



Wise Quarter  
first class IT courses

# Selenium Team141

## Ders-01

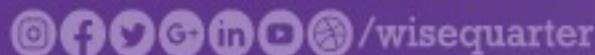
Selenium Giriş  
WebDriver Method'ları  
WebElements



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



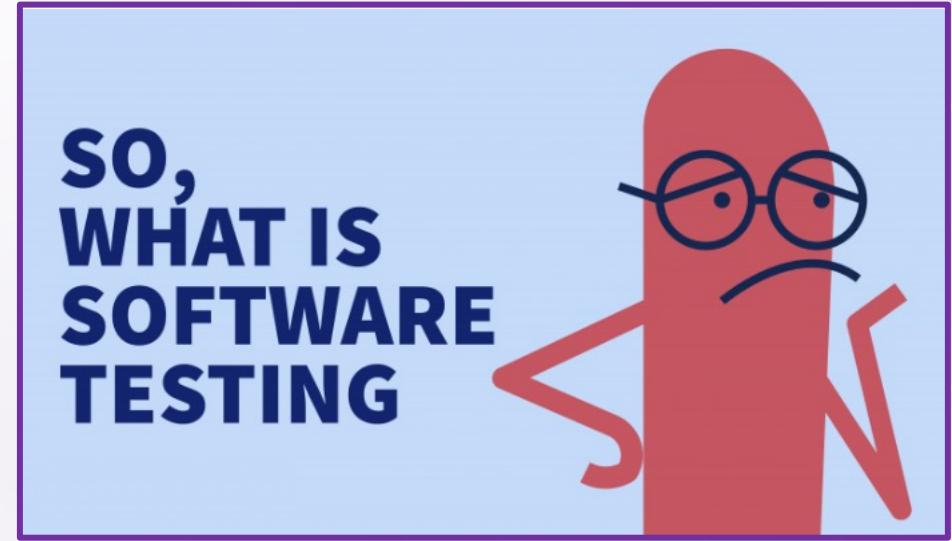
# Software Testing Nedir ?

Software testing var olan veya gelistirilmekte olan bir uygulamanin, tasirim asamasinda planlanan ozellikleri tasiyip tasimadiginin belirlenmesi icin yapılan faaliyetlerin butunudur.

Software testi icin tasirim asamasinda belirlenen sonuclar (**Expected Result**) ile uygulamanin kendisinden alınan sonuclar (**Actual Result**) karsilastirilir.

Expected ve actual result birbirine esit ise test basarili (**Test Passed**), Expected ve actual result birbirine esit degilse test basarisiz (**Test Failed**) olarak raporlanir.

Test gelistirme dongusunde developer'lar bir feature icin kod yazmaya basladiklarinda, tester'larda o feature'i analiz ederek acceptance criteria cercevesinde expected result'lari tespit etmeli, yazılımin bu ihtiyacları karşıladıgından emin olmak icin positive ve negative test senaryolari olusturmali ve bu testleri otomasyonla yapacak test case'leri olusturmalidir.



# Software Testing Neden Önemlidir ?

Gunumuzdeki rekabetci piyasa kosullari, uygulamalari bugs - free olmaya zorlamaktadir.

Ayrıca developer'larin user case'den anladıklari ile end – user'larin kullanım alışkanlıklarını her zaman uyusmayabilir.

Uygulamaya sonradan eklenen bazı feature'lar çalışan uygulamada bazı işlevleri negatif etkileyebilir.



Kullanıcının bekleyenlerini karşılamayan uygulamalar başarısız olur.

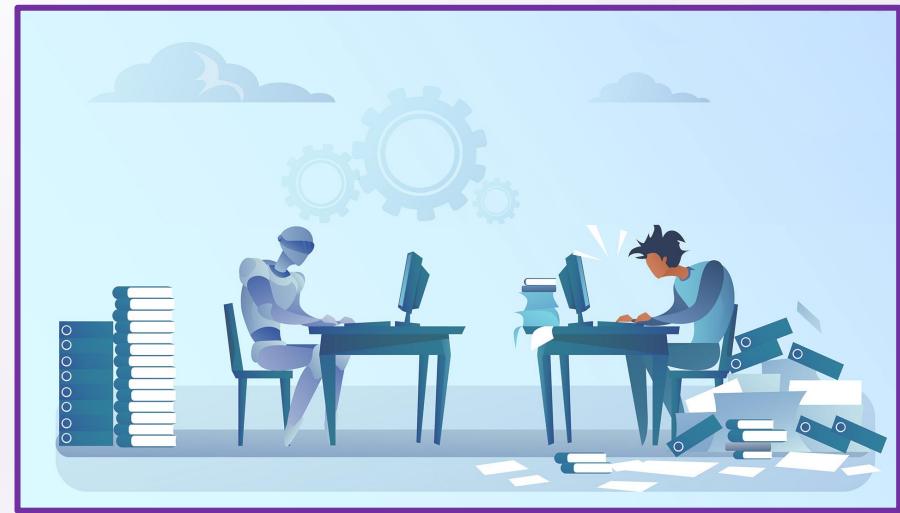
- Ürünün end-user kullanımına hazır olduğundan emin olmak
- End-user tarafından karşılaşılan sorunlarda düzeltme ve yeniden yapma maliyetlerini azaltmak
- Uygulamanın marketteki algısının üst seviyede olmasını sağlamak
- Sonradan eklenen işlevlerin eski işlevleri bozmadığından emin olmak için software testing yazılım geliştirme sürecinin vazgeçilmez bir parçası olmuştur.

# Manual Software Testing Nedir ?

Manual testing, uygulamanın planlanan sonuçlara uygun çalışıp çalışmadığını hiç bir otomasyon aracı kullanmadan, bir end-user gibi test edilmesidir.

Ancak insan gücüyle yapılan bu testler için hem çok fazla insan gücüne ve zamana ihtiyaç duyulur hem de insanın özelliklerimizden dolayı testlerde yanlışlıklar yapılabilir.

Zaman ve ihtiyaç duyulan insan gücünü azaltmak, testler çalıştırılırken ortaya çıkabilecek hataları minimum'a indirmek için test otomasyonu gereklidir.



Manual tester'lar uygulama üzerinde çok fazla zaman harcadıkları ve her adımı manual yaptıkları için uygulama bilgileri daha iyidir.

Automation tester'lar uygulamayı daha iyi anlamak ve sistem ihtiyaçlarını görmek için başlangıçta bir kaç kez manual test yapıp sonra otomasyon yapmalıdır.

# Test Otomasyonu Nedir ?

Test otomasyonu, insan gucu ile klavye ve mouse kullanilarak yapilabilecek bir yazılım testinin, bir otomasyon araci kullanilarak kodlar aracılıgi ile yapılmasidir.

Test otomasyon sayesinde

- klavye ve mouse kullanilarak yapilabilecek islemlerin cogu yapilabilir,
- yapılan islemler sonucunda gerceklesen sonuclar kaydedilebilir
- Elde edilen sonuclarla, expected sonuclar karsilastirilip, testin sonunu bulunabilir,
- Ve istenirse otomatik raporlar olusturulabilir



Otomasyonu yapılan bir test, istenen aralıklarla tekrar calistirilabilir. Hatta belirli aralıklarla olusturulan tum testler calistirilarak sistemin saglikli olarak calismaya devam ettiginden emin olunabilir (Regression Test)

Test otomasyonu insan gucu ihtiyacini azaltmasi, sorunsuz calismasi gibi ozellikleri sayesinde her gecen gun daha cok talep gormektedir.

# Automation & Manual Testing

Yandaki kod sizce nedir ?

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

Feature: US1010 herokuapp Delete testi

@heroku @sirali @pr1

Scenario: TC15 herokuapp'dan delete butonu calismali

Given kullanici "herokuappUrl" anasayfasinda

And add element butonuna basar

And kullanici 3 sn bekler

Then Delete butonu gorunur oluncaya kadar bekler

And Delete butonunun gorunur oldugunu test eder

Then Delete butonuna basar

And Delete butonunun gorunmedigini test eder

And sayfayı kapatir

Test otomasyonu sayesinde herkesin anlayacagi test case'ler olusturabilir, daha kisa surede, daha az insan kaynagi ile testlerinizi gerceklestirebilir, istediginiz raporlari otomatik olarak olusturabilirsiniz.

Manuel Test sayesinde kod bilgisi olmasa bile insanlara test yapabilir, temel test ihtiyaclarinizi karsilayabilirsiniz.

# Automation & Manual Testing

MANUAL TESTING

VS

AUTOMATION TESTING

EXECUTION TIME



PEOPLE



INFRESTRUCTURE



TOOLS



TURNAROUND TIME



TRAINING



# En Çok Kullanılan Tool'lar

En çok kullanılan test otomasyon tool'lari



# Selenium Nedir ?

## About Selenium

Selenium is a suite of tools for automating web browsers.

Selenium browser'lari otomasyon ile calistiracak tool'larin calismasi icin olusturulmus bir suite'dir.

Selenium farkli programlama dilleri ile calisarak gunumuzde kullanilan browser'larin tamamini otomasyon ile calistirabilmek icin olusturulmus class ve method'lara sahiptir.

Selenium'u kullanabilmek icin bu class'lar calisilan projeye eklenmelidir.

Selenium'un class'larini, kendi sitesinden indirecegimiz jar dosyalarini projeye ekleyerek projemize dahil edebilir veya bu isi bizim adimiza yapacak maven gibi tool'lari kullanarak class'lari direk projemize ekleyebiliriz.

# Selenium Nedir ?

**Selenium automates browsers. That's it!**

What you do with that power is entirely up to you.

Primarily it is for automating web applications for testing purposes, but is certainly not limited to just that.

Boring web-based administration tasks can (and should) also be automated as well.

Selenium browser'ları otomasyonla calistirir, bu otomasyon gucu ile ne yapacagini tamamen size kalmistir.

Selenium web uygulamalarini test etmek icin kullanilan acik kaynakli, ucretsiz bir uygulamadir.

2021 yilinda Selenium 4 piyasaya ciktigı ve Selenium'a yeni yetenekler(method'lar) kazandirdi.

Selenium, Java, Phyton, .Net gibi en çok kullanılan programlama dilleri ile kullanılabilir.

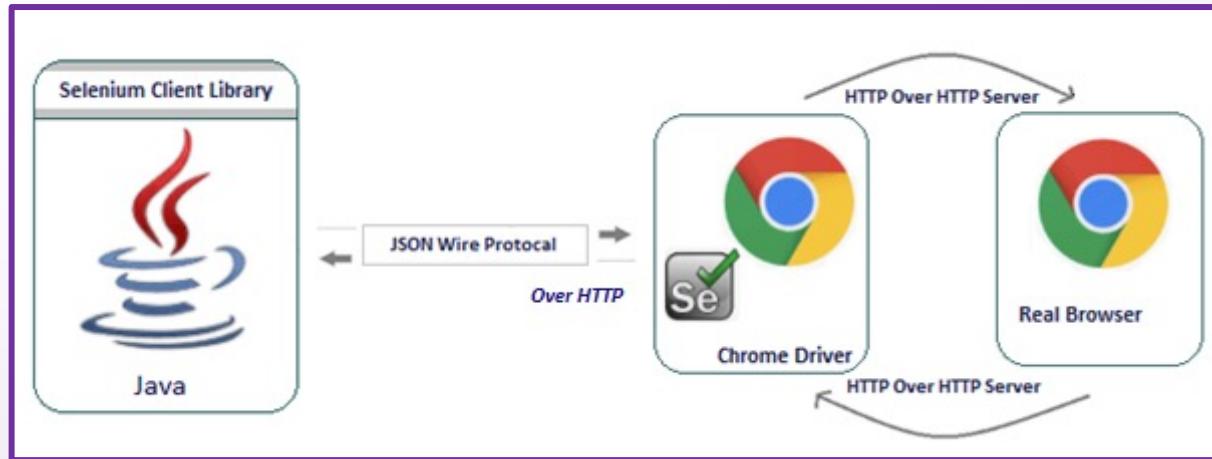
# Selenium Bileşenleri

Selenium'un dört bileşeni vardır;

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



# Selenium Nasil Calisir ?



Selenium test otomasyonunu WebDriver ile gerceklestirir.

Java ile yazdigimiz kodlar ile kullanilacak browser'a uygun bir webDriver objesi olusturulur.

Selenium kullanarak WebDriver class'indan olusturulan driver objesi bizim elimiz, gozumuz gibi calisir. Gonderildigi web sayfasinda klavye ve mouse ile yapabilecek islemleri yapar, elementlere tıklama, yazi gonderme, elementler uzerindeki yazilari alma gibi pek çok islemi gerceklestirir. Elde ettigi sonucları Java kodlarinin oldugu ortama döndürür.



# Selenium'un Avantajları

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

## Dezavantajları

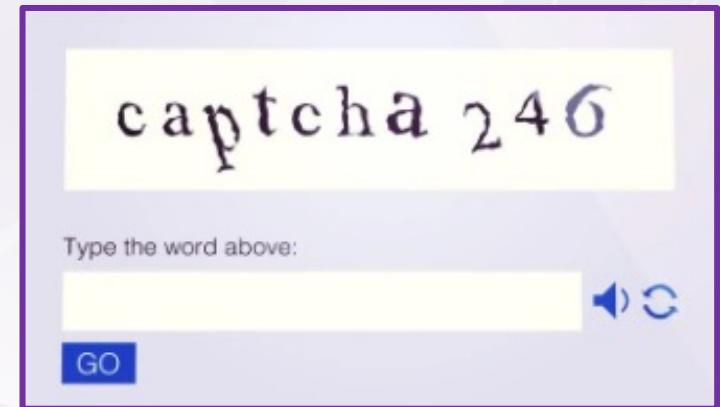
Programlama bilgisi gerektirir (Biz Java biliyoruz)

Yalnızca web tabanlı uygulamaları test eder

Profesyonel desteği sahip değil

performans testleri yapamaz

Captcha'yı asamaz(düzenleme tüm otomasyon araçları gibi)





Wise Quarter  
first class IT courses

# Selenium Team141

## Ders-02

WebDriver Method'lari

WebElements

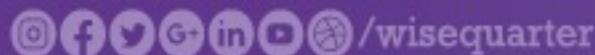
Locators



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)





# Onceki Dersten Aklimizda Kalanlar

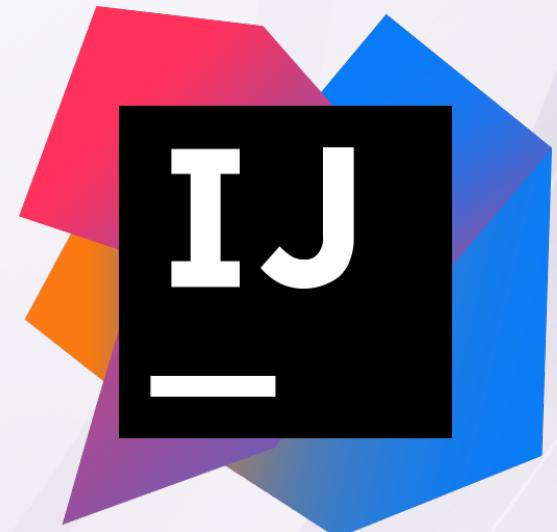
- 1- Otomasyon ile yazılım testi aslında iyi bir orkestrasyon gerektirir. Elimizdeki enstrumanları doğru kullanmalı ve mümkün olduğunda temiz kod yazarak testlerimizi çalıştırmalıyız.
  - 2- Kullanacağımız ilk enstrumanlar
    - Java : Otomasyon yapmak için kodlama bilgisine ihtiyaç vardır, biz Java kullanacağız.
    - Selenium : Browser'ları otomasyonla çalıştıracak tool'lar için bir suite'dir. Biz de selenium ile çalışacak Junit, TestNG ve Cucumber framework'leri ile çalışacağız.
- Selenium bircok programlama dili ve bir çok tool ile çalışabilir.
- Selenium'da 4 bileşen vardır ama biz testlerimizi otomasyon ile çalıştmak için Selenium WebDriver kullanacağız.
- Selenium WebDriver bizim elimiz ve gözümüzle, klavye ve mouse kullanarak yaptığımız işlemlerin çوغunu otomasyon ile yapmamıza imkan tanır
- IntelliJ : IDE java kodlarını compile edebilmek için kullanacağız
  - Framework : Ortak çalışma için dizayn edilmiş proje çerçeveleridir. Framework'ler sadece neyin nereye konulacağını söylemez, ayrıca isimizi kolaylastıracak pek çok promosyon verirler.

# IntelliJ Nedir ?

High Level programlama dilleri calisabilmek icin derleyicilere ihtiyac duyarlar.

IntelliJ IDEA 2000 yılında kurulmuş olan JetBrains firmasına ait olan, popüler bir kod gelistirme ortamı (**Integrated Development Environment**) dir.

IntelliJ uretkenligi en ust duzeye tasiyacak akilli kodlama yardimi, kod tamamlama ve ergonomic tasarim gibi ozelliklerle kod yazimini sadece verimli degil, ayni zamanda keyifli hale getirmistir.



Bir cok framework ve plugin ile calisma imkani saglar.

Kisaca, intelliJ IDE ihtiyaclarınızı tahmin eder ve sıkıcı ve tekrarlayan geliştirme görevlerini otomatikleştirir, böylece büyük resme odaklanabilirsiniz.

# Framework Nedir ?

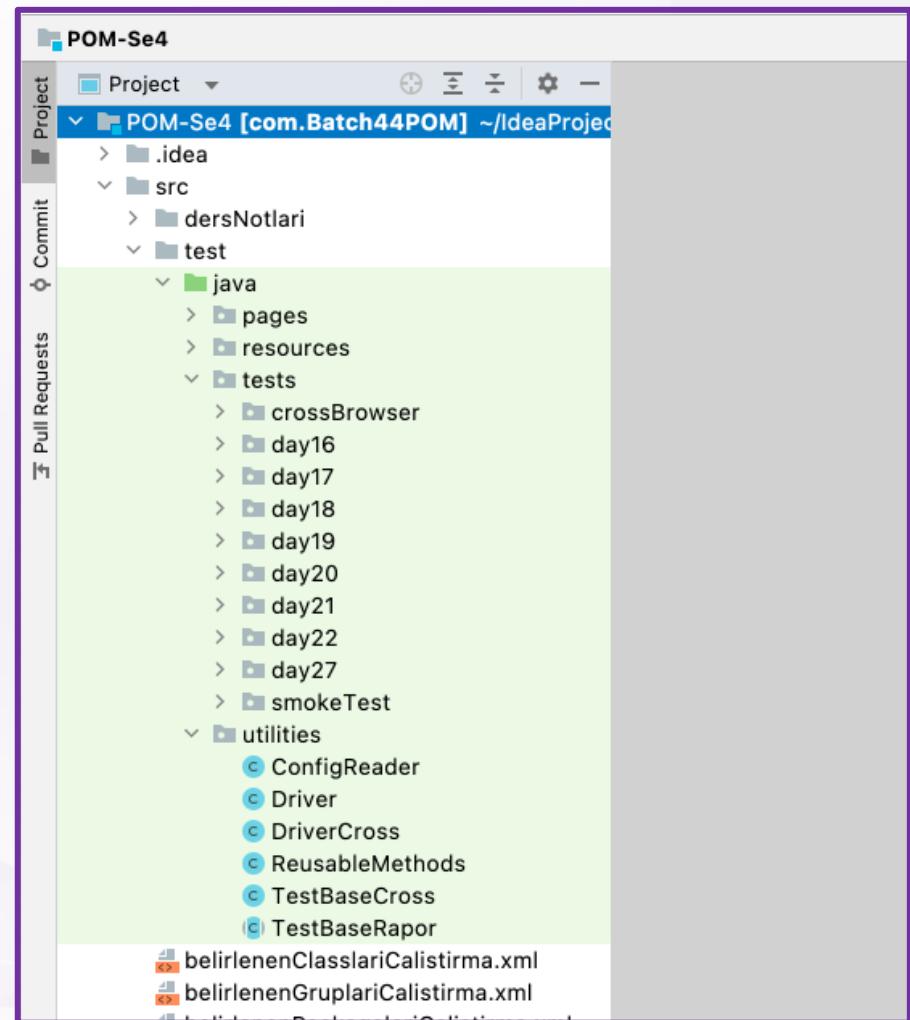
Framework, uzerine kodlarimizi yazarak projelerin olusturulabilecegi bir yapidir.

Test otomasyonu yaparken herseye sıfırdan baslayıp, herseyi sıfırdan kurgulayıp olusturmak yerine,

Herkesin anlayabileceği, test olusturma ve gozden gecirme sureclerini kolaylastirmak, tum ekibin ortak calisabilecegi bir yapı olusturmak icin test framework'lari olusturulmustur.

Framework, test yapmak icin kullandigimiz tum enstrumanlari birlestirir.

Framework ile UI, API ve Database testleri yapılandırılabilir.



# Jar Dosyaları ile Selenium Kurulumu

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

Chrome'un kendi sayfasına gidip Current stable release'i tıklayıp size uygun olanı download edin

Indirilen surum ile bilgisayarınızdaki Chrome browser surumunun aynı olduğundan emin olun

- 4) src altında resources director'si oluşturun
- 5) Bu klasor altında drivers ve libraries klasorleri oluşturun
- 6) Indirdigimiz chromedriver'i drivers klasorune, selenium-java dosyasını ise libraries klasorune çıkartın
- 7) intelliJ 'de yeni project / package / class oluşturualım ve class içinde main method oluşturun
- 8) File/Project Structure/Modules/Dependencies kısmından jar dosyalarını yükleyin

# WebDriver Objesi Olusturma

Selenium jar dosyaları ile projeye eklendiğinde, kullanmak istenen tüm browser'ların driver'larının da projeye eklenmesi gerekmektedir.

Kullanılacak browser'a ait driver projeye eklendikten sonra, her class'da bilgisayardaki browser'i yönetecek bir WebDriver objesi oluşturulur ve o obje yardımıyla WebDriver Class'ındaki hazır method'lar kullanılabilir.

WebDriver objesi oluşturmak ve objeye kullanılacak browser'a uygun değeri atamak için main method içerisinde

- 1) Java'daki setProperty("webdriver.chrome.driver", "driverPath"); ile sistem ayarları yapılır.

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

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

- 2) WebDriver driver= new ChromeDriver( );

ile webdriver objesi oluşturulur ve istenen browser'a uygun değer ataması yapılır.

# WebDriver Objesi Kullanma

Selenium ile otomasyon yapabilmenin ilk adımı WebDriver Class'ından obje oluşturmaktr.

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class C01_DriverMethods {

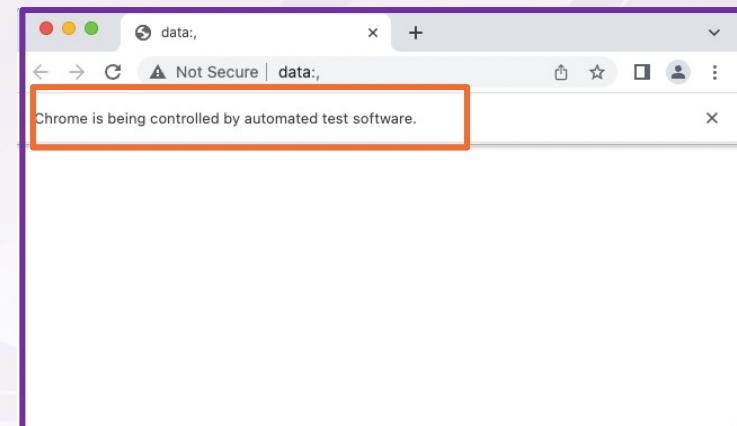
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.chrome.driver", "src/resources/chromedriver");

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

İlgili ayarları yapıp bir driver objesi oluşturduğumuzda, Selenium bu driver objesi sayesinde otomasyon yapabileceğimiz bir browser acar.

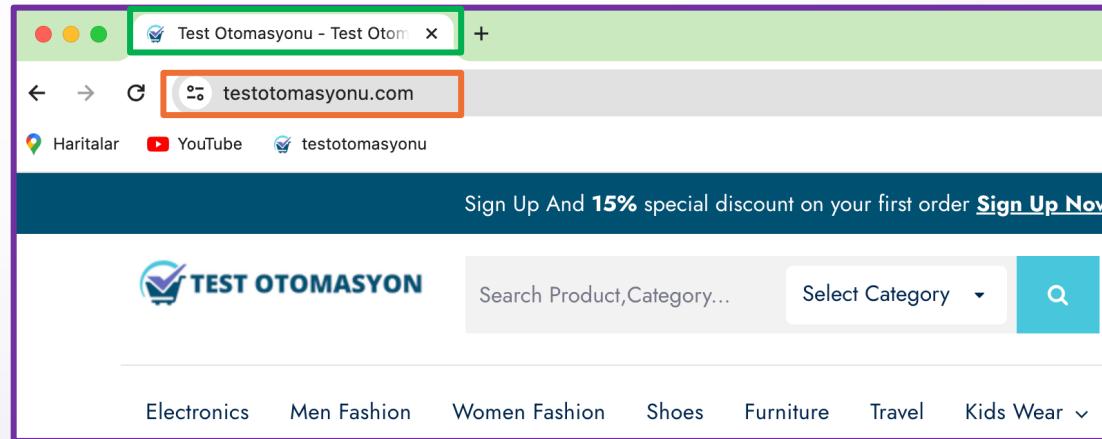
Bu browser'un Selenium tarafından kontrol edildiği yazılıdır.

Chrome dışında bir browser kullanılabaksa, o browser'in driver'ini da projeye eklemeli ve class içindeki ayarlarda o driver'in dosya yolu driver'a gösterilmelidir.





# driver.get...() Method'ları



- 1- `driver.get("https://www.testotomasyonu.com");` driver'i istenen url'e goturur.
- 2- `driver.getCurrentUrl();` Gidilen Web sayfasinin URL bilgisini döndürür.
- 3- `driver.getTitle();` Gidilen Web sayfasinin title (baslik) bilgisini döndürür.
- 4- `driver.close();` Acilmis olan driver'i kapatir
- 5- `driver.quit();` driver ile birden fazla window acilmissa, tumunu kapatir.



# driver.get...() Method'ları

## 6- driver.getPageSource()

Gidilen sayfanın kaynak kodlarını döndürür.

Sayfa kaynak kodları çok test otomasyonunda çok kullanılmaz, sadece özel olarak bu kodlarda bir kelimenin var olup olmadığı gibi özel bir test istenirse sayfa kodları String olarak kaydedilip, istenen arama yapılır.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <meta name="baseURL" content="https://testotomas...
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="csrf-token" content="cMbXU5KTMWg6M3q...
    <meta name="assetURL" content="https://testotomas...
    <meta name="frontendAssetURL" content="https://test...
    <meta name="isLogin" content="">
    <meta name="currency" content="$">
    <meta name="currentRoute" content="home">
    <meta name="direction" content=">">
    <meta name="image-loader" content="">
    <meta name="author" content="Test Otomasyon">
  </head>
  <title>Test Otomasyonu – Test Otomasyonu</title>
  <meta name="description" content="Test Otomasyonu is you...
  <meta name="keywords" content="testotomasyonu,ecommerce"...
  <!-- Favicon -->
```

## 7- driver.getWindowHandle()

CDwindow-8C07925B8CBA4C8EF3039F660C30DDA1

Açılan window'a işletim sistemi tarafından verilen unique bir değer olan **window handle** değerini döndürür.

## 8- driver.getWindowHandles()

Test sırasında driver birden fazla window actıysa , bir **Set** olarak açılan tüm window'ların **window handle** değerlerini döndürür.

# İlk Test Otomasyonu

Software testi için tasarım aşamasında belirlenen sonuçlar (**Expected Result**) ile uygulamanın kendisinden alınan sonuçlar (**Actual Result**) karşılaştırılır.

Expected ve actual result birbirine eşit ise test başarılı (**Test Passed**), Expected ve actual result birbirine eşit değilse test başarısız (**Test Failed**) olarak raporlanır.

Test aşamalarının ve test sonuçlarını **anlasılabilir** olması, testin **kısa** olmasından önemlidir.

