.00%
stepdefinitions.AmazonStepDefinitions.sonucun_iphone_icerdigini_test_eder()	2	0.122	0.142	0.244	100.00%
stepdefinitions.AmazonStepDefinitions.sonucun_tea_pot_icerdigini_test_eder()	2	0.112	0.152	0.225	100.00%
stepdefinitions.AmazonStepDefinitions.tea_pot_icin_arama_yapar()	2	4.324	4.504	8.649	100.00%
stepdefinitions.BestbuyStepDefinitions.kullanici_anasayfaya_gider(java.lang.String)	2	13.412	14.311	26.825	100.00%
stepdefinitions.CkHotelsStepDefinitions.gecerli_password_girer()	1	0.188	0.188	0.188	100.00%
stepdefinitions.CkHotelsStepDefinitions.gecerli_username_girer()	1	0.230	0.230	0.230	100.00%
stepdefinitions.CkHotelsStepDefinitions.gecersizPasswordGirer()	1	0.190	0.190	0.190	100.00%
stepdefinitions.CkHotelsStepDefinitions.gecersizUsernameGirer()	1	0.221	0.221	0.221	100.00%
stepdefinitions.CkHotelsStepDefinitions.log_in_yazisina_tiklar()	2	1.145	1.200	2.291	100.00%
stepdefinitions.CkHotelsStepDefinitions.login_butonuna_basar()	2	6.593	12.063	13.186	100.00%
stepdefinitions.CkHotelsStepDefinitions.sayfayaGirisYapilamadiginiKontrolEder()	1	0.104	0.104	0.104	100.00%
stepdefinitions.CkHotelsStepDefinitions.sayfaya_giris_yaptigini_kontrol_eder()	1	0.037	0.037	0.037	100.00%
19	52	3.260	1:27.416	2:49.553	Totals



- 1- Paralel test tum testlerimizi hep birlikte execute etmek istedigiimizde zamanı azaltmak için kullanılır. Paralel testing'de istedigimiz kadar browser'i aynı anda calistirabiliriz
- 2- Ilk yapmam gereken paralel calismasını istedigim browser sayısınca runner olusturmak
- 3- Runner class'larinin ismi TestRunner icermelidir. Farklı bir isim istiyorsak sureFire plugin'inden calistirilacak runner'ların isminde gecen ortak kismi yazmaliyiz
- 4- Calistirmak istedigimiz tum testleri paralel calistiracagımız browser sayisina gore tag'larla gruplandiririz.
- 5- pom.xml'de dependencies tagi bittikten sonra icice build, plugins taglarini acalim. Plugins tagları içinde 2 tane plugin ekleriz
 - failsafe pluginı testimiz basarisiz olsa da paralel testing devam ettirir
 - sureFire bu plugin de paralel calistirma ayarları yapilir
- 6- Daha guzel ve gelismis bir rapor almak icin pom.xml'e **maven-cucumber-reporting** pluginı eklenir
- 7- Plugin'lerin devreye girebilmesi icin, testlerimiz runner class'dan degil, IntelliJ terminal'den calistirilmalidir.
- 8- Terminal'den runnerlari calistirmak icin **mvn clean verify** yazmaliyiz



TestNG Paralel Testing

- TestNg'de paralel test xml dosyasi kullanilarak yapilir.
- Paralel test calisma suresini azaltir, dolayisiyla zaman kazanmak icin parallel test kullanilir.
- Paralel test ayni anda birden fazla test case'i eszamanli olarak calistirmak demektir.
- Xml dosyasinda belirlenen testleri belirledigimiz **level** seviyesinde belirledigimiz **thread-count** sayisinda parallel calistirir
- Classes,methods seviyesinde calistirirsak verilen tum gorevler bitene kadar baska class veya method varsa calismaya devam eder. Level olarak **Tests** secilirse testlerin tanimlanmasi gereklidir
- Cross Browser (Multi Browser) test ise farkli browserlar ile test yapmak demektir.Sirali (sequential) veya paralel yapilabilir.



- Çoklu calistirma, Parallel calistirma ve Cross Browser calistirma farkli farkli islemlerdir.
- Mesela 5 testi sirali olarak ama topluca calistirirsak → sirali çoklu calistirma
- 5 testi ayni anda calismaya baslayan iki driver'la calistirirsam
 - ➔ parallel calistirma
- 5 testi sirali olarak calistirip, ilk ucunu chrome, son ikisini firefox'da calistirirsam
 - ➔ sirali cross browser
- 5 testin ucunu chrome, ikisini firefox ile calistirip, chrome ve firefox'u ayni zamanda calistirirsam
 - ➔ parallel cross browsing test olur



```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Paralel Test 2" parallel="classes" thread-count="2">
    <test name="Class Paralel">
        <classes>
            <class name="com.techproed.tests.D26_AmazonSatirSutunSayisi"></class>
            <class name="com.techproed.tests.D26_AmazonHucreTesti"></class>
            <class name="com.techproed.tests.D25_HtmlRapor1"></class>
            <class name="com.techproed.tests.D25_WindowHandle"></class>
        </classes>
    </test>
</suite>
```



