



DERS 1

SELENIUM GIRIS
OTOMASYON TEST NEDIR?
SELENIUM PROJESİ
OLUSTURMA

Mehmet BULUTLUOZ
Elektronik Muh.

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 sadece bir jar dosyasıdır. Kurulum sırasında jar dosyalarını gördünüz.
- Selenium, otomasyon yapmak için kendi sınıflarına ve yöntemlerine sahip bir kütüphanedir..
- 2016'da piyasaya sürülen Selenium 3'ü öğreneceğiz
- Selenium'u çeşitli programlama dilleri ile yazabilirsiniz java, Python, ruby, .Net vb.



IntelliJ IDEA NEDİR

- 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 yardımı 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 islemidir.

- 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 control etmek ve ortaya çıkartmak yazılım testinin amacı olmalıdır.

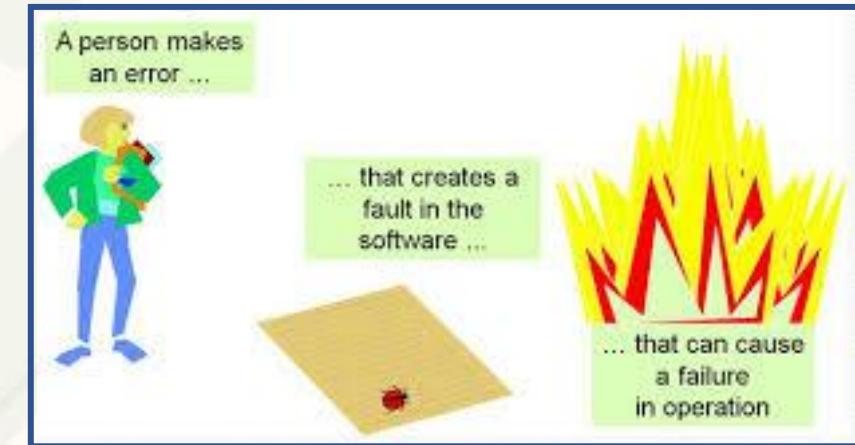
- Her User Story icin Positive ve Negative Test(ler) yapılmalıdır
- Test musteri/isletme ihtiyaçlarını karşılamak için yapılır.
- Bir uygulamayı test etmek için onceden belirlenmiş user storyler (kullanıcı hikayeleri) ve tanımlanmış acceptance criterias (kabul kriterleri) dikkate alınır.

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 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 gerektiginde 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

EXECUTION TIME



PEOPLE



INFRESTRUCTURE



TOOLS



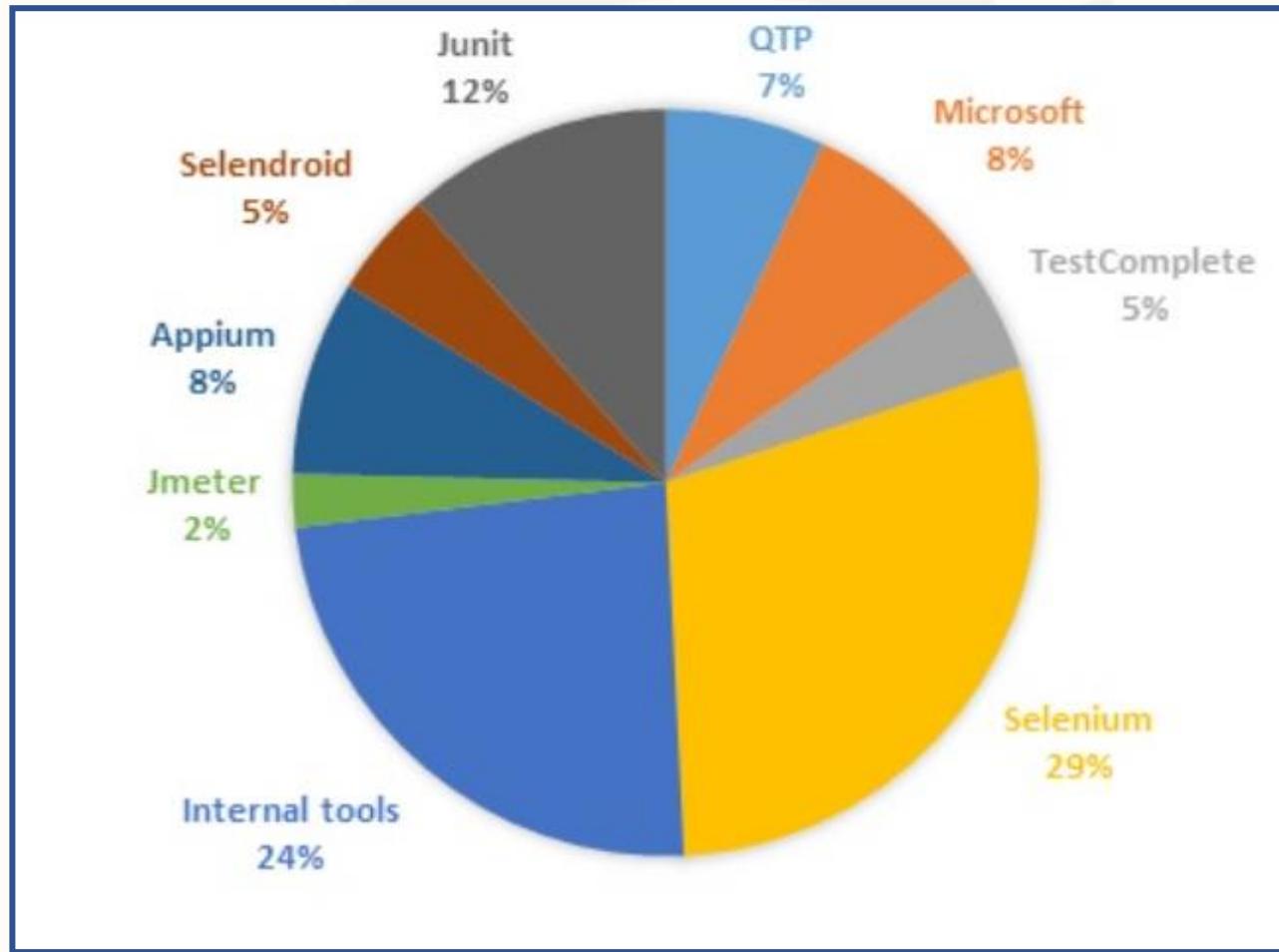
TURNAROUND TIME



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 secebirlirler.
- 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.

Basarılı Bir Otomasyon Testi için Yapılmaması Gerekenler

- Her seyi otomasyonlaştırmaya çalışmayın.

Ekipiniz % 100 otomasyon gerektiğini söyleyebilir. Ancak tüm uygulama için % 100 otomasyon mümkün değildir..

- Otomasyona Hemen Başlamayın.

Otomasyona başlamadan önce aplikasyon hakkında detaylı bilgi sahibi olun.

- Sadece otomasyon araçlarına güvenmeyin.

Zaman zaman manuel müdahaleler gerekebilir.

- Ise başladığınız ilk günden itibaren sorularınızı sormaktan çekinmeyin.

Sonuçta, hatasız ürün olusumundan siz sorumlusunuz.



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 sadece bir jar dosyasıdır. Kurulum sırasında jar dosyalarını gördünüz.
- Selenium, otomasyon yapmak için kendi sınıflarına ve yöntemlerine sahip bir kütüphanedir.
- Selenium ile birlikte APPIUM mobile testing de yapılabilir.

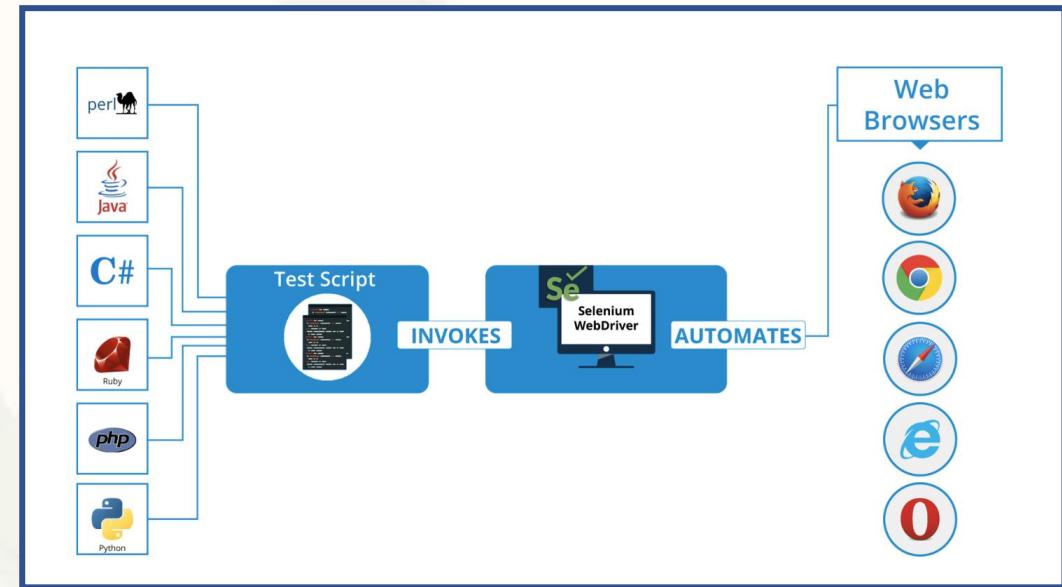


Selenium'un dört bileşeni vardır;

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

SELENIUM NASIL CALISIR?

- Driver selenium'dan komutları alır.
- Bu komutları tarayıcının API'nda dönüştürür
- Browser'lardan donen sonucları alır ve Selenium'a geri gönderir



SELENIUM'UN AVANTAJLARI NELERDIR?

- Ücretsizdir.

Open source

- Bir çok programlama dilini destekler

(Java, Python, PHP, C#, Ruby vs.)

- Çoklu işletim sistemleriyle çalışır.

(Windows, MacOS, Linux)

Multiple operating systems

- Birden çok tarayıcı ile çalışır.

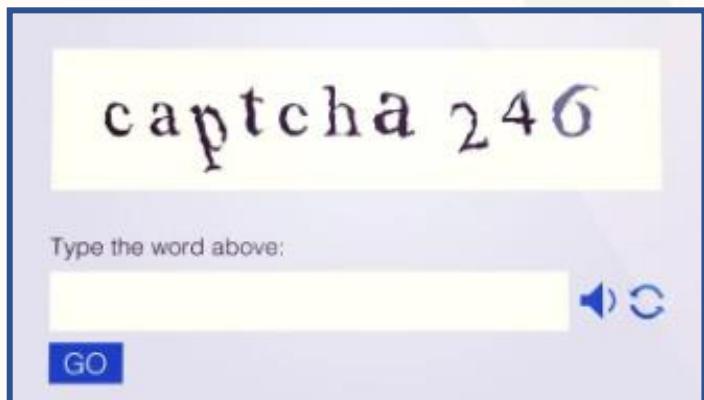
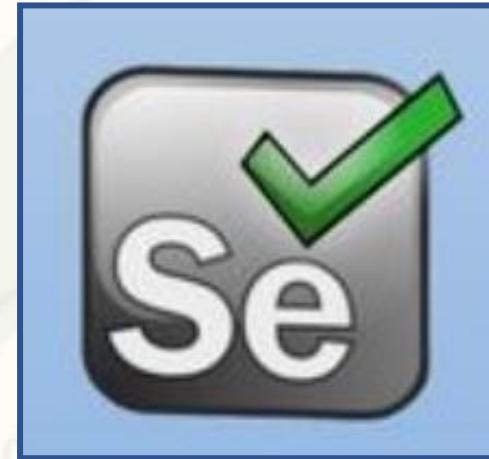
(IE, Safari, Chrome, Firefox vs.)

Multiple browsers



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 NEDİR?

- Test Otomasyon Framework (TAF), yazılım projelerindeki Test Case'lerin oluşturulma ve gözden geçirme süresini azaltmak ve QA ekibimizin isini kolaylaştırmak için tasarlanmış yapılardır.
- 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-hand sidebar, showing the main directory, .idea, src (with main and test sub-directories), resources, and target. The test directory contains a java sub-directory with a com.techproed package. Inside this package, several test classes are listed: CheckboxAndRadioButton, DropDownAmazon, DropDownHomework, FirstMavenClass, FirstTestNgClass, TestNGAnnotationsExample (which is currently selected), TestNGAnnotations, and WebTables. The right-hand panel shows the code editor for the TestNGAnnotationsExample class, which starts with a package declaration for com.techproed and a public class definition for TestNGAnnotationsExample.

```
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 yukleyelim
- 3) Browsers altında Chrome documentation linkini tiklayalim
Chrome'un kendi sayfasina gidip Current stable release'i tiklayip size uygun olani indirelim
***** buradaki surum ile bilgisayarinizdaki Chrome surumunun aynı olduğundan emin olun**
- 4) C:\Users\lenovo\Documents altında selenium dependencies klasoru 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

ALISTIRMA

1. Yeni bir class oluşturun: class name ⇒ BasicNavigations
2. Create main method
3.

```
System.setProperty("", "");  
System.setProperty("webdriver.chrome.driver","/Users/techproed/Documents/selen  
ium dependencies/drivers/chromedriver"); /MAC  
  
System.setProperty("webdriver.chrome.driver","C:\\\\Users\\\\techproed\\\\Documents\\\\  
selenium dependencies\\\\drivers\\\\chromedriver.exe"); \\\\WINDOWS
```
4. Chrome driver oluşturun

```
WebDriver driver = new ChromeDriver();
```
5. "https://www.amazon.com" adresinden amazon ana sayfasını açın.

BASIC NAVIGATIONS

`driver.get("https://www.youtube.com");` istedigimiz web sayfasina gider.

`driver.manage().window().maximize();` calistigimiz browser'i tam sayfa yapar

`driver.navigate().refresh();` calistigimiz browser'i refresh yapar (yeniler)

`driver.navigate().to("https://www.google.com/");` istedigimiz web sayfasina gider.

`get()`'den farkli olarak `navigate` ile gittigimiz sayfadan `back()` ve `forward()` komutlarini kullanabiliriz.

`driver.navigate().back();` `navigate()` ile geldigimiz bir onceki sayfaya gider

`driver.navigate().forward();` `navigate()` ile gidip geri dondugumuz sayfaya yeniden gider

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

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

Selenium Navigation Methods

1. Yeni bir Class olusturalim.(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 tam ekran yapalim
8. Sayfayı kapatalim / Tüm sayfaları kapatalim

Verify Page Title

1. Yeni bir Class olusturalim. (VerifyTitle)
2. Amazon ana sayfasina gidelim . <https://www.amazon.com/>
3. Sayfa basliginin (title) "amazon" oldugunu dogrulayin. (verify)

Verify Page URL

1. Yeni bir class olusturalim : (VerifyURLTest)
2. techproeducation ana sayfasina gidelim <https://www.techproeducation.com/>
3. Sayfa URL'inin www. techproeducation.com oldugunu dogrulayin

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

Homework

- 1.Yeni bir class olusturalim (Homework)
- 2.ChromeDriver kullanara, 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



DERS 2

WebElements Locators

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Selenium : web browserlarda otomasyon yapmamizi saglayan bir suite dir
- 2- IntelliJ : kodlarimizi calistirmamizi saglayan IDE (eclipse'in modern versiyonu gibi)
- 3- otomasyon test expected result ile actual result'in karsilastirilmasidir.
- 4- Selenium ile bir cok farkli framework kullanabiliriz, biz Junit, TestNg ve Cucumber ogrenecegiz.
- 5- Bir framework kullanmadan da testlerimizi yapabiliriz. Bunun icin oncelikle driver'l ve selenium'u IntelliJ'ye tanitmamiz gereklidir.

```
System.setProperty(driverinIsmi,driver'in yolu)
WebDriver driver = new ChromeDriver
```
- 6- driver objesi kullanarak web sayfalari arasında gezinebilir, title ve url bilgilerine ulasabilir, sayfayı maximize yapabiliriz ve sayfayı kapatabiliriz
- 7- Kodumuz çok hızlı çalışığından internet yavaş olduğunda sayfaya gitmeden işlemleri yapmaya çalışabilir ve istediğimiz işlemi yapamadığından hata verebilir. Bu durumda kodlarımızi bekletmek için Thread.sleep(mili saniye olarak zaman) yazarız
- 8- selenium ile UI, API ve database testing yapılabilir, Appium kullanarak mobile testing de yapılabilir.

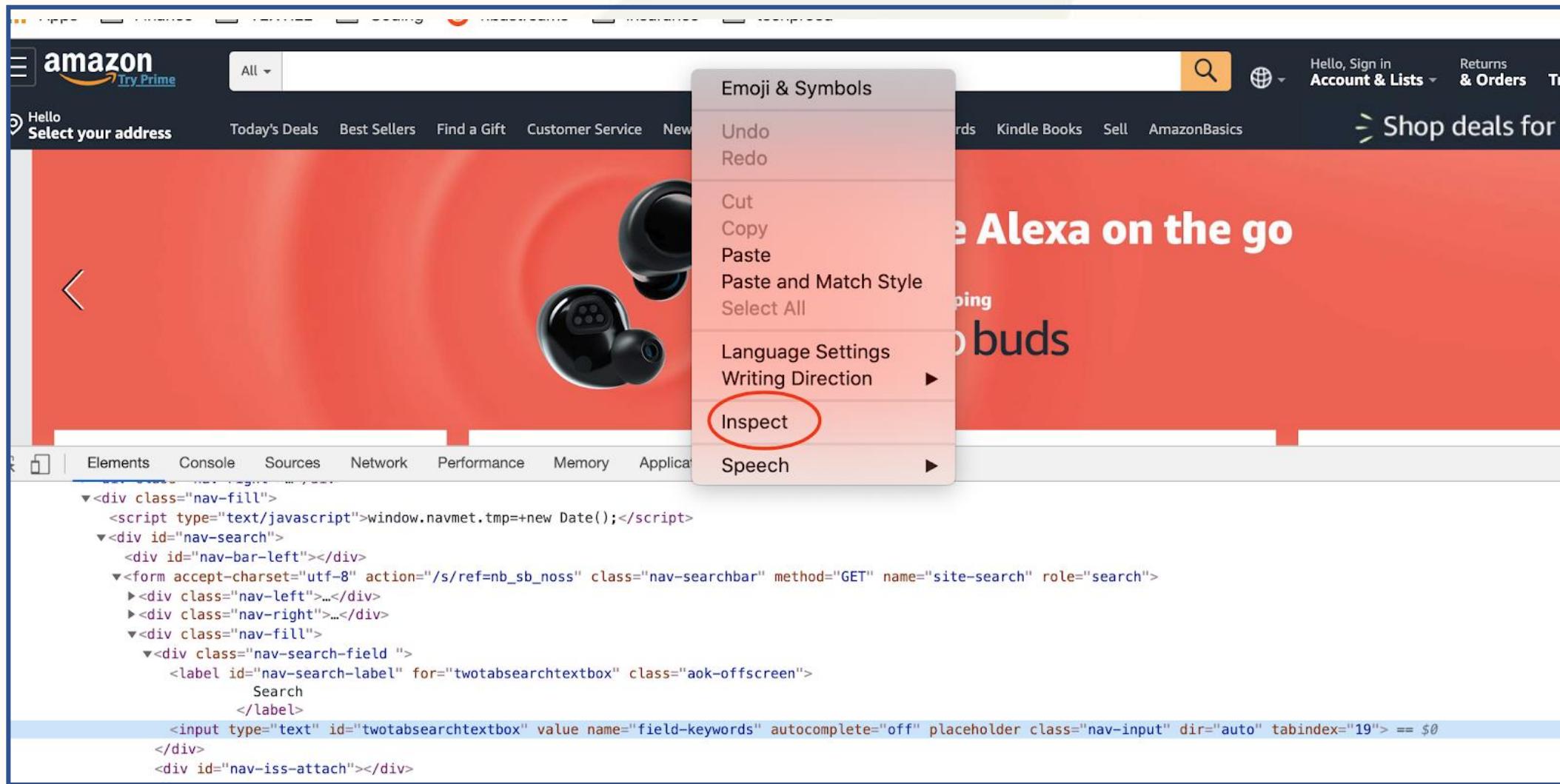
WEBELEMENTS

- 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)



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 web öğeler üzerinde eylemler gerçekleştirmek için yer buluculara 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)

Bir web elementini bulabilmek 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. css => css yazmanın birden fazla yolu vardır

LOCATORS (YER BULUCU-KONUM BELİRLEYICI)

Bir web elementini bulabilmek için 8 tane selenium locator vardır.

1. By.id => driver.findElement(By.id(""));
2. By.name => driver.findElement(By.name(""));
3. By.className => driver.findElement(By.className(""));
4. By.tagName => driver.findElement(By.tagName(""));
5. By.linkText => driver.findElement(By.linkText(""));
6. By.partialLinkText => driver.findElement(By.partialLinkText(""));
7. By.xpath => driver.findElement(By.xpath(""));
8. By.css => driver.findElement(By.css(""));

LOCATORS (YER BULUCU-KONUM BELIRLEYICI)

1. By.id() method