```
// test otomasyonu anasayfaya gidin
driver.get("https://www.testotomasyonu.com");

// URL'in https://www.testotomasyonu.com olduğunu test edin
String expectedUrl = "https://www.testotomasyonu.com/";
String actualUrl = driver.getCurrentUrl();

if (expectedUrl.equals(actualUrl)){
    System.out.println("URL testi PASSED");
} else {
    System.out.println("URL testi FAILED");
    System.out.println("Actual URL : " + actualUrl);
}
```

Örneğin; gidilen sayfanın title değerinin belirli bir kelimeyi içerdigi test edilmek isteniyorsa, expected ve actual değerler kaydedilip, bir if else blogu içerisinde istenen test yapıp, sonuc yazdırılabilir.

# WebDriver Method'lari

1. testotomasyonu sayfasina gidelim. <https://www.testotomasyonu.com/>
2. Sayfa basligini(title) yazdirin
3. Sayfa basliginin “Test Otomasyonu” icerdigini test edin
4. Sayfa adresini(url) yazdirin
5. Sayfa url’inin <https://testotomasyonu.com/> oldugunu test edin.
6. Sayfa handle degerini yazdirin
7. Sayfa HTML kodlarinda “otomasyon” kelimesi gectigini test edin
8. Sayfayı kapatın.



Wise Quarter  
first class IT courses

# Selenium Team141

## Ders-03

### WebDriver Method'lari

### WebElements

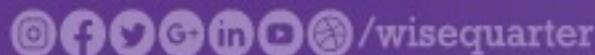
### Locators



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# driver.navigate...( ) Method'ları

9- `driver.navigate().to( url: "https://www.testotomasyonu.com");`

driver'i verile URL'e götürür. driver.get( ) ile aynı işlevi gerçekleştirir.

10- `driver.navigate().back();` Gidilen web sayfasını bir önceki sayfaya döndürür.

11- `driver.navigate().forward();` Gidilen web sayfasından navigate( ).back( ) ile bir önceki sayfaya donulmuşse yeniden ilk sayfaya götürür.

12- `driver.navigate().refresh();` İçinde olunan web sayfasını yeniler.

# driver.navigate...( ) Method'lari

1. Yeni bir Class olusturalim.CO5\_NavigationMethods
2. Youtube ana sayfasina gidelim . <https://www.youtube.com/>
3. Testotomasyonu sayfasina gidelim. <https://www.testotomasyonu.com/>
4. Tekrar YouTube'sayfasina donelim
5. Yeniden testotomasyonu sayfasina gidelim
6. Sayfayı Refresh(yenile) yapalim
7. Sayfayı kapatalim / Tüm sayfaları kapatalim

# driver.manage( )... Method'lari

13- `driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));`

driver'in gittiği sayfayı açması ve orada kullanacağı her bir web elementi bulması için tanımlanan maximum bekleme süresini tanımlar.

Wait konusu ayrı bir konu olarak anlatılacak, ancak yazılan otomasyon yapılmırken, internet bağlantısı veya bilgisayarın hızı gibi sebeplerle gecikmeler yaşanması durumunda ne yapacağını net olması için her testin başında max.bekleme süresi belirlenmelidir.

14- `driver.manage().window().maximize();`

Açılan driver'i tam sayfa yapar.

`driver.manage().window()`.... ile kullanılabilen farklı method'lar vardır ancak açılan web sayfasında tüm webelement'lerin görülebilir ve ulaşılabilir olması için her testin başında `maximize()` method'u kullanmasında fayda vardır.

# driver.manage( )... Method'ları

15-

```
driver.manage().window().fullscreen();
driver.manage().window().maximize();
driver.manage().window().minimize();
```

Acilan driver'i onceden belirlenmis standart buyukluklere getirir.

16-

```
driver.manage().window().setSize(new Dimension(width: 1000, height: 700));
driver.manage().window().setPosition(new Point(x: 100, y: 100));
```

Acilan driver'i kullanicinin istedigi ozel olculere getirir ve istenen noktaya tasir.

17-

```
driver.manage().window().getPosition();
driver.manage().window().getSize();
```

Acilan driver'in bulundugu pozisyonu ve boyutlarini döndürür.

# driver.manage( )... Method'lari

1. <https://www.testotomasyonu.com/> sayfasina gidin
2. Sayfanin konumunu ve boyutlarini yazdirin
3. Sayfayi simge durumuna getirin
4. simge durumunda 3 saniye bekleyip sayfayı maximize yapın
5. Sayfanin konumunu ve boyutlarini maximize durumunda yazdirin
6. Sayfayı fullscreen yapın
7. Sayfanin konumunu ve boyutlarini fullscreen durumunda yazdirin
8. Sayfayı kapatın

# driver.manage( )... Method'lari

1. <https://www.testotomasyonu.com/> sayfasina gidin
2. Sayfanin konumunu ve boyutlarini yazdirin
3. Sayfanin konumunu pixel olarak (50,50)'ye getirin
4. Sayfa boyutunu pixel olarak (1000,500)'ye ayarlayın
4. Sayfanin istenen konum ve boyuta geldigini test edin
5. Sayfayi kapatın

# WebDriver Method'ları

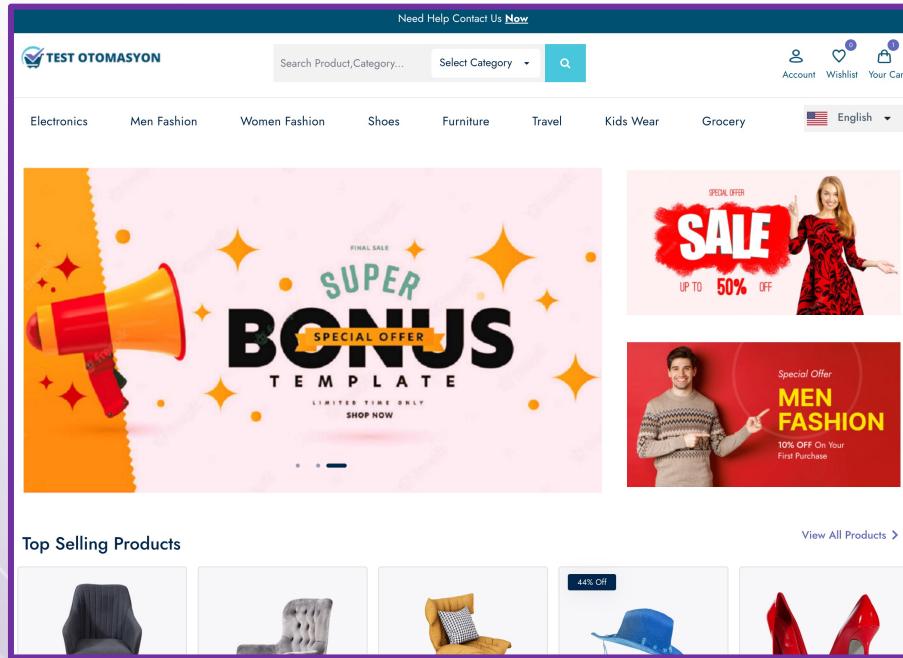
1. Yeni bir class olusturalim (Homework)
2. ChromeDriver kullanarak, facebook sayfasina gidin ve sayfa basliginin (title) "facebook" oldugunu test edin, degilse dogru basligi yazdirin.
3. Sayfa URL'inin "facebook" kelimesi icerdigini doğrulayin, icermiyorsa "actual" URL'i yazdirin.
4. <https://www.walmart.com/>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

# WebDriver Method'ları

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 testotomasyonu sayfasına gidin <https://www.testotomasyonu.com/>
5. Youtube sayfasına geri donun
6. Sayfayı yenileyin
7. Testotomasyonu sayfasına donun
8. Sayfayı tamsayfa yapın
9. Ardından sayfa başlığının "Test" içerip içermediğini (contains) test edin, Yoksa doğru başlığı(Actual Title) yazdırın.
10. Sayfa URL'sinin <https://www.testotomasyonu.com/> olup olmadığını test edin, degilse doğru URL'yi yazdırın
11. Sayfayı kapatın

# WebElements

Bir web sayfasında kullanılan herseye web element denir.



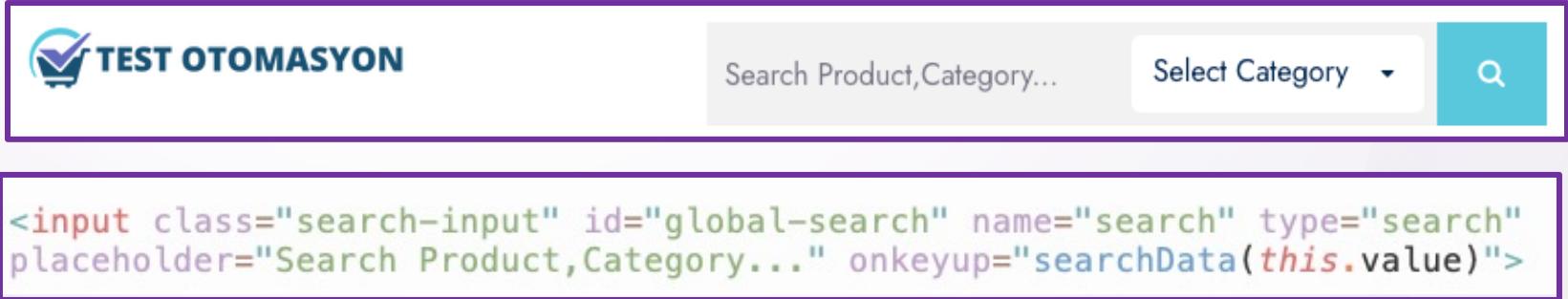
Her web element farklı özelliklerde olur. Link, açılır menü, button gibi etkilesimli web elementler olduğu gibi resim, background gibi etkilesimsiz web elementler de olur.

Gorunen her web element aslında developer'lar tarafından yazılan bir HTML kodun gorselleştirilmiş halidir.

Selenium webDriver görsel elementleri değil, HTML kodları kullanır.

Otomasyon sırasında kullanılmak istenen web elementler HTML kodları kullanılarak unique olarak webDriver'a tarif edilmelidir.

# WebElements



```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

Her bir web element yapısına uygun olarak farklı tag ve attribute'ler bulundurur.

Web elementi unique olarak tarif edebilmek için tag ve attribute'ler tekil olarak kullanılabilir.

Tekil kullanım unique tarif için yeterli olmazsa, birden fazla bilginin kombinasyonu kullanılır.

Tag : input

Attributes : type, id, value, name, autocomplete, placeholder, class, dir, tabindex, aria-label

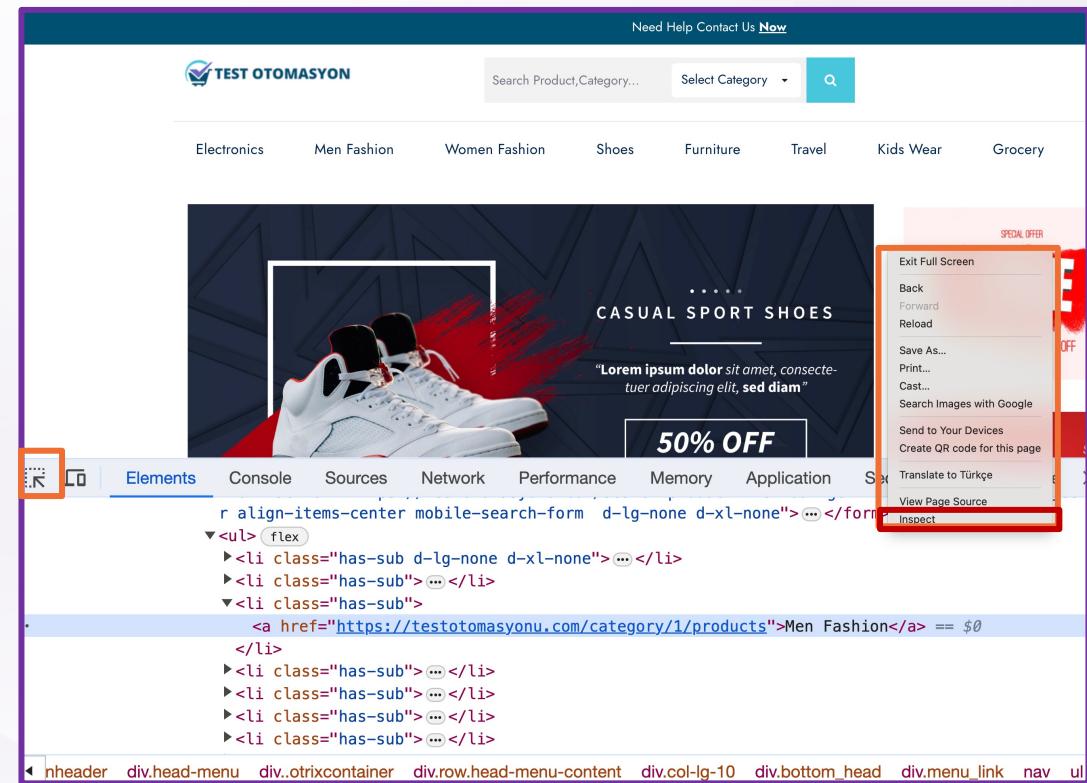
# WebElements

Web sayfasında HTML kodlarını görebilmek için mouse'da sağ click yapıp inspect/incele seçilmelidir.

Istenen web elementin HTML kodunu bulmak için, o element üzerinde yeniden sağ click yapıp inspect denebilir,

Veya menudeki → işaretini seçip, mavi iken mouse ile istenilen element seçilebilir.

HTML kodları açık iken ctrl+f tuslarına basılıncaya acılan bölümde webelement'in özellikleri aratılırsa, o özellikte kaç webelement bulunduğu görülebilir.



# Locators

Selenium LOCATORS, web sayfasındaki web öğelerini tanımlamak için kullanılır.

Selenium'da; metin kutuları, onay kutuları, linkler, radyo butonları, liste kutuları ve diğer **tüm web öğeler** üzerinde eylemler gerçekleştirmek için LOCATORS'a ihtiyacımız vardır.

Konum belirleyiciler bize web elementleri tanımlamada yardımcı olur.

Web Elementlerine ulaşmak için tag veya bazı attribute'lerin kullanıldığı 6 adet locators bulunur, bunlarla ulaşamayan webelementleri için özel olarak tanımlanan Xpath ve css locator'lari kullanılır.



## SELENIUM LOCATORS



# Locators

The screenshot shows a web application header. On the left is a logo with a checkmark icon and the text "TEST OTOMASYON". To its right is a search bar containing the placeholder "Search Product,Category...". Next to the search bar is a dropdown menu labeled "Select Category". On the far right is a teal-colored button with a white magnifying glass icon.

```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

## 1- By.id("uniqueId")

Web elementi tanimlamak icin ilk bakacagimiz locator id olabilir.

Id genellikle unique oldugu icin locate etmekte sıkça kullanılır. Ancak developer'ların aynı id ile birden fazla webelementi tanımlayabilecekleri de unutulmamalıdır.

Hangi locator kullanılırsa kullanılsın, web sayfasının HTML kodlarında locator aratılarak, unique sonuca ulaşıldığı gözlemlenmelidir.



# driver.findElement( ) Method'u



Search Product,Category...

Select Category ▾



```
<input class="search-input" id="global-search" name="search" type="search"  
placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

Unique locator'i tespit edilen web element kullanilmak icin, driver objesi ile LOCATE edilib, WebElement class'indan olusturulan objeye atanmalidir.

Driver.findElement(By.locator ("uniqueLocatorDegeri"))

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

Driver, findElement( ) ile objeyi bulamazsa NoSuchElementException verir.

findElement( ) ile locate edilib, objeye atanan WebElement testler sirasinda kullanilabilir.

webElement bir obje oldugu icin direkt yazdirilamaz, hazir method'lar kullanilarak manipule edilebilir.



Wise Quarter  
first class IT courses

# Selenium Team141

## Ders-04

Locators

Relative Locators



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



# Onceki Dersten Aklimizda Kalanlar

- 1- Selenium WebDriver bizim elimizle ve gozumuzle, klavye mouse kullanarak yaptigimiz islemleri bizim adimiza otomasyonla yapma imkani verir.
- 2- driver objesi ile direkt yapabilecegimiz islemler sinirlidir. Bir url'e gidebilir, gittigimiz web sayfasinin title'ini alabilir, acilan pencerenin buyuklugunu ve konumunu belirleyebiliriz.
- 3- Ancak driver'in web sayfasinda istedigimiz islemleri yapabilmesi icin webelementleri driver'a tarif etmemiz gereklidir.
- 4- Bir web sayfasındaki etkilesimli veya etkilesimsiz her nesne bir webelement'tir.
- 5- Web elementleri biz gorseller seklinde goruruz ama aslinda bilgisayarimiz, browser'imiz ve server'lar webelement'lere HTML kodlarina gore davranırlar.
- 6- Bizim HTML kodlarina mudahale etme, onlari degistirme veya HTML kodlari uzerindeki attribute'leri anlamlandırma gibi bir görevimiz yok.
- 7- Bizim amacimiz webelementlerin HTML kodlarini inceleyip, onlar icin driver'in bulabilecegi unique tarif edicileri(locator) olusturmaktir.
- 8- Locator'lari da Selenium bizim icin hazirlamistir. Toplam 8 locator vardir.
  - bunlardan 6 tanesi front-end developer'in hazırladigi HTML kodlarina bagimlidir.
  - XPath ve CssSelector ise her turlu webelementi locate edebilecek ozel bir tasarıma sahiptir.



# WebElement Objesi Olusturma

The screenshot shows the header of the 'TEST OTOMASYON' website. It features a logo with a checkmark and the text 'TEST OTOMASYON'. To the right is a search bar with the placeholder 'Search Product,Category...'. Next to it is a dropdown menu labeled 'Select Category' with a downward arrow, and finally a blue search button with a magnifying glass icon.

```
<input class="search-input" id="global-search" name="search" type="search"  
placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

## Arama Testi

- 1- Bir test class'i olusturun ilgili ayarlari yapin
- 2- <https://www.testotomasyonu.com> adresine gidin
- 3- urun arama kutusunu locate edin
- 4- arama kutusuna "shoe" yazdirin
- 5- arama islemini yapabilmek icin ENTER tusuna basin



# Locators

TEST OTOMASYON

Search Product,Category...

Select Category ▾

Q

```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

## 2- By.name("uniqueName")

WebElement'in HTML kodlarında name attribute'u varsa ve unique ise locate etmek için By.name( ) kullanılır

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



# Locators

CATEGORY

WOMEN	+
MEN	+
KIDS	+

### 3- By.className("uniqueClassName")

class attribute'u genellikle benzer ozellikleri barindiran web elementleri gruplandirmak icin kullanilir.

```
<div class="panel-group category-products" id="accordion">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">...</h4>
    </div>
    <div id="Women" class="panel-collapse collapse">...</div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">...</h4>
    </div>
    <div id="Men" class="panel-collapse collapse">...</div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading">...</div>
    <div id="Kids" class="panel-collapse collapse">...</div>
  </div>
</div>
```

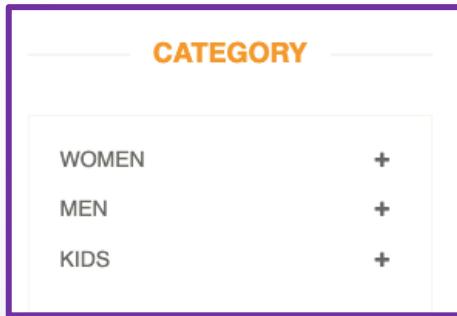
Bu sebeple class attribute'u ile yapacagimiz locate islemleri genellikle 1 web element degil, birden fazla element döndürür.

bu elementleri store edebilmek icin bir web element degil, web elementlerinden olusan bir list gereklidir.

NOT : class value'sunde bosluk (space) varsa By.className ile locate islemlerinde sorunlar yasanabilir.



# driver.findElements( ) Method'u



driver.findElements(....)

Locator'a uygun tüm web elementlerini döndürür.

```
<div class="panel-group category-products" id="accordion">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">...</h4>
    </div>
    <div id="Women" class="panel-collapse collapse">...</div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">...</h4>
    </div>
    <div id="Men" class="panel-collapse collapse">...</div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading">...</div>
    <div id="Kids" class="panel-collapse collapse">...</div>
  </div>
<div class="brands_products">
```

findElements( ) birden fazla web element döndürebileceği için dönen elementleri store etmek için bir list kullanılmalıdır.

Locator'a uyan hicbir webelement olmasa da exception olusmaz, bos bir list olusur.

List'teki tüm elementler web element oldugu için direkt yazdırılamaz, bir for-each loop kullanılarak elementlere istenen işlemler yapılabilir.



# Locators

The screenshot shows a search results page for 'phone'. At the top, there's a navigation bar with categories: Electronics, Men Fashion, Women Fashion, Shoes, Furniture, and Travel. Below the navigation is a breadcrumb trail: Home > Search Products. On the left, there's a sidebar with a 'Categories' dropdown menu containing items like Grocery, Kids Wear, Travel, Shoes, Furniture, Men Fashion, Women Fashion, and Electronics. Below the categories is a 'Price Range' input field. The main content area displays a heading '4 Products Found' above three product cards. The first card is for an 'APPLE IPhone 13 (Starlight, 128 GB)' priced at '\$799.00'. The second card is for a 'Real Phone 2 (White, 128 GB) (8 GB RAM)' priced at '\$699.00'. The third card is partially visible.

- 1- Bir test class'i olusturun ilgili ayarlari yapin
- 2- <https://www.testotomasyonu.com/> adresine gidin
- 3- arama kutusuna phone yazip aratin
- 4- Category bolumunde 8 element oldugunu test edin
- 5- Category isimlerini yazdirin
- 6- Sayfayi kapatin



# Locators

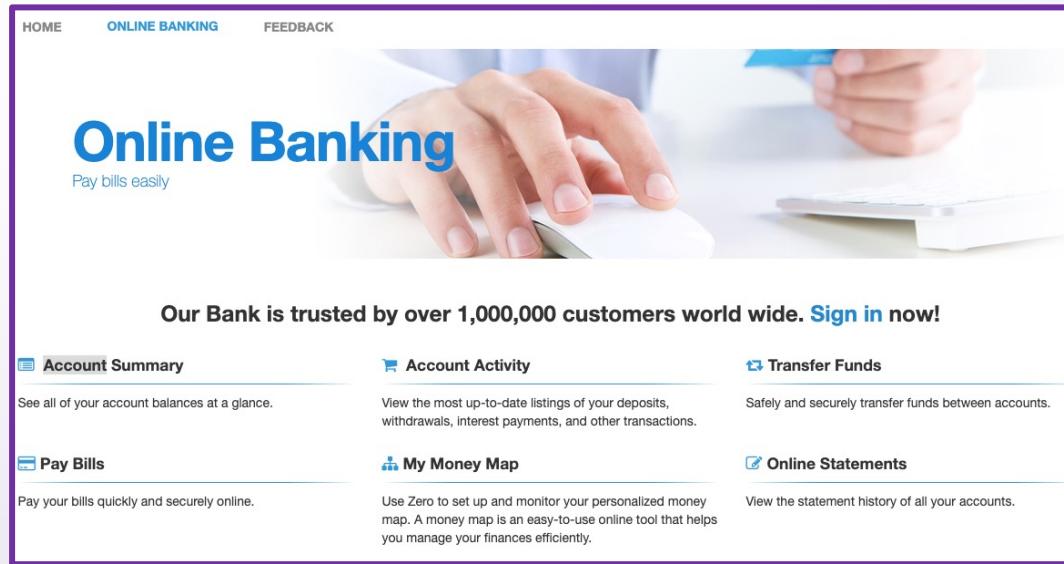
CATEGORY	
WOMEN	+
MEN	+
KIDS	+

## Automation Exercise Category Testi

- 1- Bir test class'i olusturun ilgili ayarlari yapin
- 2- <https://www.automationexercise.com/> adresine gidin
- 3- Category bolumundeki elementleri locate edin
- 4- Category bolumunde 3 element oldugunu test edin
- 5- Category isimlerini yazdirin
- 6- Sayfayı kapatın

# Locators

By.className( ) Homework



- 1- Bir test class'i olusturun ilgili ayarları yapın
- 2- <http://zero.webappsecurity.com/> adresine gidin
- 3- “ONLINE BANKING” linkine tiklayin
- 4- Resim altında 6 işlem baslığı olduğunu test edin
- 5- İşlem başlıklarında “Pay Bills” olduğunu test edin
- 6- Sayfayı kapatın

# Locators

## 4- By.tagName("tagName")

```
<input class="search-input" id="global-search" name="search" type="search"  
placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

```
List<WebElement> inputTagList =driver.findElements(By.tagName("input"));
```

Bir web sayfasında herhangi bir tagName'in unique olması nadiren karşılaşılmabilir bir durumdur.

Tag ismi ile yapılan locate'ler unique bir elemente ulaşmaktan daha çok sayfadaki tüm link'leri bulmak gibi amaclarla kullanılabilir.

Birden fazla web element döndürecegi için driver.findElements(..) ile kullanılması daha çok karşılaşılan bir durumdur.

# Locators

```
▼<a href="/view_cart"> == $0
▶<i class="fa fa-shopping-cart">...
  " Cart"
</a>
```

5- By.linkText("linkYazisininTamami")

6- By.partialLinkText("linkYazisininBirBolumu")

Sadece link'ler icin kullanilabilirler.

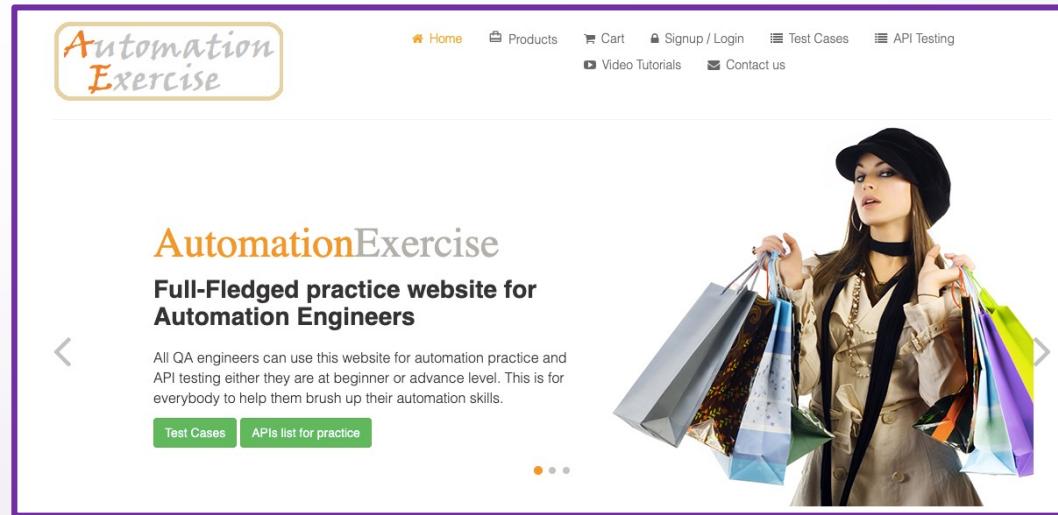
Her link uzerinde bulunan yazi kullanilarak locate yapmamizi saglar.

Link uzerinde bulunan yazi String data turunde oldugundan case sensitive'dir.

By.linkText ( ) icin bosluklar da dikkate alınarak tum metin yazılmalıdır.

Tum metnin yazılaması, yazının kısmı olarak kullanılması isteniyorsa  
By.partialLinkText ( ) kullanılmalıdır.

# Locators



## Automation Exercise Link Testi

- 1- Bir test class'i olusturun ilgili ayarlari yapin
- 2- <https://www.automationexercise.com/> adresine gidin
- 3- Sayfada 147 adet link bulundugunu test edin.
- 4- Products linkine tiklayin
- 5- special offer yazisinin gorundugunu test edin
- 6- Sayfayi kapatın

# WebElement Method'lari



TEST OTOMASYON

Search Product,Category...

Select Category ▾



```
<input class="search-input" id="global-search" name="search" type="search"  
placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

Bir web element'i locate etmek her testin vazgecilmez bir adimidir.

Webelement'i locate ettikten sonra variable atamak veya atamadan direk kullanmak test adimlarina ve belirlenen genel test stratejisine baglidir.

Webelement ile yapabilecegimiz islemler icin hazir method'lari kullaniriz.

1-

`webElement.click();`

Web element'e click yapar.

2-

`webElement.sendKeys(...keysToSend: "Istenen Metin");`

Web element'e istenen metni yollar

# WebElement Method'lari



TEST OTOMASYON

Search Product,Category...

Select Category ▾



```
<input class="search-input" id="global-search" name="search" type="search"  
placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

- 3- `webElement.submit();` Web element ile işlem yaparken ENTER tusuna basma işlemini yapar.
- 4- `webElement.sendKeys( ...keysToSend: "Istenen Metin" + Keys.ENTER);` Web element'e istenen metni yollayip, sonra ENTER tusuna basar
- 5- `webElement.isEnabled();` Web element erişilebilir ise true, yoksa false döner.
- 6- `webElement.isDisplayed();` Web element gorunuyor ise true, yoksa false döner.
- 7- `webElement.isSelected();` Web element secili ise true, yoksa false döner.

# WebElement Method'lari



TEST OTOMASYON

Search Product,Category...

Select Category ▾



```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

8- `webElement.getAttribute( name: "id");`

Webelement'in istenen attribute'unun  
degerini dondurur

9- `webElement.getLocation();`

Web element'in konumunu pixel olarak verir

10-  
`webElement.getLocation().getX();`  
`webElement.getLocation().getY();`

Web element'in konumuna ait X ve Y  
degerini pixel olarak verir

11- `webElement.getSize();`

Web element'in boyutunu pixel olarak verir

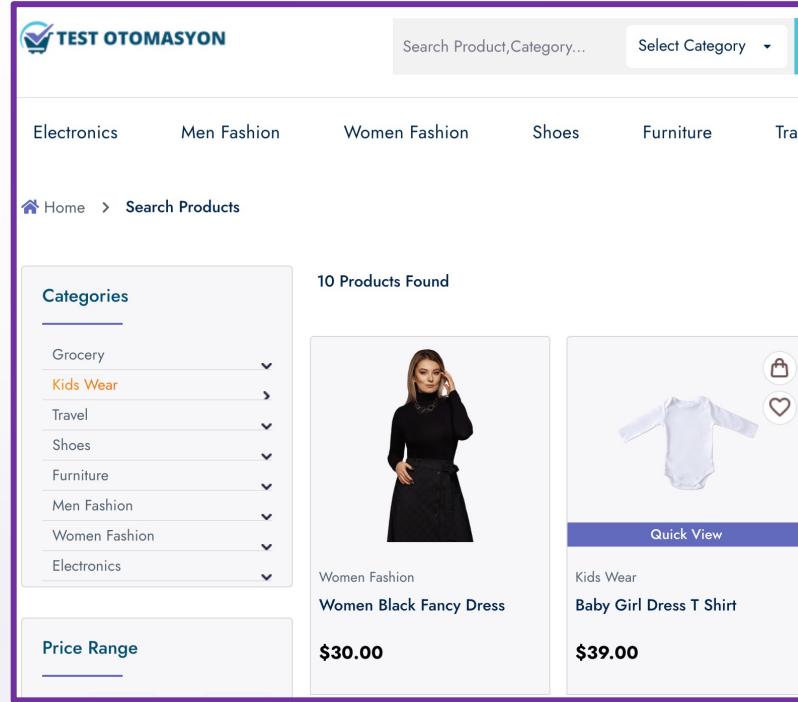
12-  
`webElement.getSize().getHeight();`  
`webElement.getSize().getWidth();`

Web elementin yukseklik ve genisligini pixel  
olarak verir.

# Locators

	<code>findElement( )</code>	<code>findElements( )</code>
websayfasında birden fazla Web Element Locator ile uyusursa	İlk elemani dondurur	Tüm elemanları dondurur
websayfasında hiçbir Web Element Locator ile uyuşmazsa	NoSuchElementException fırlatır	Exception fırlatmaz, boş bir liste dondurur
Return Type	WebElement	List<WebElement>
Elemana erişim	Direk ulaşılabilir	Liste'den index veya iterator ile ulaşılabilir

# Locators



The screenshot shows a web browser window with the title 'TEST OTOMASYON'. At the top, there is a search bar labeled 'Search Product,Category...' and a dropdown menu labeled 'Select Category'. Below the header, there are several navigation links: Electronics, Men Fashion, Women Fashion, Shoes, Furniture, Travel, and Contact Us. A breadcrumb trail indicates the user is at 'Home > Search Products'. On the left side, there is a sidebar with a 'Categories' section containing dropdown menus for Grocery, Kids Wear, Travel, Shoes, Furniture, Men Fashion, Women Fashion, and Electronics. Below this is a 'Price Range' section with a dropdown menu. The main content area displays a message '10 Products Found' above two product cards. The first card shows a woman in a black dress with the text 'Women Fashion' and 'Women Black Fancy Dress' below it, along with a price of '\$30.00'. The second card shows a baby's onesie with the text 'Kids Wear' and 'Baby Girl Dress T Shirt' below it, along with a price of '\$39.00'. Each card has a 'Quick View' button and icons for a shopping bag and a heart.

- 1- <https://www.testotomasyonu.com/> sayfasına gidin.
- 2- Arama kutusuna “dress” yazip aratin
- 3- Görüntülenen sonuçların sayısını yazdırın
- 4- Listededen ilk ürünün resmine tıklayın.
- 5- Ürün detayında dress kelimesi geçtiğini test edin.

# Locators - Xpath

Bir WebElement'i locate etmek için kullanabileceğimiz en etkin yöntemdir.

```
WebElement webElement= driver.findElement(By.xpath( xpathExpression: "bulunan xpath"));
```

Onceki 6 locator, HTML element oluşturulurken developer'in yazdığı kodlara göre yapılır.

Ornegin; HTML element'de id attribute'u varsa By.id( ) method'u kullanabilir ama developer id attribute'u koymedi ise kullanamayız.

Aynı şekilde HTML elementi bir link ise By.linkText( ) veya By.partialLinkText( ) kullanabiliriz, link degilse kullanamayız.



Xpath de HTML kodu kullanır ancak farklı kombinasyonlar kullanıldığı için dinamiktir ve her webelement için mutlaka bir xpath bulunabilir.

2 çeşit Xpath yazılabilir

1. **Absolute xpath** (mutlak)

2. **Relative xpath** (bağılı)

# Locators

## HTML kodlarda Parent – Child – Sibling ilişkisi

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

Her HTML sayfasi <Html> </Html> taglari arasina yazilir.

Bir HTML taginin arasina yeni bir tag acildiginda konum olarak bir tab icerden baslar.

HTML taginin dusey hizasina bakarak parent-child-sibling ilişkisi anlasilabilir.