Paralel Testing / Methods

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Paralel Test 1" parallel="methods" thread-count="2" >
  <test name="Smoketest Paralel" >
    <packages>
      <package name="com.techproed.smokeTest"></package>
      <package name="com.techproed.tests"></package>
    </packages>
  </test>
</suite>
```



Cross Browser Testing

1. Cross Browser test bir uygulamayı farklı browserlar ile test etmek demektir
2. Testleri sıralı (sequential) veya paralel olarak yapabiliriz
3. Çv'niz açısından Cross Browser test önemlidir çünkü ileri seviyeyi gösterir
4. Cross Browser testi yapabilmek için framework'de gerekli düzenlemeleri yapmanız gereklidir. (Bu her tester'in sahip olacağı bir özellik değildir, dolayısıyla size 1 adım one çıkarır)
5. Her bir adımı ezbere bilmek zorunda değiliz ama mantığı anlamak ve bunu sözlü olarak ifade edebilmek zorundayız





Cross Browser Testing

1. Crossbrowser testi icin yeni bir driver class'i olusturacagiz : **DriverCross**

- getDriver methoduna parameter ekleyecegiz **WebDriver getDriver(String browser)**
- if blogundan once bir satir kod ekleyecegiz
browser = browser == null ? ConfigReader.getProperty("browser") : browser;
- switch case'deki degeri degistirelim **switch(browser)**

Bu class'in gorevi xml dosyasindan parameter olarak gonderilen browser'i driver olarak tanimlamaktir. Xml dosyasindan parametre gelmezse o zaman .properties dosyasinda tanimli browser'i kullanir

2. Crossbrowser testi icin yeni bir **TestBase** class'i olusturacagiz

- Basa gelen parametreyi kullanmak icin **@Parameters("browser")** ekleyecegiz
- setup methodune parametre gonderecegiz **setUp(@Optional String browser)** burada optional yazma sebebimiz parameter gelmese de calismasini istememiz



3. Farkli browser'lar ile calistirmak istedigimiz class'lari bir package altina toplayalim
crossBrowser ve class'lari TestBaseCross clasina **extends** ile **child** olarak tanimlayalim
4. Xml dosyasi olusturalim ve cross browser icin <test> satirinin altina browser icin parametre gonderelim

```
<parameter name="browser" value="firefox"></parameter>
```

5. Paralel calistirmak istersek paralel calistirma kodlarini eklememiz yeterli



Test NG

@Data Provider

@DataProvider bir TestNG annotation'ıdır.
Dolayisiyla sadece TestNG ile kullanılır.
Veri sağlamak için kullanılır.
DDT (Data Driven Test) yapılır.

Cucumber'daki Scenario Outline
İle aynı işlev sahiptir

```
public class data {  
  
    @Test(dataProvider = "aranacaklar")  
    public void test(String aranan){  
        Driver.getDriver().get(ConfigReader.getProperty("amazon_url"));  
        AmazonPage amazonPage=new AmazonPage();  
        amazonPage.aramaKutusu.sendKeys(aranan+ Keys.ENTER);  
    }  
  
    @DataProvider(name="aranacaklar")  
    public Object[] getUrunler(){  
        String aranacaklar[] = {"Ali","Veli","Hasan","Huseyin","Yasar"};  
  