```
<input type="email" class="form-control" placeholder="Email" data-test="email"  
name="session[email]" id="session_email">
```

```
WebElement sessionEmail=driver.findElement(By.id("session_email"));
```

- Web ögesini tanımlamanın en popüler yolu id kullanmaktır.
- 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 gecirilmedir

LOCATORS (YER BULUCU-KONUM BELIRLEYICI)

2.By.name() method

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

```
WebElement passwordTextBox =driver.findElement(By.name("session[password]"));
```

- Name ve value unique ise bu metodu da kullanabilirsiniz.

LOCATORS (YER BULUCU-KONUM BELIRLEYICI)

3.By.className() method

```
<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.

LOCATORS (YER BULUCU-KONUM BELİRLEYICI)

4.By.linkText() method

```
<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 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 (YER BULUCU-KONUM BELIRLEYICI)

5.By.partialLinkText() method

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

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

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

LOCATORS (YER BULUCU-KONUM BELİRLEYİCİ)

6.By.tagName() method

```
<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 bağlantıları döndürmek için:

```
List<WebElement> linksOnThePage = driver.findElements(By.tagName("a"));
```

findElement() Method

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

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

findElements() Method

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

- Web elementlerinin bir listesini döndürür
- Locator stratejisiyle eşleşen web elementi yoksa boş bir liste döndürür.
- **NoSuchElementException** hatası vermez.
- Her Web elementi, tipki array'deki gibi 0'dan başlayan bir sayı indeksi alır.

*** **findElements()** ile **findElement()** arasındaki farklar nelerdir ?

WebElement Üzerindeki İşlemler

- Bir WebElement üzerinde eylemler gerçekleştirmek otomasyon tester'ları için çok önemlidir.
- **sendKeys** ("yazdığımız metin"); => metin kutusuna yazmak için kullanılır.
- **clear();** => Metin kutusunun içindekileri temizler.
- **submit();** => **click();** bir form veya form içindeki bir öğeyi gönderir. Genellikle form veya arama kutusundaki metni göndermek için **sendKeys()** yönteminden sonra kullanılır.
- **isDisplayed()** => Bir web elementinin gorunur olup olmadigini control eder
- **getText()** => Web elementine ait yaziyi getirir
- **Keys.ENTER()** => enter'a basar

Class Work: Login Test

1. Bir class oluşturun: LocatorsIntro
2. Main method oluşturun ve aşağıdaki görevi tamamlayın.
 - a. <http://a.testaddressbook.com> adresine gidiniz.
 - b. Sign in butonuna basin
 - 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 görüntünlendiğini(displayed) doğrulayın(verify).
3. Sayfada kaç tane link olduğunu bulun.

Class Work: Search Test

1. Bir class oluşturun : AmazonSearchTest
2. Main method oluşturun ve aşağıdaki görevi tamamlayın.
 - a.google web sayfasına gidin. [https://www. amazon.com/](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.



DERS 3

Xpath
CssSelector

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Locators : yer bulucu, testlerimizi otomasyon yapmak icin driver objesi kullaniriz, bu objenin istedigimiz webelement'ine ulasabilmesi icin web element'lerin konumunu driver'a buldururuz gereklidir. Driver a konum buldurmak icin findElement method'u ile birlikte kullandigimiz method'lara locator deriz/
- 2- 6 tane locator birbirine cok benzer olarak kullanilir. Attribute olarak varsa id, classname, name ve tagname locator olarak kullanilabilir
eger web element'imiz link ise linkText veya partialLinkText kullanilabilir
- 3- findElement ve locator kullanarak elementimizi buldugumuzda sonucun unique oldugundan emin olmaliyiz. Unique olmazsa kodumuz hata vermez ama buldugu webelement birden fazla oldugu icin hangisini kullanacagini bilemez dolayisiyla o webelement ile ilgili bir islem yapamaz
- 4- eger locator dogru degilse java noSuchElementException verir ve o satirda kodun calismasi durur. Dolayisiyla bu exception'l aldigimizda locator'larimizi gozden gecirmeliyiz

Parent-Child-Sibling Terimleri Nedir?

```
▼<tbody>
  ►<tr>...</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>...</tr>
</tbody>
```

Tags: tbody, tr, td, input, label => parent-child-sibling relationship(Ebeveyn-çocuk-kardeş ilişkisi) hakkında konuştuğumuzda, yalnızca tag adları önemlidir

Attributes: type, class, name, id, etc.

"tr", "tbody" nin çocuğudur(child) ve "tbody", "tr" in ebeveyndir (parent)

"td", "tr" in çocuğu ve "td" nin 3 kardeşi vardır

By.xpath() Method

2 cesit Xpath yazılabilir

1.**Absolute** xpath (mutlak)

2.**Relative** xpath (bağılı)

1.Absolute xpath: Parent'tan (root element) child a tek tek gidilir.

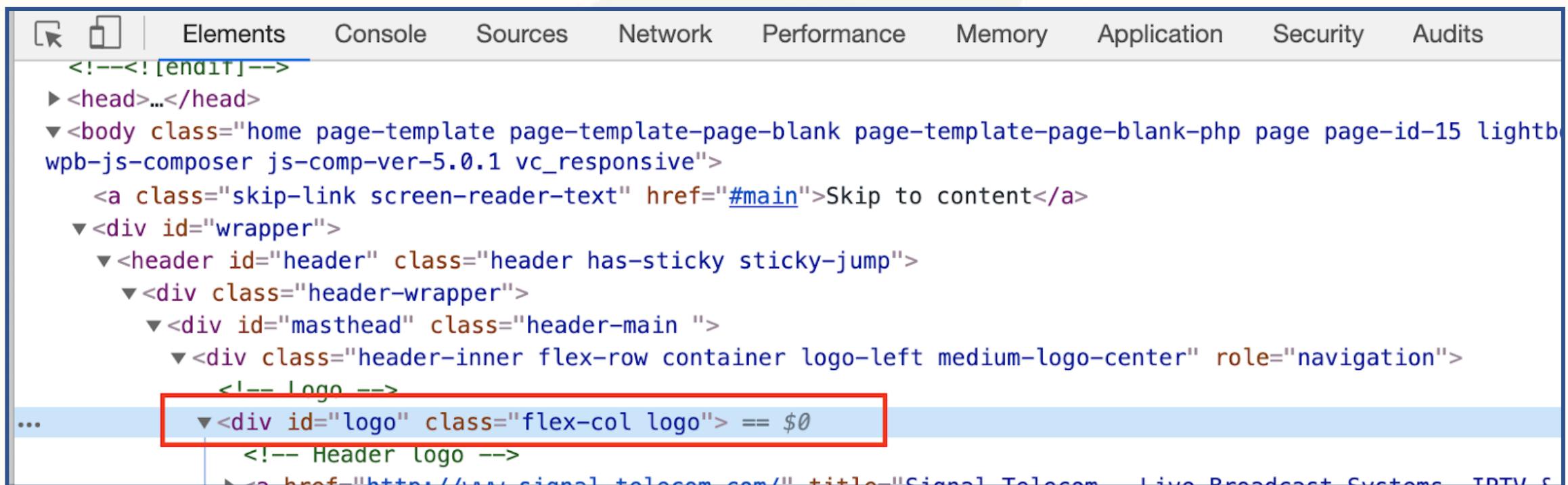
Yaygın olarak kullanılmaz.

tag'lar arasında ' / ' kullanılır.

- Xpath = //parent/child/child/child/...
- eger absolute xpath yazılırken bir yerden sonra ilerleyemiyorsak(tum webelementler sibling ise) index kullanırız. Bunun icin [] içinde index yazılır.
*** index 1 den baslar

<http://a.testaddressbook.com>

1. Absolute Xpath()



The screenshot shows the browser's developer tools with the 'Elements' tab selected. The DOM tree is displayed, starting with the root body element. A red box highlights the 'logo' div element, which has the ID 'logo' and the class 'flex-col logo'. This element is located within the masthead section of the header.

```
<!--<!--[endif]-->
> <head>...
<body class="home page-template page-template-page-blank page-template-page-blank-php page page-id-15 lightbox wpb-js-composer js-comp-ver-5.0.1 vc_responsive">
  <a class="skip-link screen-reader-text" href="#main">Skip to content</a>
  <div id="wrapper">
    <header id="header" class="header has-sticky sticky-jump">
      <div class="header-wrapper">
        <div id="masthead" class="header-main ">
          <div class="header-inner flex-row container logo-left medium-logo-center" role="navigation">
            <!-- Logo -->
            <div id="logo" class="flex-col logo"> == $0
              <!-- Header Logo -->
              <a href="http://www.signal-telecom.com/" title="Signal Telecom - Live Broadcast Systems TDTVS"...
...</div>
```

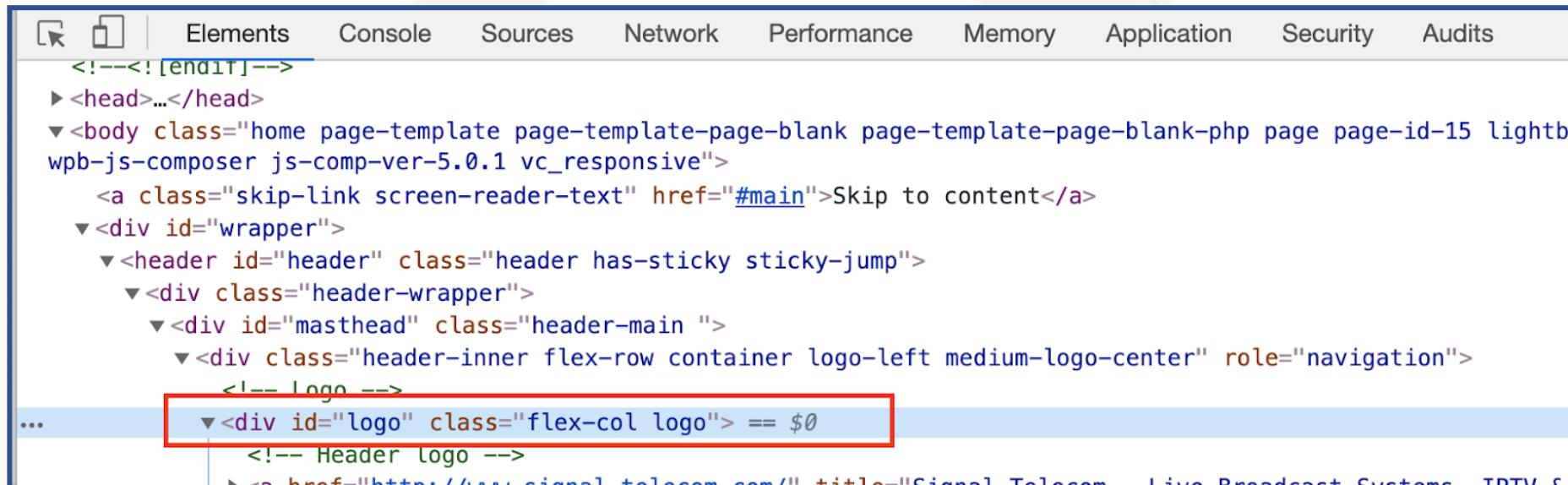
(Absolute xpath = //body/div/header/div/div/div/div)

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

2.Relative Xpath()

Yaygın olarak kullanılır. **//** kullanarak belirli bir elemente gidebilirsiniz.
Xpath yazmanın birden çok yolu vardır.

*** `//tagName[@attributelsmi='attributeValue'];`



```
<!--<! [endIT]-->
> <head>...</head>
<body class="home page-template page-template-page-blank page-template-page-blank-php page page-id-15 lightb wpb-js-composer js-comp-ver-5.0.1 vc_responsive">
    <a class="skip-link screen-reader-text" href="#main">Skip to content</a>
    <div id="wrapper">
        <header id="header" class="header has-sticky sticky-jump">
            <div class="header-wrapper">
                <div id="masthead" class="header-main ">
                    <div class="header-inner flex-row container logo-left medium-logo-center" role="navigation">
                        <!-- Logo -->
... <div id="logo" class="flex-col logo"> == $0
                        <!-- Header logo -->
                    <a href="https://www.signal-telecom.com/" title="Signal Telecom - Live Broadcast Systems - TDTV S
...>
```

Relative xpath = `//div[@id='logo']` veya `//div[@class='flex-col logo']`
(Absolute xpath = `//body/div/header/div/div/div`)

2.Relative Xpath()

- `//tagName[@attributelsmi='attributeValue']`; => önemli!
- `//tagName` sadece tagname ile locate etmek için
- `//*[@type='email']` tag ismi önemli olmadan attribute'u type olarak "email" değeri alan elementleri verir
- **Exact Text**(Belirli bir text) ile element bulma:
 - `//tagname[.= 'text name']` ⇒ Belirli bir tagname , herhangi bir attribute, belirli bir text
 - `//*[@.= 'text name']` ⇒ Herhangi bir tag, belirli bir text.
 - `//*[@text()='exact text with extra space and all']` ⇒ Herhangi bir tag, belirli bir text
- **Belirli bir metni içeren** bir öğeyi bulmak için şunları kullanabiliriz:
 - `//*[@contains(text(),'piece of text')]`;
- **Birden fazla attribute yazabiliriz**
 - `//tag[@attribute1='value 1' and attribute2='value2']`

Example: `//div[@id='logo' or class='flex-col logo']`

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- Delete butonunun gorunur olmadigini test edin

8- By.cssSelector() Method

Css selector xpath'e benzer. Üç ana tip kullanılır

```
<input type="email" class="form-control" placeholder="Email" data-test="email" name="session[email]" id="session_email">
```

- **css = tagName[attribute name='value']**
 - `driver.findElement(By.cssSelector("input[name='session[email]']"));`
- **css="tagName#id value" or just css="#id value"=>yalnızca id value ile çalışır**
 - `driver.findElement(By.cssSelector("input#session_email"));`
- **css="tagName.class value" or just css=".class value"=>yalnızca class value ile çalışır**
 - `driver.findElement(By.cssSelector("input.form-control"));`

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.



DERS 4

Maven PROJE OLUSTURMA

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- daha once ogrendigimiz 6 locator daha spesifik webelementler icin kullanilabiliyordu.
- 2- xpath ve cssSelector ise tum webelementler icin kullanilabilir
- 3- xpath ve cssSelector her webelementinde bulunmasi zorunlu olan 3 ozelligi kullanarak unique webelement'e ulasmaya calisir. Bu uc ozellik tag ismi, attribute ismi ve attribute degeri'dir.
- 4- Eger bu ucluyu yazdigimizda birden fazla element bulunuyorsa index kullanabiliriz.
Index 1'den baslar
- 5- eger bir weblement link olmadigi halde text'e sahipse ozel bir xpath ile bu webelementi locate edebiliriz
`//tagIsmi[text()='textin tamami']`
- 6- locator yazilirken tagismi onemli degilse yerine * yazilabilir
- 7- xpath ve cssselector birbirine çok benzemekle birlikte xpath daha çok kullanılır



- Maven bir Java derleme aracıdır (build tool). Maven proje otomasyon ve yönetim aracıdır (automation and management tool).
- Maven ile tüm otomasyon sürecini yönetmek için maven projesi oluşturabilirsiniz.
- Maven, konfigürasyon için pom.xml dosyasını kullanır. Bu dosya projeyi build etmek için gerekli bütün bilgileri içerir (dependencies) .
- Maven, java uygulamalarını derlememize(compile), çalıştırılmamıza(run) ve yayinallyamamıza(deploy) yardımcı olur.
- Ant ve Gradle gibi başka araçlar da var ama Maven en popüler olanı.
- Apache Software Foundation

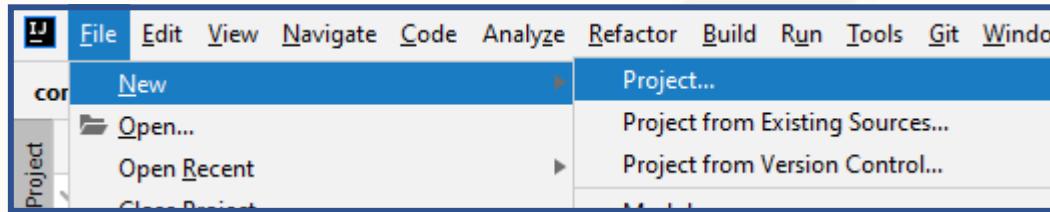


Neden Maven?