# Locators

## 1. Absolute Xpath

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

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

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

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



Wise Quarter  
first class IT courses

# Selenium Team141

## Ders-05

Locators

Relative Locators

Maven



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



# Locators

## 2.Relative Xpath

```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

Bir web element'in 3 bileseni bulunur.

1- Tag : input

2- Attributes : type, id, value, name, autocomplete, placeholder, class, dir, tabindex, aria-label

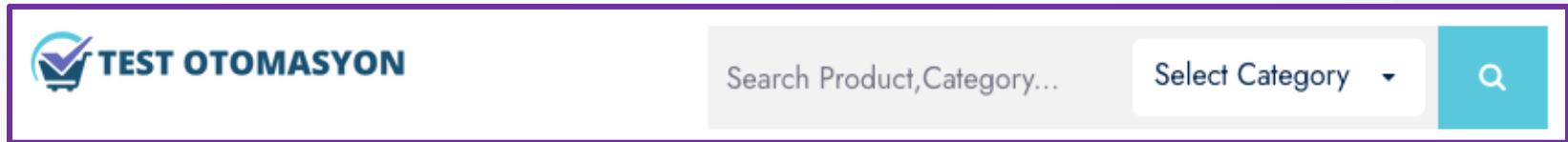
3- Attribute Values : text, twotabsearchtextbox, field-keywords .....

Relative Xpath bu 3 bilesenin belirlenen sekilde birlikte kullanilmasi ile olusur. Her Xpath ile unique bir sonuc elde edilemeyebilir ancak unique bir deger mutlaka bulunur.

//tagName[@attributelsmi='attributeValue']

# Locators

## 2.Relative Xpath



```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

Unique deger icin 3 bilesenin tamami kullanilmak zorunda degildir.

Hedef unique olarak webelement'i locate etmektir. Daha az bilesenle de unique degere ulasabilen Xpath'ler de kullanilabilir.

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

Sadece tag ismi ile

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

tag ismi farketmeksizin

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

Attribute farketmeksizin

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

Attribute value farketmeksizin

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

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

# Locators

## 2. Relative Xpath

Link disindaki bazi webelementler'inde de text bulunabilir.

Bu text'ler o webelement'e ozel oldugu icin unique bir xpath elde etmek icin kullanisli olabilirler.

Text ile locate yazmak icin kullanilan genel syntax :

```
//tagName[text()='yazinin tamami']"
```

Genel xpath kullanimina uygun olarak tagname veya attribute ismi yazilmadan da text ile xpath yazilabilir.

```
//tagname[.='yazinin tamami']"
```

```
//*[.='yazinin tamami'] "
```

Metnin sadece bir kismi kullanilacaksa

```
//*[contains(text(),'yazinin bir bolumu')] "
```



# Locators

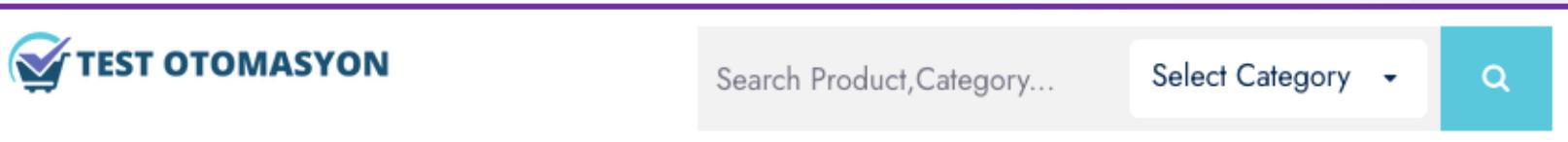
The screenshot shows a top navigation bar with the TEST OTOMASYON logo, a search bar containing 'Search Product,Category...', a 'Select Category' dropdown, and a magnifying glass icon. Below the navigation is a horizontal menu with links: Electronics, Men Fashion, Women Fashion, Shoes, Furniture, and Travel. A large central box contains the text 'Add/Remove Elements' and a blue 'Add' button.

## Relative Xpath Soru

- 1- [https://testotomasyonu.com/addremove/ adresine gidin](https://testotomasyonu.com/addremove/)
- 2- Add Element butonuna basin
- 3- Remove butonu'nun gorunur oldugunu test edin
- 4- Remove tusuna basin
- 5- "Add/Remove Elements" yazisinin gorunur oldugunu test edin

# Locators

## 8.cssSelector



```
<input class="search-input" id="global-search" name="search" type="search" placeholder="Search Product,Category..." onkeyup="searchData(this.value)">
```

cssSelector de xpath'e benzer bir kullanıma sahiptir. Tag ismi, attribute ismi ve attribute value ile yapılacak kombinasyonlarla oluşturulur.

```
driver.findElement(By.cssSelector("input[id='twotabsearchtextbox']"));
```

Hedef unique olarak webelement'i locate etmektir. Daha az bilesenle de unique degere ulasılabilen Xpath'ler de kullanılabilir.

cssSelector özellikle class ve id attribute'leri ile kısa şekilde yazılabilir

```
driver.findElement(By.cssSelector("#twotabsearchtextbox"));
```

Id attribute ile

```
driver.findElement(By.cssSelector(".nav-input nav-progressive-attribute"));
```

Class attribute ile



# Tekrar Sorusu

- 1- bir class olusturun
- 2- <https://www.testotomasyonu.com/> adresine gidin
- 3- Browseri tam sayfa yapin
- 4- Sayfayı “refresh” yapın
- 5- Sayfa basliginin “Test Otomasyonu” ifadesi icerdigini test edin
- 6- Furniture linkine tiklayin
- 7- price range filtresinde min degere 40, max degere 200 yazip filtreleyin
- 8- filtreleme sonucunda urun bulunabildigini test edin
- 10-ilk urunu tiklayin
- 11- Urun fiyatinin 40 ile 200 arasında oldugunu test edin
- 12-Sayfayı kapatın

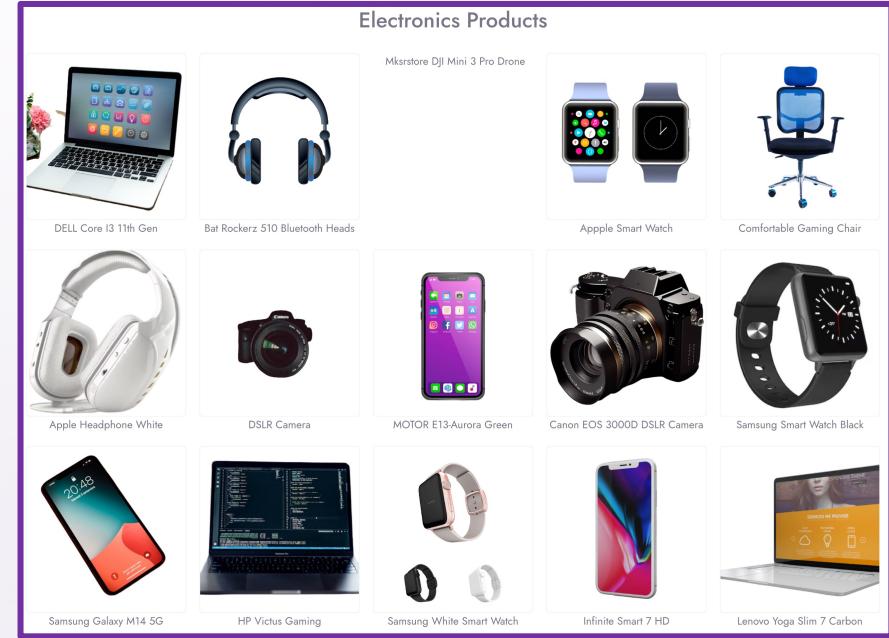
# Relative Locators

## Relative Locators nedir ?

Bir web elementi direk locate edemedigimiz durumlarda gunluk hayatimizda kullandigimiz sekilde o web elementi etrafindaki web elementlerin referansi ile tarif edebiliriz.

Bu ozellik Selenium 4 ile gelen yeniliklerden biridir.

Ornegin yandaki resimde ulasamadigimiz bir webelement olursa, etrafinda locate edilebilen webelementlerden faydalananip relative locator tanimlayabiliriz.

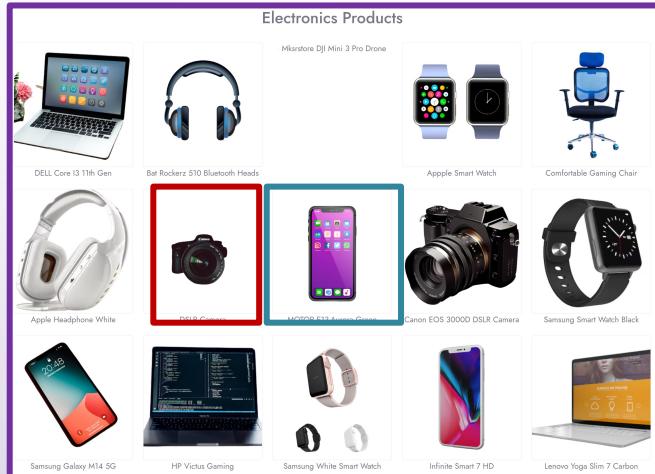


Amac **hedef webelement'e** ait bir ozellik ve **etrafinda** locate edilebilir bir web element ile hedefimizdeki webelement'i locate etmektir.

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

# Relative Locators

## Class Work: Relative Locators



- 1- Ulaşmak istediğiniz webelement'i belirleyin
- 2- Hedef webelement'in HTML kodlarını bulun ve kullanacağınız bir attribute veya tag'i kopyalayın
- 3- Hedef webelement'in etrafından locate edebileceğimiz bir webelement'i locate edin
- 4- Hedef webelement'in ozelligini ve locate ettigimiz webelementi kullanarak locate yapın

```
// https://testotomasyonu.com/relativeLocators sayfasına gidin
driver.get("https://testotomasyonu.com/relativeLocators");
// DSLR Camera elementini locate edin
WebElement dslrCameraElementi = driver.findElement(By.id("pic7_thumb"));

// DSLR camera webelementi ve relative locator kullanarak
// Motor E13 telefonu locate edin

WebElement motorE13Elementi =
    driver.findElement(RelativeLocator.with(By.tagName("img")).toRightOf(dslrCameraElementi));

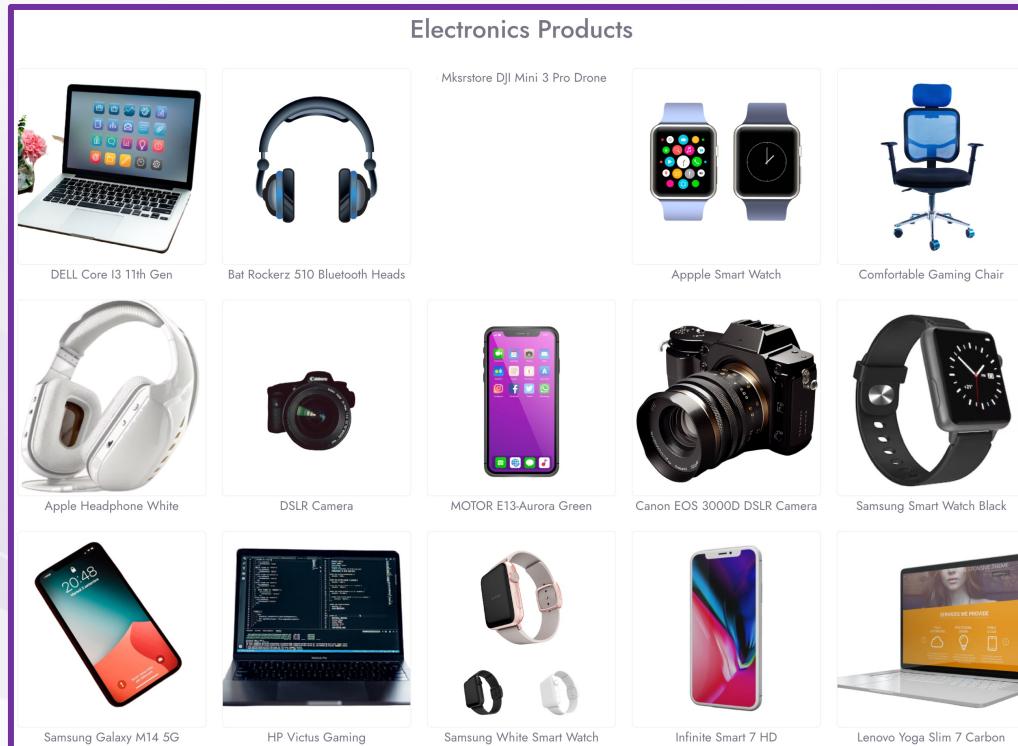
// DSLR camera webelementi ve relative locator kullanarak
// Hp Victus bilgisayarı locate edin

WebElement hpVictusElementi =
    driver.findElement(RelativeLocator.with(By.id("pic12_thumb")).below(dslrCameraElementi));
```

# Relative Locators

## Class Work: Relative Locators

- 1 ) <https://testotomasyonu.com/relativeLocators> adresine gidin
- 2 ) DSLR Camera'yi etrafındaki elementleri kullanarak 3 farklı relative locator ile locate edip tiklayın
- 3 ) Acılan ürünün DSLR Camera olduğunu test edin.



# Relative Locators

## Homework: Relative Locators

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



# Maven



Apache Maven yazılım projelerinin kolay anlasılmasına ve yönetilmesine odaklanmış bir tool'dur.

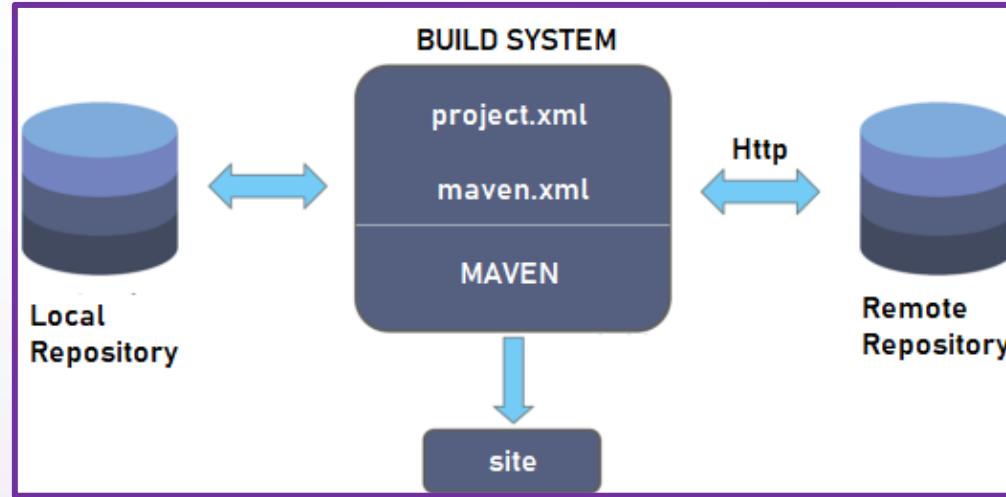
Proje nesne modeli (POM) konseptine dayalı olarak Maven, bir projenin inşasını, raporlamasını ve dokümantasyonunu merkezi bir bilgi parçasından(**dependency**) yönetebilir.

- 1- Projeleri oluşturma için standart belirleme,
- 2- projenin nelerden olduğunu net olarak tanımlama,
- 3- proje bilgilerini yayinallyamanın kolay bir yolunu bulma
- 4- JAR dosyalarını farklı projeler arasında paylaşma

Amaçları çerçevesinde başlayan bir çalışma sonucu ortaya çıktı.

Sadece Java tabanlı projeleri oluşturmak ve yönetmek için kullanılabilecek bir araçtır.

# Maven



Maven bir Java oluşturma aracıdır (**build tool**). Maven proje otomasyon ve yönetim aracıdır (**automation and management tool**).

Maven, konfigürasyon için pom.xml dosyasını kullanır.

pom.xml projenin insası , raporlaması ve dokümantasyonu için gerekli bütün bilgileri içerir ( dependencies , plugins v)

Maven ile çalışan tüm projeler aynı konfigürasyonu kullandığı için yeni bir projede çalışmaya başlandığında projenin anlaşılması çok kolay olacaktır.

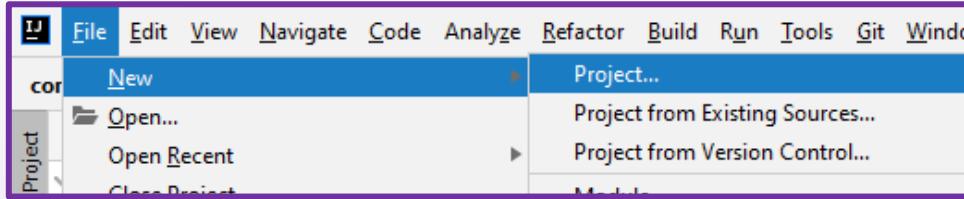
# Neden Maven ?

- 1- Proje yönetiminde oluşturma, belgeleme, yayınılama ve dağıtım gibi tüm süreçlerin yönetilmesine yardımcı olur.
- 2- Projenin ve yapılm sürecinin performansını artırır.
- 3- Jar dosyalarını ve diğer bağımlılıkları (dependencies) indirme görevi otomatik olarak yapılır
- 4- Gerekli tüm bilgilere kolay erişim sağlar
- 5- Geliştiricinin, bağımlılıklar, süreçler vb. hakkında endişelenmeden farklı ortamlarda bir proje oluşturmasını kolaylaştırır.
- 6- Open source olduğundan ücretsizdir, geniş bir kullanıcı tabanı olduğu için karşıtlan sorunlara çözüm bulmakta zorluk yaşanmaz.





# Maven Proje Olusturma

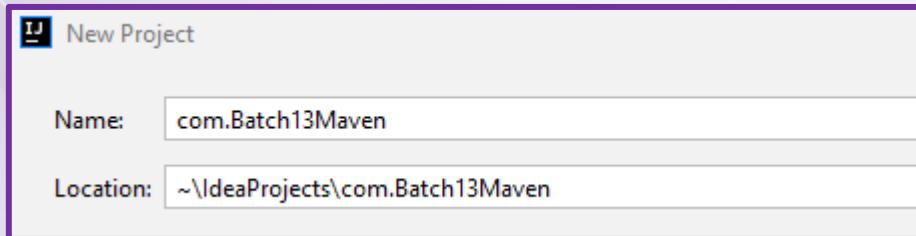


1. Create Project: File -> New -> Project



2. Select Maven -> click next

Java versiyonunun bilgisayarinizdaki version ile ayni oldugunu control edin



3. Name: com.projeAdi -> click finish

EnableAutoImport sorarsa click

4. Package olusturun, name : day04

5. Class olusturun, name : FirstMavenClass

# pom.xml

```
<?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>SeleniumInt</artifactId>
<version>1.0-SNAPSHOT</version>

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

<dependencies>
  <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
  <dependency>...</dependency>

  <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
  <dependency>...</dependency>
</dependencies>
</project>
```

Project Object Model (pom) Maven projesinin temelini oluşturur.

pom.xml proje hakkında bazi bilgiler ve Maven tarafından projeyi olusturmak icin kullanılan konfigurasyon detaylarini gösterir.

Bir projenin hangi framework'u kullandigini anlamak icin pom.xml'e bakmak yeterlidir.

POM'da belirtilebilecek yapılandırmalardan bazıları proje bağımlılıkları(dependencies), yürütülebilecek eklentiler(plugin) veya hedefler(goal), yapı profilleri vb. Proje sürümü, açıklama, geliştiriciler (artifact id, group id, version) ve benzeri gibi diğer bilgiler de belirtilebilir.

Projeye eklenecek dependency'ler mvnrepository.com'dan bulunabilir.

Istenen dependency icin version secilirken guncellik, stabil versiyon olma ve kullanılma sayiları dikkate alınmalıdır.



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

<dependencies>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency...>

    <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager -->
    <dependency...>
</dependencies>
</project>
```

- 1- pom.xml'de </properties> kapanis tagi ile </project> kapanis tagi arasinda <dependencies> acilis ve </dependencies> kapanis tag'larini olusturun.
- 2- mvnrepositories.com adresinden WebDriverManager ve Selenium Java dependency'lerini kopyalayip, dependencies taglari arasina yapistirin
- 3- Bu dependency'leri projenize ilk defa yuklediginiz icin versiyon bolumleri kirmizi cikacaktir. Kirmiziliği gidermek icin 4. adimi takip edin.

<version>4.5.0</version>

# pom.xml'e Dependency Ekleme



4- projenin sag ust kisminda bulunan Maven yazisini tiklayin, acilan bolumde yenile butonuna tiklayin ve yuklediginiz dependency'lerin projenize eklendigini kontrol edin.

The screenshot shows an IDE interface with several tabs at the top: Test.java, pom.xml (SeleniumInt), C03\_linkText.java, C05\_noSuchElementExc.java, C06\_webElementMethodlari.java, C01\_Xpath.java, C01\_Xpath2.java, and Maven. The Maven tab is active, showing the Maven tool window. The left pane of the Maven window displays the project structure: Seleniumpoint, Lifecycle, Plugins, Dependencies, and two entries under Dependencies: org.seleniumhq.selenium:selenium-java:4.4.0 and io.github.bonigarcia:webdrivermanager:5.3.0. The right pane shows notifications. The main code editor on the left contains the pom.xml file with the following content:

```
<?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/pom-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>SeleniumInt</artifactId>
    <version>1.0-SNAPSHOT</version>

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

    <dependencies>
        <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
        <dependency>
            <groupId>org.seleniumhq.selenium</groupId>
            <artifactId>selenium-java</artifactId>
            <version>4.4.0</version>
        </dependency>

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

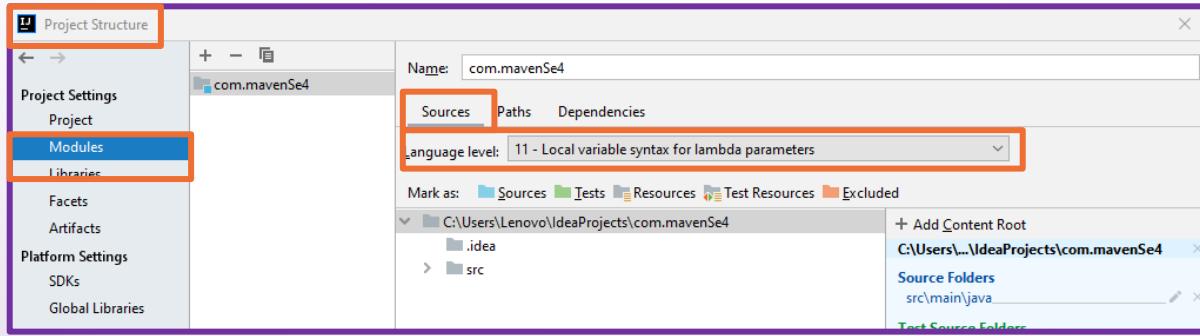
```

The dependencies section is highlighted with a red box. The entire pom.xml file is also highlighted with a red box.

# JDK8 Oncesi Versiyonlar'i Kullanma



Wise Quarter  
first class IT courses

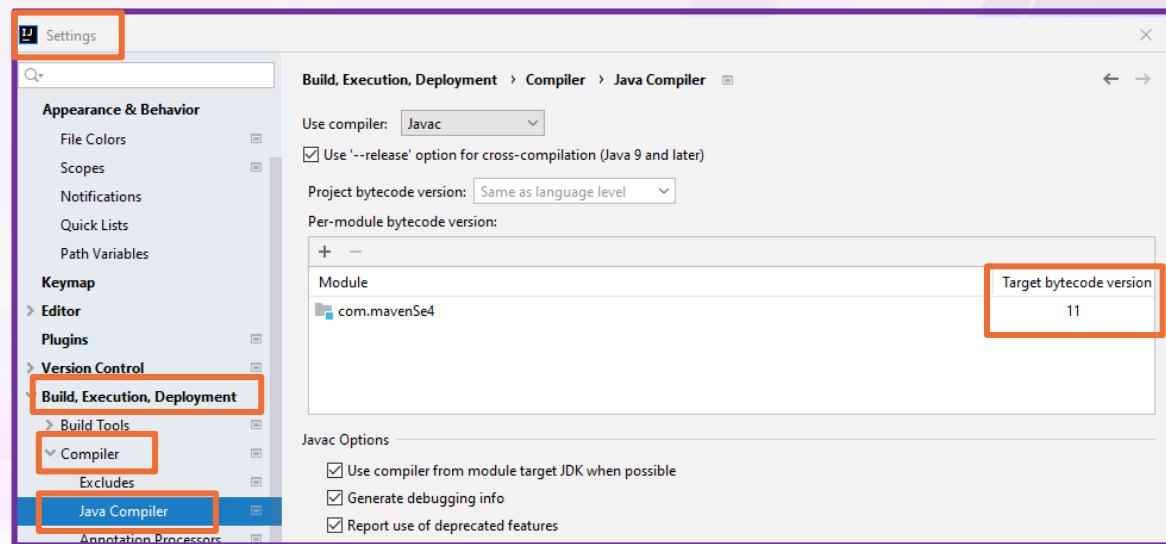


## File

## Project Structure Modules Sources

Language level : min 8 yapin, bilgisayarinizda kurulu java 8 veya ustу ise java versiyonu aynи olmali

File  
Settings  
Build,Execution,Deployment  
Compiler  
Java Compiler  
Target bytecode version : min 8 yapin, bilgisayarinizda kurulu java 8 veya ustу ise java versiyonu aynи olmali





# Maven WebDriver Oluşturma

Maven ile Selenium Java ve WebDriverManager dependency'lerini projemize eklediği için, her seferinde driver.exe dosyasını driver'a tanıtma mecburiyeti kalmadı.

```
System.setProperty("webdriver.chrome.driver", "src/resources/chromedriver");
WebDriver driver=new ChromeDriver();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));
driver.manage().window().maximize();
```

Bundan sonra driver ayarları yapılarken ilk satırda sistem ayarlarını bonigarcia WebDriverManager kullanarak yapacağız.

```
WebDriverManager.chromedriver().setup();
WebDriver driver=new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));
```

Chrome之外のブラウザを使用する場合は、1. と 2. 行で Chrome ではなくその他のブラウザを選択する必要があります。

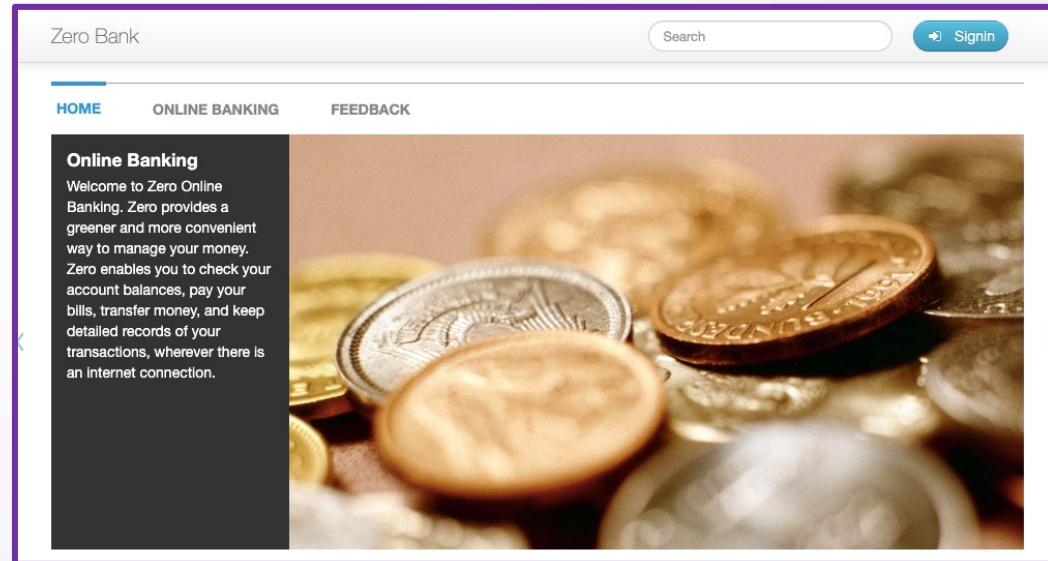
# Maven ClassWork



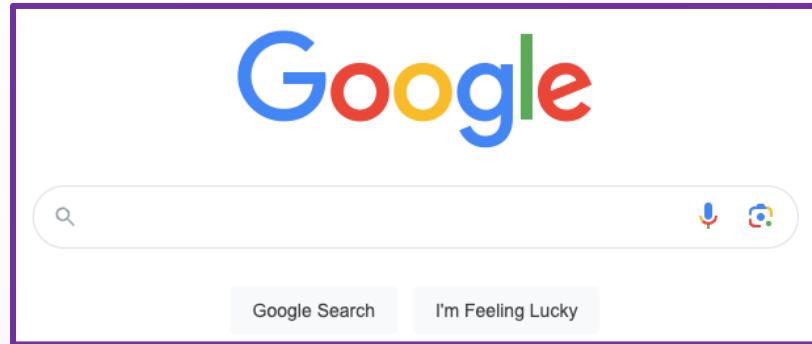
- 1- testotomasyonu.com anasayfasina gidelim
- 2- arama kutusunu locate edelim
- 3- “phone” ile arama yapalim
- 4- Bulunan sonuc sayisini yazdiralim
- 5- Ilk urunu tiklayalim
- 6- Urunun stokta var oldugunu test edin

# Maven ClassWork

1. <http://zero.webappsecurity.com> sayfasina gidin
2. Signin buttonuna tiklayin
3. Login alanine “username” yazdirin
4. Password alanina “password” yazdirin
5. Sign in buttonuna tiklayin
6. Back tusu ile sayfaya donun
7. Online Banking menusunden Pay Bills sayfasina gidin
8. amount kismina yatirmak istediginiz herhangi bir miktarı yazin
9. tarih kismina “2023-09-10” yazdirin
10. Pay buttonuna tiklayin
11. “The payment was successfully submitted.” mesajinin ciktigini test edin



# Maven Tekrar Testi



- 1- C01\_TekrarTesti isimli bir class olusturun
- 2- <https://www.google.com/> adresine gidin
- 3- cookies uyarisini kabul ederek kapatın
- 4- Sayfa basliginin “Google” ifadesi icerdigini test edin
- 5- Arama cubuguna “Nutella” yazip aratin
- 6- Bulunan sonuc sayisini yazdirin
- 7- sonuc sayisinin 10 milyon’dan fazla oldugunu test edin
- 8- Sayfayı kapatın

# Maven Tekrar Testi

Swag Labs

Username

Password

Login

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



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-04

Maven

Junit Framework

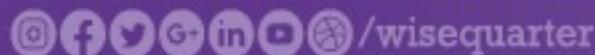
Junit Assertions



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)





# Onceki Dersten Aklimizda Kalanlar

Selenium'da orkestrasyon yapiyoruz. Elimizdeki enstrumanlar

1-Java : programlama dili

-IntelliJ : IDE(Compiler), Java gibi high level programlar icin compiler gerekir.

2- Selenium : Browserlari otomasyonla calistiracak tool'lar icin suite

- WebDriver : bizim elimiz ve gozumuzle, klavye mouse kullanarak yaptigimiz islemleri bizim adimiza yapar

- WebElement : Bir web sayfasindaki etkilesimli veya etkilesimsiz her sey

- WebElement'leri driver'in bulabilmesi icin onlari oncelikle locator ile locate edip, findElement(s) ( ) ile bulmasini ve bize getirmesini sagliyoruz.

- Selenium kendi WebDriver'ina sahiptir. Ancak sirketler farkli sebeplerle kendilerine ait WebDriver'lar kullanabilirler. Biz de simdilik ucretsiz olan Boni Garcia WebDriver kullanacagiz.

3- Maven : Build tool. Projelere ihtiyacimiz olan dependency'leri kolaylikla ekleyebilmek, update edebilmek ve projeleri daha Rahat anlayabilmek icin gelistirilmistir

4- Framework : Ortak calisma duzeni kurmak icin cizilen cercevelerdir. Sadece duzenleme ile sinirli kalmazlar, bize pek çok kolaylik saglarlar.

# JUnit

Java ile olusturulabilecek en temel Test Framework'udur.

Open-source bir kutuphanedir.

Developer'lar tarafindan yapılan unit test'ler icin de kullanilir.

Testlerimizi yaparken bize kolaylik saglamak amaciyla olusturulan Assert, Test, Before, After gibi bir çok class'a sahiptir.

JUnit, Maven altyapisini kullanir. JUnit framework icin ihtiyacimiz olan kutuphaneleri mvn.repository.com adresinden bulabilecegimiz dependency'ler ile projemize ekleyebiliriz.

JUnit'de testler icin ihtiyac duyulan pek çok olsa da TestNG next generation olarak daha fazla ozellik ile piyasaya cikmis ve piyasayı domine etmistir.



JUnit5 eski versiyonlardan farklı olarak birden fazla dependency kullanır.

**JUnit Jupiter:** JUnit 5'in yeni test motorudur. Bu modül, Java 8'den itibaren gelen özellikleri kullanarak test yazmayı kolaylaştırır. Yeni annotasyonlar ve genişletilebilir mimari sayesinde JUnit Jupiter, testlerinizi esnek ve okunabilir hale getirir.

**JUnit Platform:** JUnit 5 testlerinin çalıştırılmasını, raporlanmasını ve entegrasyonunu sağlayan temel altyapıyı sağlar.

**JUnit Vintage:** Bu modül, JUnit 3 ve JUnit 4 testlerinin JUnit 5 platformunda çalışmasını sağlar. Mevcut testlerinizi yükseltirken uyumluluk ve geçiş sürekliliği sağlar.