        return aranacaklar;  
    }  
}
```



Soru 1

1. <https://qa-environment.koalaresorthotels.com/> sayfasina gidelim
2. Cucumber parametre ,cucumber scenario outline ve TestNg framework @Dataprovider kullanarak asagidaki gorevi tamamlayin
 - Login tusuna basin
 - Asagidaki 5 kullanici adi ve sifreyi deneyin ve login olmadigini test edin
 - Manager - Manager
 - Manager1- Manager1
 - Manager2 - Manager2
 - Manager3 - Manager3
 - Manager4 - Manager4



Soru 2

1. <http://automationpractice.com/index.php> sayfasina gidelim
2. Cucumber ile asagidaki testi yapalim

Given user web sayfasinda

And user sign in linkine tiklar

And user Create and account bölümüne email adresi girer

And Create an Account butonuna basar

And user kisisel bilgilerini ve iletisim bilgilerini girer

And user Register butonuna basar

Then hesap olustugunu dogrulayin



Soru 3

1. <http://automationpractice.com/index.php> sayfasina gidelim
2. Cucumber ile asagidaki testi yapalim
 - Given user web sayfasinda
 - And user sign in linkine tiklar
 - And email kutusuna @isareti olmayan email adresi yazar ve enter'a tiklar
 - Then error mesajinin "Invalid email address" oldugunu dogrulayin



Soru 4

1. <http://demo.guru99.com/test/web-table-element.php> sayfasina gidelim
2. Cucumber framework'de US1012_Guru_Web_Tables olusturun
3. Scenario : TC_16_kullanici_liste_alabilmeli asagidaki testi yapin

Given user web sayfasinda

Then Company listesini consola yazdirir

And DCB Bank'in listede oldugunu test eder



Soru 5

1. <http://demo.guru99.com/test/web-table-element.php> sayfasina gidelim
2. Cucumber framework'de US1012_Guru_Web_Tables altinda

Scenario : TC_17_kullanici_sirket_Prev_Close_alabilmeli olusturun ve asagidaki testi yapin

Given user web sayfasinda

And "Istenen Sirket" Prev.Close degerini yazdirir



Soru 6

1. <http://demo.guru99.com/test/web-table-element.php> sayfasina gidelim
2. Cucumber framework'de US1012_Guru_Web_Tables altinda

Scenario : TC_18_kullanici_satir_sutun_degeri_ile_yazi_alabilmeli olusturun ve asagidaki testi yapin

Given user web sayfasinda

And “Istenen Satir”, “Istenen Sutun” daki yaziyi yazdirir



Soru 7

1. <http://demo.guru99.com/test/web-table-element.php> sayfasina gidelim
2. Cucumber framework'de US1012_Guru_Web_Tables altinda
Scenario : TC_18_kullanici_sutun_basligi_ile_liste_alabilmeli olusturun ve asagidaki testi yapin
Given user web sayfasinda
And "Istenen Baslik", sutunundaki tum degerleri yazdirir



Test NG

Genel Tekrar

Soru 8

- 1) Yeni bir class olusturalim D34_readExcel
- 2) Baskentler excelini framework'e ekleyelim ve excelle ilgili islemleri yaparak dosyayı kullanabilir hale getirelim
 - 1.satirdaki 2.hucreye gidelim ve yazdiralim
 - 1.satirdaki 2.hucreyi bir string degiskene atayalim ve yazdiralim
 - baskenti Jakarta olan ulke ismini yazdiralim
 - Ulke sayisini bulalim
 - Fiziki olarak kullanılan satır sayısını bulun
 - Tüm bilgileri map olarak kaydedelim
 - baskenti Jakarta olan ulkenin tüm bilgilerini yazdiralim
 - Satır ve sutun bilgisi ile hucre yazdiralim



Test NG

Genel Tekrar

Soru 9

Yeni bir Class olusturalim D34_WriteExcel

1) Yeni bir sutun olusturalim

- baslik satirinda ilk bos hucreye yeni bir cell olusturalim
- Olusturdugumuz hucreye "ulke nufusu" yazdiralim
- 2.satir ulke nufusu kolonuna "1,5 milyar" yazdiralim
- 8.satir nufus kolonuna 250 milyon yazdiralim
- Dosyayı kaydedelim
- Dosyayı kapatelim



Soru 10

Yeni bir Class olusturalim Day36_ExplicitlyWait

- 1) <https://demoqa.com/browser-windows> adresine gidin
- 2) Alerts'e tiklayin
- 3) On button click, alert will appear after 5 seconds karsisindaki click me butonuna basin
- 4) Allert'in gorunur olmasini bekleyin
- 5) Allert uzerindeki yazinin "This alert appeared after 5 seconds" oldugunu test edin
- 6) Ok diyerek alerti kapatin



Soru 11

Yeni bir test methodu olusturun

- 1) <https://demoqa.com/dynamic-properties> adresine gidin
- 2) "Will enable 5 seconds" butonunun enable olmasini bekleyin
- 3) "Will enable 5 seconds" butonunun enable oldugunu test edin



Soru 12

Yeni bir test methodu olusturun

- 1) <https://demoqa.com/dynamic-properties> adresine gidin
- 2) "Will enable 5 seconds" butonunun enable olmasini bekleyin
- 3) "Will enable 5 seconds" butonunun enable oldugunu test edin



Soru 13

Yeni bir test methodu olusturun

<https://demoqa.com/dynamic-properties> adresine gidin

- 1) "Visible After 5 seconds" butonunun gorunur olmasini bekleyin
- 2) "Visible After 5 seconds" butonunun gorunur oldugunu test edin



Soru 14

Yeni bir test methodu olusturalim

https://the-internet.herokuapp.com/add_remove_elements/ adresine gidin

- 1) "Add Element" butona basin
- 2) "Delete" butonu gorunur oluncaya kadar bekleyin
- 3) "Delete" butonunun gorunur oldugunu test edin
- 4) Delete butonuna basarak butonu silin
- 5) Delete butonunun gorunmedigini test edin