- Tekrarlanabilir derlemeler / yürütütmeler.(Repeatable builds/executions.)
- Maven 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



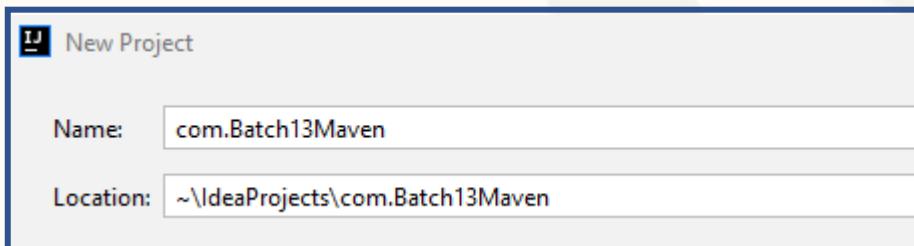
ILK Maven PROJESINI OLUSTURMA



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



2. Select Maven -> click next



Name: com.Batch30Maven -> finish -> click on
(EnableAutoImport)

3.Package name : day04 -> Right click on java create the package

4.class 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, sürücüler 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



POM XML DOSYASINA DEPENDENCIES EKLEME

- 1) Mvnrepository.com adresine gidelim
- 2) WebDriverManager aratalim (En guncel ve en cok kullanılan 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



POM XML DOSYASINA DEPENDENCIES EKLEME

The screenshot shows a development environment with two main windows. On the left is the code editor displaying the `pom.xml` file for a project named `com.asdf`. The code editor has syntax highlighting and shows the following XML structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>org.example</groupId>
    <artifactId>com.asdf</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <!-- 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>
        <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
        <dependency>
            <groupId>io.github.bonigarcia</groupId>
            <artifactId>webdrivermanager</artifactId>
            <version>3.8.1</version>
        </dependency>
    </dependencies>
</project>
```

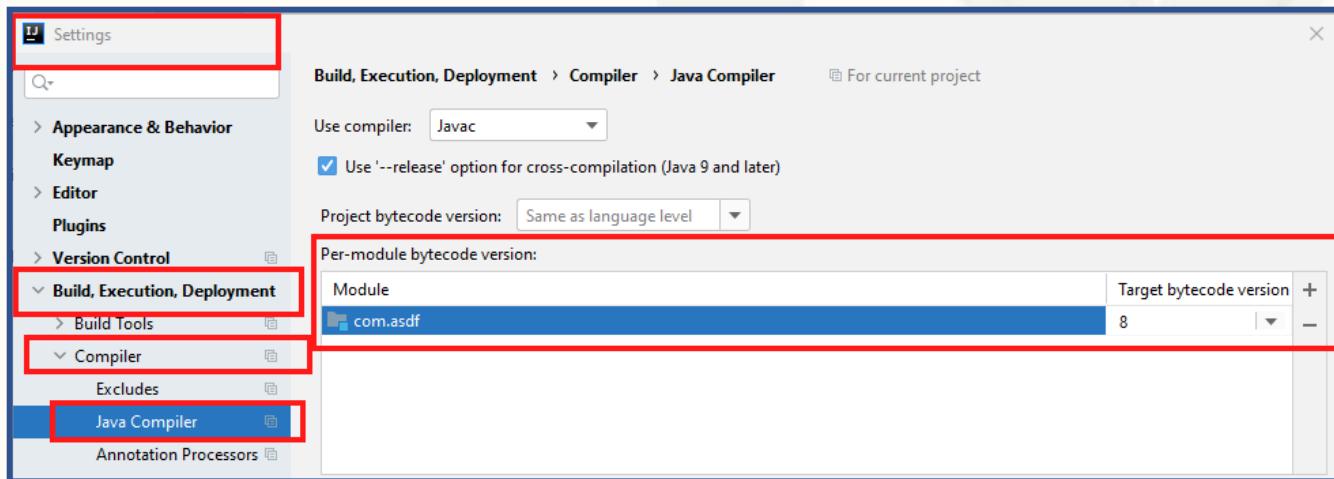
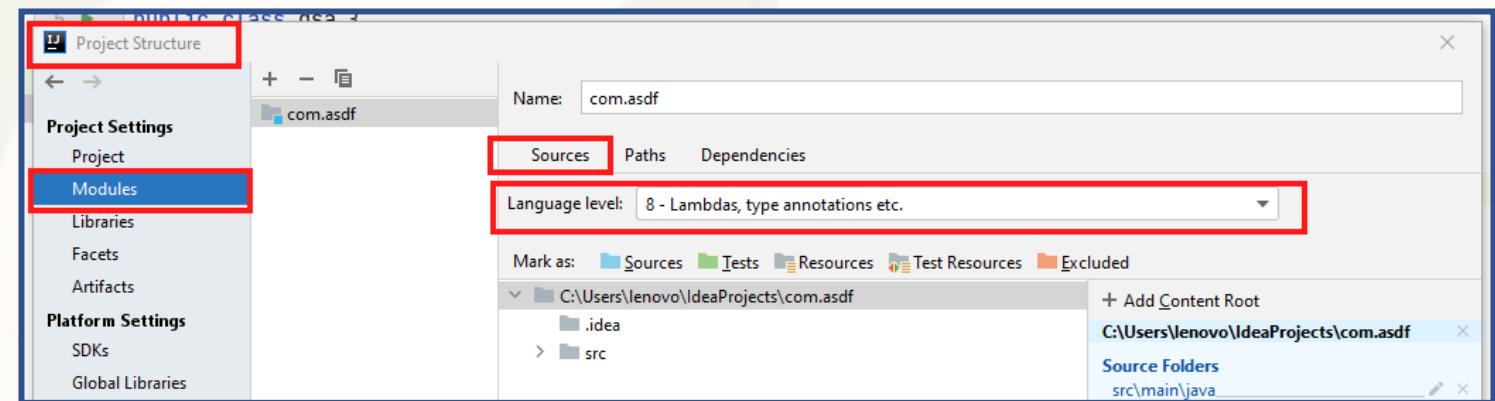
The right side of the interface is the Maven tool window, which displays the project structure and available Maven goals. A red box highlights the `Dependencies` section under the `com.asdf` project node, which lists the following dependencies:

- org.seleniumhq.selenium:selenium-java:3.141.59
- io.github.bonigarcia:webdrivermanager:3.8.1

Maven™

CLASS AYARLARI

File
Project Structure
Modules
Sources
Language level : 8 yapalim



File
Settings
Build,Execution,Deployment
Compiler
Java Compiler
Target bytecode version 8 yapalim



CLASS WebDriver AYARLARI

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

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-Test01 isimli bir class olusturun
- 2- <https://www.walmart.com/> adresine gidin
- 3- Browseri tam sayfa yapin
- 4-Sayfayı “refresh” yapın
- 5- Sayfa basliginin “Save” ifadesi icerdigini control edin
- 6-Sayfa basliginin “Walmart.com | Save Money.Live Better” a esit oldugunu control ediniz
- 7- URL in walmart.com icerdigini control edin
- 8-“Nutella” icin arama yapiniz
- 9- Kac sonuc bulundugunu yaziniz
- 10-Sayfayı kapatın



Soru 2

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



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

DERS 5

Junit FRAMEWORK OLUSTURMA

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Maven : javada yazdigimiz kodlari derlemek icin kullanılan bir tool'dur. project management tool
 - 2- Maven'in calisabilmesi icin pom.xml dosyasina ilgili dependencies eklenmelidir. Biz baslangic olarak olmazsa olmazimiz 2 dependency'yi pom.xml'e ekledik
 - selenium.java
 - WebDriverManager (Bony Garcia)
 - 3- pom.xml'e ekleyecegimiz dependencies'i mvnrepository.com adresinden yukleyebiliriz
 - 4- bir dependency icin yüklenecek versiyonu secerken stabil versiyonlar içerisinde en güncel ve en çok kullanılan versiyonu tercih ederiz.
 - 5- pom.xml sayesinde her class'da yeniden webdriver yolunu yüklemeye ve projelere jar dosyası yüklemeye gerek kalmaz
 - 6- implicitly wait : Bir class'daki her sayfa yükleme ve her locate işlemi için max. olarak parametre olarak kullandığımız sure kadar bekler. İşlem tamamlanınca surenin dolmasını beklemeden sonraki koda gecer
-

JUnit

- Java ile en temel framework JUnit ile oluşturulabilir.
- Developerlar unit testleri calistirmak icin kullanirlar.
- Biz testlerimizi yapmak icin JUnit'in ileri sürümü olduğundan TestNG framework oluşturup kullanacagiz.
- <https://mvnrepository.com/> sitesinden dependency leri alabilir ve test framework oluşturmak için Junit annotationlarini kullanabiliriz.
- En gelismis ve guncel Framework olarak Cucumber kurdugumuzda Junit'i yine kullanacagiz

JUnit

Annotations

- Java dilinde 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.
- Java dilinde notasyonlar "@" işaretıyla başlarlar. Annotationlar ile derleyiciye talimatlar verebiliriz.

En çok kullanılan Junit annotation'ları

@Test

@BeforeClass @AfterClass

@Before , @After

@Ignore

JUnit

Annotations

@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.

JUnit

Annotations

@Before ve @After

- Before notasyonu, her test metodundan önce çalışır. Örneğin bir sayfa ile test yapıyorsak ve her testten önce o sayfaya gitmemiz gerekiyorsa @before kullanabiliriz. @before notasyonunun kullanıldığı metoda genelde setup() ismi verilir
- After notasyonu, her test metodundan sonra çalışmaktadır. Örneğin test sırasında kullandığımız sayfanın kapatılması gibi. @after notasyonunun kullanıldığı metoda genelde teardown() ismi verilir
- @BeforeClass ve @AfterClass notasyonları test sürecinde bir kere çalışırken, @Before ve @After notasyonları her test metodunun başında ve sonunda çalışmaktadır.

JUnit

Annotations

@BeforeClass ve @AfterClass

- BeforeClass notasyonu, tüm testler öncesi yapılması gereken işlemlerde kullanılır (precondition). Örneğin test metodlarımız çalışmadan driver olusturup tüm methodlarda kullanabilirim.
- AfterClass notasyonu, tüm testler tamamlandıktan sonra yapılması gereken işlemlerde kullanılır. Örneğin actigimiz sayfayı kapatmak veya elde ettigimiz test sonuçlarını raporlamak gibi.
- Her iki metodunda oluşturulurken **public static void** olarak tanımlanması gerektiğini bilmemiz gerekmektedir.

Class Work

- 1-Test01 isimli bir class olusturun
- 2- <https://www.amazon.com/> adresine gidin
- 3- Browseri tam sayfa yapin
- 4-Sayfayı "refresh" yapın
- 5- Sayfa basliginin "Amazon" ifadesi icerdigini control edin
- 6-Sayfa basliginin "Amazon.com. Spend less. Smile more." a esit oldugunu control ediniz
- 7- URL in amazon.com icerdigini control edin
- 8-"Nutella" icin arama yapiniz
- 9- Kac sonuc bulundugunu yaziniz
- 10-Sayfayı kapatın

JUnit

CheckBox

1. Bir class oluşturun : CheckBoxTest
2. Gerekli yapıyı 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

JUnit

Radio Button

1. Bir class oluşturun : RadioButtonTest
2. Gerekli yapıyi olusturun ve aşağıdaki görevi tamamlayın.
 - <https://www.facebook.com> adresine gidin
 - “Create an Account” button'una basin
 - “radio buttons” elementlerini locate edin
 - Secili degilse cinsiyet butonundan size uygun olani secin

JUnit

DERS 6

Junit ASSERTIONS

Mehmet BULUTLUOZ
Elektronik Muh.

JUnit

Assertions: (Verification)

- JUnit framework'ünde test senaryomuzu çalıştırıldıktan sonra beklediğimiz sonuçların olup olmadığını tespit etmek için assert sınıfının fonksiyonlarını kullanırız.
- JUnit, belirli koşulları test etmek için isimleri assert ile başlayan statik metotları sağlar.
- Bu metotlarla beklenen ve gerçek değerleri karşılaştırarak testimizi sonuçlandırırız. Bu kıyaslama başarısız olursa AssertionException ile hata mesajı verilir.

Assert.**assertEquals**(actualResult,expectedResult)

Assert.**assertTrue**(yas>65) olduğunu doğrulayın

Assert.**assertFalse**(isim.equals("Ayse")) olmadığını doğrulayın

- Assertion fail olursa içinde bulunduğu testin kalan kısmını calistirmaz

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)

Assert.**assertTrue**(C>D)

Assert.**assertFalse**(B>D)

Assert.**assertEquals**(A,D)

Assert.**assertTrue**(C>B)

Assert.**assertFalse**(B>A)

JUnit

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 görüntüledigini test edin
 - FrancaisLinkTest => Fransızca Linkin görüntülediğini test edin

JUnit

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" olmadigini doğrulayın

JUnit

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 OZET

Assert, bir test senaryosunun PASS veya FAILED durumunu belirlemede kullanılan yararlı bir yöntemdir. Secilen metot ve yazılan **boolean kosul'a** göre test sonucu belirlenir.

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

En çok kullandığımız 3 Assert metodu;

1) Assert.assertTrue(**kosul**)

Yazılan kosul'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 ASSERT OZET

2) Assert.assertFalse(**koşul**)

Yazilan 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**)

Yazilan expected ile actual esit ise test PASS, yoksa test FAILED olur

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

True

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

False

TestNG

DERS 7

TestNG Framework Olusturma
Priority
Handle Dropdown

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Assertion : verification/dogrulama , Assert class'indan hazir static method'lar kullanarak test islemlerini az kodla yapabiliriz
- 2- E çok kullanılan assertion method'lari
 - Assert.assertEquals(prm1,prm2) : yazılan parametreler eşit ise test PASS
 - Assert.assertTrue(Boolean prm) : Boolean işleminin sonucu true ise test PASS
 - Assert.assertFalse (Boolean prm) : Boolean işleminin sonucu false ise test PASS
- 3- Assert method'lari kullandığımızda assertion failed olursa o satırdan sonrası çalışmaz (Execution stops)

TestNG

- Junit'in yeni 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 versiyonları gereklidir
- TestNG ile ilgili dokumanlara asagidaki adresden ulasilabilir

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

TestNG

- 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.
- TestNG ile daha fazla before ve after methodları kullanabilir
- 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 kullanılmasına imkan tanır.
- Error mesajlarının daha detaylı bir şekilde gösterilmesini sağlar.
- Multi-Browser Test
- Kullanisli HTML veya xml raporları olusturmaya yarar



Nasıl Yuklenir?

- 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.

```
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.0.0</version>
    <scope>test</scope>
</dependency>
```

TestNG

Proje Oluşturma

1. Create Project: File -> New -> Project-> Select maven -> click next
2. Isim: com.batch30TestNG->finish->cikarsa EnableAutoImport'u tiklayın
3. com.techproed'a Right click yapın ve "create package" seçin.

package ismi : **tests**

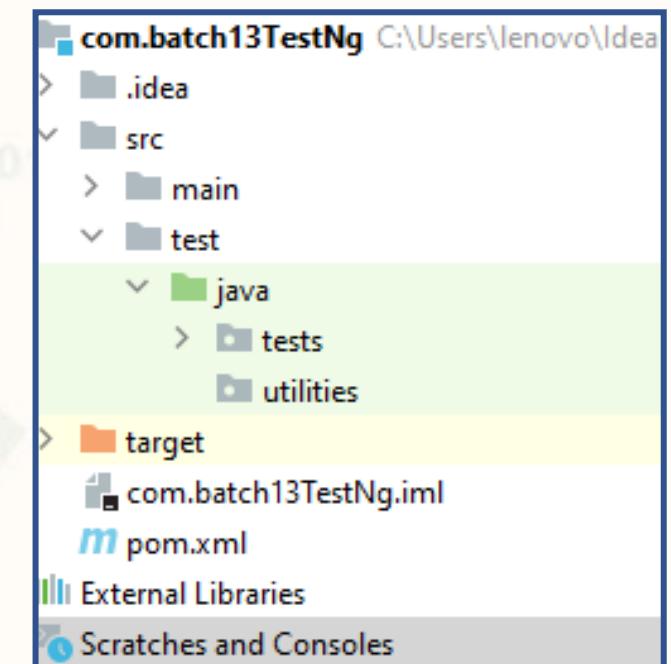
4. Bir daha com.techproed'a Right click yapın ve "create package" seçin.
package name : **utilities**

5. Right click on tests package and create a new package
package name: **day07**

6. Add the dependencies on you pom.xml file:

Get the dependencies mvnrepository.com

- a. selenium java,
- b. webdrivermanager,
- c. testNG



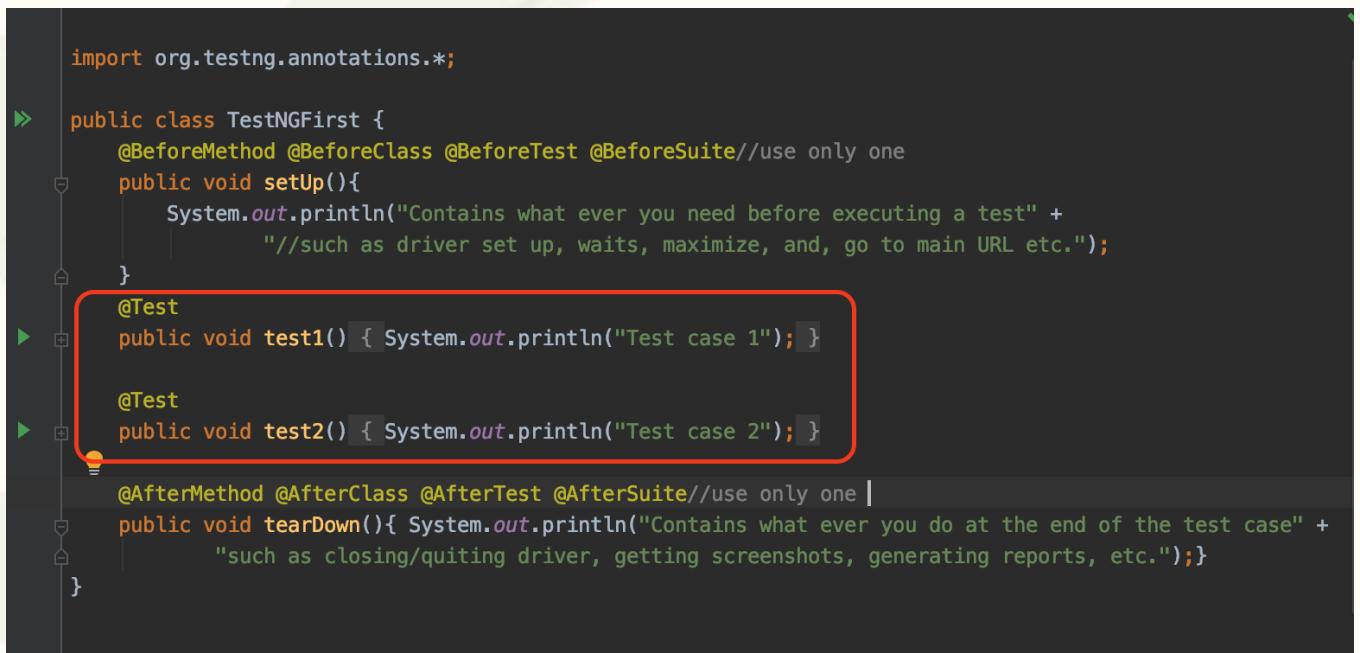
TestNG

@Test annotation

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

`@Test`