```
</properties>

<dependencies>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java -->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>4.19.1</version>
    </dependency>
    |
    <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
    <dependency>
        <groupId>org.junit.jupiter</groupId>
        <artifactId>junit-jupiter-api</artifactId>
        <version>5.10.2</version>
        <scope>test</scope>
    </dependency>
    |
    <!-- https://mvnrepository.com/artifact/org.junit.platform/junit-platform-launcher -->
    <dependency>
        <groupId>org.junit.platform</groupId>
        <artifactId>junit-platform-launcher</artifactId>
        <version>1.10.2</version>
        <scope>test</scope>
    </dependency>
```

## Annotations

JUnit ile Java class'larinin vazgecilmez ogesi olan main method ihtiyaci ortadan kalkar

Annotations ile compiler'a talimatlar verilebilir

Annotations kodların daha iyi anlasilabilmesi ve daha iyi bir yapı kurulabilmesi icin gerekli meta-data (basit bilgi)'lari Java'ya iletmek icin kullanılır.

Testler sirasinda en cok kullanılan Junit notasyonları

@Test

@Disabled

@BeforeEach , @AfterEach

@BeforeAll , @AfterAll



## @Test ve @Disabled Annotations

@Test notasyonu bir method'un bagimsiz olarak calisabilmesini saglar.

@Test notasyonu olan method'un calismasi icin Java'daki gibi method call yapilmasina ihtiyac yoktur, bagimsiz olarak calistirilabilir.

Class level'daki run butonu ile class'daki tum test method'lari da birlikte calistirilabilir.

Junit'in method'lari calistirma sirasi ongorulemez. Bunun icin her test method'u bagimsiz olarak calistirilabilir olmalidir.

@Disabled notasyonun tanimli olduğu metotlar test sirasinda calistirilmayacaktir. Ayrca istenilirse @ Disabled("açıklama") şeklinde yazilarak metodun neden test edilmesini istemedigimizide yazabiliriz.

```
@Test  
@Disabled  
public void youtubeTesti(){...}
```

```
@Test  
public void test0tomasyonuTesti(){...}
```

```
@Test  
public void wisequarterTesti(){...}
```



# JUnit

## @Test Annotation

JUnit standart olarak, calistirilan her test method'unun sorunsuz olarak calisip calismadigini raporlar.

Calistirilan testlerin kac tanesinin passed, kac tanesinin failed oldugunu yazar.

Passed olan testler yesil tik ile, failed olan testler kirmizi carpi isareti ile, ignore edilen testler ise gri bir daire uzerine bir cizgi cekilerek isaretlenir,

Ancak JUnit'in dogru raporlama yapabilmesi icin Assert class'indan method'lar kullanilmalidir. Aksi takdirde sadece test method'unun sorunsuz calismis oldugunu raporlar.

```
@Test  
public void test01(){  
    System.out.println("test01");  
}
```

```
@Test @Ignore  
public void test02(){  
    System.out.println("test02");  
}
```

```
@Test  
public void test03(){  
    Assert.assertEquals( expected: 5, actual: 7);  
    System.out.println("test03");  
}
```

The screenshot shows a JUnit test run interface. On the left, a tree view lists the test classes and methods. The 'test01' method under 'C01\_JUnitIlikTest' is marked with a green checkmark and labeled '1ms'. The 'test02' method is marked with a grey circle and labeled '0 ms'. The 'test03' method is marked with a red X and labeled '4 ms'. To the right, the results are summarized: 'Tests failed: 1, passed: 1, ignored: 1 of 3 tests - 5ms'. Below this, the output for each test is shown. For 'test01', it says 'test01'. For 'test02', it says 'Test ignored.'. For 'test03', it shows an assertion error message: 'java.lang.AssertionError: Expected :5 Actual :7 <Click to see difference>'.

# JUnit

## @BeforeEach - @AfterEach Annotations

@BeforeEach ve @AfterEach notasyonuna sahip method'lar her @Test method'undan once ve sonra calisirlar.

Bu method'lar sayesinde driver olusturulurken yaptigimiz ayarlari ve test method'u bittikten sonra driver'in kapatilmasi gibi gorevler icin tekrar tekrar kod yazmak mecburiyeti ortadan kalkar.

Genellikle @BeforeEach notasyonu kullanan method icin **setup**, @AfterEach notasyonu kullanan method icin **teardown** isimleri kullanilir.

```
WebDriver driver;
```

```
@BeforeEach
public void setUp(){
    WebDriverManager.chromedriver().setup();
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));
    System.out.println("setup method'u calisti");
}
```

```
@AfterEach
public void teardown(){
    System.out.println("teardown method'u calisti");
    ReusableMethods.bekle(saniye: 2);
    driver.quit();
}
```

```
@Test
public void youtubeTesti(){...}
```

```
@Test
public void testOtomasyonuTesti(){...}
```

```
@Test
public void wisequarterTesti(){...}
```

## Test Vs Test Method'u

Test method'u @Test notasyonu kullanilarak olusturulan, tek basina da veya baska test methodlari ile birlikte calistirilabilen bir test case'dir.

Test ise genellikle birden fazla method, class veya package icerebilen, belirli bir amac icin calistirilan test method'larinin bütünüdür.

Ornek olarak, smoke test, regression test veya end2end testlerini söyleyebiliriz.

```
WebDriver driver;
@Before
public void setUp(){
    WebDriverManager.chromedriver().setup();
    driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));
}
@Test
public void amazonTest(){
    driver.get("https://www.amazon.com");
    System.out.println(driver.getTitle());
}
@Test
public void facebookTest(){
    driver.get("https://www.facebook.com");
    System.out.println(driver.getTitle());
}
@Test
public void bestbuyTest(){
    driver.get("https://www.bestbuy.com");
    System.out.println(driver.getTitle());
}
@After
public void tearDown(){
    driver.close();
}
```



# JUnit

## @BeforeAll - @AfterAll Annotations

Eger bir class'daki birden fazla test method'u icin driver'in bir kere olusturulmasi ve tum testleri bittikten sonra driver'in kapatilmasi yeterli olacaksa kullanilirlar.

Boylece driver'i tekrar tekrar acip kapatmak zorunda kalinmaz.

**NOT :** @BeforeAll ve @AfterAll notasyonu kullanacak method'lar static olmak zorundadir.

```
static WebDriver driver;  
List<WebElement> bulunanUrunElementleriList;
```

```
@BeforeAll  
public static void setup(){  
    WebDriverManager.chromedriver().setup();  
    driver = new ChromeDriver();  
    driver.manage().timeouts().implicitlyWait  
    driver.manage().window().maximize();  
}
```

```
@AfterAll  
public static void teardown(){  
    ReusableMethods.bekle( saniye: 2);  
    driver.quit();  
}
```

```
@Test  
public void test01(){...}
```

```
@Test  
public void test02(){...}
```

```
@Test  
public void test03(){...}
```

## Annotations

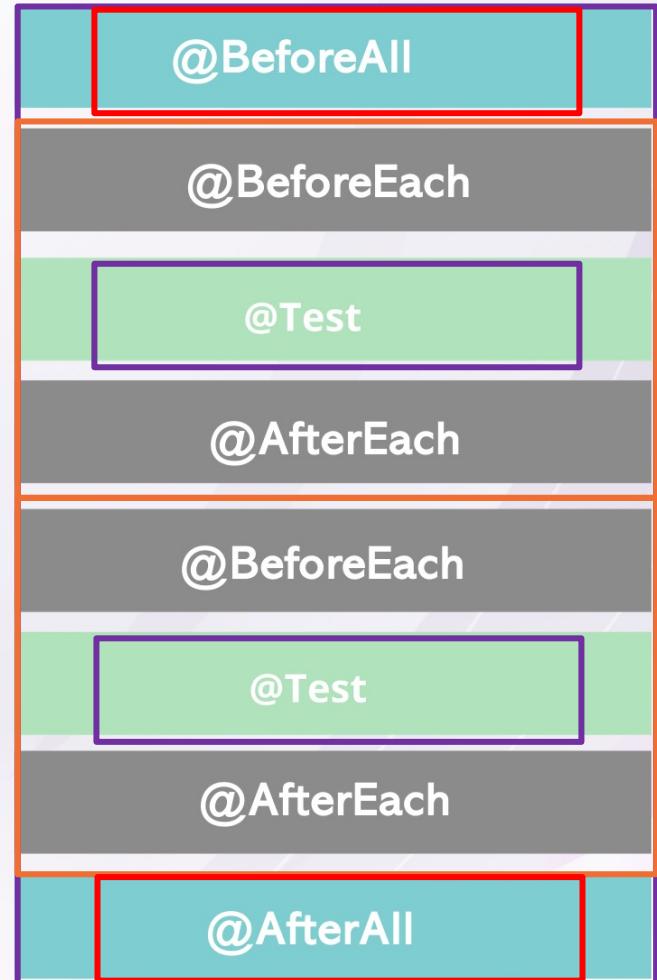
Bir class'da hem @BeforeEach - @AfterEach notasyonlari, hem de @BeforeAll-@AfterAll notasyonlari birlikte kullanilabilir.

Birlikte kullanimda method'larin yapacakları islemler ve yapıları, çalışma sırası ve notasyon özelliklerine göre düzenlenmelidir.

Ornek : 2 Test method'u

@BeforeEach - @AfterEach notasyonlari

@BeforeAll-@AfterAll notasyonlari olan bir class calistirilirsa, yandaki gibi bir calisma olacak ve toplam 8 method calistirilmis olacaktir.





Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-05

Junit Assertions  
Dropdown Menu



+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



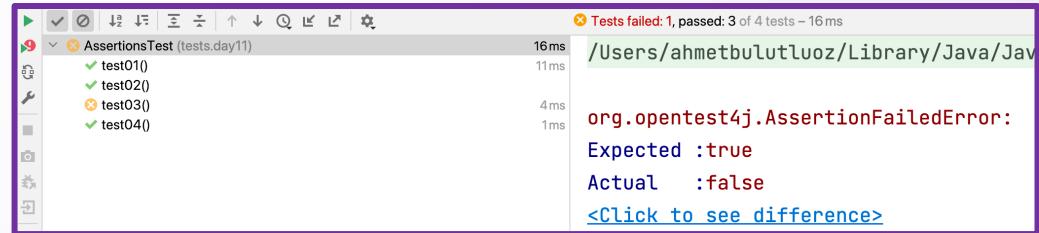
# Onceki Dersten Aklimizda Kalanlar

- 1- Framework'ler calisma ortamini duzenlemenin yaninda bizim icin faydalı ve kolaylik saglayici islevler saglarlar.
- 2- Junit ile hayatimize giren en buyuk yenilik notasyonlardir. @ ile baslayan notasyonlar, framework'un isleyisine mudahale edecek kucuk bilgilere sahiptirler.
- 3- @Test : siradan bir method cagrilmadan calismaz. @Test notasyonu eklenen method'lar ise bagimsiz olarak calistirilabilir. Bu da bizi main method bagimlilikinden kurtarir.
- 4- @Disabled : yazildigi test notasyonunun ignore edilmesini saglar
- 5- @BeforeEach ve @AfterEach : Her test method'undan once calismasini istedigimiz siradan bir method varsa, @BeforeEach ile, her test method'undan sonra calismasini istedigimiz siradan bir method varsa da @AfterEach ile isaretleriz.
- 6- JUnit bir test method'unun failed veya passed olmasina test method'u calisirken kodların bir sorunla karsilasip karsilasmadigina gore karar verir.  
Eger kodlar hic bir sorunla karsilasmadan sona kadar calisirsa, (if-else ile yaptigimiz test failed olsa bile) test PASSED olarak isaretlenir.  
Eger kodlar calisirken bir sorunla karsilasilar ve exception olusursa testi FAILED olarak isaretler.



Junit framework Assertions method'ları kullanıldığında, her test method'unun FAILED veya PASSED olduğunu otomatik olarak raporlar.

```
public class AssertionsTest {  
    int a = 10;  
    int b = 20;  
    int c = 30;  
  
    @Test  
    public void test01(){  
        Assertions.assertEquals(c, actual: a+b);  
    }  
    PASSED  
  
    @Test  
    public void test02(){  
        Assertions.assertNotEquals(c,b);  
    }  
    PASSED  
  
    @Test  
    public void test03(){  
        Assertions.assertTrue(condition: c==b);  
    }  
    FAILED  
  
    @Test  
    public void test04(){  
        Assertions.assertFalse(condition: c==b);  
    }  
    PASSED
```



Junit Assertions Class'indaki static method'lar, kendilerine verilen değerlerin method ismi ile uyumlu olup olmadığını kontrol ederler.

Secilen assertion method'u ile içine yazılan değerlerin uyumlu olup olmamasına göre FAILED veya PASSED sonucu üretir.

## Assertions

JUnit ile assertion icin en cok kullanilan 4 method vardir.

```
Assertions.assertEquals(c,a+b);
Assertions.assertNotEquals(c,b);
Assertions.assertTrue(c==b);
Assertions.assertFalse(c==b);
```

Kullanilacak method secilirken, assert isleminin icerisine yazilan degerler ile islem sonucunun ne olmasi beklendiği dusunulmeli, assert method'u beklenen sonuca uyumlu olarak secilmelidir.

Sonucun 20 oldugunu test edin → Assertions.assertEquals( sonuc , 20 )

Sonucun 20 olmadigini test edin. → Assertions.assertNotEquals(sonuc,20)

Sayinin 50'den buyuk oldugunu test edin. → Assertions.assertTrue( sayi>50)

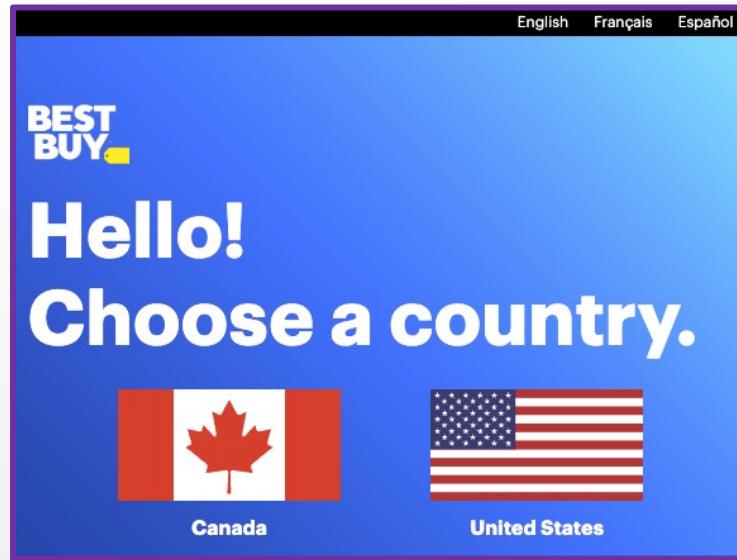
Sayinin 50'den buyuk olmadigini test edin. → Assertions.assertFalse( sayi>50)



```
public class AssertionsTest {  
    String a = "Ali";  
    String b = "ALI";  
    String c = "Ali";  
  
    @Test  
    public void test01(){  
  
        Assertions.assertEquals( a , c ); → Passed  
         equals  
  
        Assertions.assertNotEquals( b , c ); → Passed  
         not equals  
  
        Assertions.assertTrue ( a.equals(b) ); → Failed  
         false  
  
        Assertions.assertFalse( b.equals(c) ); → Passed  
         false  
  
        Assertions.assertEquals( a.toUpperCase(),b ); → Passed  
         equals  
  
        Assertions.assertNotEquals( b,c.toUpperCase() ); → Failed  
         equals  
  
        Assertions.assertTrue( a.equalsIgnoreCase(b) ); → Passed  
         true  
    }  
}
```

# JUnit

## Assertions

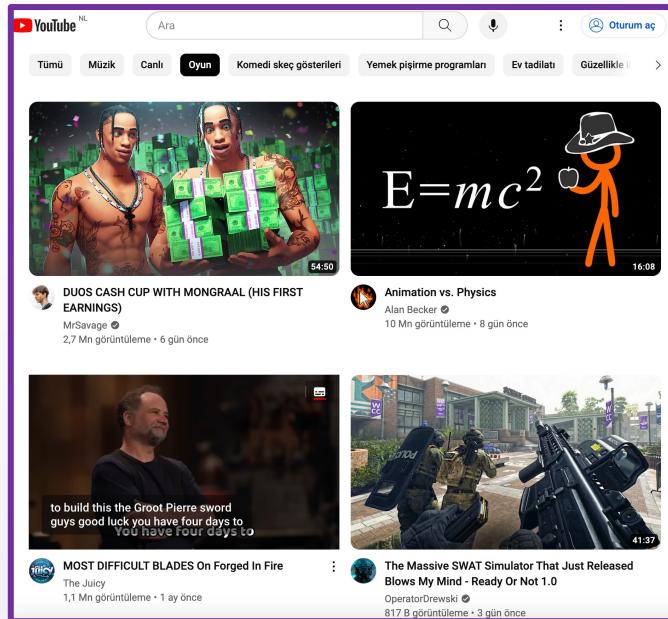


<https://www.bestbuy.com/> Adresine gidin farkli test method'lari olusturarak asagidaki testleri yapin

- Sayfa URL'inin <https://www.bestbuy.com/> 'a esit oldugunu test edin
- titleTest => Sayfa başlığının "Rest" içermediğini(contains) test edin
- logoTest => BestBuy logosunun görüntünlendiğini test edin
- FrancaisLinkTest => Fransizca Linkin görüntünlendiğini test edin

# JUnit

## Assertions



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

## Check Box

Gerekli yapıyı olusturun ve aşağıdaki görevi tamamlayın

- a. Verilen web sayfasına gidin.

<https://testotomasyonu.com/form>

- b. Sirt Agrisi ve Carpinti checkbox'larini secin

- c. Sirt Agrisi ve Carpinti checkbox'larinin seçili olduğunu test edin

- d. Seker ve Epilepsi checkbox'larinin seçili olmadigini test edin

İsim	<input type="text" value="İsim"/>
Doğum Tarihi	<input type="text" value="Gün"/> <input type="text" value="Ay"/>
Cinsiyet	<input type="radio"/> Kadın <input type="radio"/> Erkek <input type="radio"/> Diğer
Son Bir Yılda Yaşadığınız Rahatsızlıklar	
<input type="checkbox"/> Bölgesel Öksürük <input type="checkbox"/> Nefes Darlığı <input type="checkbox"/> Göğüs Ağrısı <input type="checkbox"/> Çarpıntı <input type="checkbox"/> Sırt Ağrısı <input type="checkbox"/> İshal veya Kabızlık <input type="checkbox"/> Eklem Ağrısı	
Düzenle(Açıklayınız)	<input type="text" value="Rahatsızlıklar..."/>
Aşağıdaki Hastalıklardan Hangisine Sahipsiniz?	
<input type="checkbox"/> Kalp Hastalığı <input type="checkbox"/> Şeker <input type="checkbox"/> Böbrek Yetmezliği <input type="checkbox"/> Mide Ülseri <input type="checkbox"/> İşitme Kaybı <input type="checkbox"/> Sinir Sistemi Rahatsızlıkları <input type="checkbox"/> Epilepsi	
Düzenle(Açıklayınız)	<input type="text" value="Rahatsızlıklar..."/>

## Check Box

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  
e. Checkbox1 ve Checkbox2'nin seçili olduğunu test edin

**Specialty Food Type**

- GMO-Free
- Organic
- USDA Organic

**Calories Per Serving**

- 0 Calories
- Up to 40 Calories
- 40 to 100 Calories
- 100 to 200 Calories
- 200 to 300 Calories

**Fat Calories Per Serving**

- 40-100 Calories

**Nutrition Facts Per Serving**

- Fat Free (<0.5g)
- Low Fat (<3g)
- Free of Saturated Fat (<0.5g)
- Free of Trans Fat (0g)
- Cholesterol Free (<2mg)
- Sodium Free (<5mg)
- Low Sodium (<140mg)
- Carbohydrate Free (<0.5g)
- Sugar Free (<0.5g)
- Dietary Fiber (>10g)
- Protein (>10g)

**Food Diet Type**

- Vegan

**Availability**

- Include Out of Stock



# JUnit

## Radio Buttons

İsim

Doğum Tarihi

Cinsiyet

Kadın    Erkek    Diğer

Gerekli yapıyi olusturun ve aşağıdaki görevi tamamlayın.

- a. Verilen web sayfasına gidin.

<https://testotomasyonu.com/form>

- b. Cinsiyet Radio button elementlerini locate edin
- c. İki farklı test method'u oluşturup yazidan veya direkt buton'dan size uygun olanı seçin
- d. Seçtiniz radio button'un seçili, ötekilerin seçili olmadığını test edin

## Radio Buttons

Gerekli yapıyi olusturun ve aşağıdaki görevi tamamlayın.

- Verilen web sayfasına gidin.

<https://facebook.com>

- Cookies'i kabul edin
- Create an account buton'una basın
- Radio button elementlerini locate edin ve size uygun olanı seçin
- Sectiginiz radio button'un seçili, ötekilerin seçili olmadığını test edin

### Kaydol

Hızlı ve kolaydır.

Adın  Soyadın

Cep telefonu numarası veya e-posta

Yeni şifre

Doğum Tarihi  18 Ara 2023

Cinsiyet  Kadın  Erkek  Özel

Hizmetimizi kullanan kişiler senin iletişim bilgilerini Facebook'a yüklemesi olabilir. Daha fazla bilgi al.

Kaydol düğmesine tıklayarak, Koşullarımı kabul etmiş olursun. **Gizlilik İlkesimizde** verilerini nasıl topladığımız, kullandığımız ve paylaştığımız hakkında ve **Çerezler İlkesimizde** çerezleri ve benzer teknolojileri nasıl kullandığımız hakkında bilgi alabilirsin. Bizden SMS bildirimleri alabilirsin ve bunu istediğiniz zaman durdurabilirsin.

**Kaydol**

# JUnit

## TestBase Class

Her test method'u calisirken webDriver objesine ve bu obje icin ayarlara ihtiyac vardır. Bu islemleri her class icin yeniden yasmak yerine Java'daki **Inheritance** ozelligi kullanilabilir.

Bu islemlerin yapildigi **@Before...** ve **@After...** notasyonuna sahip method'lar olusturulan bir TestBase class'ina konulabilir.

Testlerin yapilacagi class'lar extends keyword ile TestBase class'ina child class yapip, oradaki setup ve teardown method'lari direk kullanilabilir.

TestBase class'inda setup ve teardown method'lari disinda tekrar tekrar yapacagimiz islemleri yapan hazir method'lar da konulabilir.

Olusturulan driver objesi sadece child class'larin kullanabilmesi icin protected yapilabilir

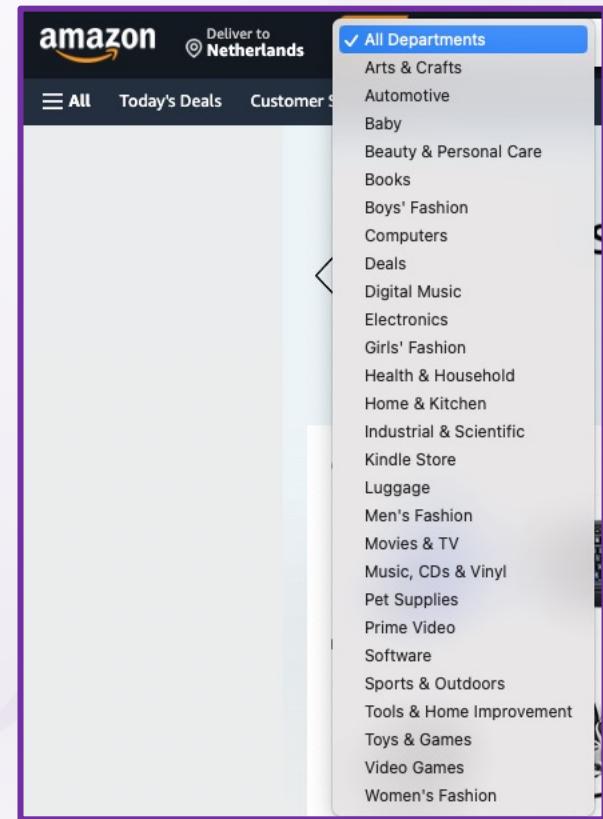
```
public class TestBaseEach {  
  
    public WebDriver driver;  
  
    @BeforeEach  
    public void setup(){  
        WebDriverManager.chromedriver().setup();  
        driver = new ChromeDriver();  
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));  
        driver.manage().window().maximize();  
    }  
  
    @AfterEach  
    public void teardown() { driver.quit(); }  
}
```

## Handle Dropdown

Dropdown(acilir menu) ozel bir HTML kodu ile olusturulur.

HTML sayfalarda farkli acilir menuler yapilabilir. Dropdown'i digerlerinden ayiran tag'inin **<select>** olmasi ve secilebilecek opsiyonların da **<option>** tag'i ile olusturulmasidir.

Selenium dropdown menu ile islem yapilabilmesi icin ozel bir Select class'i olusturmustur. Select class'indan olusturacak obje yardimi ile bu class'daki method'lar kullanilabilir.



## Handle Dropdown

Dropdown'daki opsiyonlardan birini secmek icin 3 islem yapilir.

1- Dropdown webelement'i locate edin

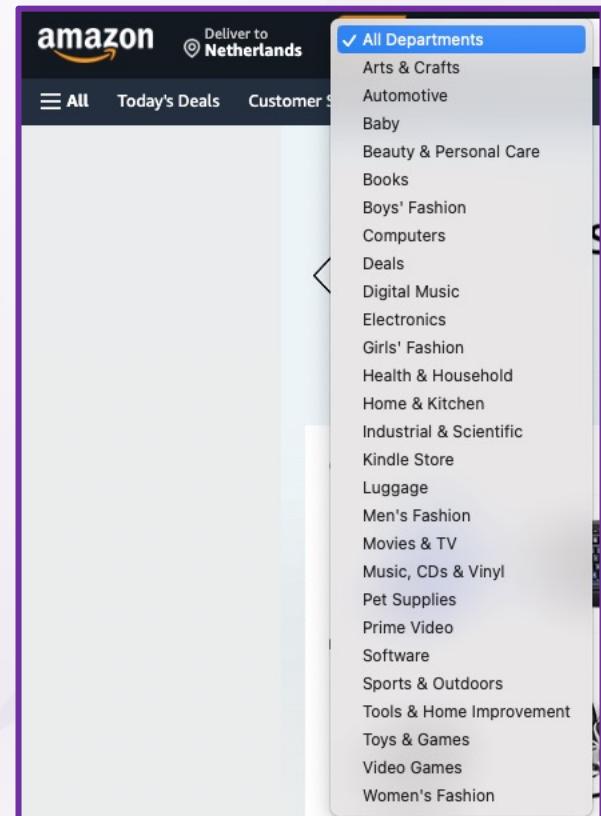
```
WebElement ddm= driver.findElement(By.id("searchDropdownBox"));
```

2- Select class'indan bir obje olusturun ve locate edilen dropdown elementi parametre olarak yazin

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

3- select objesi ile Select class'inda bulunan method'lardan uygun olani ile istediginiz option'i secin.

```
select.selectByVisibleText("Electronics");
```



## Select Class Method'ları

1- istenen option'i secme

```
select.selectByIndex(1);  
select.selectByValue("2");  
select.selectByVisibleText("Option 1");
```

2- Bir option secildikten sonra secilen option'i  
webelement olarak döndürme

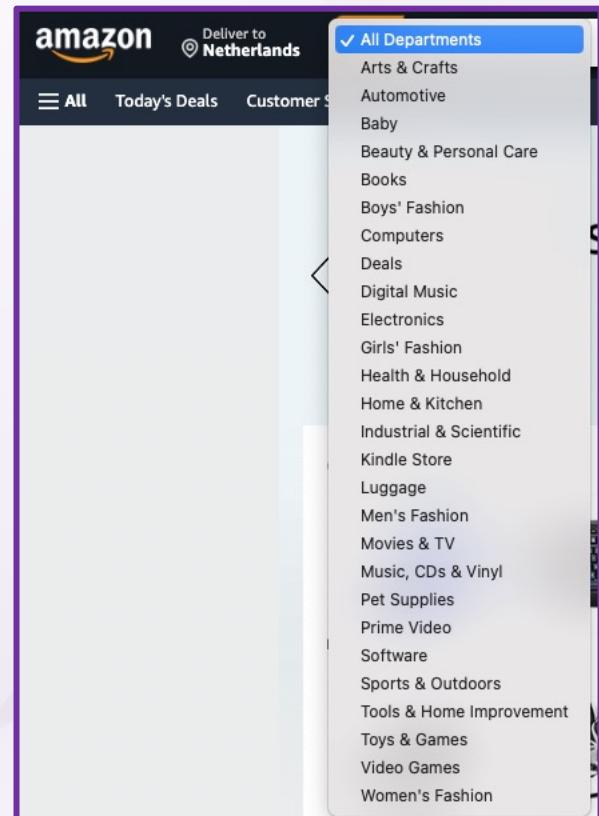
```
select.getFirstSelectedOption()
```

3- Bir option secildikten sonra secilen option'i text  
olarak döndürme

```
select.getFirstSelectedOption().getText()
```

4- Tum option'ları döndürüp kaydetme

```
List<WebElement> optionsList= select.getOptions();
```



## Handle Dropdown

Ön Kayıt Formu

İsim	Soyisim	
<input type="text" value="İsim"/>	<input type="text" value="Soyisim"/>	
Doğum Tarihi		
<input type="text" value="Gün"/>	<input type="text" value="Ay"/>	<input type="text" value="Yıl"/>

- <https://testotomasyonu.com/form> adresine gidin.
1. Dogum tarihi gun seçeneğinden index kullanarak 5'i secin
  2. Dogum tarihi ay seçeneğinden value kullanarak Nisan'i secin
  3. Dogum tarihi yil seçeneğinden visible text kullanarak 1990'i secin
  4. Secilen değerleri konsolda yazdirin
  5. Ay dropdown menüdeki tum değerleri(value) yazdirın
  6. Ay Dropdown menusunun boyutunun 13 olduğunu test edin

# JUnit

## Handle Dropdown Homework

### Dropdown List

✓ Please select an option

Option 1

Option 2

- <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 boyutunun 4 olduğunu test edin

# JUnit

## Handle Dropdown Homework

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, back tusuna basarak sayfaya donun
6. Online banking menusunden Pay Bills sayfasına gidin
7. “Purchase Foreign Currency” tusuna basin
8. “Currency” drop down menusunden Eurozone’u secin
9. “amount” kutusuna bir sayı girin
10. “US Dollars” in secilmədigini test edin
11. “Selected currency” butonunu secin
12. “Calculate Costs” butonuna basin sonra “purchase” butonuna basin
13. “Foreign currency cash was successfully purchased.” yazısının çıktılarını kontrol edin.



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-06

Javascript Alerts

Basic Authentication

Iframe



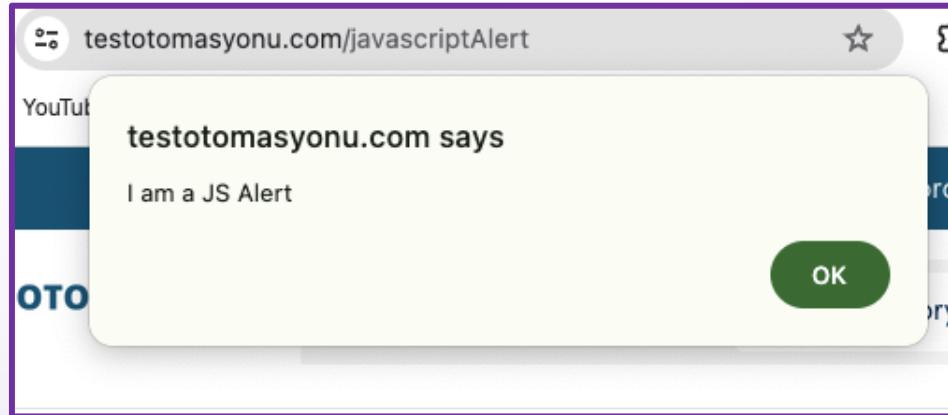
The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

@ /wisequarter

## JS Alerts



### Alert Nedir?

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

### 1- HTML Alerts

Bir alert çıktığında sağ click ile inspect yapabiliyorsak html alert'dir ve extra bir işlem gereklidir.

### 2- Js Alerts

Js alerts inspect yapılamaz, ekstra işlemi ihtiyaç vardır.

## JS Alerts

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

2. Confirmation Alert : Bu onay uyarısı bir tür işlem yapma izni ister. Alert onaylanyorsa 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://testotomasyonu.com/javascriptAlert>

## JS Alerts

1. `accept()` : Alert üzerindeki OK butonuna basmak için kullanılır.

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

2. `Dismiss()` : Alert üzerindeki Cancel butonuna basmak için kullanılır.

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

3. `getText()` : Alert üzerindeki yazıyı döndürür.

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

4. `sendKeys("istenen yazı")` : Alert üzerindeki text kutusuna istenilen metni yazdırır.

```
driver.switchTo().alert().sendKeys( keysToSend: "Deneme");
```

**Click for JS Alert**

**Click for JS Confirm**

**Click for JS Prompt**



# JUnit

3 test method'u olusturup asagidaki gorevi tamamlayin

## 1. Test

- <https://testotomasyonu.com/javascriptAlert> adresine gidin
- 1.alert'e tiklayin
- Alert'deki yazinin "I am a JS Alert" oldugunu test edin
- OK tusuna basip alert'i kapatin

## 2. Test

- <https://testotomasyonu.com/javascriptAlert> adresine gidin
- 2.alert'e tiklayalim
- Cancel'a basip, cikan sonuc yazisinin "You clicked: Cancel" oldugunu test edin

## 3. Test

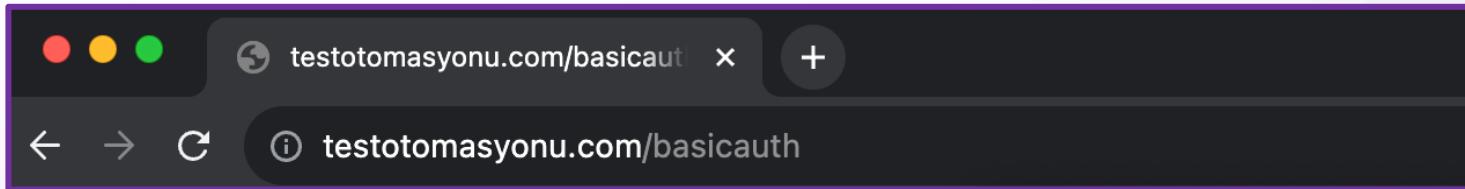
- <https://testotomasyonu.com/javascriptAlert> adresine gidin
- 3.alert'e tiklayalim
- Cikan prompt ekranina "Abdullah" yazdiralim
- OK tusuna basarak alert'i kapatyalim
- Cikan sonuc yazisinin Abdullah icerdigini test edelim

Click for JS Alert

Click for JS Confirm

Click for JS Prompt

## Basic Authentication



### Authentication Nedir?

Kısaca, herhangi bir internet kullanıcısının, uygulamanın ya da programın, söz konusu sisteme dahil olup olamayacağını belirleyen formu ifade eder.

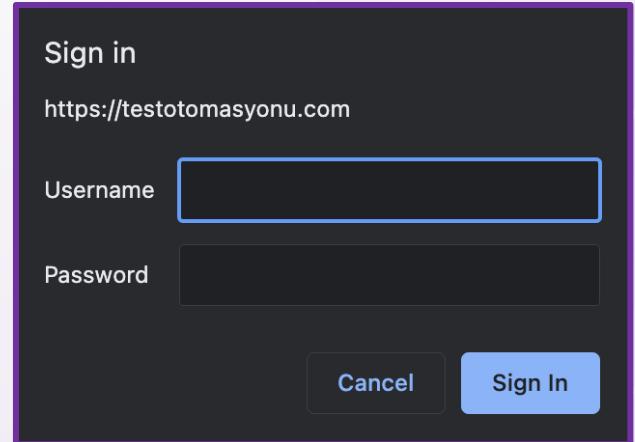
Uygulama ana sayfalarındaki kullanıcı adı ve password istemek de bir authentication'dır.

End user'lar için tasarılanmayan uygulamalarda(Ornegin API sorgularında) bu authentication HTML komutları ile de yapılabilir.

Bu authentication'i yapabilmek için uygulamanın kullanıcılarına authentication'i nasıl yapacağına dair bilgilendirme yapmış olması gereklidir.

Ornegin yandaki uygulama için authentication aşağıdaki gibi yapılabilir.

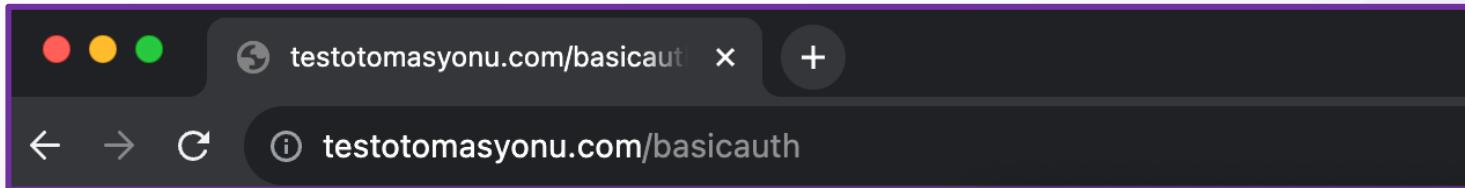
<https://username:password@URL>





# JUnit

## Basic Authentication



- 1- Bir class olusturun : BasicAuthentication
- 2- <https://testotomasyonu.com/basicauth> sayfasina gidin
- 3- asagidaki yontem ve test datalarini kullanarak authentication'i yapin

Html komutu : https://username:password@URL

Username : membername  
password : sunflower

- 4- Basarili sekilde sayfaya girildigini dogrulayin

`https://membername:sunflower@testotomasyonu.com/basicauth`

Sign in

https://testotomasyonu.com

Username

Password

# JUnit

## Handle IFrame

HTML kodlarda kullanılan `<iframe>` tag'i bir HTML sayfasının içerisinde başka bir HTML sayfası gömmek(embed) için kullanılır.

Iframe'ler genellikle videoları, haritaları ve diğer medyaları bir web sayfasına gömmek için kullanılır. Ancak sadece bunlarla sınırlı degildir, her turlu HTML sayfası `<iframe>` tag'i ile kullanılabilir.

`<iframe>` tag'i içerisinde header ve body bulunur. Bir HTML sayfasında iframe varsa, HTML kodlar içerisinde birden fazla header veya body olacaktır.

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



# JUnit

## Handle IFrame

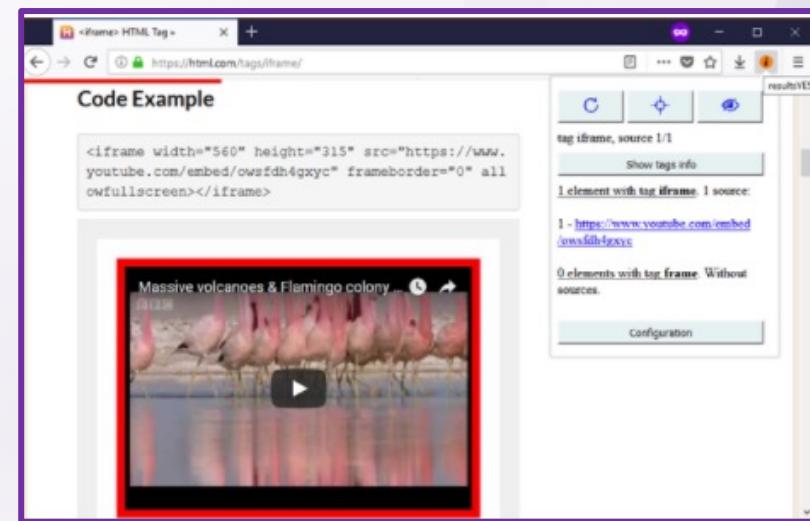
Bir websayfasında locate işlemi doğru yapıldığı halde istenen webelement'e ulaşamıyorsa, aranan elementin bir iframe içinde olup olmadığı kontrol edilmelidir.

Bir iframe içerisindeki webelementi kullanabilmek için driver'i iframe'e switch yapmak gereklidir.

Webdriver'i istenen iframe'e switch yapabilmek için iframe'i driver'a tanıtmak gereklidir. Bu tanıtma 3 farklı yolla yapılabilir.

1) Iframe'i webelement olarak locate ederek

```
driver.switchTo().frame(iframeElementi);
```



2) Iframe'in id veya name attribute value'su kullanılarak

3) Iframe'in index'i biliniyorsa, index kullanılarak

# JUnit

## Handle IFrame

Iframe icerisindeki webelement'e ulasmak icin iframe'e switch edildigi gibi, iframe icerisine gecis yaptiktan sonra iframe disindaki bir elemente erisebilmek icin de yeniden iframe'den anasayfaya gecis yapmak gereklidir.

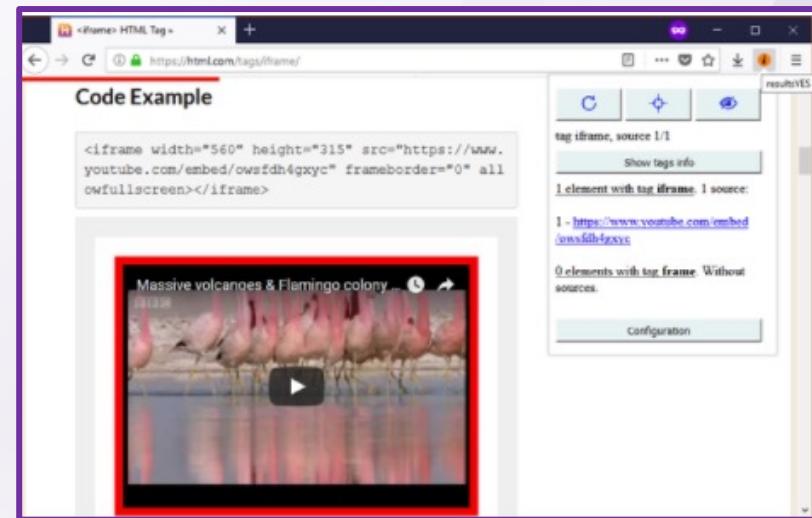
Iframe icerisindeyken oradan cikmak icin 2 yontem kullanilabilir.

1- Direk anasayfaya gecis yapmak icin

```
driver.switchTo().defaultContent();
```

2- Icice birden fazla iframe varsa, bir ust iframe'e cikmak icin

```
driver.switchTo().parentFrame();
```



# JUnit

## Handle IFrame

Sale Up To 50%  
Here are some products.

**Electronics Products**



DELL Core I3 11th Gen      Bat Rockerz 510 Bluetooth Heads

**Fashion**



Men Slim Fit Casual Shirt      Men Slim Fit Solid Casual Shirt

- 1- <https://testotomasyonu.com/discount> adresine gidin
- 2- Electronics Products yazisinin gorunur oldugunu test edin
- 3- Dell bilgisayar urun isminin 'DELL Core I3 11th Gen' oldugunu test edin
- 4- Sagdaki bolumde gorunen urunler arasinda 'Men Slim Fit' içeren en az 1 urun oldugunu test edin
- 5- 'Fashion' yazisinin gorunur oldugunu test edin
- 6- 'Here are some products' yazisinin gorunur oldugunu test edin
- 7- Sayfayı kapatın

# JUnit

## Handle IFrame

### An iFrame containing the TinyMCE WYSIWYG Editor



The screenshot shows a web-based WYSIWYG editor. At the top, there's a menu bar with 'File', 'Edit', 'View', and 'Format'. Below the menu is a toolbar with icons for back, forward, search, bold (B), italic (I), and alignment (left, center, right, full). The main content area contains the placeholder text 'Your content goes here.' In the bottom left corner of the content area, there's a small 'P' icon. In the bottom right corner, it says 'POWERED BY TINY' with a tinyMCE logo.

- 1 ) <https://the-internet.herokuapp.com/iframe> adresine gidin.
- 2 ) Bir metod olusturun: iframeTest
  - "An IFrame containing...." textinin erisilebilir oldugunu test edin ve konsolda yazdirin.
  - TextBox'a "Merhaba Dunya!" yazin.
  - TextBox'in altinda bulunan "Elemental Selenium" linkinin gorunur oldugunu dogrulayin ve yazisini konsolda yazdirin.



# JUnit

## Handle IFrame

The screenshot shows a demo page for testing on the Guru99 website. The main content area displays a video player for an 'AUTOMATION PROCESS' tutorial. Below the video, a horizontal bar says 'All provided FREE!!!'. To the right of the video player, there is an iframe containing a slide titled 'FOLLOWING STEPS ARE FOLLOWED IN AN AUTOMATION PROCESS' with five colored arrows pointing right: red (Test tool selection), green (Define scope of automation), purple (Planning, Design and development), blue (Test Execution), and orange (Maintenance). A 'Watch on YouTube' button is visible at the bottom left of the video player.

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

# JUnit

## Handle Windows

### Opening a new window

[Click Here](#)

Powered by [Elemental Selenium](#)

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

Bir HTML sayfasında test yaparken, bazen isteyerek veya bir link tiklayarak yeni bir tab veya windows açılabilir.

Test sırasında test için yapılan tüm eylemleri webdriver objesi yaptığı için, yeni açılan sayfada işlem yapılabilmesi için webdriver'in yeni sayfaya geçis yapması gereklidir.

Selenium4 ile yeni gelen bir özellik olarak, test sırasında yeni bir tab veya window açılabilir.

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

Bu method kullanıldığında yeni sayfa/tab driver ile açıldıgından driver otomatik olarak yeni sayfaya geçis yapar.



# JUnit

## Handle Windows

- Yeni bir class olusturun: WindowHandle
- testotomasyonu anasayfa adresine gidin.
- Sayfa'nin window handle degerini String bir degiskene atayin
- Sayfa title'nin “Otomasyon” icerdigini test edin
- Yeni bir tab olusturup, acilan tab'da wisequarter.com adresine gidin
- Sayfa title'nin “Wise Quarter” icerdigini test edin
- Yeni bir window olusturup, acilan sayfada walmart.com adresine gidin
- Sayfa title'nin “Walmart” icerdigini test edin
- Ilk acilan window'a donun ve testotomasyonu sayfasina dondugunuzu test edin



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-07

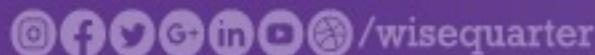
Switching Windows  
Actions Class  
Faker Class



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)





# Onceki Dersten Aklimizda Kalanlar

1- driver.switchTo( ).... driver'i farkli ortamlara gecirmek icin kullanilir

2- switchTo( ) ile kullanabilecegimiz ilk WebElement turu JavaScript alerts'dir.

JavaScript alerts calistiginda baska WebElement'lere erisimi ve driver'in calismasini kilitler. Bu durumda switchTo( ) ile alert'e gecis yapip var olan 4 method'dan uygun olani kullanabiliriz

JavaScript alerts ile islem yapacagimiz her adimda switchTo( ) kullanmaliyiz.

3- switchTo( ) ile switch yapip kullanabilecegimiz ikinci WebElement iFrame'dir.

iFrame'ler HTML sayfa icinde farkli bir sayfa olusturmayı saglar.

iFrame icindeki bir webElement'e ulasmak icin once iFrame'e switch yapilmalidir.

Js alert'ten farkli olarak iFrame'e bir kere gecis yapildi mi, oradan cikincaya kadar driver iFrame'de kalir.

4- switchTo( ) ile Window'lardir.

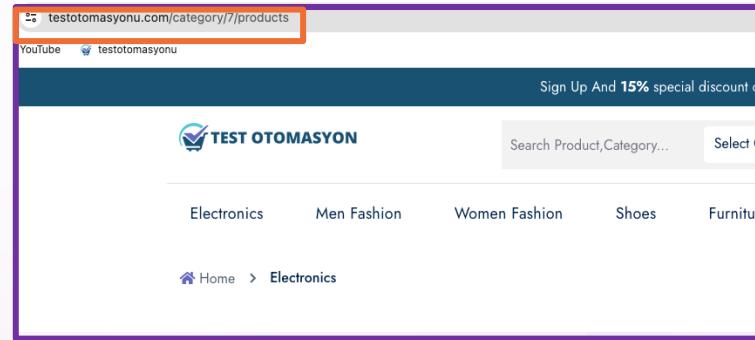
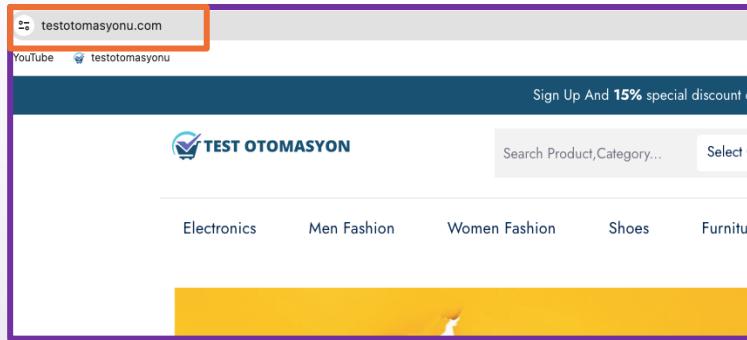
Gecis yapacagimiz window'un WindowHandle degerini kullanarak gecis yapabiliriz.

Eger window'u olustururken switchTo( ).newWindow( ) kullanirsak driver otomatik olarak yeni window'a gecer,

ancak kullandigimiz bir link KONTROLUMUZ DISINDA yeni bir window aciyorsa once acilan window'un WindowHandleDegerini bulmaliyiz.

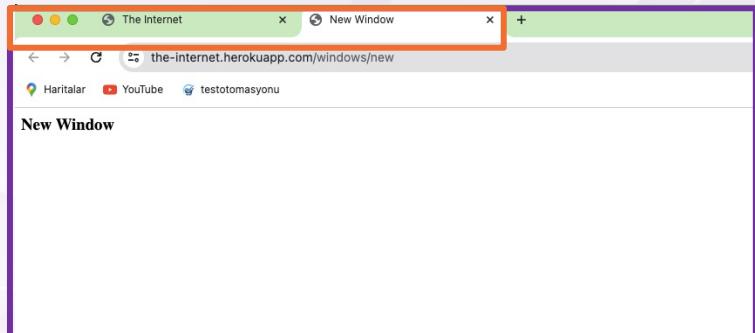
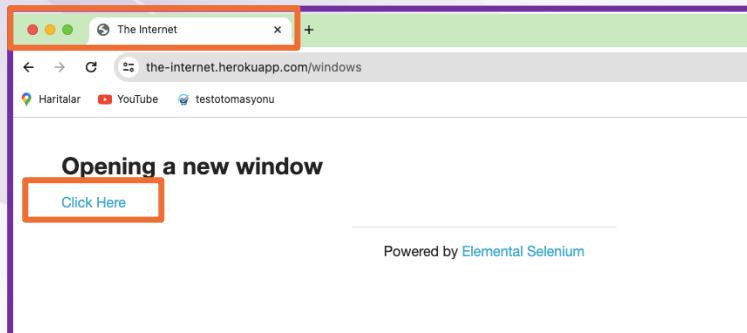
# JUnit

## Handle Windows



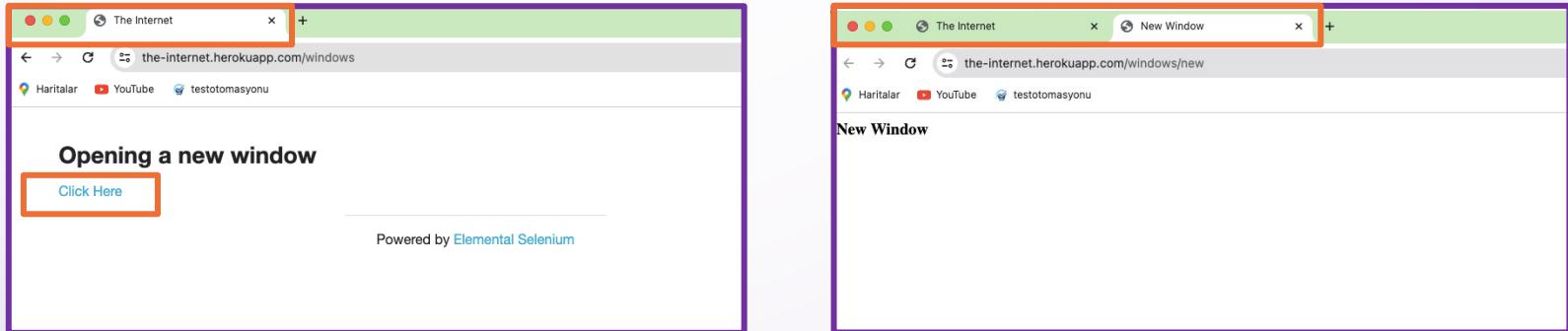
Verilen görevi yaparken tıkladığımız bir linkin, aynı pencerede kalarak bizi başka bir url'e getirmesi beklenir. Orneğin Electronics linkini tıkladığımızda yeni bir tab veya window açılmaz, aynı sayfada electronics sayfasına yönlendiriliriz.

Fakat, bazı sayfalarda bir linki tıkladığımızda, aynı sayfada acmak yerine yeni bir TAB veya yeni bir window açabilir.



# JUnit

## Handle Windows



Kontrolümüz disinda bir tab/window acildiginda selenium webdriver yeni tab/window'a gecmez, ilk sayfada kalir.

Selenium webdriver ile bir tab/window'a gecis yapabilmek icin o tab/window'un windowHandle degerine ihtiyac vardir.

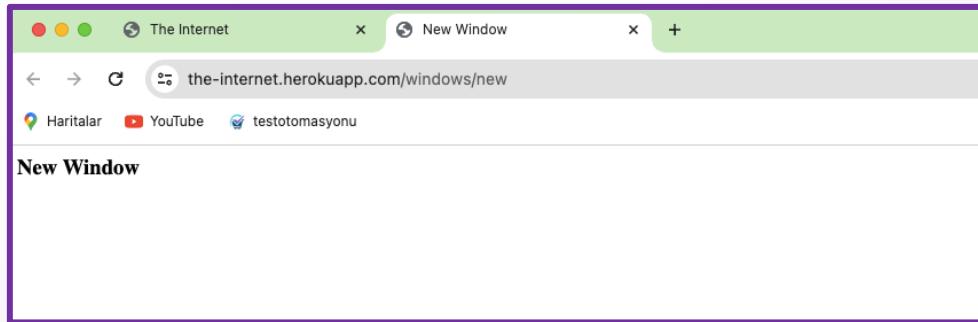
```
driver.switchTo().window(ilkSayfaHandleDegeri);
```

Bir window'un window handle degeri getWindowHandle( ) ile alinabilir, ancak bu method'u calistirabilmek icin once o sayfada olmak gerekir.

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

# JUnit

## Handle Windows



Ikinci sayfa kontrolümüz disinda aciliyorsa, o sayfaya gecmek icin window handle degerini driver ilk sayfada iken bulmamiz gereklidir. Bunun icin hazir bir method yoktur ama yazacagimiz kod ile 3 adimla bu islemi yapmamiz mumkundur.

1- Ilk sayfada iken, o sayfanin window handle degerini kaydedin.

```
String ilkSayfaHandleDegeri= driver.getWindowHandle();
```

2- Ikinci window acildiktan sonra, acik tum window'larin handle degerlerini bir set olarak kaydedin.

```
Set<String> windowHandlesSeti= driver.getWindowHandles();
```

3- For-each loop ile set'deki windowHandle degerlerini kontrol edip, ilk sayfanin window handle degerine esit olmayani, ikinci sayfanin window handle degeri olarak kaydedin.

# JUnit

## Handle Windows

Sale Up To 50%  
Here are some products.

**Electronics Products**



DELL Core I3 11th Gen



Bat Rockerz 510 Bluetooth Heads

**Fashion**



Men Slim Fit Casual Shirt



Men Slim Fit Solid Casual Shirt

- 1- <https://testotomasyonu.com/discount> adresine gidin
- 2- Electronics Products yazisinin gorunur oldugunu test edin
- 3- Dell bilgisayar urun isminin 'DELL Core I3 11th Gen' oldugunu test edin
- 4- Dell bilgisayar'a tiklayip acilan sayfada urun fiyatinin \$399.00 oldugunu test edin.
- 5- Ilk sayfaya donun ve Fashion yazisinin gorunur oldugunu test edin
- 6- Sayfayi kapatın

Add/Remove Elements

Add

Please click for Electronics Products

- <https://testotomasyonu.com/addremove/> adresine gidin.
- Sayfadaki textin “Add/Remove Elements” olduğunu doğrulayın.
- Sayfa başlığının(title) “Test Otomasyonu” olduğunu doğrulayın.
- ‘Please click for Electronics Products’ linkine tıklayın.
- Electronics sayfasının açılmasını test edin
- Bulunan ürün sayısının 16 olduğunu test edin
- İlk actığınız addremove sayfasının olduğu window'a donun
- Url'in addremove içerdigini test edin



# JUnit

## Handle Windows

### Opening a new window

[Click Here](#)

Powered by [Elemental Selenium](#)

- <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 test edin
- Sayfadaki textin “New Window” olduğunu doğrulayın.
- Bir önceki pencereye geri döndükten sonra sayfa başlığının “The Internet” olduğunu test edin

Actions class'i kullanılarak mouse ve klavye ile yapabilecek tüm islevler gerçekleştirilebilir.

**Actions Class** birçok kullanışlı mouse ve klavye method'una sahiptir.

- Çift tıklama (double click),
  - sürükleme ve bırakma(drag and drop)
  - mouse'u objeye götürme  
(move to element)
- gibi karmaşık mouse eylemleri



veya Keyboard ile yapabileceğimiz pageUp, pageDown, shift, arrowDown gibi işlemleri Actions class'ından object ureterek driver ile yapabiliriz.

# JUnit

## Actions Class

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

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

2. Adım: Üzerinde çalışmak istediğiniz WebElement locate edilir.