```
public void test1(){
    WebDriverManager.chromedriver().setup();
    WebDriver driver= new ChromeDriver();
    driver.get("https://www.google.com/");
}
```



```
import org.testng.annotations.*;

public class TestNGFirst {
    @BeforeMethod @BeforeClass @BeforeTest @BeforeSuite//use only one
    public void setUp(){
        System.out.println("Contains what ever you need before executing a test" +
                           "//such as driver set up, waits, maximize, and, go to main URL etc.");
    }

    @Test
    public void test1() { System.out.println("Test case 1"); }

    @Test
    public void test2() { System.out.println("Test case 2"); }

    @AfterMethod @AfterClass @AfterTest @AfterSuite//use only one
    public void tearDown(){ System.out.println("Contains what ever you do at the end of the test case" +
                                              "such as closing/quiting driver, getting screenshots, generating reports, etc.");}
}
```



@Before @After Annotations

- **@BeforeSuite @AfterSuite** => runs before/ after all tests in this suite
- **@BeforeTest @AfterTest.** => run before/after all the test methods after Test
- **@BeforeGroups @AfterGroups** =>run before/after any specific test group
- **@BeforeClass @AfterClass** =>run before/ after all the test methods in a test class

NOT : Bu notasyonlar kullanilinca her method icin ayri ayri calismaz, soylendigi sekilde sadece bir kere calisir. (JUnitteki @BeforeClass ile ayni sekilde)

- **@BeforeMethod @AfterMethod** =>run before/ after each test method

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

TestNG

@Before @After Annotations

```
import org.testng.annotations.*;

public class TestNGFirst {
    @BeforeMethod @BeforeClass @BeforeTest @BeforeSuite//use only one
    public void setUp(){
        System.out.println("Contains what ever you need before executing a test" +
                           "//such as driver set up, waits, maximize, and, go to main URL etc.");
    }

    @Test
    public void test1() { System.out.println("Test case 1"); }

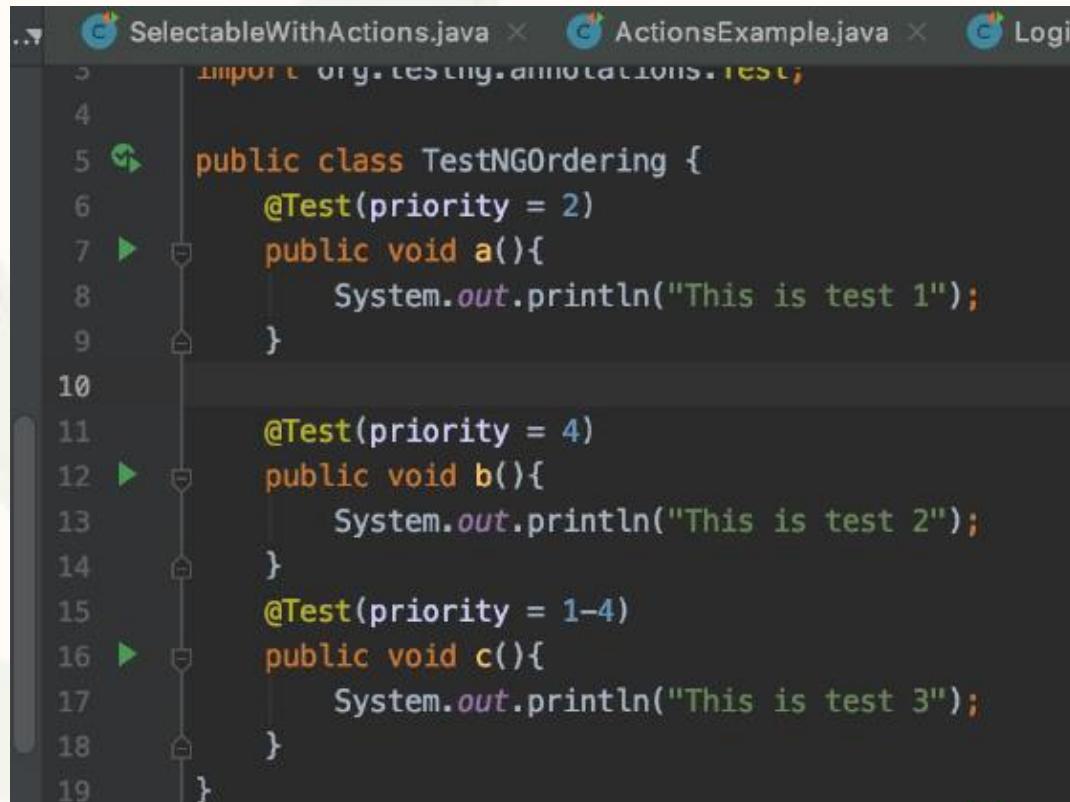
    @Test
    public void test2() { System.out.println("Test case 2"); }

    @AfterMethod @AfterClass @AfterTest @AfterSuite//use only one
    public void tearDown(){ System.out.println("Contains what ever you do at the end of the test case" +
                                              "such as closing/quiting driver, getting screenshots, generating reports, etc.");}
}
```

TestNG

Priority

- 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 oncelikle o calisir, sonra priority yazan testler siralanir



The screenshot shows a Java code editor with three tabs: 'SelectableWithActions.java', 'ActionsExample.java', and 'Login'. The code in 'SelectableWithActions.java' demonstrates TestNG ordering:

```
1 import org.testng.annotations.Test;
2
3 public class TestNGOrdering {
4
5     @Test(priority = 2)
6     public void a(){
7         System.out.println("This is test 1");
8     }
9
10    @Test(priority = 4)
11    public void b(){
12        System.out.println("This is test 2");
13    }
14
15    @Test(priority = 1-4)
16    public void c(){
17        System.out.println("This is test 3");
18    }
19}
```

TestNG

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 erişilebilir olduğunu test edin (isEnabled())
 - wrongTitleTest => Sayfa basliginin “youtube” olmadığını doğrulayın

TestNG

Handle Dropdown

- Adım1: Dropdown menüyü herhangi bir locator ile locate edin.

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

- Adım 2: Yeni bir "Select" objesi oluşturun ve daha önce locate ettigimiz WebElement'i parameter olarak yeni objeye ekleyin

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

- Adım 3: Dropdown için kullanılan 3 yöntemden biriyle dropdown menusundeki elemanlardan istediğinizini seçin

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

TestNG

Handle Dropdown

Dropdown menusundeki elementleri 3 sekilde secebiliriz

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

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

TestNG

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.

TestNG

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

TestNG

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 seçin
9. "amount" kutusuna bir sayı girin
10. "US Dollars" in secildigini test edin
11. "Selected currency" butonunu seçin
12. "Calculate Costs" butonuna basin sonra "purchase" butonuna basin
13. "Foreign currency cash was successfully purchased." yazısının çıktığını kontrol edin.

TestNG

dependsOnMethods

- ✓ Bu yontem, bir metodun diğer bir metoda bağlı olmasını sağlamak için kullanılır.
- ✓ Örnekte, searchTest metodu HomePage metoduna bağlıdır. Yani, HomePage başarılı olursa searchTest'in çalıştırılacağı anlamına gelir.
- ✓ Diğer bir deyişle, HomePage başarısız olursa searchTest'in çalışmayaceği (searchTest yok sayılır) anlamına gelir.
- ✓ Yalnızca searchTest yöntemini çalıştırırsak, TestNG önce HomePage metodunu çalıştırır. HomePage başarılı olursa searchTest yürütülür

The screenshot shows a Java code editor with the following code:

```
8  public void login(){
9      System.out.println("This is login test");
10 }
11
12 @Test
13 public void HomePage(){
14     System.out.println("This is home page test");
15     Assert.assertTrue( condition: false); //failing this test
16 }
17
18 @Test(dependsOnMethods = "HomePage")//Meaning searchTest method will run
19 //if HomePage passes. Will be skipped if HomePage method fails.
20 public void searchTest() {
21     System.out.println("This is search test");
22 }
23
24 @Test
25 public void resultTest() {
26     System.out.println("This is test result test");
27 }
```

The code uses the `@Test` annotation with the `dependsOnMethods` parameter set to "HomePage". This means that the `searchTest()` method will only be executed if the `HomePage()` method passes. If `HomePage()` fails, `searchTest()` will be skipped.

Below the code, the IDE's test results window shows the execution details:

- Tests failed: 1, passed: 2, ignored: 1 of 4 tests – 52 ms
- mymaven 52 ms
- TestNG 52 ms
 - hom 39 ms
 - login 9 ms
 - resul 4 ms
 - searc 0 ms
- This is home page test



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 tiklayın ve fiyatinin \$16.83 oldugunu test edin

TestNG

DERS 8

Assertions
Soft Assert

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- priority : TestNG biz mudahale etmesek testleri alfabetik siraya (Natural Order) gore yapar. Eger istedigimiz siralama ile calistirmak istersek test method'larina priority degerleri atayabiliriz. Priority degeri kucuk olan once calisir.
Eger priority degeri girilmeyen method'lar varsa, onlari 0 olarak Kabul eder ve diger degerlere gore 0'in sirasi geldiginde natural order'a uygun olarak calistirir.
- 2- DropDown menuden secim yapmak icin Select class\indan obje olusturup kullanmamiz gerekir. Dropdown menuden istenen option secimini 3 adimda yapariz
 - Dropdown menunun oldugu Webelement'i locate ederiz
 - Locate ettigimiz bu webelement'ini parameter olarak kullanip select objesi olustururuz
 - Select class'inda var olan hazir static method'lardan istedigimizi kullanarak option'lardan secimimizi yapariz
- 3- DependsOnMethods : Birden fazla test method'u calistirirken method'lar birbirine bagimli ise dependsOnMethods kullaniriz. Baglanilan test calisip, assertion PASS olmazsa baglanan test calismaz (ignore olur).
Eger 2 test bu method'la baglanmissa biz baglanan method'u tek basina calistirmak istedigimizde once gidip baglanilan method'u calistirir ama 2'den fazla ise zincir olarak calismaz.

TestNG

Assertions

Otomasyonun temel noktalarından biri Assertions'dır.

- Her bir test case'den sonra bir tür Assertion veya Verification kullanmalıyız.
- Bu bir nevi ileri seviye if else statement'dır.
- TestNG ile iki çeşit Assertion yapmak mümkündür.

```
public class Assertions_Hard_And_Soft extends TestBase {  
  
    @Test  
    public void hardAssert(){  
        driver.get("https://www.google.com/");  
        String googleTitle=driver.getTitle();  
        Assert.assertEquals( actual: "google",googleTitle);  
        System.out.println("Running after hard assert");  
    }  
  
    @Test  
    public void softAssert(){  
        driver.get("https://www.google.com/");  
        String googleTitle=driver.getTitle();  
        SoftAssert softAssert=new SoftAssert(); //create soft assert object  
        System.out.println("Beginning of soft assert");  
        softAssert.assertEquals( actual: "google",googleTitle);  
        System.out.println("Running after first assert");  
        softAssert.assertEquals( actual: "3", expected: "5");  
        System.out.println("Running after second assert");  
        softAssert.assertTrue( condition: true);  
        System.out.println("Running after third assert");  
        softAssert.assertAll(); //This is very important and main reason for soft assertion.  
        //We use .assertAll() at the end. This marks which soft asserts passed, and which ones failed.  
        System.out.println("Running after soft assertAll() assert");  
    }  
}
```

TestNG

Assertions

1. HARD ASSERT

JUnit'te Öğrendiğimiz Assertion Türü

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

Bu assertion başarısız olursa, test method'nun geri kalanını durdurur ve yürütmez.

Tester'lar olarak test case'in neden başarısız olduğunu hemen anlamak için hard assertion'u tercih ederiz.

2. SOFT ASSERT (VERIFICATION)

Eğer softAssert başarısız olursa test method'unun geri kalanını durdurmaz ve yürütmeye devam eder. If else statement'da olduğu gibi.

TestNG

Soft Assert

- SoftAssert doğrulama (verification) olarak da bilinir.
- SoftAssert kullanabilmemiz için object create etmemiz gereklidir.
 - **1.Adım:** SoftAssert objesi olusturalim

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

- **2.Adım:** İstedigimiz verification'ları yapalim

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

- **3.Adım:** SoftAssert'in durumu raporlamasını isteyelim

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

TestNG

Soft Assert & Hard Assert

- Benzerlikleri

SoftAssert ve HardAssert method'ları TestNG'den gelmektedir. Kullanma amaçları farklı fakat method'lar aynıdır.

- Farkları

- Eğer Assertion fail olursa test method'larının execute etmesi durur. Ve FAILED olarak tanımlanır.
- Eğer Verification FAIL olursa test method'ları execute etmeye devam eder.

TestNG

Soft Assert Class Work

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 olduğunu 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)"

TestNG

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

TestNG

DERS 9

Handle Alerts
Handle iFrame
Test Base Class

Mehmet BULUTLUOZ
Elektronik Muh.

TestNG

Handle Alerts

Alert Nedir?

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

Click for JS Alert

Click for JS Confirm

Click for JS Prompt

- 1. Simple Alert :** Bu basit alert ekranda bazı bilgiler veya uyarılar görüntüler. Ok denilerek kapatılır
- 2. Confirmation Alert :** Bu onay uyarısı bir tür işlem yapma izni ister. Alert onaylanıyorsa OK, onaylanmıyorsa Cancel butonuna basılır.
- 3. Prompt Alert :** Bu Prompt Uyarısı kullanıcıdan bazı girdilerin girilmesini ister ve selenium webdriver metni sendkeys ("input....") kullanarak girebilir.

https://the-internet.herokuapp.com/javascript_alerts

TestNG

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

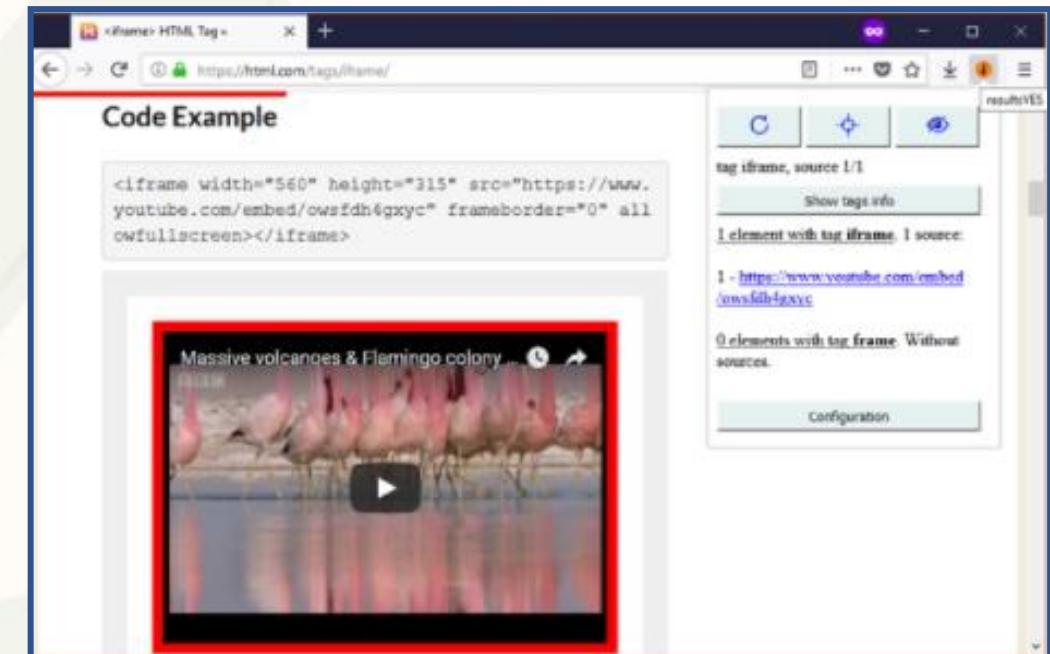
- Bir class olusturun: **D12_Alerts**
- https://the-internet.herokuapp.com/javascript_alerts adresine gidin.
- Bir metod olusturun: **acceptAlert**
 - 1. butona tıklayın, uyarıdaki OK butonuna tıklayın ve result mesajının “You successfully clicked an alert” olduğunu test edin.
- Bir metod olusturun: **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.

TestNG

Handle Iframe

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/>

TestNG

Handle Iframe

- Bir sayfada iframe varsa, Selenium bir iframe içindeki elementleri doğrudan göremez.
- Switching to iframe: iframe geçmenin 3 yolu vardır;
 - **index ile :**
`driver.switchTo().frame(index of the iframe); //index start from 0`
 - **id veya name value ile**
`driver.switchTo().frame("id of the iframe");`
 - **WebElement ile**
`driver.switchTo().frame(WebElement of the iframe);`

TestNG

Handle Iframe

Iframe'den cikmak icin

`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/>

TestNG

Handle Iframe Class Work

- Bir class olusturun: D12_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.

TestNG

Handle Iframe Class Work

- 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

TestNG

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.
- oluşturduğumuz 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().timeouts().pageLoadTimeout(10, TimeUnit.SECONDS);
        driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);
    }
    @AfterMethod
    public void tearDown(){
        driver.quit();
    }
}
```

TestNG

DERS 10

Handle Windows
Actions Class

Mehmet BULUTLUOZ
Elektronik Muh.

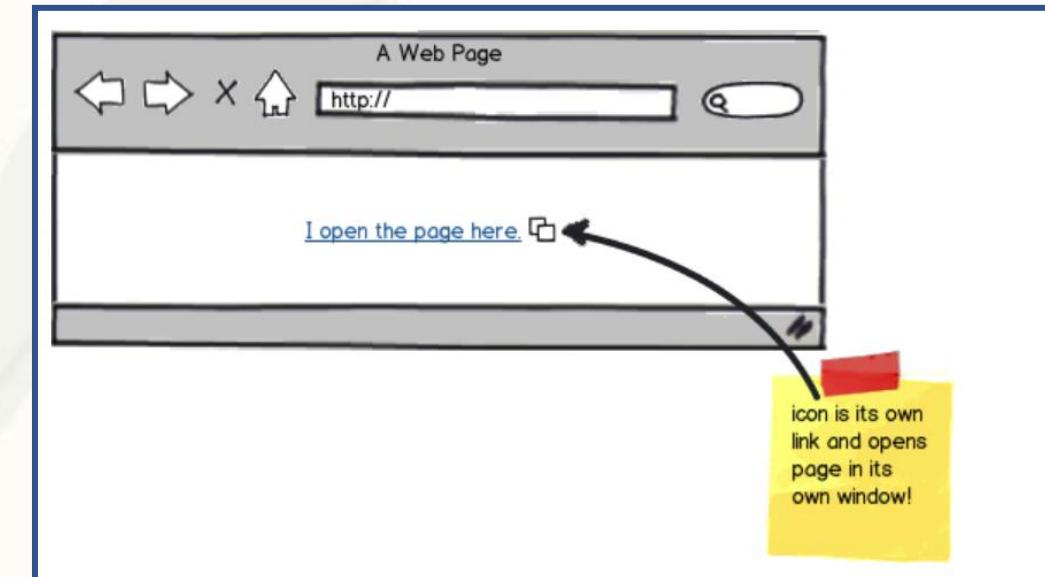
Onceki Dersten Aklimizda Kalanlar

- 1- JS alert : karsimiza cikan her alert JS Alert degildir. Eger sag click yaptigimizda inspect yapabiliyorsak o alert HTML alert'dur ve siradan bir webelement olarak handle edileilir. Ancak sag click yapamiyorsak o alert JS Alert'dur ve handle edebilmek icin once switchTo().alert methodunu kullanmamiz ve sonra yapacagimiz islemi (accept, dismiss, getText, sendKeys) yapabiliriz
- 2- iFrame : Bir web sayfasi icerisinde baska bir web sayfasi varsa iFrame ile HTML kodu yazilir. Iframe icerisindeki bir web elemente ulasmak icin once switchTo.iFrame diyerek ilgili iFrame'e gecis yapmamiz gerekir.
iFrame'e gecis icin 3 yonten vardir. Index, id veya name, webelement
iFrame'e gecis yaptiktan sonra yeniden ana sayfaya donmek istersek iFrame'den cikmamiz gerekir. Bunun icin defaultContent(en ustte) veya parentFrame (bir ust) kullanilabilir.
- 3- TestBase Class : Bizim framework olarak temel hedefimiz tekrar eden kodlari ve test datalarini tekrar tekrar yazmaktan kurtulmak. Bunun ilk adimi olarak test base class'ini kullaniyoruz.
Test Base class'inda webDriver, beforeClass(), afterClass(), beforeMethod() veya afterMethod() method'lari olabilir. Ihtiyacimiza gore birden fazla testBase class'i olusturabiliriz.

TestNG

Tabs(Sekmeler) Handle Windows

- Bazen bir butona tıkladığımızda, başka bir sekmede(tab) yeni bir pencere açılır.
- Birden fazla pencereyle uğraşırken driver'ı pencereler arasında değiştirmemiz gereklidir.
- Selenium WebDriver, WebDriver objesi başlatıldığından her pencereye bir alfanumerik kimlik atar. Bu benzersiz(unique) alfanumerik kimliğe pencere tanıtıcısı(window handle) denir. Selenium, birkaç pencere arasında kontrolü değiştirmek için bu benzersiz kimliği kullanır.



TestNG

Tabs(Sekmeler) Handle Windows

- **switchTo()** => pencereler arasında geçiş yapma(switch) metodu
- **getWindowHandle()** => WebDriver'ın o anda kontrol ettiği sayfanın window handle'ini almak için kullanılır. String return eder.
- **getWindowHandles()** =>Açık olan tüm pencerelerin window handle'larini almak icin kullanılır. Set return eder.
- Syntax: **driver.switchTo().window(driver.getWindowHandle());**

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

TestNG

Handle Windows Class Work

- Tests package'ında yeni bir class olusturun: D13_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.

Handle Windows Ozet

Birden fazla window acilan durumlarda ilk sayfadan baslayarak adim adim her sayfanin handle degerini alip kaydetmemiz gerekir

- 1.Adim : sadece 1 sayfa var iken o sayfanin windowhandle degerini alip String bir degiskene kaydediyoruz
- 2.Adim : yeni sayfa acildiktan sonra acik olan tum sayfalarin windowHandle degerlerini alip bir Set'e atama yapiyoruz.
- 3.Adim : Set'deki elemanlardan 1.sayfa handle'ina esit olmayani ikinci sayfanin window Handle degeri olarak bir String'e atama yapiyoruz.
- 4.Adim: hem 1.sayfanin hem de 2.sayfanin window handle degerleri elimizde oldugu icin istedigimiz sayfaya gecis yapabiliriz
`driver.switchTo().window("gecmek istedigimiz sayfanin handle degeri")`

TestNG

Actions Class

- Fare ve klavye eylemlerini gerçekleştirmek için Actions class'ı kullanmamız gereklidir.
- Actions Class birçok kullanışlı fare ve klavye işlevine sahiptir.
- Çift tıklama (double click), sürükleme ve bırakma(drag and drop), fareyi hareket ettirme (mouse actions) gibi karmaşık fare eylemleri için veya Keyword ile yapabileceğimiz pageUp, pageDown, shift, arrowDown gibi işlemleri Actions classından object ureterek yaparız.



TestNG

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 sağ tıklamak için.

actions.contextClick(element).perform();

perform() En sonda action'i gerçekleştirmek için kullanılır

```
@Test
public void RightClick() {
    driver.get("http://demo.guru99.com/test/simple_context_menu.html");
    WebElement rightClick=driver.findElement(By.xpath("//*[text()='right click me']"));
    Actions actions=new Actions(driver);
    actions.contextClick(rightClick).perform(); //pass the webelement that you want to right click.
    WebElement editTextOption=driver.findElement(By.xpath("//*[text()='Edit']"));
    editTextOption.click();
    driver.switchTo().alert().accept();
}

@Test
public void doubleClick() {
    driver.get("http://demo.guru99.com/test/simple_context_menu.html");
    WebElement doubleClick=driver.findElement(By.xpath("//*[text()='Double-Click Me To See Alert']"));
    Actions actions=new Actions(driver);
    actions.doubleClick(doubleClick).perform(); //pass the web element that you want to right click.
    driver.switchTo().alert().accept(); //switching and accepting alert.
}
```

TestNG

Mouse Base Actions Mouse Aksiyonları

- **doubleClick ()**: Öğeye çift tıklama yapar
- **clickAndHold ()**: Fareyi serbest bırakmadan uzun tıklama yapar
- **dragAndDrop ()**: Öğeyi bir noktadan diğerine sürüklər ve bırakır
- **moveToElement ()**: Fare işaretçisini ögenin ortasına kaydırır
- **contextClick ()**: Fare üzerinde sağ tıklama yapar



TestNG

Mouse Base Actions Class Work

- 1- Yeni bir class olusturalim: C03_MouseActions1
- 2- https://the-internet.herokuapp.com/context_menu sitesine gidelim
- 3- Cizili alan uzerinde sag click yapalim
- 4- Alert'te cikan 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: D14_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

TestNG

Mouse Base Actions Class Work

Yeni bir class olusturalim: C05_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

TestNG

DERS 11

Actions Class
File İşlemleri
Synchronization

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Window Handle : Birden fazla window acilan durumlarda ilk sayfadan baslayarak adim adim her sayfanin handle degerini alip kaydetmemiz gerekir
 - 1.Adim : sadece 1 sayfa var iken o sayfanin windowhandle degerini alip String bir degiskene kaydediyoruz
 - 2.Adim : yeni sayfa acildiktan sonra acik olan tum sayfalarin windowHandle degerlerini alip bir Set'e atama yapiyoruz.
 - 3.Adim : Set'deki elemanlardan 1.sayfa handle'ina esit olmayani ikinci sayfanin window Handle degeri olarak bir String'e atama yapiyoruz.
 - 4.Adim: hem 1.sayfanin hem de 2.sayfanin window handle degerleri elimizde oldugu icin istedigimiz sayfaya gecis yapabiliriz

driver.switchTo().window("gecmek istedigimiz sayfanin handle degeri")

- 2- Actions class : Klavye ve mouse ile manuel olarak yaptigimiz islemleri otamasyon ile yapabilmemiz icin hazirlanmis olan bir class'dir.
 - Bu class'daki hazir method'lari kullanmak icin oncelikle bu class'dan obje olusturmaliyiz ve bu objeye parameter olarak driver'imizi yazmaliyiz.
 - Uzerinde islem yapacagimiz webelementimizi locate etmeliyiz
 - actions objesi ve webElementi kullanarak istedigimiz aksiyonu yapip sonuna .perform() eklemeliyiz

TestNG

Keyboard Base Actions Klavye Aksiyonları

Klavye aksiyonları ile klavyedeki tusları kontrol edebiliriz. Bunun için 3 tane aksiyon kullanırız.

- **sendKeys ()**: Öğeye bir dizi anahtar gönderir
- **keyDown ()**: Klavyede tuşa basma işlemi gerçekleştirir
- **keyUp ()**: Klavyede tuşu serbest bırakma işlemi gerçekleştirir



TestNG

Keyboard Base Actions Class Work

1- Bir Class olusturalim C01_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





Keyboard Base Actions

Class Work

1- Bir Class olusturalim C02_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

TestNG

Keyboard Base Actions Home Work

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*

TestNG

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");` bulunulan klasörün yolunu (Path) verir
 - `System.getProperty ("user.home");` kullanıcı 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



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

TestNG

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



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.

TestNG

DERS 12

Synchronization
Faker Kutuphanesi

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

1- Keyboard actions : klavyede çok fazla tuş olduğu için her tuş için ayrı method yazmak yerine 3 method'la tüm tuşları kullanabiliriz

- sendKeys(istenen tuş) istenen tusa sadece 1 defa basmak için kullanılır
- keyDown : istenen tusa basılı tutmamızı sağlar
- keyUp : basılı olan tustan elimizi kaldırırmamızı sağlar

2- File işlemleri : Selenium sadece web uygulamaları için kullanılır, bilgisayarımızda bulunan dosyalara erişim konusunda selenium ile işlem yapamayız/
Bilgisayarımızdaki dosyalara erişim ve onları değiştirmek istediğimizde Java kullanırız.
Dosya erişiminde kullanılan exist, fileInputStream, fileOutputStream gibi komutları kullanabilmek için dosya yoluna ihtiyaç duyuyoruz

Her bilgisayarın dosya yolu farklı olduğundan bu işlemlerin dinamik olarak tüm bilgisayarlarda çalışması adına dosya yolundaki değişken kısımları Java vasıtasiyla alırız

`System.getProperty("user.home")`

Home kısmından sonra dosya yolu genellikle ortak olduğundan o kısmı hard coding ile yazabiliriz

`System.getProperty("user.dir")` bu komut ile de testlerimizi çalıştırdığımız clasorun dosya yolunu alabiliriz

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.
- Bu sorunu çözmek için, sürücümüzün(driver) yavaşlaması için ek bir bekleme(wait) yapmamız gereklidir.
 - **Thread.sleep** : Javadan gelir ve kodları yazılan sure kadar bekletir.
 - **Implicitly Wait**: Sayfadaki tüm öğeler için global bir zaman aşımıdır(timeout).
 - **Explicitly Wait**: Çoğunlukla belirli öğeler için belirli bir koşul(expected condition) için kullanılır.

TestNG

IMPLICITLY WAIT

- Bir sayfadaki tüm öğeler için belirli bir süre sürücü (driver) bekler. Tester'lar genellikle otomasyon frameworklerinde sürücü nesnesini (driver object) varsayılan olarak implicitly wait ile bekletir. Implicitly wait'i TestBase class'ımızda kullanıyoruz.
- Implicitly wait'de, tıklanabilirlik(clickability), görünürlük(visibility) vb. gibi beklenen bir koşul yoktur.

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

- Bu, sürücünün (driver) sayfadaki herhangi bir öğe(web element) için 10 saniye beklemesini istediğimiz anlamına gelir.
- Web element 10 saniyeden kısa sürede yüklenirse driver bulur ve devam eder. Örneğin, Web element 3 saniye içinde yüklenirse, driver sadece 3 saniye bekleyecektir ve bir sonraki satırı geçecektir.
- Web element 10 saniye içinde yüklenmezse, test case başarısız olur ve **NoSuchElementException** uyarısı verir.

TestNG

EXPLICIT WAIT

- Beklenen bir durum(expected condition) olduğunda explicit wait kullanılır. Bu, daha fazla bekleme süresi gerektiren belirli öğeler(web element) için kullanılır.
- İlk olarak belirli miktarda bekleme süresi ile wait object create edilir.

wait object'i olusturma

```
WebDriverWait wait = new WebDriverWait(driver, 20);
```

WebElement icin wait object'i kullanma

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

TestNG

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()

TestNG

Explicit Wait Class Work

1. Bir class olusturun : C01_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 doğrulayın.
6. Add buttonuna basin
7. It's back mesajinin gorundugunu test edin

TestNG

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.

TestNG

Explicit Wait Class Work

1. Bir class olusturun : ExplicitlyWaitTest
2. Bir metod olusturun : enableTest()
3. <https://demoqa.com/dynamic-properties> adresine gidin.
4. Will enable 5 seconds'in etkin olmadigini(enabled) test edin
5. Will enable 5 seconds etkin oluncaya kadar bekleyin ve enabled oldugunu test edin
6. Bir metod olusturun : visibleTest()
7. Ayni sayfaya tekrar gidin, Visible After 5 Seconds butonunun gorunmesini bekleyin, ve gorunur oldugunu test edin

TestNG

Faker Kutuphanesi

1. "https://facebook.com" Adresine gidin
2. "create new account" butonuna basin
3. "firstName" giris kutusuna bir isim yazin
4. "surname" giris kutusuna bir soyisim yazin
5. "email" giris kutusuna bir email yazin
6. "email" onay kutusuna emaili tekrar yazin
7. Bir sifre girin
8. Tarih icin gun secin
9. Tarih icin ay secin
10. Tarih icin yil secin
11. Cinsiyeti secin
12. Isaretlediginiz cinsiyetin secili, diger cinsiyet kutusunun secili olmadigini test edin.
13. Sayfayı kapatın



Actions Class Home Work

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



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

TestNG

DERS 13

POM
Page Object Model

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Syncronization : Kodlarimizla internet arasında zamanlama farkları olusursa aradaki senkronizasyonu saglamak icin wait'ler kullanılır
 - Thread.sleep() : kodları yazılan sure kadar bekletir. Sure konusunda kesindir, az veya çok beklemez
 - Implicitly wait : Tüm test case'imizi kapsayan esnek bir beklemedir. Sayfanın yüklenmesi veya locate edilecek herbir element için yazılıaan max sure kadar bekler, o sure içerisinde sayfa yüklenemez veya webelement locate edilemezse hata verir
 - explicitly wait : esnek olmakla beraber sadece bir element veya bir işlem için tanımlanmış beklemedir. Bunu kullanabilmek için oncelikle wait objesi oluşturmamız gereklidir. Wait objesi ile birlikte kullanılan method'a bağlı olarak önce locate edilip sonra beklenebilir veya iki işlem tek satırda beraberce yapılabilir
- 2- Faker Kutuphanesi : Pom.xml e ekledigimiz faker kutuphanesi sayesinde isim, soyisim, mail, doğum tarihi gibi bilgileri selenium'a ürettirebiliyoruz.
Bunun icin faker objesi uretip, bu objeyi kullanarak FAKER class'ından hazır method'lara ulaşabiliyoruz.

TestNG

Java'dan Hatırlamamız Gerekenler

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

A- inheritance (Miras) kullandığımız 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 methodları kullanabiliriz. (Test Base gibi)

B- Object olusturarak istedigimiz class'a ulasabilir ve o class'daki variable ve methodları object'imizi aracılığıyla kullanabiliriz
ornek: Okul class'ına ulaşmak istiyorsak

Okul okul=new Okul(); diyerek object uretir, sonra
okul.variable diyerek variable'lara
okul.method() diyerek de methodlara ulaşabiliriz

C- Eger kullanacagimiz variable veya method static ise object olusturmadan direkt class ismi ile variable veya method'a ulaşabiliriz.