```
WebElement accountListElementi= driver.findElement(By.xpath( xpathExpression: "xpath"));
```

3.Adım : Ardından bu webelement üzerinde action gerçekleştirilir.

Örneğin Mouse'u istenen webelement'in üzerine getirmek için

```
actions.moveToElement(accountListElementi).perform();
```

**NOT :** Action Class'ini her kullanmak istedigimizde yeniden obje olusturmamız gerekmeyez.

**NOT 2 :** action objesi'ni bir kere olusturunca, istediginiz kadar action. ile baslayan komut yazar ve calismasi icin sonuna perform( ) yazariz.

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

# JUnit

## Mouse Base Actions

**doubleClick ( )**: WebElement'e çift tıklama yapar

**clickAndHold ( )**: WebElement üzerinde click yapılı olarak bizden komut bekler.

**dragAndDrop ( )**: WebElement'i bir noktadan diğerine sürüklər ve bırakır

**moveToElement ( )**: Mouse'u istedigimiz WebElement'in üzerinde tutar

**contextClick ( )**: Mouse ile istedigimiz WebElement'e sağ tıklama yapar.



# JUnit

## Actions Class

### Context Menu

Custom additions in the context menu are additional options that appear when you right-click.

Right click on the image below. Clicking it triggers a JavaScript alert.



- 1- <https://testotomasyonu.com/click> sitesine gidin
- 2- “DGI Drones” üzerinde sağ click yapın
- 3- Alert’ta çıkan yazının “Tebrikler!... Sağ click yaptınız.” olduğunu test edin.
- 4- Tamam diyerek alert’ı kapatalım

# JUnit

## Actions Class

### Context Menu

Context menu items are custom additions that appear in the right-click menu.

Right-click in the box below to see one called 'the-internet'. When you click it, it will trigger a JavaScript alert.

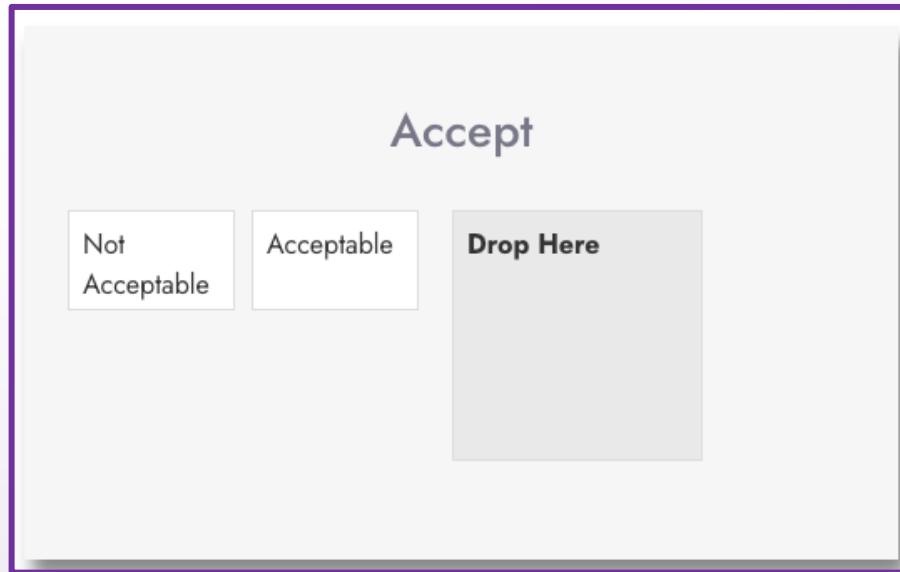


Powered by [Elemental Selenium](#)

- 1- Yeni bir class olusturalim: MouseActions1
- 2- [https://the-internet.herokuapp.com/context\\_menu](https://the-internet.herokuapp.com/context_menu) sitesine gidin
- 3- Cizili alan uzerinde sag click yapin
- 4- Alert'te cikan yazinin "You selected a context menu" oldugunu test edin.
- 5- Tamam diyerek alert'i kapatalim
- 6- Elemental Selenium linkine tiklayalim
- 7- Acilan sayfada h1 taginda "Elemental Selenium" yazdigini test edelim

# JUnit

## Actions Class



- 1- <https://testotomasyonu.com/droppable> adresine gidelim
- 2- Accept bolumunde “Acceptable” butonunu tutup “Drop Here” kutusunun ustune birakalim
- 3- “Drop here” yazisi yerine “Dropped!” oldugunu test edin
- 4- Sayfayı yenileyin
- 5- “Not Acceptable” butonunu tutup “Drop Here” kutusunun ustune birakalim
- 6- “Drop Here” yazisinin degismedigini test edin



# JUnit

## Actions Class

Simple    Accept    Prevent Propagation    Revert Draggable

Drag me

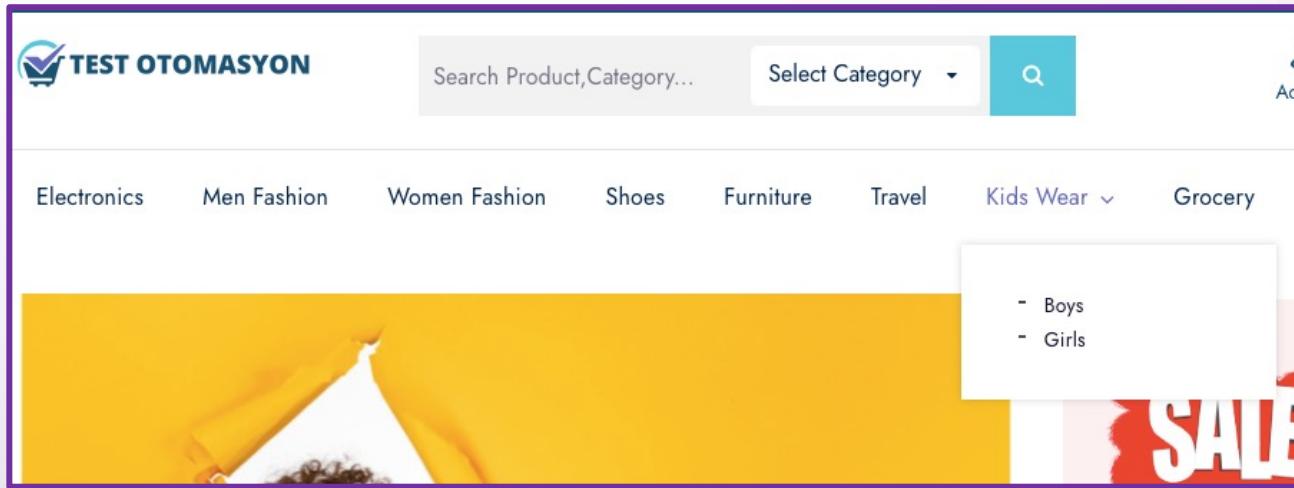
Drop here

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



# JUnit

## Actions Class



- 1- <https://www.testotomasyonu.com/> adresine gidin
- 2- “Kids Wear” menusunun acilmasi icin mouse'u bu menunun ustune getirin
- 3- “Boys” linkine basin
- 4- Acilan sayfadaki ilk urunu tiklayin
- 4- Acilan sayfada urun isminin “Boys Shirt White Color” oldugunu test edin

# JUnit

## Keyboard Base Actions

Action Class'indaki hazır method'lar ile klavyedeki tuslar kontrol edilebilir.

Klavyede çok fazla tuş vardır ama tüm tuşlar 3 temel işlev ile kontrol edilebilir.

1 ) **sendKeys ( )**: Öğeye bir dizi anahtar gönderir

2 ) **keyDown ( )**: Klavyede tuşa basma işlemi gerçekleştirir

3 ) **keyUp ( )** : Klavyede tuşu serbest bırakma işlemi gerçekleştirir



# JUnit

## Keyboard Base Actions



- 1- <https://www.testotomasyonu.com> sayfasina gidelim
- 2- Arama kutusuna actions method'larini kullanarak "DELL Core I3" yazdirin ve Enter'a basarak arama yaptirin
- 3- Bulunan urun isminde "DELL Core I3" bulundugunu test edin

# JUnit

## Keyboard Base Actions

Register Now

First Name \*

Last Name \*

Email Address \*

Password \*

Confirm Password \*

Sign Up

- 1- <https://www.testotomasyonu.com> adresine gidelim
- 2- Account linkine tiklayin
- 3- Sign Up linkine basalim
- 4- Ad, soyad, mail ve sifre kutularina deger yazalim ve Sign Up butonuna basalim
- 5- Kaydin olusturuldugunu test edin



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-08

Actions Class

Faker Class

File Testleri

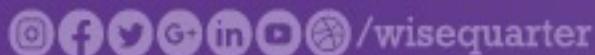
Selenium Waits



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# JUnit

## Faker Kutuphanesi

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

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

### Kaydol

Hızlı ve kolaydır.

|Adın

Soyadın

Cep telefonu numarası veya e-posta

Yeni şifre

Doğum Tarihi ?

25

Ara

2023

Cinsiyet ?

Kadın

Erkek

Özel

Hizmetimizi kullanan kişiler senin iletişim bilgilerini Facebook'a yüklemiş olabilir. [Daha fazla bilgi al.](#)

Kaydol düğmesine tıklayarak, **Koşullarımızı kabul etmiş** olursun. **Gizlilik İlkemizde** verilerini nasıl topladığımız, kullandığımız ve paylaştığımız hakkında ve **Çerezler ilkemizde** çerezleri ve benzer teknolojileri nasıl kullandığımız hakkında bilgi alabilirsin. Bizden SMS bildirimleri alabilirsin ve bunu istediğiniz zaman durdurabilirsin.

**Kaydol**

## Keyboard Base Actions

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

**Kaydol**

Hızlı ve kolaydır.

Adın  Soyadın

Cep telefonu numarası veya e-posta

Yeni şifre

Doğum Tarihi  25  Ara  2023

Cinsiyet  Kadın  Erkek  Özel

Hizmetimizi kullanan kişiler senin iletişim bilgilerini Facebook'a yüklemiş olabilir. [Daha fazla bilgi al.](#)

Kaydol düğmesine tıklayarak, [Koşullarımızı kabul etmiş olursun.](#) [Gizlilik İlkemizde](#) verilerini nasıl topladığımız, kullandığımız ve paylaştığımız hakkında ve [Çerezler İlkemizde](#) çerezleri ve benzer teknolojileri nasıl kullandığımız hakkında bilgi alabilirsin. Bizden SMS bildirimleri alabilirsin ve bunu istediğiniz zaman durdurabilirsin.

**Kaydol**

## Keyboard Actions Homework



- 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



## Keyboard Actions Homework



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

# JUnit

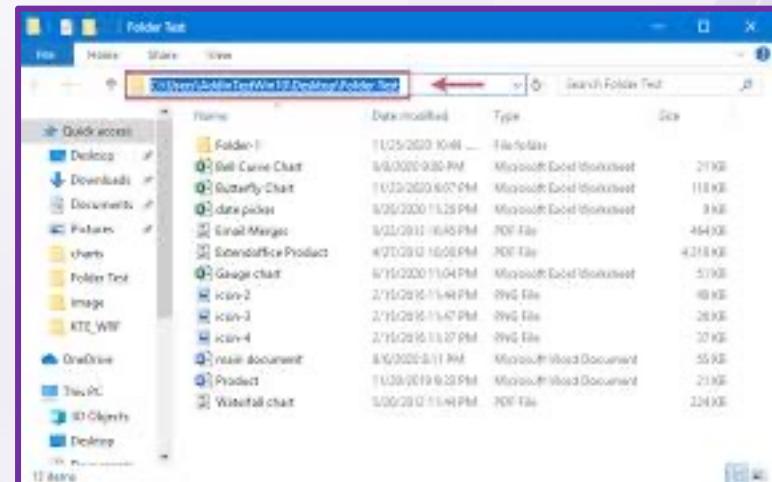
## File Exist

Selenium webDriver objesi üzerinden calisir ve local bilgisayard işlem yapamaz. Local bilgisayardaki dosyalara erişmek veya test etmek için JAVA kullanılabilir.

Local bilgisayarda bir dosyaya ulaşmak ve olup olmadığını(exist) kontrol etmek için dosya yoluna ihtiyaç vardır.

Her bilgisayarın ismi ve kullanıcı ismi farklı olacağınından, bir bilgisayarda yazılan dosya yolu başka bilgisayarda çalışmaz.

Java'daki getProperty( ) method'u ile her bilgisayarda farklı olan kısım, testin çalıştığı bilgisayardan alınabilir. Bu temel path'den sonrası tüm bilgisayarlarda aynı olacağı için kod dinamik olur.



## File Exist

getProperty( ) method'u iki farkli parametre ile calisabilir.

- 1- System.getProperty ( "user.dir"); icinde bulunulan klasörün yolunu (Path) verir
- 2- System.getProperty ( "user.home"); bilgisayarimizda bulunan user klasörünü verir

```
String dosyaYolu= System.getProperty("user.home")+"Desktop/FileTesti/deneme.txt";
```

Seklinde oluşturulup kaydedilen dosya yolu dinamik olduğu için kodların calisacagi tum bilgisayarlarda sorunsuz kullanılabilir.

String olarak dosya yolu oluşturulan bir dosyanın bilgisayarda var olup olmadığını test etmek için Files class'ından exist( ) method'u kullanılır.

```
Files.exists(Paths.get(dosyaYolu))
```

Bu kod bize boolean bir sonuç döndürür. Bu sonuc kullanılarak test gerçekleştirilebilir.

## File Exist

1. <https://the-internet.herokuapp.com/download> adresine gidelim.
2. logo.png dosyasını indirelim
3. Dosyanın başarıyla indirilip indirilmediğini test edelim

```
@Test
public void downloadTesti(){

    driver.get("https://the-internet.herokuapp.com/download");

    driver.findElement(By.xpath( xpathExpression: "//*[text()='logo.png']")).click();

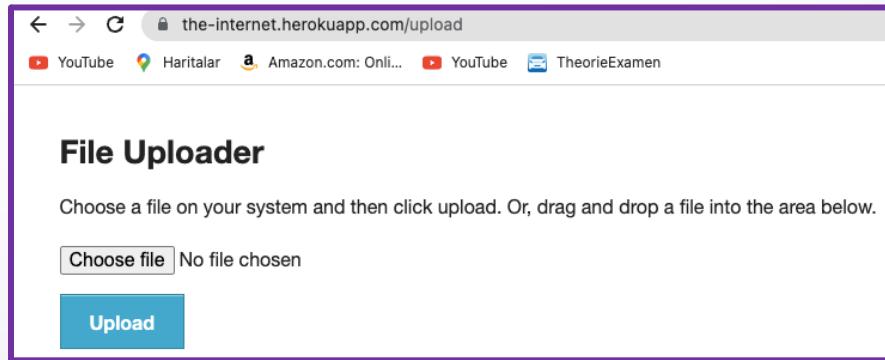
    bekle( beklenenSaniye: 5);

    String dosyaYolu= System.getProperty("user.home")+ "/Downloads/logo.png";

    Assert.assertTrue(Files.exists(Paths.get(dosyaYolu)));
}
```

# JUnit

## File Upload



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

Local bilgisayardaki bir dosyayı bir web uygulamasına yüklemek için, bilgisayarda dosyalar arasında gezinmek ve dosyayı tıklayarak seçmek gerekebilir.

Ancak selenium ile local dosyalara ulaşamayacağı için yine java'daki dosya kullanma yöntemleri kullanılabilir. Dosyayı yüklemek için

1- Dosya yolunu oluşturup kaydedin

2- Choose file butonunu locate edip dosya yolunu bu element'e gonderin

3- Upload butonunu locate edip tıklayın

# JUnit

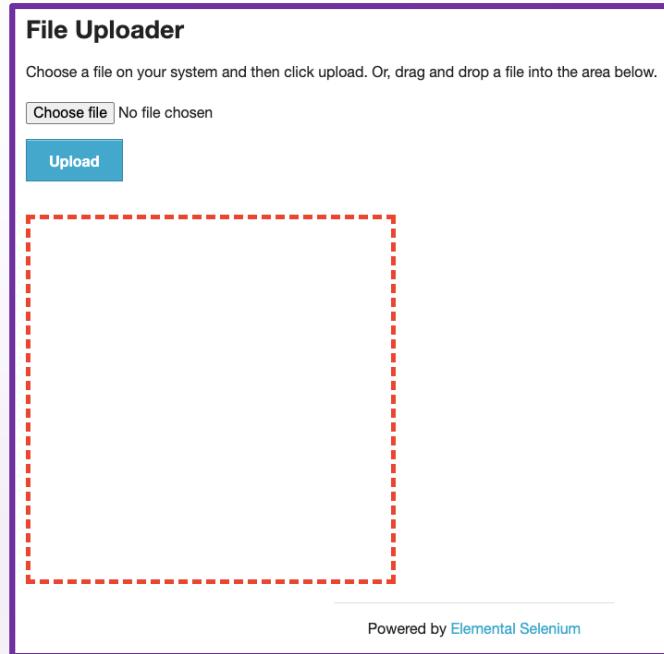
## File Upload

**File Uploader**

Choose a file on your system and then click upload. Or, drag and drop a file into the area below.

No file chosen

Powered by [Elemental Selenium](#)



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

# JUnit

## Synchronization- Waits

Synchronization(Senkronizasyon), UI (kullanıcı arayüzü) üzerinde planlanan bir testin sorunsuz calismasi icin mutlaka dikkate alınması gereken bir konudur.

Bir sayfanın uygulama sunucusu veya web sunucusu çok yavaşsa veya internet ağı çok yavaşsa, web sayfasındaki öğelerin (webelement) yüklenmesi beklenenden uzun sürebilir.

Bu durumda, komut dosyanız (test script) öğeyi bulmaya çalıştığında, öğeler yüklenmez.

Bu yüzden test komut dosyası(test script) öğeyi bulamaz ve başarısız olur ve kod NoSuchElementException verip, calismayı durdurur.



# JUnit

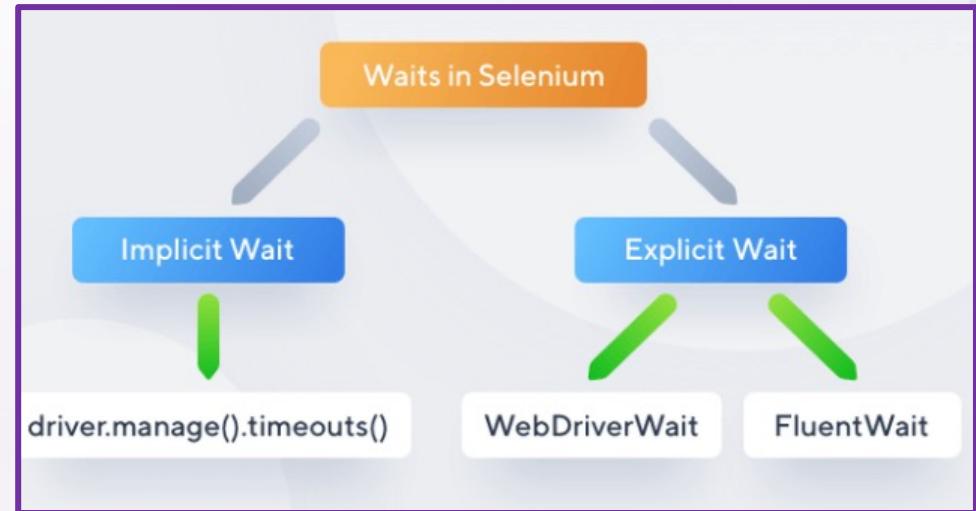
## Synchronization- Waits

Driver ile cihaz veya internet arasında yasanın senkronizasyon sorunlarını çözmek için driver'i belirli yöntemler ile bekletmek(wait) gereklidir.

### 1 ) Java tabanlı wait

**Thread.sleep** : Javadan gelir ve kodları yazılan süre kadar bekletir. Sure dolduktan sonra alt satırdan işlem devam eder

### 2 ) Selenium tabanlı wait'ler



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

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

# JUnit

## Implicitly Wait

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

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

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

Implicitly wait TestBase class'da kullanılabilir.

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



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

## Implicitly Wait

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

Webelement 10 saniyeden kısa sürede yüklenirse driver bulur ve devam eder.

Örneğin, Webelement 3 saniye içinde yüklenirse, driver sadece 3 saniye bekleyecektir ve bir sonraki satırı geçecektir.

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



# JUnit

## Explicitly Wait

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

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

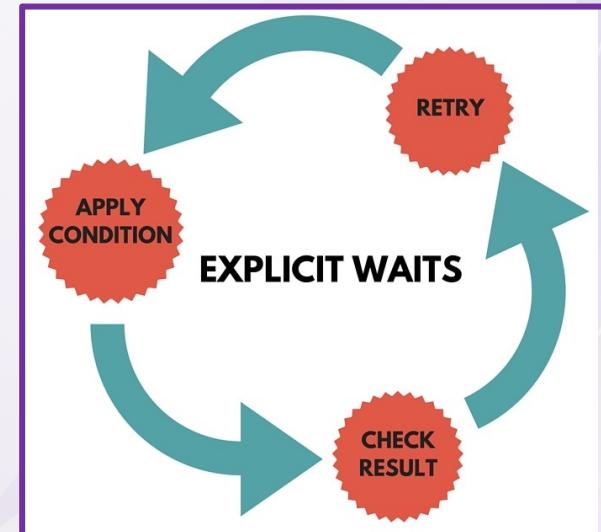
Nadiren karşılaşılan ve daha uzun bekleme süresi gerektiren işlem uygulanan webelementler için explicitly wait kullanılır.

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

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

Explicit wait'de hem webelement, hem de beklenen condition kullanılır. Cunku olmayan bir webelement'in locate edilmesi mümkün olamayabilir.

```
WebElement itsBackElementi= wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("message")));
```



## 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( )

# JUnit

## Explicit Wait

**Dynamic Controls**

This example demonstrates when elements (e.g., checkbox, input field, etc.) are changed asynchronously.

Remove/add

A checkbox

**Remove**

---

Enable/disable

**Enable**

Powered by Elemental Selenium

Iki tane metod olusturun : implicitWaitTest , explicitWaitTest

Iki metod icin de asagidaki adimlari test edin.

1. [https://the-internet.herokuapp.com/dynamic\\_controls](https://the-internet.herokuapp.com/dynamic_controls) adresine gidin.
2. Textbox'in etkin olmadigini(enabled) doğrulayın
3. Enable butonuna tıklayın ve textbox etkin oluncaya kadar bekleyin
4. Textbox'in etkin oldugunu(enabled) doğrulayın.
5. "It's enabled!" mesajinin goruntulendigini doğrulayın.

# JUnit

## Explicit Wait

**Dynamic Controls**

This example demonstrates when elements (e.g., checkbox, input field, etc.) are changed asynchronously.

Remove/add

A checkbox

**Remove**

---

Enable/disable

**Enable**

Powered by Elemental Selenium

1. [https://the-internet.herokuapp.com/dynamic\\_controls](https://the-internet.herokuapp.com/dynamic_controls) adresine gidin.
2. Remove butonuna basin.
3. “It’s gone!” mesajinin goruntulendigini doğrulayın.
4. Add buttonuna basin
5. It’s back mesajinin gorundugunu test edin

# JUnit

## Actions Class Homework



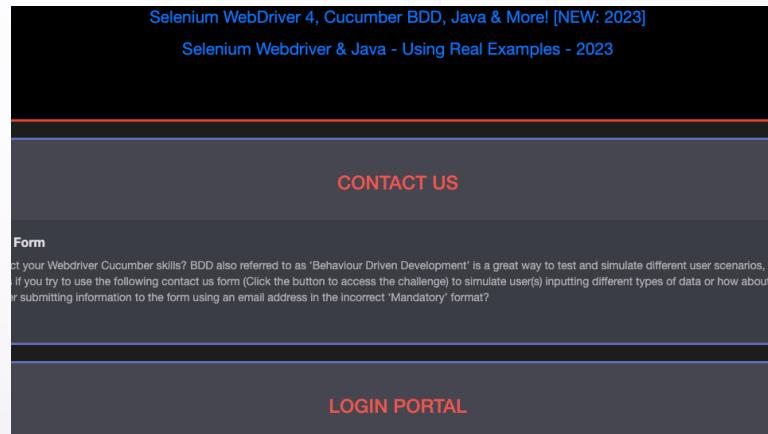
1. "http://webdriveruniversity.com/Actions" sayfasina gidin
2. "Hover over Me First" kutusunun ustune gelin
3. "Link 1" e tiklayin
4. Popup mesajini yazdirin
5. Popup'i tamam diyerek kapatin
6. "Click and hold" kutusuna basili tutun
7. "Click and hold" kutusunda cikan yaziyi yazdirin
8. "Double click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

## Iframe Homework



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 Homework



- 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



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-09

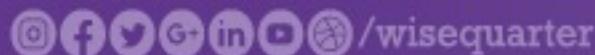
Cookies  
Web Tables



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# JUnit

## Cookies

Çerezler, belirli kullanıcıları tanımlamak ve bu kullanıcıların göz atma deneyimini iyileştirmek için kullanıcının bilgisayarı ile web sunucusu arasında takas edilen, kullanıcı adı ve parola gibi küçük veri parçalarını içeren dosyalardır.

İnternette gezinirken ziyaret ettiğiniz web sayfaları, bilgisayarınıza ve telefonunuza küçük bilgi dosyaları kaydeder. Bu dosyalar telefon veya bilgisayarınızın hafızasında saklanır. Daha sonra aynı siteleri ziyaret ettiğinizde bu kayıtlı bilgi dosyaları sayesinde siteler sizi tanıyabilir.

Bilgileriniz bu dosyalara yazıldığından dolayı tekrar aynı web sayfalarını ziyaret ettiğinizde bilgilerinizi yeniden girmeye gerek duymazsınız.

Cookies, kişisel bilgiler de dahil olmak üzere birçok bilgiyi içerebilir. Web siteleri, ancak sizin izin verdığınız bilgilere erişebilir. Bu web sayfaları, sizin vermediğiniz bilgilere erişemez ya da bilgisayarlarınızdaki diğer dosyaları görüntüleyemez.



# JUnit

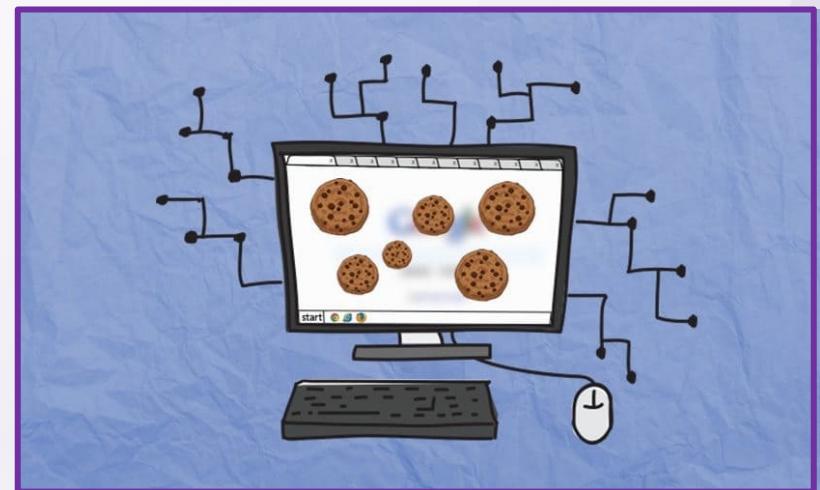
## Cookies

Birkaç farklılıkla, siber dünyadaki çerezlerin oturum çerezleri ve kalıcı çerez olmak üzere iki çeşidi vardır.

Oturum çerezleri yalnızca bir web sitesinde gezinirken kullanılır. Bunlar rastgele erişimli bellekte saklanır ve hiçbir zaman sabit sürücüye yazılmasın. Oturum sona erdiğinde oturum çerezleri otomatik olarak silinir.

Kalıcı çerezler bir bilgisayarda sonsuza kadar kalır ancak birçoğunun bir son kullanma tarihi olup bu tarihe gelindiğinde otomatik olarak kaldırılırlar.

Üçüncü taraf çerezler daha sıkıntılıdır. Bunlar, genellikle kullanıcıların halihazırda gezindiği web sayfalarındaki reklamlarla bağlantılı olduklarından bu sayfalardan farklı web siteleri tarafından oluşturulur.



# JUnit

## Cookies

Selenium ile cookies otomasyonu yapabiliyoruz.

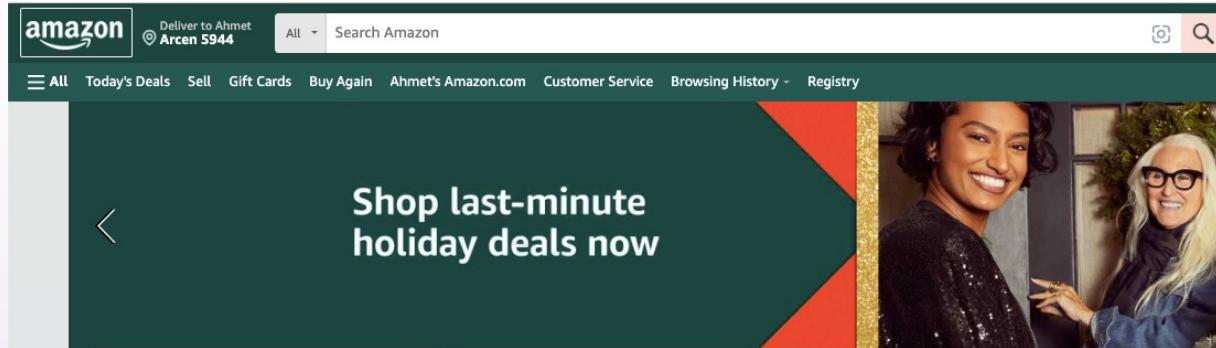
```
driver.manage().cook
    (m) addCookie(Cookie cookie)           void
    (m) getCookies()                     Set<Cookie>
    (m) deleteAllCookies()              void
    (m) deleteCookie(Cookie cookie)      void
    (m) getCookieNamed(String name)     Cookie
    (m) deleteCookieNamed(String name)   void
    ...
    Ctrl+Down and Ctrl+Up will move caret down and up in the editor  Next Tip
```

Driver.manage( ). method'u ile cookie'leri

- listeleyebilir
- Isim ile cagirabilir
- Yeni cookie ekleyebilir
- Var olanlari ismi silebilir
- Var olan tum cookie'leri silebiliriz

# JUnit

## Cookies



- 1- amazon anasayfaya gidin
- 2- tum cookie'leri listeleyin
- 3- Sayfadaki cookies sayisinin 5'den buyuk oldugunu test edin
- 4- ismi i18n-prefs olan cookie degerinin USD oldugunu test edin
- 5- ismi “en sevdigim cookie” ve degeri “cikolatali” olan bir cookie olusturun ve sayfaya ekleyin
- 6- eklediginiz cookie'nin sayfaya eklendigini test edin
- 7- ismi skin olan cookie'yi silin ve silindigini test edin
- 8- tum cookie'leri silin ve silindigini test edin

# JUnit

## Web Tables

Promotional Products				
Product Name	Category	Price	Actions	
 DELL Core i3 11th Gen 8 GB/256 GB SSD/32 GB EMMC Storage/Ubuntu	Electronics	\$399.00	<a href="#">Go</a>	
 Samsung White Smart Watch	Electronics	\$40.00	<a href="#">Go</a>	
 Medium 25 L Laptop Backpack For Office/College/Travel (Black, Yellow)	Travel	\$99.00	<a href="#">Go</a>	

```
<table>
  <thead>
    <tr>
      <th></th>
    </tr>
  </thead>

  <tbody>
    <tr>
      <td></td>
    </tr>
  </tbody>
</table>
```

Diagram illustrating the structure of an HTML table:

- The outermost element is the `<table>` tag, labeled as **table**.
- The first section is the `<thead>` tag, labeled as **header**.
- The first row in the header is the `<tr>` tag, labeled as **Header row**.
- The first cell in the header row is the `<th>` tag, labeled as **Header data**.
- The second section is the `<tbody>` tag, labeled as **body**.
- The first row in the body is the `<tr>` tag, labeled as **row**.
- The first cell in the body row is the `<td>` tag, labeled as **data**.

## Web Tables

Promotional Products			
Product Name	Category	Price	Actions
 DELL Core i3 11th Gen 8 GB/256 GB SSD/32 GB EMMC Storage/Ubuntu	Electronics	\$399.00	<a href="#">Go</a>
 Samsung White Smart Watch	Electronics	\$40.00	<a href="#">Go</a>
 Medium 25 L Laptop Backpack For Office/College/Travel (Black, Yellow)	Travel	\$99.00	<a href="#">Go</a>

- 1."<https://testotomasyonu.com/webtables>" adresine gidin
- 2.Web table tum body'sini yazdirin
3. Web tablosunda "Comfortable Gaming Chair" bulundugunu test edin
4. Web table'daki satir sayisinin 5 oldugunu test edin
5. Tum satirlari yazdirin
6. Web table'daki sutun sayisinin 4 oldugunu test edin
7. 3.sutunu yazdirin
8. Tablodaki basliklari yazdirin
9. Satir ve sutunu parametre olarak alip, hucredeki bilgiyi döndüren bir method olusturun
10. 4.satirdaki category degerinin "Furniture" oldugunu test edin

Promotional Products				
Product Name	Category	Price	Actions	
 DELL Core i3 11th Gen 8 GB/256 GB SSD/32 GB EMMC Storage/Ubuntu	Electronics	\$399.00	<a href="#">Go</a>	
 Samsung White Smart Watch	Electronics	\$40.00	<a href="#">Go</a>	
 Medium 25 L Laptop Backpack For Office/College/Travel (Black, Yellow)	Travel	\$99.00	<a href="#">Go</a>	

1. "https://testotomasyonu.com/webtables2" sayfasina gidin
2. Headers da bulunan basliklari yazdirin
3. 3.sutunun basligini yazdirin
4. Tablodaki tum datalari yazdirin
5. Tabloda kac tane cell (data) oldugunu yazdirin
6. Tablodaki satir sayisini yazdirin
7. Tablodaki sutun sayisini yazdirin
8. Tablodaki 3.kolonu yazdirin
9. Tabloda " Category" si Furniture olan urunun fiyatini yazdirin
10. Bir method olusturun, Test sayfasindan satir ve sutun verildiginde datayi yazdirinsin

## Web Tables

First Name	Last Name	Age	Email	Salary	Department	Action
Cierra	Vega	39	cierra@exam...	10000	Insurance	 
Alden	Cantrell	45	alden@exam...	12000	Compliance	 
Kierra	Gentry	29	kierra@exam...	2000	Legal	 

1. "https://demoqa.com/webtables" sayfasina gidin
2. Headers da bulunan basliklari yazdirin
3. 3.sutunun basligini yazdirin
4. Tablodaki tum datalari yazdirin
5. Tabloda kac tane bos olmayan 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. bir method olusturun, satir ve sutun sayisi girildiginde datayi yazdirlisin



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-10

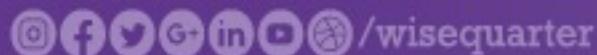
Excel Automation  
Get Screenshot



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)





# Onceki Dersten Aklimizda Kalanlar

- 1- Cookies : hem Server hem de client tarafinda cookie'ler tutulabilir. Temel amac server-client arasında karsilikli guven ve isbirligidir.
- 2- Cookies legal olarak kullanilan bir cookie ise endise etmeye gerek yoktur ama 3.taraf cookie'lere dikkat edilmelidir.
- 3- Bizim konu ile ilgimiz bir sayfaya gittigimizde farkli sebeplerle sayfadaki cookie'leri otomasyonla yonetme kabiliyetimizdir. driver.manage.cookie... ile istedigimiz islemleri yapabiliriz.
- 4- Web tables : Bir web sayfasinda tablo biciminde konumlandirilan webelementlere denir.
- 5- Klasik HTML kodlari kullanilrsa bir tablo header(<thead>) ve body(<tbody>)'den olusur. Hem header hem de body satir(<tr>)' lara sahip olabilir. Her bir satirin icinde de data(<td> veya <th>) olur.
- 6- Klasik HTML tablolarda taglari kullanarak webelementleri locate edebiliriz. Her bir tablo satirlardan, her satir da datalardan olustugu icin. Bir dataya ulasmak istersek once satira, oradan da dataya gideriz. Web tablolarinda sutun yapılanmasi yoktur. 3.sutun demek yerine her satirin 3.datasi demeyi tercih ederiz.
- 7- Klasik HTML kodlarla olusturulmazsa, HTML kodlari inceleyerek, developer'in body, satir ve datalari nasil formullestirdigini anlamali ve ona gore bir locate planlamasi yapmaliyiz.

## Excel Automation

Excel icin daha once inceledigimiz Web Table yapisina benzer bir yapı vardır.

Excel'de bir hucredeki bilgiye ulasmak icin dosya/sayfa/satir/sutun sirasiyla ilerlemeliyiz

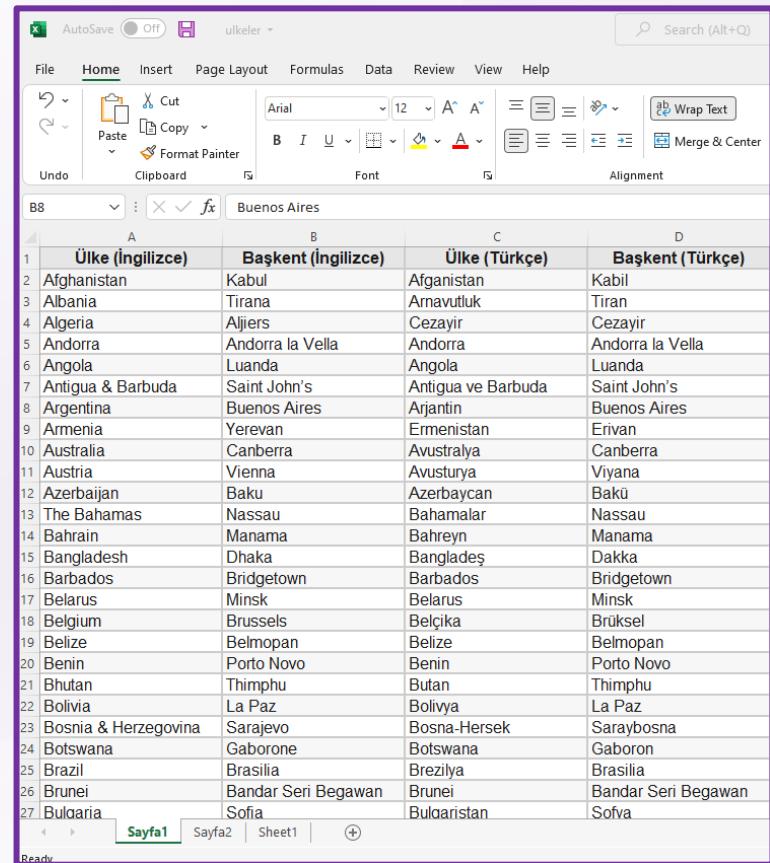
Excel ile ilgili otomasyonda web table'da oldugu gibi sutun yapisi yoktur, ihtiyac duyarsak kodla sutunu elde edebiliriz ancak bir dataya ulasmak icin sutun kullanamayiz

**Workbook** excel dosyamız

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

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

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



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

# JUnit

## Apache POI

Apache POI, microsoft ofis dokumanlarına erişmek için kullanılan Java API'ıdır.

Poi.apache.com sayfasından official dokumanlar incelenebilir.

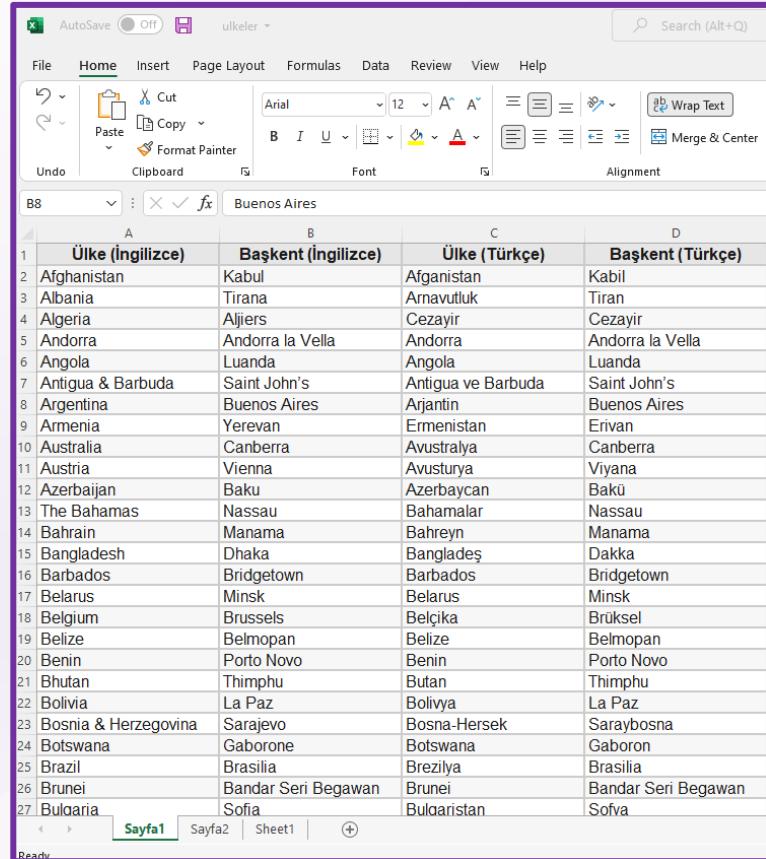
```
<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi</artifactId>
    <version>5.2.2</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.apache.poi/poi-ooxml -->
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-ooxml</artifactId>
    <version>5.2.2</version>
</dependency>
```

## Read Excel

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

- 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



The screenshot shows an Excel spreadsheet with four columns: 'Ülke (İngilizce)', 'Başkent (İngilizce)', 'Ülke (Türkçe)', and 'Başkent (Türkçe)'. The data includes 27 rows of countries and their capital cities, listed from Afghanistan to Bulgaria. The first row contains the column headers. The second row shows 'Afghanistan' with 'Kabul' as both the English and Turkish name. The third row shows 'Albania' with 'Tirana' as both the English and Turkish name. The fourth row shows 'Algeria' with 'Algiers' as both the English and Turkish name. The fifth row shows 'Andorra' with 'Andorra la Vella' as both the English and Turkish name. The sixth row shows 'Angola' with 'Luanda' as both the English and Turkish name. The seventh row shows 'Antigua & Barbuda' with 'Saint John's' as both the English and Turkish name. The eighth row shows 'Argentina' with 'Buenos Aires' as both the English and Turkish name. The ninth row shows 'Armenia' with 'Yerevan' as both the English and Turkish name. The tenth row shows 'Australia' with 'Canberra' as both the English and Turkish name. The eleventh row shows 'Austria' with 'Vienna' as both the English and Turkish name. The twelfth row shows 'Azerbaijan' with 'Baku' as both the English and Turkish name. The thirteenth row shows 'The Bahamas' with 'Nassau' as both the English and Turkish name. The fourteenth row shows 'Bahrain' with 'Manama' as both the English and Turkish name. The fifteenth row shows 'Bangladesh' with 'Dhaka' as both the English and Turkish name. The sixteenth row shows 'Barbados' with 'Bridgetown' as both the English and Turkish name. The seventeenth row shows 'Belarus' with 'Minsk' as both the English and Turkish name. The eighteenth row shows 'Belgium' with 'Brussels' as both the English and Turkish name. The nineteenth row shows 'Belize' with 'Belmopan' as both the English and Turkish name. The twentieth row shows 'Benin' with 'Porto Novo' as both the English and Turkish name. The twenty-first row shows 'Bhutan' with 'Thimphu' as both the English and Turkish name. The twenty-second row shows 'Bolivia' with 'La Paz' as both the English and Turkish name. The twenty-third row shows 'Bosnia & Herzegovina' with 'Sarajevo' as both the English and Turkish name. The twenty-fourth row shows 'Botswana' with 'Gaborone' as both the English and Turkish name. The twenty-fifth row shows 'Brazil' with 'Brasilia' as both the English and Turkish name. The twenty-sixth row shows 'Brunei' with 'Bandar Seri Begawan' as both the English and Turkish name. The twenty-seventh row shows 'Bulgaria' with 'Sofia' as both the English and Turkish name.

## Write Excel

- 1) Yeni bir Class olusturalim WriteExcel
- 2) Yeni bir test method olusturalim writeExcelTest()
- 3) Adimlari takip ederek Sayfa 1'deki 1.satira kadar gidelim
- 4) 5.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 kapatalım

## Get Screenshot / Tüm Sayfa

1.Adım : Bir TakeScreenshot objesi olusturup driver'imizi TakeScreenshot'a cast yapalim

```
TakesScreenshot tss= (TakesScreenshot) driver;
```

2.Adım : kaydettigimiz ekran goruntusunu projede istedigimiz yere kaydedebilmek icin path ile yeni bir File olusturalim

```
File tumSayfaSShot= new File( pathname: "target/ScreenShot/tumSayfaScreenshot.jpeg");
```

3.Adım : Takescreenshot objesini kullanarak getScreenshotAs( ) methodunu calistiralim ve gelen resmi gecici bir file'a assign edelim

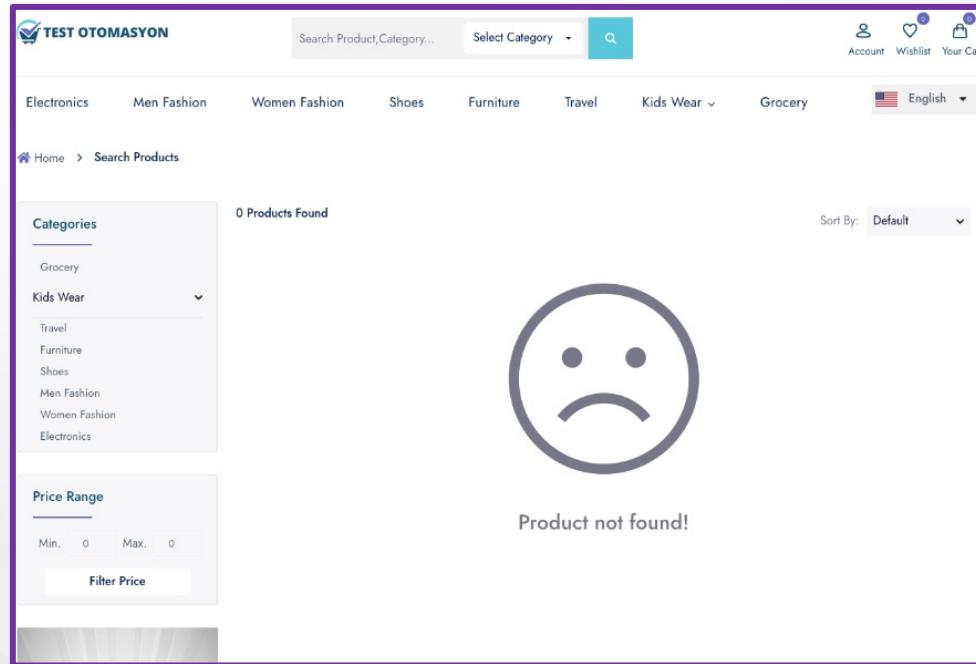
```
File geciciResim= tss.getScreenshotAs(OutputType.FILE);
```

4.Adım : Kaydettigimiz goruntuyu, saklamak istedigimiz dosyaya kopyalayalim

```
FileUtils.copyFile(geciciResim,tumSayfaSShot);
```

# JUnit

## Get Screenshot



- 1- testotomasyonu anasayfaya gidin
- 2- Nutella icin arama yapin
- 3- Arama sonucunda urun bulunamadigini test edin
- 4- Arama sonuc sayfasinin fotografini cekip, jpeg olarak kaydedin



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-11

Get Screenshot

Js Executors

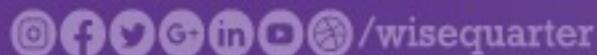
TestNG Framework



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



## Get Screenshot / Spesific Webelement

Selenium 4 ile gelen guzel ozelliklerden bir tanesi de istedigimiz WebElement'in fotografini almamiza imkan tanimasi

1.Adım : Istenilen webelement'i locate edin

```
WebElement sonucYaziElementi= driver.findElement(By.xpath( xpathExpression: "locator"));
```

2.Adım : kaydettigimiz ekran goruntusunu projede istedigimiz yere kaydedebilmek icin path ile yeni bir File olusturalim

```
File istenenElementSShot= new File( pathname: "target/ScreenShot/SonucyazisiScreenshot.jpeg");
```

3.Adım : Istenen webelement'i kullanarak getScreenshotAs( ) methodunu calistirralim ve gelen resmi gecici bir file'a assign edelim

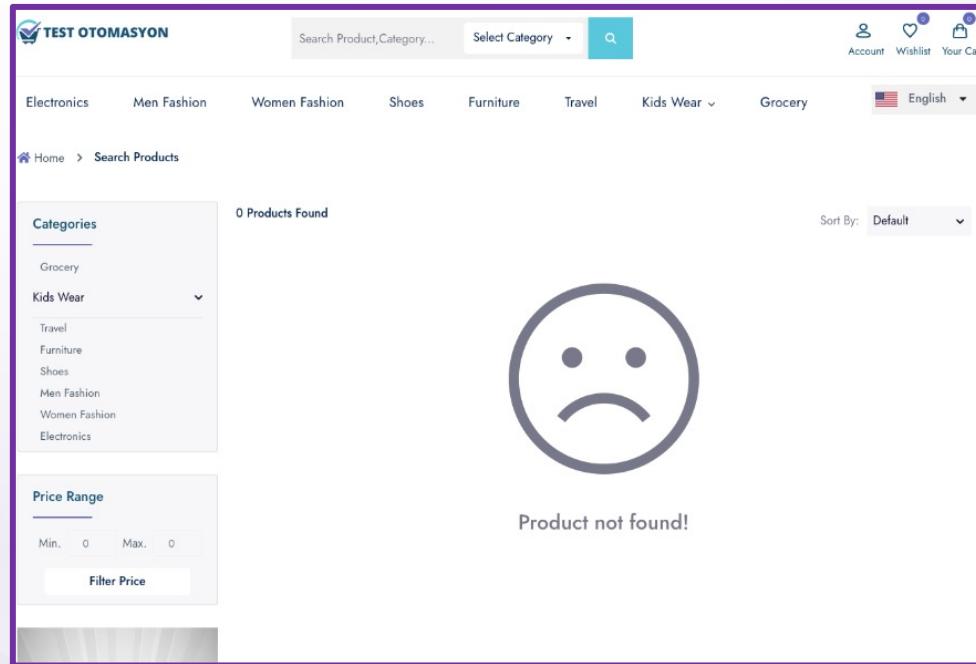
```
File geciciResim= sonucYaziElementi.getScreenshotAs(OutputType.FILE);
```

4.Adım : Gecici resmi, saklamak istedigimiz dosyaya kopyalayalim

```
FileUtils.copyFile(geciciResim,istenenElementSShot);
```

# JUnit

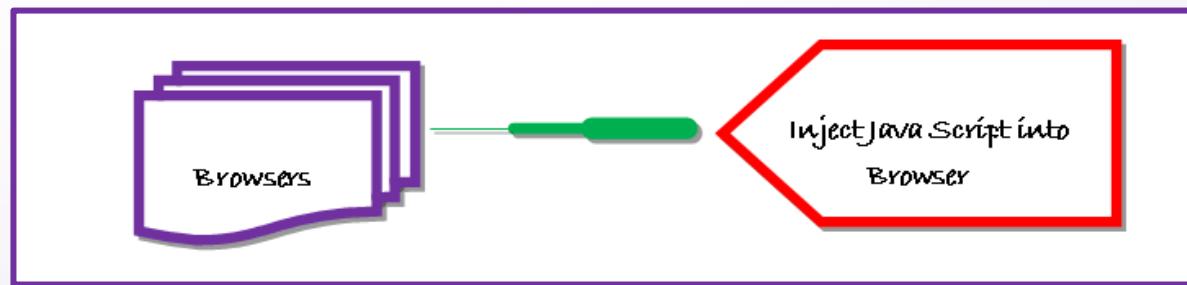
## Get Screenshot



- 1- testotomasyonu anasayfaya gidin
- 2- phone icin arama yapin
- 3- Arama sonucunda urun bulunabildigini test edin
- 4- Arama sonuc elementinin fotografini cekip, jpeg olarak kaydedin

# JUnit

## JS Executors



Selenium ile web elementlerinin kontrollerini sağlarken selenium komutları yetersiz kalabilir veya sorunlarla karşılaşabiliriz.

Bu durumlarda alternatif olarak üstesinden gelmek için JavascriptExecutor class'ını dahil edebiliriz.

JavaScript HTML kodlara direk erişip yönetebilen bir script dili olduğundan bize çok fazla kolaylık sağlayabilir.

# JUnit

## JS Executors

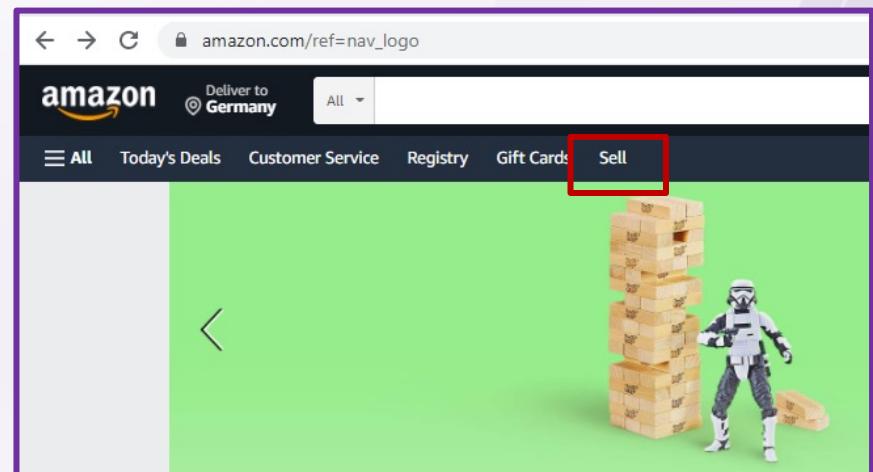
1.Adım : JavascriptExecutor kullanmak için ilk adım olarak driver'imizi JavascriptExecutor'a cast edelim

```
JavascriptExecutor jse= (JavascriptExecutor) driver;
```

2.Adım : Kullanacağımız WebElement'i locate edelim

3.Adım : Js driver ile executeScript methodunu calistiralim, icine parameter olarak ilgili script ve webelement'i yazalim

```
jse.executeScript( script: "ilgili script", ...args: "web element");
```



Ornegin Sell elementine click yapmak için

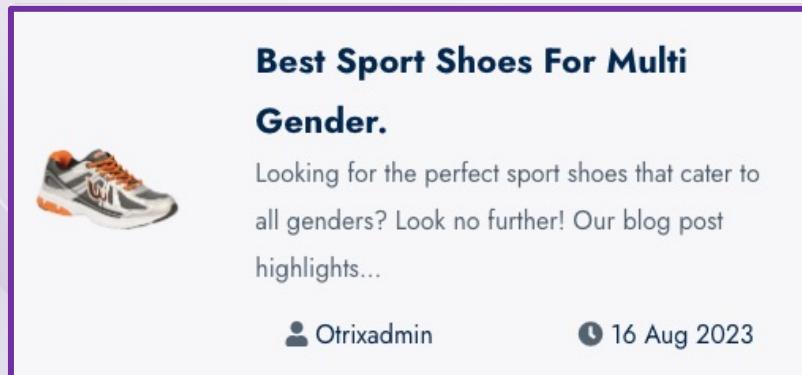
```
jse.executeScript("arguments[0].click();",sellLinki);
```

## JS Executors

Istenen Webelement'e kadar asagi inmek icin

```
jse.executeScript("arguments[0].scrollIntoView();",hedefWebelement);
```

Ornegin, testotomasyonu anasayfasina gidin alt kisimda bulunan “*Best Sport Shoes*” bolumune inmek istedigimizde



**Best Sport Shoes For Multi Gender.**

Looking for the perfect sport shoes that cater to all genders? Look no further! Our blog post highlights...

• Otrixadmin      16 Aug 2023

actions ile pageDown yapabiliriz ancak bilgisayarlarin cozunurlukleri farkli oldugundan her bilgisayarda calisacak bir kod yazamayiz.

Ama scrollIntoView ile yazdigimiz kod, tum bilgisayarlarda sorunsuz calisir.

JS alert kullanarak uygulama'dan kullaniciya mesaj vermek icin

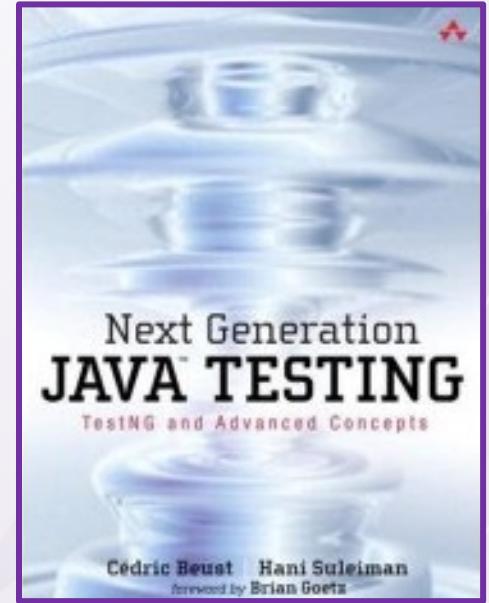
```
jse.executeScript("alert('yasasinnnn');");
```

# TestNG

## Nedir ?

TestNG, JUnit ve NUnit'ten ilham alan ancak onu daha güçlü ve kullanımı daha kolay hale getiren bazı yeni işlevler sunan bir test framework'udur, örneğin:

- Annotations
- Testlerinizi, kurallarını kendinizin belirleyeceği yöntemlerle, toplu çalıştırılabilme
- Esnek test konfigürasyonu.
- Veri odaklı test desteği (@DataProvider ile).
- Parametreler için destek.
- Çeşitli tool ve plug-ins tarafından desteklenir (Eclipse, IDEA, Maven, vb...).



TestNG, tüm test kategorilerini kapsayacak şekilde tasarlanmıştır: birim(unit), işlevsel(functional), uçtan uca(E2E), entegrasyon, vb...

# TestNG

## Neler Saglar ?

TestNG tester'lara daha fazla kontrol imkani verir ve testleri daha etkili yapmamizi saglar.

Tester'lar TestNG'yi etkili bir framework tasarlamak ve test case'leri TestNG annotation'ları ile organize etmek için kullanırlar.

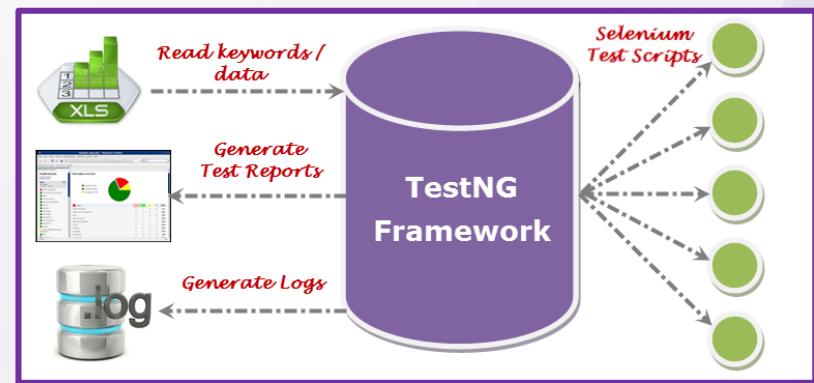
Test caseleri siralama ozelligi (**priority**) ve test caselerin birbirine bagimlilikti (**dependsOnMethod**) bize testleri organize etmeye yardım eder.

Yazilan test case'lerin paralel olarak çalıştırılmasına, birden fazla browser kullanılmamasına imkan tanır.

Error mesajlarının daha detaylı bir şekilde gösterilmesini sağlar.

Paralel ve Cross-Browser Test yapmamiza imkan tanır

Kullanicili HTML veya xml raporları olusturma imkani vardır





# TestNG

## Nasıl Yuklenir ?

Mvnrepository.com adresinden org.testng dependency eklemek yeterlidir.

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

# TestNG

## Annotations

### @Test Annotation

En çok kullanılan TestNG notasyonudur.  
Bir method'u bagimsiz olarak calisabilecek  
bir test method'u haline getirir.

Test notasyonu ile birlikte kullanılabilecek

- priority
- dataProvider
- groups

gibi bircok ozellik bulunur

@Test	Marks a class or a method as part of the test.
alwaysRun	If set to true, this test method will always be run even if it fails.
dataProvider	The name of the data provider for this test method.
dataProviderClass	The class where to look for the data provider. If no class is specified, the current class is used.
dependsOnGroups	The list of groups this method depends on.
dependsOnMethods	The list of methods this method depends on.
description	The description for this method.
enabled	Whether methods on this class/method are enabled.
expectedExceptions	The list of exceptions that a test method is expected to throw.
groups	The list of groups this class/method belongs to.
invocationCount	The number of times this method should be invoked.
invocationTimeOut	The maximum number of milliseconds this test should take to run.
priority	The priority for this test method. Lower priorities will be run first.
successPercentage	The percentage of success expected from this method.
singleThreaded	If set to true, all the methods on this test class are run sequentially. This attribute used to be called sequential (now deprecated).
timeOut	The maximum number of milliseconds this test should take to run.
threadPoolSize	The size of the thread pool for this method. The maximum number of threads used by this method.

## @Before @After Annotations

TestNG testlerimizi calistirirken yapacagimiz on hazirliklari koordine edebilmek icin daha fazla secenek sunmaktadır.

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

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

JUnit'deki @Before ve @After notasyonlarina benzer her test method'undan once ve sonra calisir.

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

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

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

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

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

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

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

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

Belirlenen grplardan once ve sonra calisir.

# TestNG

## Priority

TestNG test method'larini alfabetik siraya gore calistirir.

Eger hangi test method'unun hangi sira ile calisacagini belirlemek isterseniz priority kullanabilirsiniz.

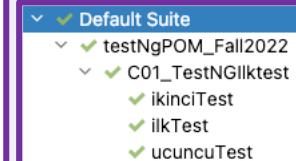
Kurallar:

1- Priority kucukden buyuge dogru calisir.

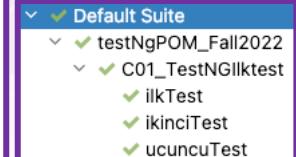
2- Priority verilmeyen test method'unun priority degeri 0 olarak kabul edilir.

3- Esit priority degerine sahip olan test method'lari isim sirasina gore calisir.