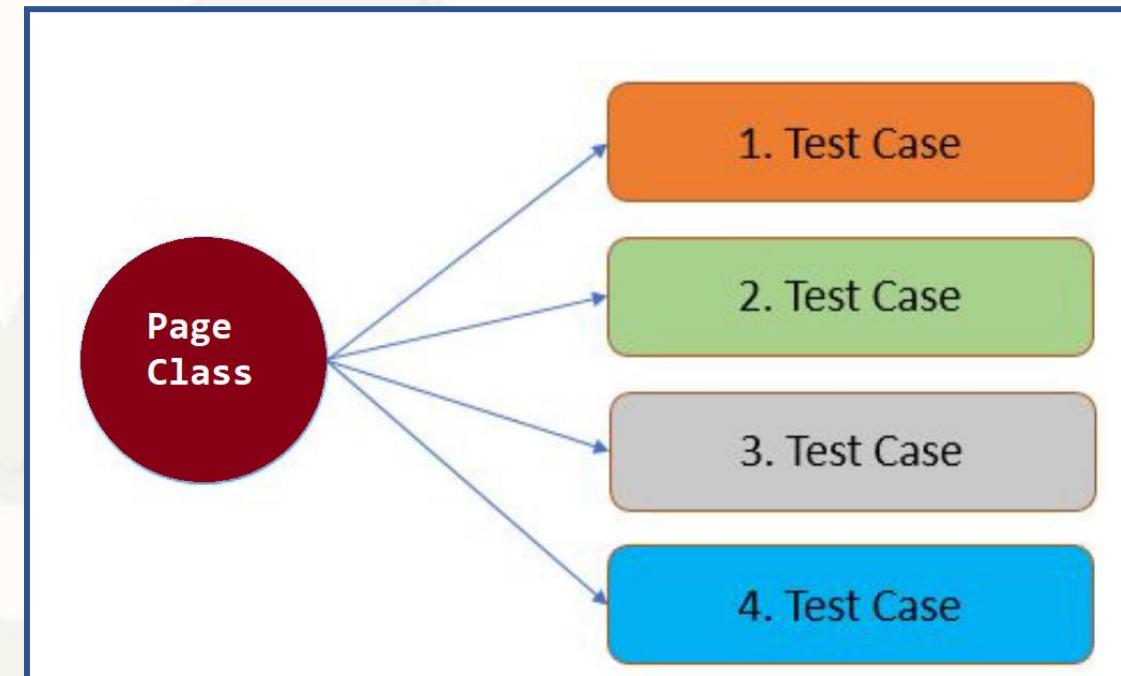
Okul.staticVariable ile variable'lara
Okul.staticMethod() ile methodlara ulaşabiliriz.

TestNG

Page Object Model

Framework design

- 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 ulaşabilir veya test dataları ile ilgili bir değişiklik yapmamız gerektiginde sadece kaynak dosyadan bir değeri değiştirek tüm test case'leri güncellemis oluruz.





Page Object Model

Framework design

- Bu 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.
- Framework un verimliliğini artırmak için core Java ve Selenium konseptini kullanarak temel olarak page classları ve test classları oluşturacağız.
- Tüm şirketler page object model dizaynini kullanmaz, ancak herkes bunu bilir ve daha da popüler hale gelmektedir.



Page Object Model

Framework design

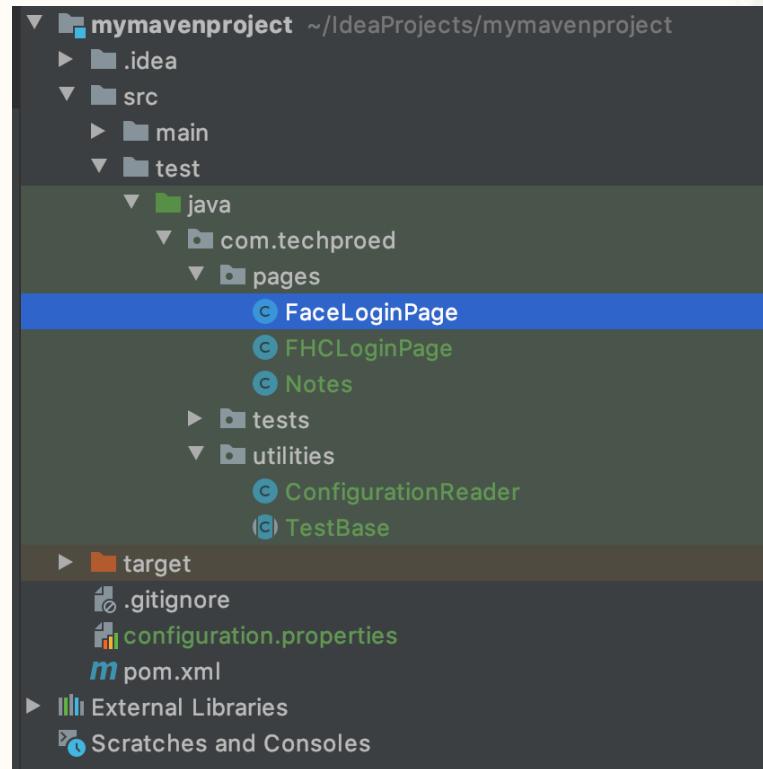
Page Object Model Nedir?

- Test case'leri ve kodları oluştururken Otomasyon framework'un (STLC nin parçası) korunması zaman içinde zor olabilir.
- Bir uygulamanın(application) işlevselligi değiştiğinde, kodu düzeltmek için framework kontrol etmelidir.
- Page Object Design daha bağımsız test senaryoları oluşturmamıza yardımcı olur, böylece test komut dosyalarında(script) hata ayıklamak daha kolay olacaktır.
- Page Object Design zorlukları azaltmak amacıyla uygulamaları test etmek için kullanılan bir automation framework design dir.
- Page Object Design objectleri, classları, methodları, dataları, etc yeniden kullanılmayı mümkün kılar. Böylece testerler daha az iş yapar.
- Sadece bir kez create edip birden çok kez kullanıyoruz.
- Daha iyi bir tasarım, testin yürütme süresini daha hızlı hale getirir.

TestNG

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

TestNG

Page Object Model

Framework design

facebook

Facebook tanıdıklarınla iletişim kurmanı
ve hayatında olup bitenleri paylaşmanı
sağlar.



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

TestNG

Page Object Model

Framework design

PAGE CLASS

```
public class FaceLoginPage {  
  
    WebDriver driver;  
    public FaceLoginPage(WebDriver driver) { //We need this  
        this.driver=driver; //to connect the page elements  
        PageFactory.initElements(driver, page: this);  
    }  
  
    @FindBy(id = "email")  
    public WebElement username;  
  
    @FindBy(id = "pass")  
    public WebElement password;  
  
    @FindBy(id = "u_0_b")  
    public WebElement loginButton;  
  
    public void login(String userid, String pass){  
        username.sendKeys( ...charSequences: userid);  
        password.sendKeys( ...charSequences: pass);  
        loginButton.click();  
    }  
}
```

Face Log In
Page Element

Face Log In
Page Method

TEST CLASS

```
public class LogIn_PageObjectModel extends TestBase {  
  
    @Test  
    public void loginWithPOM(){  
        driver.get("https://www.facebook.com/");  
        FaceLoginPage faceLoginPage=new FaceLoginPage(driver);  
        faceLoginPage.username.sendKeys( ...charSequences: "user name");  
        faceLoginPage.password.sendKeys( ...charSequences: "password");  
        faceLoginPage.loginButton.click();  
        faceLoginPage.login( userid: "username", pass: "userpassword");  
    }  
}
```

TestNG

Page Object Model Page Factory

1. **PageFactory**, page object dizayni icin bir önemli classtır.
2. **PageFactory.initElements(driver,this);**

this => page instance

driver => bizim gonderecegimiz driver
3. Page objelerini **instantiate(ilk deger atama)** için page classlarında **PageFactory** kullanıyoruz.
4. Aslinda PageFactory, elementlere ilk degeri atayan **initElements()** metodunu destekler.

```
public class LoginPage {
    WebDriver driver=new ChromeDriver();
    //It is a good practice to put PageFactory in the class constructor.
    //Page factory instantiate the page object class and its elements.
    public LoginPage(){
        PageFactory.initElements(driver, page: this);
    }
    //used to mark an create an element in the page class to be directly called on the test classes.
    @FindBy(id="login")
    public WebElement username;

    @FindBy(id = "password")
    public WebElement password;

    @FindBy(xpath = "//button[@type='submit']")
    public WebElement loginButton;

    public void login(String user,String pass){
        username.sendKeys(user);
        password.sendKeys(pass);
        loginButton.click();
    }
}
```

TestNG

Page Object Model @FindBy Annotation

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 LoginPage {
    WebDriver driver=new ChromeDriver();
    //It is a good practice to put PageFactory in the class constructor.
    //Page factory instantiate the page object class and its elements.
    public LoginPage(){
        PageFactory.initElements(driver, page: this);
    }
    //used to mark an create an element in the page class to be directly called on the test classes.
    @FindBy(id="login")
    public WebElement username;

    @FindBy(id = "password")
    public WebElement password;

    @FindBy(xpath = "//button[@type='submit']")
    public WebElement loginButton;

    public void login(String user,String pass){
        username.sendKeys(user);
        password.sendKeys(pass);
        loginButton.click();
    }
}
```

TestNG

Page Object Model Class Work

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

TestNG

DERS 14

POM
Test Datalarini Kullanma
Properties File
Configuration Reader

Mehmet BULUTLUOZ
Elektronik Muh.

TestNG

Page Object Model Class Work

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

TestNG

Page Object Model Class Work

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

TestNG

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.

- Temel olarak key(anahtar) ve value(değer) çiftlerini kullanırız ve ihtiyaç duyduğumuzda onları cagırırız(url,credentials,browser,environments,...)
 - **url=https://www.fhctrip.com/**
 - **password=Man1ager2!**
 - **browser=chrome**
 - **name=Ali**
 - **username=manager**

The screenshot shows a Java project named "mymavenproject" in an IDE. The project structure is as follows:

- mymavenproject (~\IdeaProject)
- .idea
- src
 - main
 - test
 - java
 - com.techproed
 - pages
 - tests
 - utilities
 - ConfigurationReader.java
 - TestBase.java
 - target
 - .gitignore
 - configuration.properties

The code for ConfigurationReader.java is:

```
public class ConfigurationReader {  
    //We write the config properties  
    private static Properties prop;  
    //I am making static because I  
    static { //to initialize we use  
        String path="configuration.properties";  
        try {  
            FileInputStream file=new FileInputStream(path);  
            prop=new Properties();  
            prop.load(file);  
            file.close(); //closing  
        } catch (Exception e) {  
            //System.out.println("Error"+e.printStackTrace());  
        }  
    }  
}
```

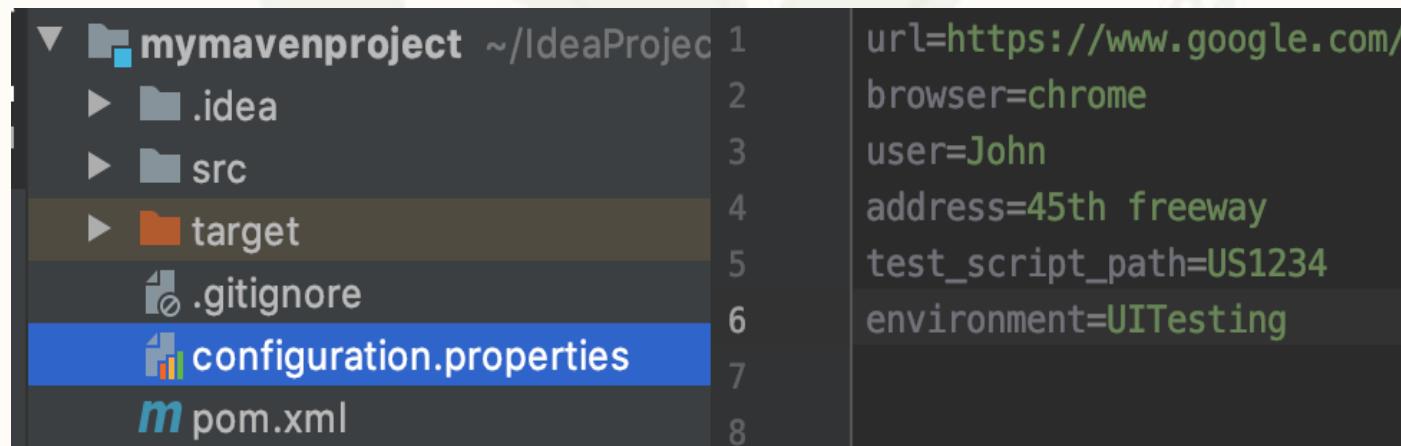
TestNG

Page Object Model Properties File

configuration.properties file Olusturmak icin project'imize sag click yapin

New =>File => configuration.properties

- properties file test datalari saklar.
- Bu dosyayı kullanmanın amacı,kodu sabit(hard coded) değil, dinamik yapmaktadır, böylece herkesverileri kolayca değiştirebilir ve test senaryolarını yürütmek



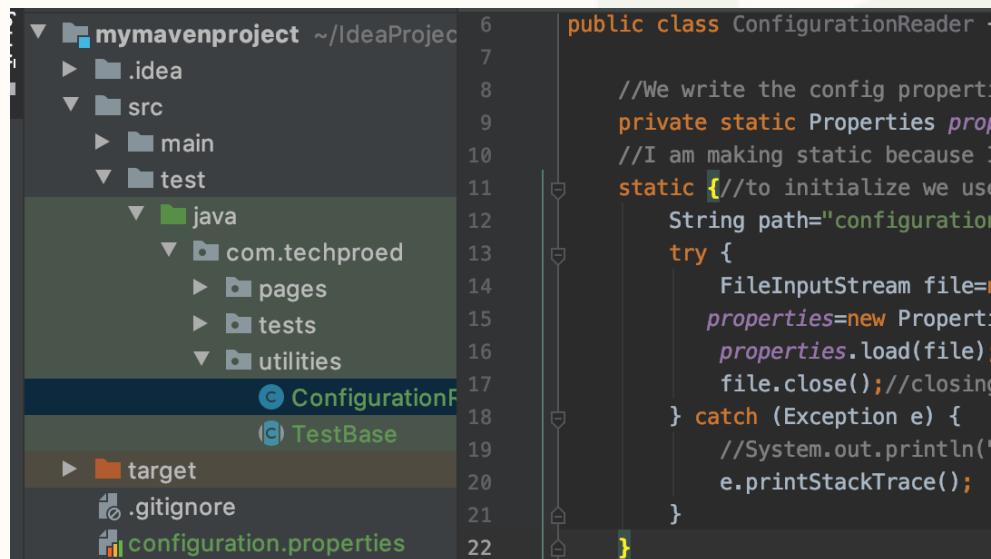
```
1 url=https://www.google.com/
2 browser=chrome
3 user=John
4 address=45th freeway
5 test_script_path=US1234
6 environment=UITesting
```

TestNG

Page Object Model Properties File

ConfigurationReader class

Configuration.properties dosyasından veri okumak için ayrı bir java class'i oluşturuyoruz.
Bu class'da test class'larinden ulaşacağımız **Static** değişkenler ve bloklar bulunur.



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

- mymavenproject (~/IdeaProject)
- .idea
- src
 - main
 - test
 - java
 - com.techproed
 - pages
 - tests
 - utilities
 - ConfigurationReader.java
 - TestBase.java
 - target
 - .gitignore
 - configuration.properties

On the right, the content of ConfigurationReader.java is displayed:

```
public class ConfigurationReader {  
    //We write the config properties  
    private static Properties prop;  
    //I am making static because I  
    static { //to initialize we use  
        String path="configuration.properties";  
        try {  
            FileInputStream file=new FileInputStream(path);  
            prop=new Properties();  
            prop.load(file);  
            file.close(); //closing  
        } catch (Exception e) {  
            //System.out.println("Error"+e.getMessage());  
            e.printStackTrace();  
        }  
    }  
}
```

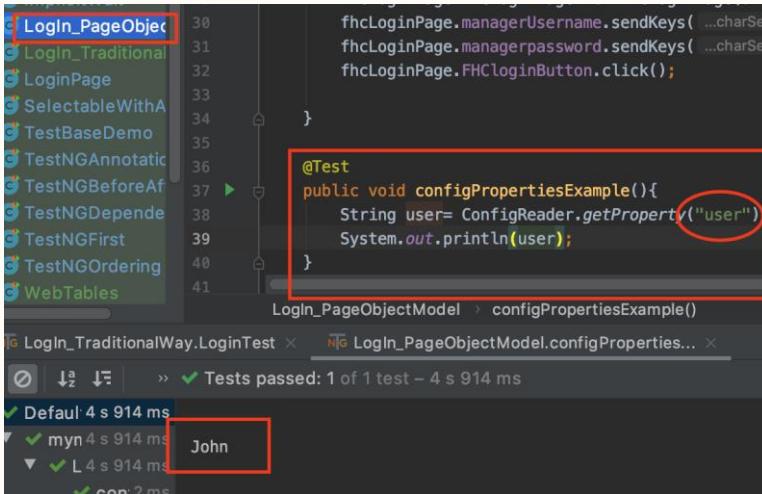
Interview Sorusu : Framework'unuzde static'i
nerede kullanıyorsunuz?

Cevap: getProperty yöntemi statiktir, bu yüzden
statik olarak kullanabiliriz

TestNG

Page Object Model Config Reader Data Flow

Test Class



```
30     fhcLoginPage.managerUsername.sendKeys("charSe...charSe...");  
31     fhcLoginPage.managerpassword.sendKeys("charSe...charSe...");  
32     fhcLoginPage.FHCloginButton.click();  
33 }  
34 }  
  
@Test  
public void configPropertiesExample(){  
    String user= ConfigReader.getProperty("user");  
    System.out.println(user);  
}  
  
Login_PageObjectModel > configPropertiesExample()  
  
Tests passed: 1 of 1 test – 4 s 914 ms
```

Not: Properties dosyasında olmayan bir anahtar(key) alırsak, null değeri döndürür . Örn: getProperty ("country") dosyada ülke yok, bu nedenle null değerini döndürür

ConfigurationReader Class

```
import java.io.FileInputStream;  
import java.util.Properties;  
  
public class ConfigurationReader {  
  
    //We write the config properties one and keep using it.  
    private static Properties properties;//We use Properties class to handle properties i...  
    //I am making static because I want it to run every time I call. I wan tit to belong  
    static {}//to initialize we use static block.  
    String path="configuration.properties";//path of the config properties file  
    try {  
        FileInputStream file=new FileInputStream(path);//To open an external file we  
        properties=new Properties();//initializing the configurationFile  
        properties.load(file);// Loading the file  
        file.close();//closing the file after loading. This is not mandatory  
    } catch (Exception e) {  
        //System.out.println("Path doesn't exist");// we can send a message if file d...  
        e.printStackTrace();  
    }  
  
    public static String getProperty(String key) {  
        return properties.getProperty(key);  
    }  
  
    //Testing if I can read from the configuration.properties file. Key and Value pairs  
    //    public static void main(String[] args) {  
    //        String a=properties.getProperty("url");  
    //        System.out.println(a);  
    //    }  
}
```

Configuration.properties file

```
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
```

TestNG

DERS 15

POM
Driver Class
Smoke Test
E2E Test

Mehmet BULUTLUOZ
Elektronik Muh.

TestNG

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.

```
public class Driver {  
  
    static WebDriver driver;//IQ:Why WebDriver static?  
  
    public static WebDriver getDriver(){  
        if (driver==null){//if driver is not pointing  
            WebDriverManager.chromedriver().setup();  
            driver=new ChromeDriver();  
        }  
        return driver;  
    }  
}
```

Basic Driver Class

```
public class SingletonDriver {  
  
    @Test  
    public void test(){  
        Driver.getDriver().get("https://www.google.com/");  
    }  
}
```

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



Page Object Model Driver Class

Singleton Pattern (Tekli Kullanım)

- Herhangi bir Java classında yalnızca bir obje oluşturamamıza izin veriliyorsa, bu tur classlara singleton class denir. Yani singleton pattern(tekli kullanım) bir Class'ı tek bir instance ile kısıtlayan bir software dizayn kalıbidır.
- Genel olarak aynı gereksinimimiz olduğunda, ve buna her ihtiyac duyuldugunda ayrı bir obje oluşturulması tercih edilmez. Singleton class; kullanıcıyı Webdriver'ın gerekli olduğu tüm örnekler(instance) için aynı objeyi kullanmaya zorlar.
- Sadece bir obje oluşturmalı ve buna her ihtiyac duyduğumuzda kullanmalıyız. Bu performans ve bellek kullanımı için de faydalıdır.
- Tüm framework'te veya tüm classlarda koordine için tek bir obje gerekiğinde faydalıdır.

TestNG

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.
- Sonra TestBasede yaptığımız gibi “wait” koyabiliriz.
- Ardından driver'i kapatıyoruz.
- Şu andan itibaren TestBase Classını değil Driver Classını kullanacağız.

```
public class Driver {  
  
    private Driver(){//so we cannot create another object. So we can have single instance}  
    }  
  
    static WebDriver driver;//IQ:Why WebDriver static? Our utilities are static  
    |  
    public static WebDriver getDriver(){  
        if (driver==null){//if driver is not pointing nothing then assign ChromeDriver  
        switch (ConfigReader.getProperty("browser")){//We will get the browser from config file  
            case "chrome":  
                WebDriverManager.chromedriver().setup();  
                driver=new ChromeDriver();  
                break;  
            case "firefox":  
                WebDriverManager.firefoxdriver().setup();  
                driver=new FirefoxDriver();  
                break;  
            case "ie":  
                WebDriverManager.iedriver().setup();  
                driver=new InternetExplorerDriver();  
                break;  
        }  
        }  
        driver.manage().timeouts().implicitlyWait( 10, TimeUnit.SECONDS);  
        driver.manage().timeouts().pageLoadTimeout( 10,TimeUnit.SECONDS);  
        driver.manage().window().maximize();  
        return driver;  
    }  
  
    public static void closeDriver(){//After we are done with the driver we close it  
        if (driver!=null){//quick when driver is pointing chrome, firefox, etc  
            driver.quit();  
            driver=null;//setting back to null, so making sure driver set to null again  
        }  
    }  
}
```

TestNG

Smoke Test

<http://qa-environment.koalaresorthotels.com> sayfasinda Smoke test olusturup yapacagiz

Bunun icin 3 test yapacagiz

- 1- Pozitif Login Test
- 2- Negatif Login Test
- 3- E2E Testi (system testi)

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.

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.

TestNG

Smoke Test

USER STORY

No: US001

Name : Dogru kullanici adi ve sifre ile sayfaya giris yapabilmeli

ACCEPTANCE CRITERIA:

- <https://qa-environment.concorthotel.com/> adresine gidip
- dogru kullanici adi ve sifre girdigimde sayfaya giris yapabilmeliyim

TestNG

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

TestNG

Smoke Test

1) smokeTest paketi altında yeni bir Class olustur: NegativeTest

3 Farkli test Methodunda 3 senaryoyu test et

- yanlisSifre
- yanlisKulllanici
- yanlisSifreKullanic

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

E2E Testing / Create Hotel Test

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 tıklayın
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.

TestNG

DERS 16

POM Ozet
E2E Test
Web Tables

Mehmet BULUTLUOZ
Elektronik Muh.

TestNG

(POM) Ozet

1- Page Object Model : Testlerimizi daha kolay ve düzenli olarak hazırlamamız ve çalıştırımız 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 methodları da bu class'da oluşturabiliriz.

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

TestNG

(POM) Ozet

- **configuration.properties** : Bu dosyayı testlerimizde kullanacağımız url,test dataları gibi kullanıcının aldığımız 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 güncellebiliriz.

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.



(POM) Ozet

-**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 olusturmamasi icin once if ile driver'in atamasi yapilmis mi control ederiz, atama yapilmissa ayni driver ile devam eder, atama yapilmamissa yeni bir driver olusturur ve deger atayip test sayfasina doneriz.

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



E2E Testing / Create Room Test

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..

TestNG

Web Tables

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 data
        </th>   </tr>   </thead>

  <tbody>
    =====>>>table body
    <tr>
      <td>
        =====>>>table row
        </td>   </tr>   </tbody>

  </table>
```

TestNG

Web Tables

ID	Name	Email	Role	Created Date	Phone	Action
2	manager	manager@st.com	manager	22.02.2020	(545) 345-3453	Manage
3	customerservice	customerservice@st.com	customerservice	14.01.2020	(543) 545-4354	Customer
4	manager2	manager2@fhctrip.com	manager	21.01.2020	(543) 434-3243	Manager
5	manager3	gfgfdgd@dfds.gfdd	manager	13.01.2020	fdsfsd fds fds	Manager

nts Console Sources Network Performance Memory Application Security Audits

```
> <div class="row">...</div>
  <div class="table-scrollable">
    <table class="table table-striped table-bordered table-hover dataTable no-footer" id="datatable_ajax" "datatable_ajax_info" role="grid">
      <thead>...</thead>
      <tbody>
        <tr role="row" class="odd">
          <td>2</td>
          <td class="sorting_1">manager</td> == $0
          <td>manager@st.com</td>
          <td>manager</td>
          <td>22.02.2020</td>
          <td>(545) 345-3453</td>
          <td>Manager</td>
        <td>...</td>
```

TestNG

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.satırdaki(row) elementleri konsolda yazdırın.

TestNG

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'l 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.satirdaki(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.

TestNG

DERS 17

Web Tables
Read Excel

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Web Tables : HTML kodlarda tablo seklinde design yapmak icin kullanilir.
- 2- Temel olarak 2 bolumden olusur, header ve body
- 3- Iki bolumde de satirlar ve satirlarda datalar olur.
- 4- Genellikle bolumlerin, satirlarin ve datalarin tag'leri sabittir.
 - header bolumu thead, satirlar tr, datalar th
 - body bolumu tbody, satirlar tr, datalar td
- 5- Web tablosunda herhangi bir dataya ulasmak icin once bolumu, sonra satir ve data numarası girilir
`//tbody//tr[2]//td[3]` bu body bolumu 2.satirdaki 3.datayı verecektir
- 6- // ile / arasında soyle bir fark vardir. // ile child ve grandchilds arasında arama yapılırken, / ile sadece childs arasında arama yapılır
- 7- Locator'lar // ile baslar sonrasında // veya / kullanimina ihtiyaca gore karar verilir
- 8- Web tablolarında sutun yapisi yoktur, eger bir sutunu yazdirmak istersek her satirdaki o sutun numarasına ait datayı icine koyacagımız bir liste olusturur ve tum datalari o listeye ekleriz.
- 9- liste olusturmak icin de @FindBy notasyonu kullanilir, sadece alt satiri degistirmemiz yeterli olur. (public List<WebElement> dorduncuSutun gibi)

TestNG

Web Tables Class Work

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

TestNG

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

TestNG

Excel'in Yapisi

- Excel icin daha once inceledigimiz Web Table yapisina benzer bir yapı vardır.
- Java ile exceldeki data'lara ulasmak için **Workbook/Sheet/Row/Cell** (Calisma kitabı/tab/satir/hucre) yapisi kullanılır.
- Kodlamamız acisinden sutun yapisi yoktur, ihtiyac duyarsak kodla sutunu elde edebiliriz.
- **Workbook** excel dosyamız
- **Sheet** Her açık excel sekmesi (Sheet1, Sheet2, etc)
- **Row(satir)** Java, yalnızca içerisinde veri varsa satırları sayar. Default olarak, Java perspektifinden satır sayısı 0'dır
- **Cells (hucre)** Java her satıra bakar ve yalnızca hücrede veri varsa hücre sayısını sayar.

TestNG

Apache POI

- Apache POI, microsoft ofis dokumanlarına erişmek için kullanılan Java API'idir.
- Poi.apache.com official dokumanlar buradadır.
- Excel kullanmak icin;
<https://mvnrepository.com/> gidin ,
apache poi dependency'lerini alın, ve pom.xml dosyanıza ekleyin.

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



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 fileInputStream objesini girelim
10. **WorkbookFactory.create(fileInputStream)**
11. Worksheet objesi olusturun **workbook.getSheetAt(index)**
12. Row objesi olusturun **sheet.getRow(index)**
13. Cell objesi olusturun **row.getCell(index)**



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



DERS 18

Write Excel
TestNG Xml Files

Mehmet BULUTLUOZ
Elektronik Muh.

Onceki Dersten Aklimizda Kalanlar

- 1- Read excel : excel dosyalari web uygulamasinda degil bilgisayarimizda oldugu icin, driver ile degil Java'dan fileInputStream ile excel'e ulastik
 - ilk once dosya yolunu String bir degiskene atadik
 - FIS ile dosya yolunu kullanip dosyaya eristik
 - FIS'i parameter olarak kullanip workbook olusturduk
 - workbook.(istenenMethod) ile sayfaya (sheet)
 - sheet .(istenenMethod) ile istedigimiz satira (row)
 - row .(istenenMethod) ile istenen hucreye ulastik (cell)
- 2- cell objesi CELL class'indan gelir bir String olmadigi icin String manipulation method'lari uygulanamaz ama sout() icine yazilarak konsolda goruntulenebilir
- 3- eger her seferinde bu kadar obje olusturmak istemezsek, workbook'a kladar ilerleyip sonra workbook. .(istenenSheetMethod). .(istenenRowMethod). .(istenenCellMethod) ile istenen cell'e ulasabiliriz
- 4- satir sayilar ve cell numaralari index ile ulasilabilir
- 5- eger son satiri degil de reel olarak kullanilan satir sayisini istiyorsak getPhysicalNumberOfRows() method'unu kullanabiliriz

TestNG

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

TestNG

XML File Oluşturma

Xml dosyasi framework'umuzdeki belirli testleri veya tum testleri otomatik olarak calistirmak icin kullandigimiz otomasyon dosyasidir.

-Olusturma

-Calistirma

Belirli package, class,method(lar) icin

-include, exclude

```
<!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>
```

TestNG xml ile ilgili tum dokumantasyon icin : <https://testng.org/doc/documentation-main.html#testng-xml>

TestNG

XML File Oluşturma

- Testng framework'de xml dosyası 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 fazlası için: <https://testng.org/doc/documentation-main.html>

The screenshot shows an IDE interface with a project structure on the left and an XML editor on the right. The project structure includes a .idea folder, a src folder with main and test subfolders, and a pom.xml file. The test folder contains a java subfolder with com.techproed, pages, smoke, tests, utilities, and pageObjectTests subfolders. The smoke folder is circled in red. The XML editor shows the smoketest.xml file with the following content:

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="smoke test automation" verbose="2">
  <test name="Login tests">
    <packages>
      <package name="com.techproed.smoke"></package>
    </packages>
  </test>
</suite>
```

A tooltip on the right side of the XML editor says: "Give path of a package and use this syntax tp execute a certain package with TestNG.xml file".

TestNG

Belirli Class'lar Nasıl Çalıştırılır?

- Yeni bir xml dosyasi olusturalim :
belirliClasslariCalistirma
- <package> attribute yerine <classes> ve
sonra <class> attribute kullanalim

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="smoke test automation" verbose="2">
  <test name="smoke tests">
    <classes>
      <class name="com.techproed.smokeTest.NegativeGlbSignInTest"></class>
      <class name="com.techproed.smokeTest.NegativeTest"></class>
    </classes>
  </test>
</suite>
```

The screenshot shows the IntelliJ IDEA interface with the TestNG plugin. On the left, the project structure is visible with several test packages like 'smoke' and 'tests'. In the center, the XML configuration file 'runningcertainclasses.xml' is displayed:

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="smoke test automation" verbose="2">
  <test name="include exclude class test">
    <classes>
      <class name="com.techproed.smoke.IncludeExcludeTestNG"/>
    </classes>
  </test>
  <test name="actions double and right click class">
    <classes>
      <class name="com.techproed.smoke.ActionsDoubleAndRightClick"/>
    </classes>
  </test>
</suite>
```

On the right, the 'Run' tool window shows the execution results for the 'smoke test automation' suite. It lists six passed tests: 'include exclude class', 'IncludeExcludeTestNG', 'apiTest', 'dataBaseTest', 'mobileTest', and 'webTest'. Below that, it shows the 'actions double and right click class' suite with two passed tests: 'ActionsDoubleClick' and 'RightClick'. The total summary is: Total tests run: 6, Passes: 6, Failures: 0, Skips: 0.



Belirli Method'lar Nasıl Çalıştırılır?

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

```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
<suite name="smoke test automation" verbose="2">
  <test name="smoke tests">
    <classes>
      <class name="com.techproed.tests.D12_Alerts">
        <methods>
          <include name="acceptAlert"></include>
        </methods>
      </class>
    </classes>
  </test>
</suite>
```

TestNG

Belirli Gruplar Nasıl Çalıştırılır?

- Yeni bir xml dosyasi olusturalim : belirliGruplariCalistirma
- Group calistirmak icin hem grup adini tanimlamak hem de nerede arayacagini 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>
```



Butun Testler Nasıl Çalıştırılır?

- 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>
```

TestNG

DERS 19

TestNG
Html Reports

Mehmet BULUTLUOZ
Elektronik Muh.

TestNG

Html Reports

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

Pom.xml dosyamıza aeventstack dependency'sini ekliyoruz.

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

TestNG

Html Reports

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 extentReports;*** 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 Açıma"); bilgi sadece adım eklemek içindir

TestNG

Html Reports

Testimize rapor olusturma adimlari

- 1- Test class'ini extends ile TestBaseRapor Class'ina child yapalim
- 2- *extentTest=extentReports.createTest("Test ismi", "Tanim");* rapor olusturalim
- 3- Gerekli/istedigimiz satirlara extentTest.info("Aciklama") ekleyelim
- 4- Assert olan satirda aciklamayı extentTest.pass ile yapabiliriz

```
@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.
```

TestNG

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

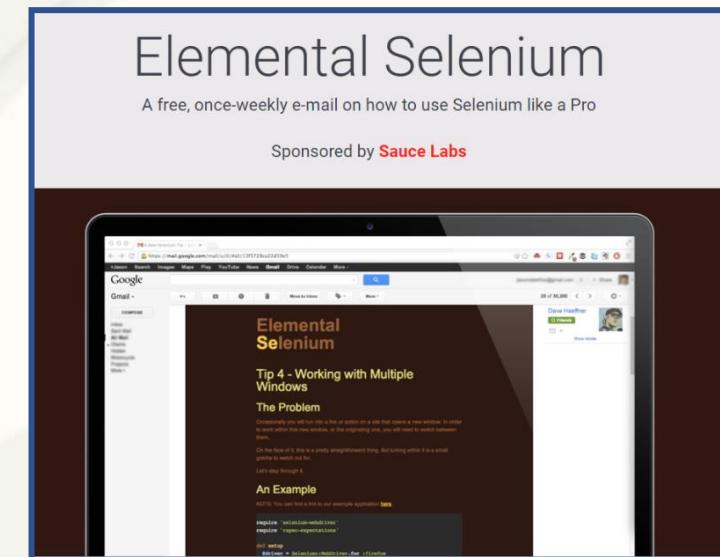
Tests

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

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

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

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





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 arasına yazıyoruz

```
</dependencies>
```



TestNG Ornek Proje Olusturma

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>
```



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

TestNG Ornek Proje Olusturma

- 7) Pom Dosyasini oluşturma işlemimiz bitti. Kutuphaneler kırmızı renkte olabilir. Sağ tarafta Maven yazan sekmede Reload oklarına tıklayıp beklediğimiz de hata gitmiş oluyor
- 8) Kullanıcının gordugu arayuzde test ederiz. (UI)
Kullanacağımız paketleri uygun isimlerde test-Java bolumun içerisinde oluşturuyoruz
- 9) Java ya sağ tıklarız new package – com (paketin ismi) yazarız
- 10) com package'ına sağ tıklarız – new package – techproed (paketin ismi) yazarız
- 11) artık projemiz com. techproed
Bu package'in altına frameworkumuzun package'larini yolosturuyoruz. Bunlar
 - A-pages
 - B-smokeTest
 - C-tests
 - D-utilities



TestNG Ornek Proje Olusturma

12) Resources paketi olusturma:

Java'ya sag tikliyoruz-new-package -->resources (yeni package)

Bu resources paketinin altina dokumanlarimizi copy-paste ederiz

13) configuration.properties dosyasi olusturma

En yukarda Projemize sag tikliyoruz new- File ' a tikliyoruz

Ismi önemli değil ama uzantisi MUTLAKA .properties olmalı

Ismi configuration.properties yaziyoruz

Bu dosyanin içine Data'larimizi key=value seklinde yaziyoruz

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 Classi oluşturuyoruz. Bu class configuration.properties deki dosyalarimizi okumak için bir araci

TestNG

TestNG Ornek Proje Olusturma

15) ConfigReader Classinda :

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

Bu objeyi sadece bu class ta kullanacagini icin private yapmamiz önerilir

2-Properties objesini kullanmak uzere bir static blok kurmaliyiz. 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);  
    }  
}
```

TestNG Ornek Proje Olusturma

16) Driver class'ini düzenliyoruz

Singleton class : object olusturulmasi kontrol altina alınan (genelde izin verilmeyen) classdir. Bunun icin baska classlarda Driver clasından 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

TestNG Ornek Proje Olusturma

18) pages package inin altında kullanacagimiz her websayfasi icin bir page Class'i 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);  
}
```

```
@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;
```

b) Ardinda Locate islemlerimizin tamamini yaziyoruz bu sayfaya



TestNG Genel Tekrar Soru Cozumu

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 altinda yeni bir class olusturun: D26_AmazonSatirSutunSayisi
- Bu class'in altinda 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



TestNG Genel Tekrar Soru Çözumu

Soru 2 :

- AmazonPage sayfasında istedigim satır ve sutun sayısı ile çağrıdigimda bana hucredeki 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



TestNG Genel Tekrar Soru Çözumu

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



TestNG Genel Tekrar Soru Çözumu

Soru 4 :

- D26_AmazonSatirSutunSayisi class'indan satirSayisi() testini ve D26_AmazonHucreTesti class'indan hucretesti() testini rapor alacak şekilde hazırlayın ve 3.sorudaki xml dosyası ile çalıştırıp raporu oluşturun



DERS 20

Cucumber Framework Olusturma

Mehmet BULUTLUOZ
Elektronik Muh.



- 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 tiklar

And gecerli username girer

And gecerli password girer

And Login butonuna basar

Then sayfaya giriş yaptığını kontrol eder

And kullanıcı sayfayı kapatır



- BDD(behaviour driven development) (Davranış güdümlü geliştirme)- ilk olarak behavior(davranis) veya functionalitileri yaziyorsunuz (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 icin 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,

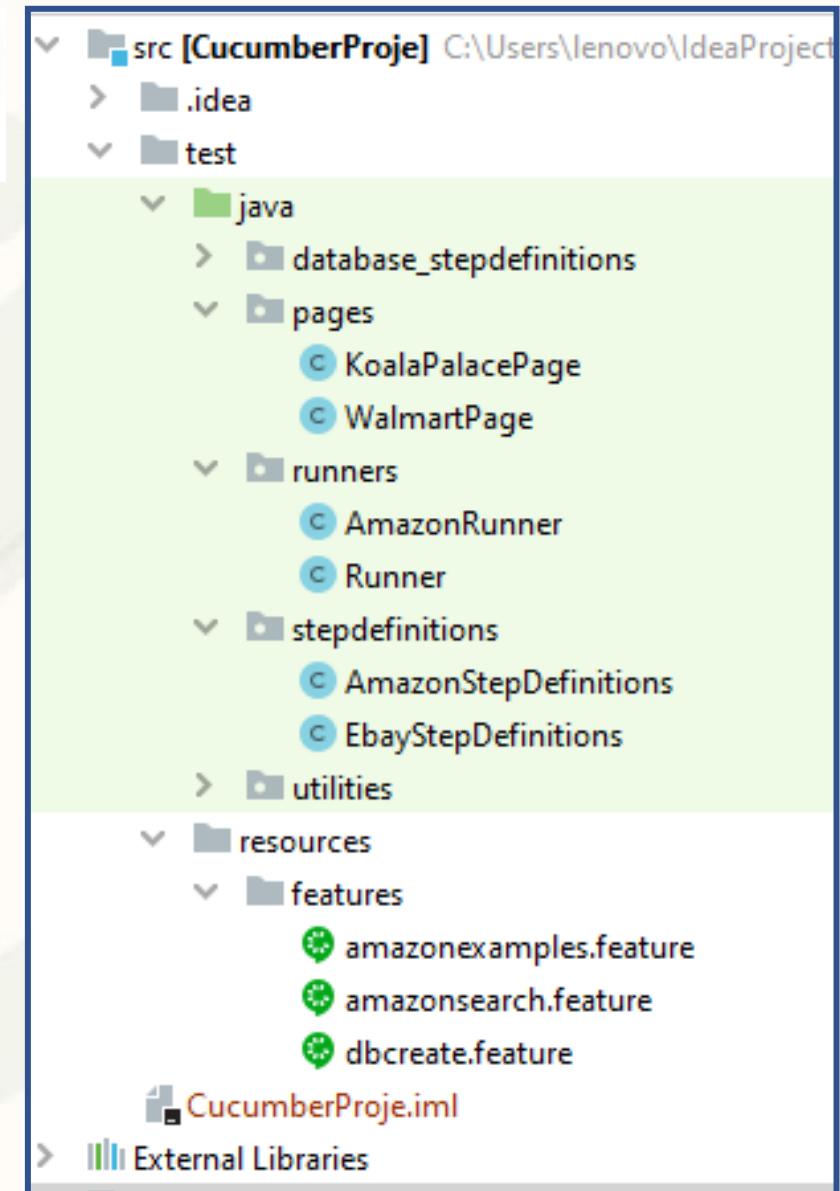
When, And anahtar kelimeleri ile olayı

Then anahtar kelimesi ile de sonuç tanımlanır.

Given-When-And-Then adımları ile Scenario oluşturulur.



- 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 Olusturma

1. **Create Project:** File -> New -> Project-> Select maven -> click next
2. Name: batch30Cucumber->finish
3. Add Dependencies =>Selenium-java, webdrivermanager, cucumber java, cucumber junit
4. Click Maven => click “Enable auto-reload after any changes” (Reload)
5. Javanin sürümüyle alakali sorunları halletmek icin compiler dependency yuklenebilir

```
<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

Farklı senaryoların başında ortak adımlarımız varsa:

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

```
Feature: First Feature
  Background: User search for amazon on hte google page
    Given user is on the google page
    And user search for amazon

  @googlesearch
  Scenario: TC02_Google search test
    Then user should should see amazon link on the search result
    Then clear the search box
    #Scenario 2 Search for flower and check if the the page related images
    Given user search for flowers on google page
    Then verify the page has flower

  @googlesearch
  Scenario: TC04_Google search test 2
    When user click on the first link
    Then user verify the amazon page displays

  This two step will run
  before each scenario
  on the same feature file
  In this example there are
  two Scenario: keyword
  So Background will
  run twice
```

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



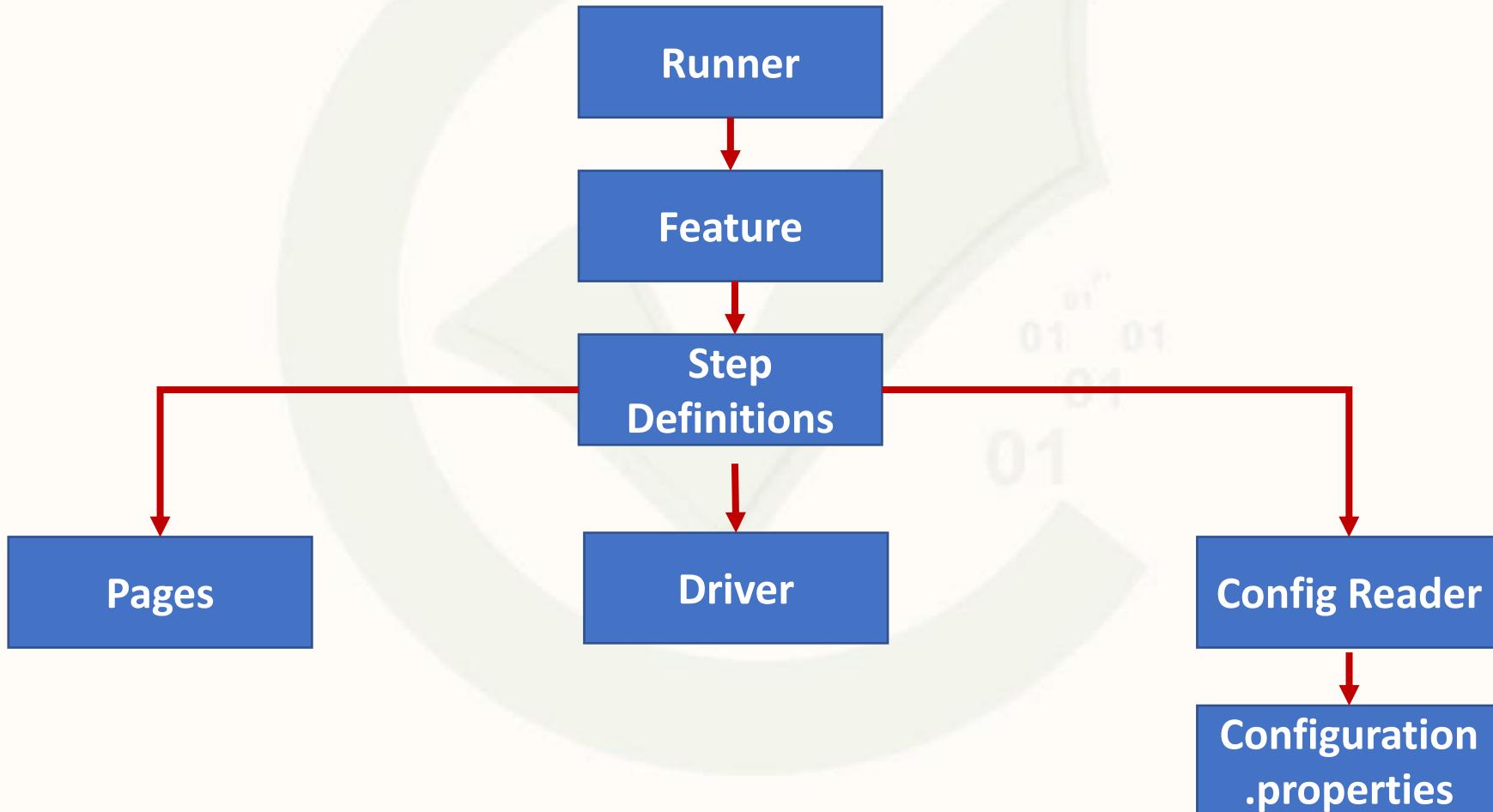
DERS 21

@tags, dryRun,
Parametre ile kullanma
Scenario Outline

Mehmet BULUTLUOZ
Elektronik Muh.



Onceki Dersten Aklimizda Kalanlar





@tags

- Tagları yalnızca belirli senaryoları (scenario) çalıştmak için kullanırız.
- Tagları senaryolarımızı grüplendirmek için kullanabiliriz (smoke test, regression test, vs.)

```
@CucumberOptions(  
    features = "src/test/resources/features",  
    glue = "stepdefinitions",  
    dryRun = false,  
    tags="@amazon"  
)
```

```
Feature: US1001 amazon page search  
  
@amazon @search @apple  
Scenario: TC01 amazon arama testi  
  
Given kullanıcı amazon sayfasına gider  
And "apple" için arama yapar  
Then sonuçların "apple" içerdigini test eder  
Then sayfayı kapatır
```



Feature File'i Parametre ile Kullanma

- Parametreleştirmek ve dinamik hale getirmek için feature file da çift tırnak " " kullanırız.
- Şimdi sadece " " içindeki değeri değiştirerek test datalarını feature file dan kontrol edebiliriz.
- Bu framework'u daha dinamik hale getirir, yani kodumuz artık hard coded degildir diyebiliriz.

```
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

  Scenario: TC02 tea pot araması testi
    Given kullanıcı amazon sayfasına gider
    And tea pot için arama yapar
    Then sonuçların tea pot içerdigini test eder

  Scenario: TC03 flower araması testi
    Given kullanıcı amazon sayfasına gider
    And flower için arama yapar
    Then sonuçların flower içerdigini test eder
    Then sayfayı kapatır
```



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



Feature File'i Parametre 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. (**|elmal|,larmut|...** gibi)

```
@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 : US1001 de kullandigimiz feature dosyasi altinda yeni bir Scenario olusturalim TC04 ve orada yaptigimiz aramayı **Scenario Outline** kullanarak farkli urunler icin yapalim



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

```
@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: **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|
```



cucumber

Class Work

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

Kullanici adi	Password
Manager5	Manager5!
Manager6	Manager6!
Manager7	Manager7!
Manager8	Manager8!
Manager9	Manager9!

Feature: US1009 Ck Hotels Login

Scenario: TC11 kullanici gecerli bilgilerle giris yapar
Given kullanici ConcordHotels ana sayfasinda
Then Log in yazisina tiklar
And gecersiz username girer
And gecersiz password girer
And Login butonuna basar
Then sayfaya giris yapilamadigini kontrol eder
And kullanicinin sayfasi 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



DERS 22

HTML Rapor Ekleme

Mehmet BULUTLUOZ
Elektronik Muh.



Html Rapor Ekleme

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

```
@CucumberOptions (
    plugin={"html:target\\cucumber-reports.html"},
    features = "src/test/resources/features",
    glue = "stepdefinitions",
    dryRun = false,
    tags="@amazon"
)
```

```
@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 Concert Hotels Login

Scenario: TC12 kullanici gecerli bilgilerle giriş yapar

Given kullanici ConcertHotels ana sayfasinda

Then Log in yazisina tiklar

And gecerli username girer

And gecerli password girer

And Login butonuna basar

Then sayfaya giriş yaptıgını kontrol eder

And kullanicı sayfayı kapatır

Feature: US1001 Ck Hotels Login

@wip

Scenario: TC01 kullanici gecerli bilgilerle giriş 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 giriş yaptıgını kontrol eder

And kullanicı sayfayı kapatır



cucumber

Class Work

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

Feature: US1009 Ck Hotels Login

Scenario: TC11 kullanici gecerli bilgilerle giriş yapar

Given kullanici Ck Hotels ana sayfasında

Then Log in yazısına tıklar

And gecersiz username girer

And gecersiz password girer

And Login butonuna basar

Then sayfaya giriş 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



Yeni Raporlar Ekleme

- Plugin ekleyerek yeni raporlar da olusturabiliriz
- Tester'lar icin onemli olan rapor Html olsa da json ve xml formatinda da rapor almak mumkundur.
- Ayrıca maven-cucumber-reporting plugin yüklemek istersek pom.xml'e plugin ekleyebiliriz

```
@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",  
    dryRun = false,  
    tags="@amazon"  
)
```

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



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.
- Cucumber 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();  
}
```





DERS 23

Cucumber Paralel
Calistirma

Mehmet BULUTLUOZ
Elektronik Muh.



Paralel Testing

- Paralel testing: Birden fazla browser'in es zamanlı çalıştırılmasıdır.
- Cucumber ile parallel test çalıştırırmak testing'ye göre daha zordur.
- Paralel çalıştırılmak için birden fazla Runner Class'ına ihtiyacımız var
- Cucumber'da, testleri paralel olarak çalıştırılabilmek için bazı eklentilere(plugin) ve yapılandırmalara ihtiyacımız var.
- 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'ina sahip olmalisiniz. Class isimleri TestRunner ile bitmelidir
 - a. SmokeTestRunner
 - b. FirstTestRunner
 - c. SecondTestRunner
2. maven-failsafe-plugin eklentisi ekleyin.(Belirli testler başarısız olduktan sonra testleri çalıştırılmaya 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>
```

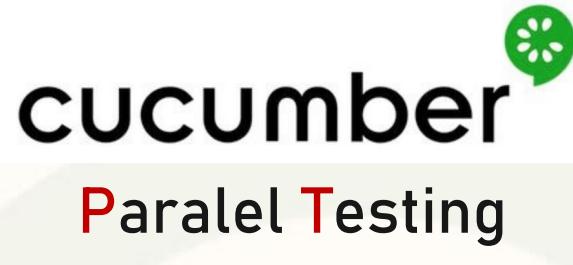


cucumber

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>
```



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

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	Steps					Scenarios			Features		
	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	1:27.416	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

TestNG

DERS 24

TestNG
Paralel Testing

Mehmet BULUTLUOZ
Elektronik Muh.



Cucumber Paralel Testing'de Ne Ogrendik

- 1- Paralel testin tum testlerimizi hep birlikte execute etmek istedigiimizde zamanı azaltmak icin kullanılır. Paralel testing'de istedigimiz kadar browser'i aynı anda calistirabiliriz
- 2- Ilk yapmam gereken paralel calismasini istedigim browser sayisinda runner olusturmak
- 3- Runner class'larinin ismi TestRunner icermelidir. Farkli bir isim istiyorsak sureFire plugin'inden calistirilacak runner'larin isminde gecen ortak kismi yazmaliyiz
- 4- Calistirmak istedigimiz tum testleri paralel calistiracagimiz browser sayisina gore tag'larla gruplandiririz.
- 5- pom.xml'de dependencies tagi bittikten sonra icice build, plugins taglarini acalim. Plugins taglari icinde 2 tane plugin ekleriz
 - failsafe plugini testimiz basarisiz olsa da paralel testing devam ettirir
 - sureFire bu plugin de paralel calistirma ayarlari yapilir
- 6- Daha guzel ve gelismis bir rapor almak icin pom.xml'e [maven-cucumber-reporting](#) plugini 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.

TestNG

Paralel Testing

- Çoklu calistirma, Parallel calistirma ve Cross Browser calistirma farkli farkli islemlerdir.
- Mesela 5 testi sirali olarak ama topluca calistirirsak → sirali çoklu calistirma
- 5 testin yarisini bir driverla, yarisini baska driverla calistirip, iki driver'l birlikte 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 birlikte calistirirsam → parallel cross browsing test olur



Paralel Testing / Classes

```
<!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>
```

TestNG

DERS 25

TestNG
Cross Browser Testing

Mehmet BULUTLUOZ
Elektronik Muh.



Onceki Dersten Aklimizda Kalanlar

- TestNG parallel calistirma, cucumber'a gore daha kolay ancak cucumber'daki raporlari bize vermedigi icin cucumber paralele calistirma tercih edilebilir
- Paralel calistirmak icin, sirali calistirma uzerine hazırlanmis xml dosyasında suite isminden sonra bir bosluk bırakıp parallel keyword'u aktif hale getiriyoruz, sonrasında thread'ler arasında iş bolusumunun hangi seviyede(test, class, method...) olacağını yazıyoruz. Sonra bir bosluk daha bırakıp thread-count keywordunu seçip, değer olarak aynı anda çalışmaya başlamasını istediğimiz thread sayısını seçiyoruz
- Execution olduktan sonra konsoolda çıkacak yazıları azaltmak veya artırmak için verbose keyword seçiyoruz. Değer olarak 1 ile 10 arasında bir değer giriyoruz. 1 en az yazı, 10 en çok yazı.

TestNG

Cross Browser Testing

1. Cross Browser test bir uygulamayı farklı browserlar ile test etmek demektir
2. Testleri sıralı (sequential) veya paralel olarak yapabiliriz
3. Cv'niz açısından Cross Browser test önemlidir cunku 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



TestNG

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

TestNG

Cross Browser Testing

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

TestNG

@Data Provider

@DataProvider bir TestNG annotation'ıdır.

Dolayısıyla 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[] getUruler(){  
        String aranacaklar[] = {"Ali","Veli","Hasan","Huseyin","Yasar"};  
  
        return aranacaklar;  
    }  
}
```

Genel Tekrar

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

Genel Tekrar

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

Genel Tekrar

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 "Istelenen Satir", "Istelenen Sutun" daki yaziyi yazdirir

Genel Tekrar

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 "Istelenen Baslik", sutunundaki tum degerleri yazdirir

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

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 kapatalım

Soru 10

Yeni bir Class olusturalim Day36_ExplicitlyWait

- 1) <https://demoqa.com/browser-windows> adresine gidin
- 2) Alerts'e tiklayın
- 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 kapatın

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