```
@Test  
public void ilkTest(){  
    driver.get("https://www.amazon.com");  
}  
  
@Test  
public void ikinciTest(){  
    driver.get("https://www.youtube.com");  
}  
  
@Test  
public void ucuncuTest(){  
    driver.get("https://www.wisequarter.com");  
}
```

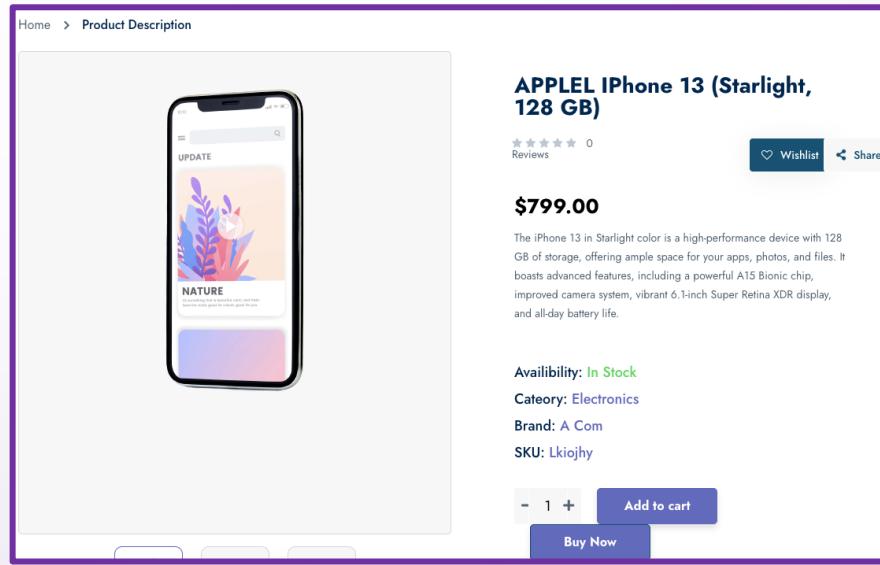


```
@Test(priority = 1)  
public void ilkTest(){  
    driver.get("https://www.amazon.com");  
}  
  
@Test(priority = 12)  
public void ikinciTest(){  
    driver.get("https://www.youtube.com");  
}  
  
@Test(priority = 35)  
public void ucuncuTest(){  
    driver.get("https://www.wisequarter.com");  
}
```



# TestNG

## Priority



Home > Product Description

**APPLEL IPHONE 13 (Starlight, 128 GB)**

Reviews 0

Wishlist Share

**\$799.00**

The iPhone 13 in Starlight color is a high-performance device with 128 GB of storage, offering ample space for your apps, photos, and files. It boasts advanced features, including a powerful A15 Bionic chip, improved camera system, vibrant 6.1-inch Super Retina XDR display, and all-day battery life.

Availability: In Stock

Category: Electronics

Brand: A Com

SKU: Lkiojhy

- 1 + Add to cart

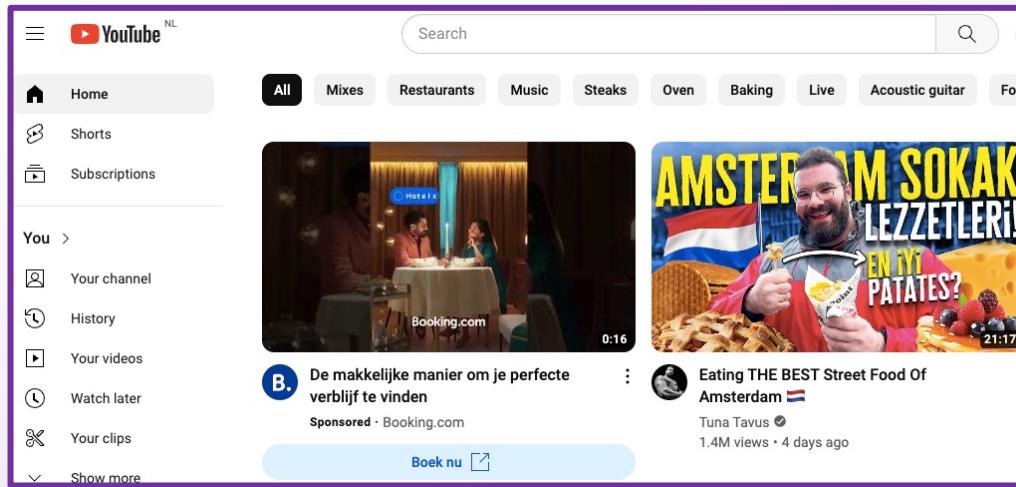
Buy Now

- <https://www.testotomasyonu.com/> adresine gidin. Olusturacaginiiz 3 farkli test method'unda asagida verilen görevleri yapin.

1. **Test** : Testotomasyonu ana sayfaya gittiginizi test edin
2. **Test** : search Box'i kullanarak “phone” icin arama yapin ve arama sonucunda urun bulunabildigini test edin
3. **Test** : ilk urunu tiklayin ve urun isminin case sensitive olmaksizsin phone icerdigini test edin

# TestNG

## Priority



1) <https://www.youtube.com> adresine gidin

2) 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

## dependsOnMethods

dependsOnMethods test method'larinin siralamasi ile ilgili degildir. Siralamayi priority ile tanimlayabiliriz.

dependsOnMethods ile baska method'a baglanan bir method calismaya baslamadan once baglandigi method'un calisip, PASSED oldugunu kontrol eder.

- Bagli oldugu test calisti ve PASSED olduysa testi **calistirir**.
- Bagli oldugu test calisti ve FAILED olduysa testi **ignore eder**.
- Bagli oldugu test calismadi ise oncelikle **bagli oldugu testi calistirir**, sonucuna gore kendisi calisir veya ignore eder

Bagli oldugu testi calistirma sadece 2 method icindir. 3 Method birbirine baglandiginda, 3.method calistirilmak istendiginde 1.method'a kadar gitmez.

```
@Test  
public void test01(){  
  
    System.out.println("1.test calisti");  
}  
  
@Test(dependsOnMethods = "test01")  
public void test02(){  
  
    System.out.println("2.test calisti");  
}  
  
@Test(dependsOnMethods = "test02")  
public void test03(){  
  
    System.out.println("3.test calisti");  
}
```

# TestNG

## dependsOnMethods

```
@Test  
public void amazonTesti(){...}  
  
@Test(priority = 2,dependsOnMethods = "amazonTesti")  
public void nutellaTesti(){...}  
  
@Test(priority = 5,dependsOnMethods = "nutellaTesti")  
public void aramaSonucTesti(){...}
```

- <https://www.testotomasyonu.com/> adresine gidin. Olusturacaginiiz 3 farkli test method'unda asagida verilen görevleri yapin.
  1. Test : Testotomasyonu ana sayfaya gittiginizi test edin
  2. Test : 1.Test PASSED ise search Box'i kullanarak “phone” icin arama yapin ve arama sonucunda urun bulunabildigini test edin
  3. Test : 2.Test PASSED ise ilk urunu tiklayin ve urun isminin case sensitive olmaksizin phone icerdigini test edin

## Java'da Class Uyelerini Kullanma

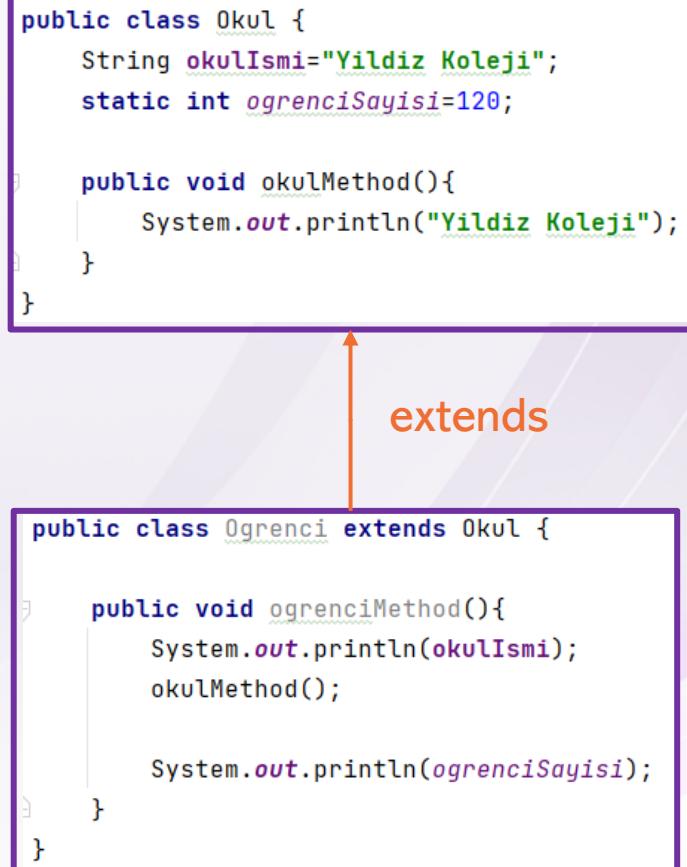
Java ile olusturulan bir proje'de farkli class'daki class uyelerine erisim ve kullanma farkli yontemlerle yapilabilir.

### 1- inheritance (Miras)

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

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

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





## Java'da Class Uyelerini Kullanma

### 2- Object olusturarak

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

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

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

### 3- Static Class Uyeleri :

Eger kullanacagimiz variable veya method static ise object olusturmadan direk class ismi ile variable veya method'a ulasabiliriz.

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

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



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-12

### TestNG Framework



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter

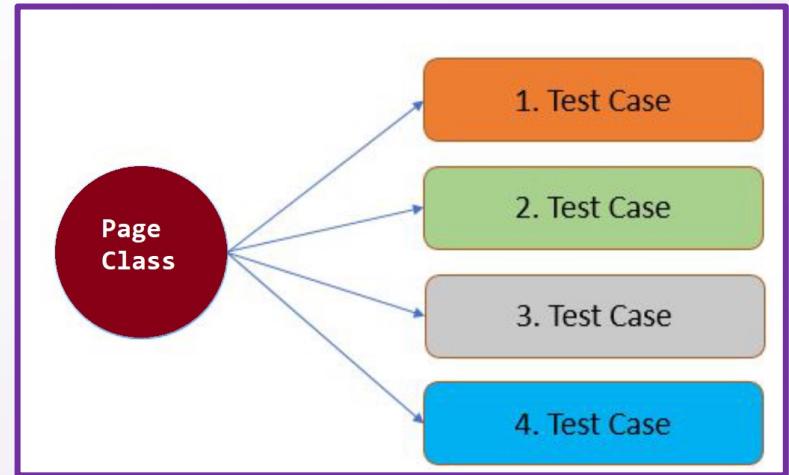


# Onceki Dersten Aklimizda Kalanlar

- 1- TestNG : JUnit'den ilham almakla birlikte, JUnit'in cozemedigi konulara el atmak, tester'larin daha etkin oldugu bir framework olusturup Java ile yapılan testleri Next Generation'a tasimak amaciyla ortaya cikmistir.
- 2- Daha fazla notasyon, notasyolarin yaninda onlari destekleyecek priority, dependsOnMethods gibi ek ozellikler, istenen konfigurasyonlara uygun olarak toplu test calistirma, parallel ve/veya cross-browser calistirma, bazi yardimci kutupaneler kullanarak raporlama,dataProvider ile “data driven testing” gibi pek cok ozelligi barindirir.
- 3- Guncel olarak is yerimizde kullanabilecegimiz bir framework'tur. Olusturulan yapı sayesinde testleri yazmak, test datalarini(kullanilacak browser, kullanici adi ve sifreleri, url'ler, aranacak kelimeler) ve locate'leri yonetmek daha kolay olmaktadır.
- 4- TestNG testleri yazdiktan sonra maintenance(surdurme) kolayligi da amaclar.
- 5- Driver, locate'ler ve test datalari icin Java'daki OOP Conzept cercevesinde ayri class'lar ve dosyalar olusturularak. Bunlarda yapılacak degisikliklerin kolayca yapılmasını ve testlerin update edilmesini saglar.
- 6- Kullananan class'lar ve bu class'ların OOP cercevesinde kullanım bicimi TestNG ve POM tarafından standartlastirilmistir. Boylece ekipteki degisikliklerde framework'u anlamak ve uyum saglamak da kolay olmaktadır.

## Page Object Model Framework Design

Bir sirkette test framework'u olusturdugumuzda kullanici adi, sifresi, gidilecek web adresi gibi test datalari tum testler icin gecerlidir. Ayrıca surekli kullanmamız gereken variable ve method'lar olacaktır.



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 duydugumuzda bu verilere kolayca ulasabilir veya test datalari ile ilgili bir degisiklik yapmamız gerektiginde sadece kaynak dosyadan bir degeri degistirerek tum test case'leri guncellemis oluruz.

# TestNG

## Framework Yapisi

TestNG ile olusturacagimiz bu framework, isyerinde kullanilabilecek kullanisli bir framework olacak.

TestNG ile uygulayacagimiz POM (Page Object Model) icin Test'lerimizin oldugu class'lar disinda bazi package ve class'lar daha olusturulacaktir.

Olusturulacak POM (Page Object Model) ile

- tum ekip ile sorunsuz sekilde calisabilecegimiz,
- test datalarini kolayca yonetebilecegimiz
- Kod tekrarlarinden kurulacagimiz

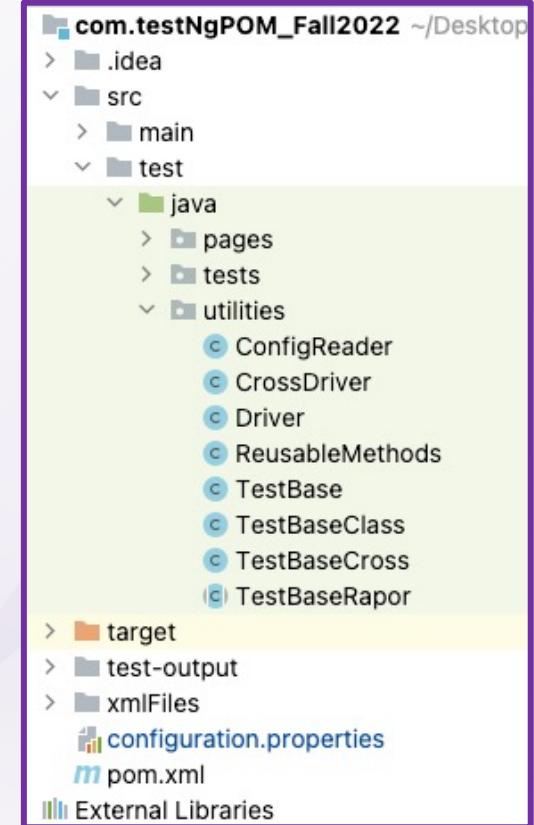
bir yapı olusturulacaktır.

1- File menusunden New, Project secip yeni proje olusturun.

2- src/test/java package'i altinda iki package olusturun

- tests
- utilities

3- Yeni olusturdugumuz tests package'i altinda gunluk package ve test class'i olusturun



## Page Object Model Framework Design

POM çok popüler bir **Framework Design Pattern** 'dir.

Test suitlerimizde çok fazla testimiz olduğunda, test caseleri ve kodları korumak daha karmaşık hale gelir.

Bu nedenle,

sürdürülebilir(**maintainable**),

yeniden kullanılabilir(**reusable**),

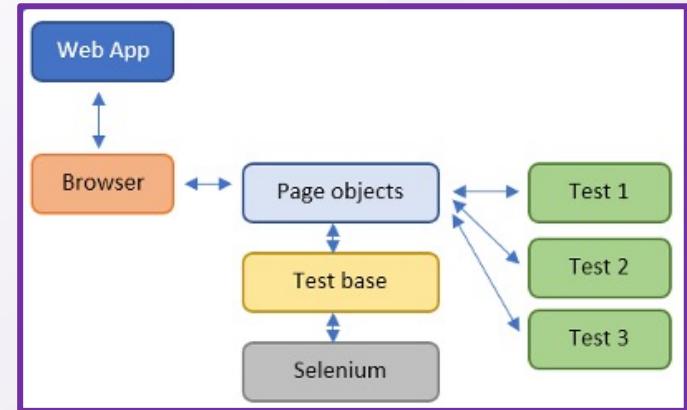
daha hızlı(**faster**),

anlaşılabilir( **understandable**)

daha iyi bir framework dizaynına ihtiyacımız vardır.

Page object model ile, sayfaya özgü elementleri veya methodları **page class** icinde tutar, ve bunları gerçek **test class**larından uzak tutarız.

POM ile ihtiyacımız olan class üyelerini sadece bir kez create edip birden çok kez kullanabiliriz.



## Page Object Model Framework Design

Framework'un verimliliğini artırmak için core Java ve Selenium konseptini kullanarak temel olarak **page classları** ve **test classları** oluşturacağız.

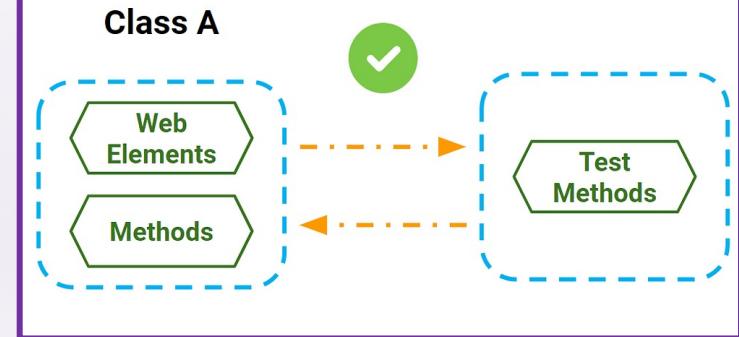
Tüm şirketler page object model dizaynını kullanmaz, ancak herkes bunu bilir ve daha da popüler hale gelmektedir.

Daha iyi bir tasarım, testin yürütme süresini daha hızlı hale getirir.

Bir uygulamanın(application) işlevselligi değiştiğinde, kodu düzeltmek için framework kontrol edilmesini ve gerekli düzeltmelerin yapılmasını kolaylaştırır.

Page Object Design daha bağımsız test senaryoları oluşturmamıza yardımcı olur, böylece test komut dosyalarında(script) hata ayıklamak daha kolay olacaktır.

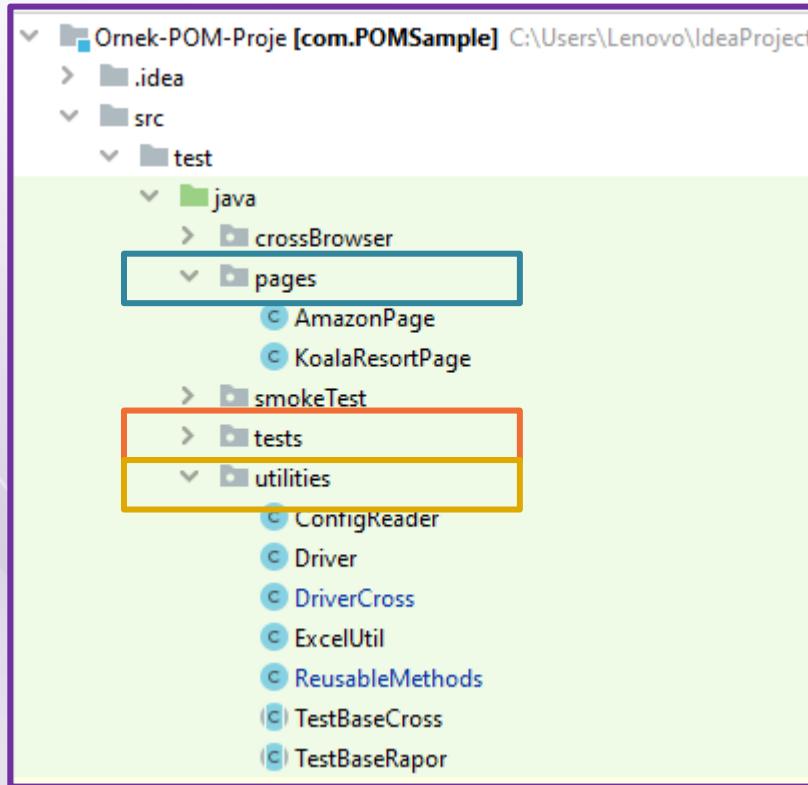
Design Patterns in Automated Testing series  
**Page Object Model**





# 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

## Page Object Model Framework Design

Login Now

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

Don't have an Account? [Sign Up.](#)

<https://www.testotomasyonu.com/>

Calistigimiz uygulamanin login ile girilen bir uygulama oldugunu dusunelim.

Yaptiginiz tum testlerde oncelikle login olmaniz gerekecektir.

Bu durumda her test icerisinde username, password kutularini ve login butonunu locate etmek zorunda kalirsınız

Yine her test icin gecerli kullanici adi ve sifre bilgilerini girmeniz gerekir.

Gecerli kullanici adi ve sifre bilgisi test datalari her degistiginde kullanilmis oldugu testleri bulmak ve tamamini guncellemek gerekecektir.

POM locator'lari sadece 1 kere yapip, ihtiyacimiz oldugunda kullanma, test datalarini bir kere olusturup kullanacagimiz zaman oradan kullanip, update yapacagimiz zaman oradan update yapma gibi imkanlar tanir.

## Page Object Model / Basic 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.

Su anki haliyle TestBase Class'imizla ayni olacak ama inheritance kullanmak yerine static method'lar ile driver'i kullanacagiz. Driver class'ini daha sonra tum browser'lar ile calisacak hale getirecegiz

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

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

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

## Page Object Model / @FindBy Annotation

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

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

```
public class HotelMyCampPage {  
    public HotelMyCampPage(){  
  
        PageFactory.initElements(Driver.getDriver(), page: this);  
    }  
  
    @FindBy(xpath = "//a[text()='Log in']")  
    public WebElement ilkLoginLinki;  
  
    @FindBy(xpath = "//input[@id='UserName']")  
    public WebElement usernameBox;  
  
    @FindBy(xpath = "//input[@id='Password']")  
    public WebElement passwordBox;  
  
    @FindBy(xpath = "//input[@id='btnSubmit']")  
    public WebElement loginButonu;  
  
    @FindBy(xpath = "//div[@class='validation-summary-errors']")  
    public WebElement girisYapilamadiYaziElementi;  
  
    @FindBy(xpath="//span[text()='ListOfUsers']")  
    public WebElement basariliGirisYaziElementi;
```

## Page Object Model / Page Factory

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

1. PageFactory, page object dizayni icin bir önemli classtır.
2. Page objelerini instantiate(ilk deger atama) için page classlarında PageFactory kullanıyoruz.
3. PageFactory.initElements(driver,this);  
this => page instance  
driver => bizim gonderecegimiz driver
4. Aslinda PageFactory class'ina, elementlere ilk degeri atayan initElements( ) metodunu kullanmak icin ihtiyacimiz var

```
public class HotelMyCampPage {  
    public HotelMyCampPage(){  
  
        PageFactory.initElements(Driver.getDriver(), page: this);  
    }  
  
    @FindBy(xpath = "//a[text()='Log in']")  
    public WebElement ilkLoginLinki;  
  
    @FindBy(xpath = "//input[@id='UserName']")  
    public WebElement usernameBox;  
  
    @FindBy(xpath = "//input[@id='Password']")  
    public WebElement passwordBox;  
  
    @FindBy(xpath = "//input[@id='btnSubmit']")  
    public WebElement loginButonu;  
  
    @FindBy(xpath = "//div[@class='validation-summary-errors']")  
    public WebElement girisYapilamadiYaziElementi;  
  
    @FindBy(xpath="//span[text()='ListOfUsers']")  
    public WebElement basariliGirisYaziElementi;
```



# TestNG

## Page Object Model

### Page Class

```
@FindBy(xpath = "//a[text()='Log in']")
public WebElement loginLinki;

@FindBy(xpath = "//input[@id='login-email']")
public WebElement emailKutusu;

@FindBy(xpath = "//input[@id='login-password']")
public WebElement passwordKutusu;

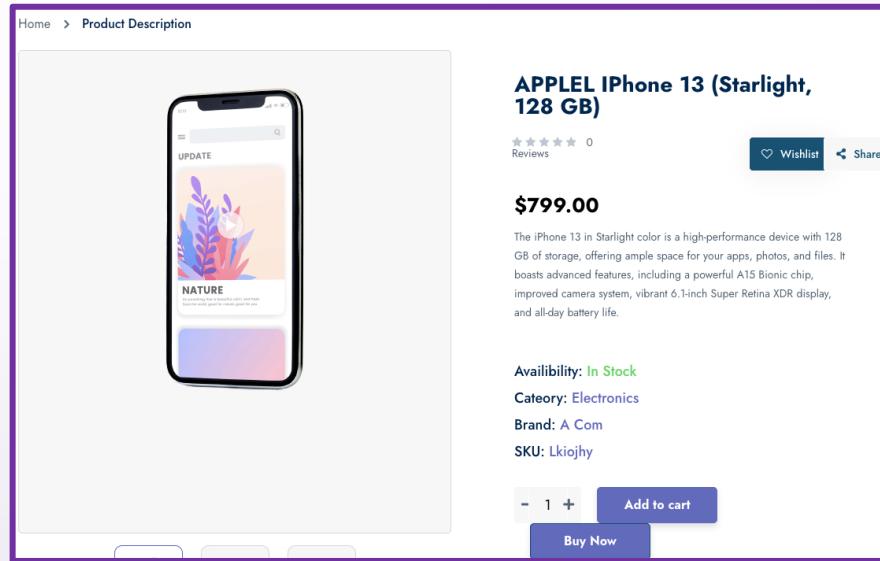
@FindBy(xpath = "//button[text()='Login']")
public WebElement loginButonu;
```

```
@Test
public void pozitifLoginTesti(){
    // Mycoursesdemmy anasayfasına gidel
    Driver.getDriver().get(ConfigReader.getProperty("myUrl"));
    // login linkine basin
    MyCoursesdemmyPage myCoursesdemmyPage=new MyCoursesdemmyPage();
    myCoursesdemmyPage.loginLinki.click();
    // Kullanici email'i olarak valid email girin
    myCoursesdemmyPage.emailKutusu.sendKeys(ConfigReader.getProperty("myGecerliEmail"));
    // Kullanici sifresi olarak valid sifre girin
    myCoursesdemmyPage.passwordKutusu.sendKeys(ConfigReader.getProperty("myGecerliPassword"));
    // Login butonuna basarak login olyun
    myCoursesdemmyPage.loginButonu.click();
    // Basarili olarak giris yapildigini test edin

    Assert.assertTrue(myCoursesdemmyPage.coursesLinki.isDisplayed());

    ReusableMethods.bekle(saniye: 2);
    Driver.closeDriver();
}
```

## Arama Testi



- 1- testotomasyonu anasayfaya gidin
- 2- phone icin arama yapin
- 3- Arama sonucunda bulunan urun sayisinin 3'den cok oldugunu test edin
- 4- ilk urunu tiklayin
- 5- acilan urun sayfasinda, urun isminde case sensitive olmadan phone bulundugunu test edin.

## Page Object Model

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



The screenshot shows a login form with a purple border. It contains two input fields: 'E-posta veya Telefon Numarası' (Email or Phone Number) and 'Şifre' (Password). Below these is a large blue button labeled 'Giriş Yap' (Log In). To the right of the password field is a link 'Şifreni mi Unuttun?'. At the bottom is a green button labeled 'Yeni Hesap Oluştur' (Create New Account).

## Page Object Model

Doğum Tarihi

Gün	Ay	Yıl
-----	----	-----

- 1 - <https://testotomasyonu.com/form> adresine gidin
- 2- Dogum tarihi gun seçenekinden index kullanarak 5'i secin
- 3- Dogum tarihi ay seçenekinden value kullanarak Nisan'i secin
- 4- Dogum tarihi yil seçenekinden visible text kullanarak 1990'i secin
- 5- Secilen değerleri konsolda yazdirin
- 6- Ay dropdown menüdeki tum değerleri(value) yazdirın
- 7- Ay Dropdown menusunun boyutunun 13 olduğunu test edin

## Page Object Model

- 1 - <https://testotomasyonu.com/form> adresine gidin
- 2- Sirt Agrisi ve Carpinti checkbox'larini secin
- 3- Sirt Agrisi ve Carpinti checkbox'larinin seçili olduğunu test edin
- 4- Seker ve Epilepsi checkbox'larinin seçili olmadigini test edin
- 5- sayfayı kapatın

Son Bir Yılda Yaşadığınız Rahatsızlıklar

- Bölgesel Öksürük
- Nefes Darlığı
- Göğüs Ağrısı
- Çarpıntı
- Sırt Ağrısı
- İshal veya Kabızlık
- Eklem Ağrısı

Diger(Açıklayınız)

Rahatsızlıklar...

Aşağıdaki Hastalıklardan Hangisine Sahipsiniz?

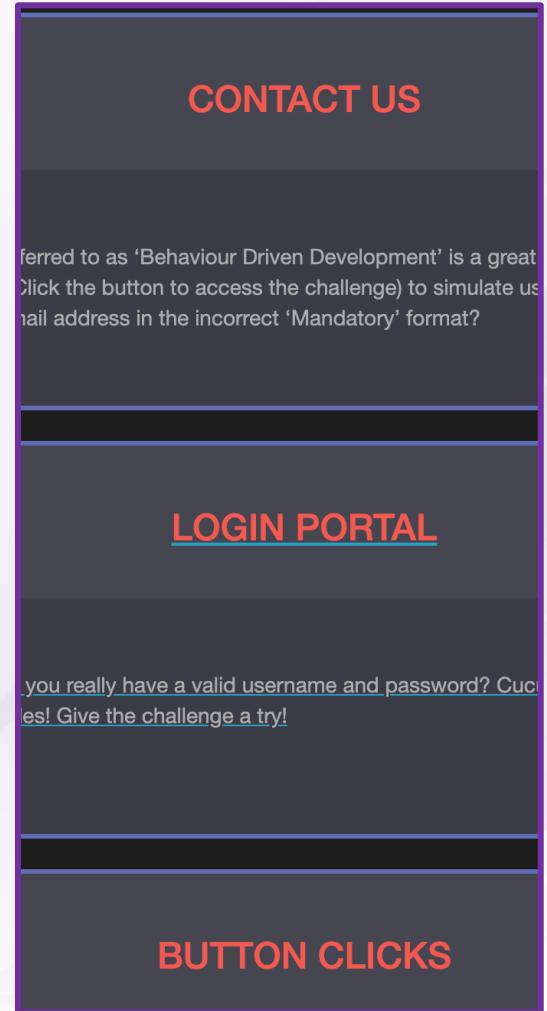
- Kalp Hastalığı
- Şeker
- Böbrek Yetmezliği
- Mide Ülseri
- İşitme Kaybı
- Sinir Sistemi Rahatsızlıkları
- Epilepsi

Diger(Açıklayınız)

Rahatsızlıklar...

## Page Object Model

- 1."<http://webdriveruniversity.com/>" adresine gidin
- 2."Login Portal" a kadar asagi inin
- 3."Login Portal" a tiklayın
- 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 kapatın
- 9.Ilk sayfaya geri donun
- 10.Ilk sayfaya donuldugunu test edin



The screenshot shows a dark-themed web application interface. It consists of three main sections separated by horizontal lines:

- CONTACT US**: Contains text about Behaviour Driven Development and a challenge related to email address validation.
- LOGIN PORTAL**: Contains a login form with fields for username and password, and a message asking if the user has a valid account.
- BUTTON CLICKS**: A section at the bottom.

# TestNG

## POM Properties File

configuration.properties Test datalarini tuttugumuz .properties uzantılı bir dosyadır.

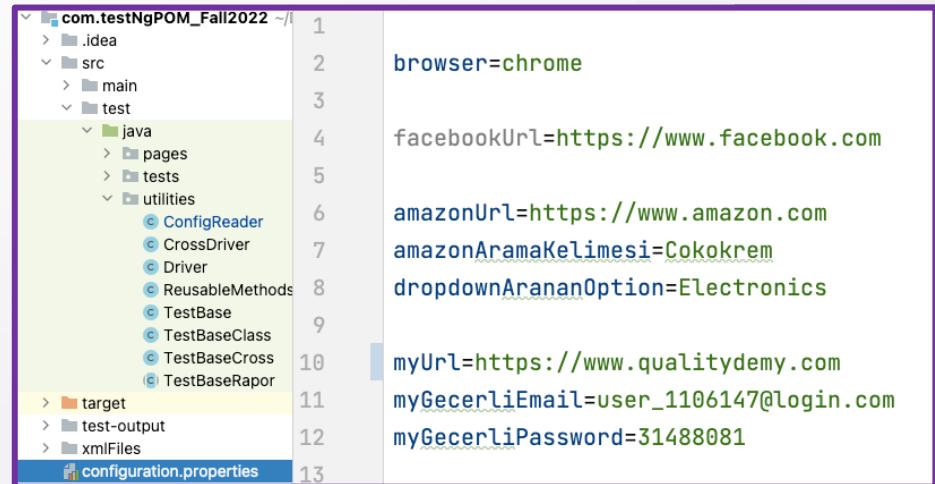
Bu dosya , test datalarina erisimi ve test datalarinda yapılan değişikliklerin tum test case'lere islenmesini kolaylastirir.

Örneğin :

Her test method'unda  
driver.get("https://www.amazon.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 key kullanarak value cagırırız

- url=https://www.amazon.com/
- browser=chrome
- username=manager
- password=3145665
- name=Ali



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

- com.testNgPOM\_Fall2022
- .idea
- src
  - main
  - test
    - java
      - pages
      - tests
      - utilities
        - ConfigReader
        - CrossDriver
        - Driver
        - ReusableMethods
        - TestBase
        - TestBaseClass
        - TestBaseCross
        - TestBaseRapor
    - target
    - test-output
    - xmlFiles
- configuration.properties

On the right side, the contents of the configuration.properties file are displayed:

```
browser=chrome
facebookUrl=https://www.facebook.com
amazonUrl=https://www.amazon.com
amazonAramaKelimesi=Cokokrem
dropdownArananOption=Electronics
myUrl=https://www.qualitydemy.com
myGecerliEmail=user_1106147@login.com
myGecerliPassword=31488081
```

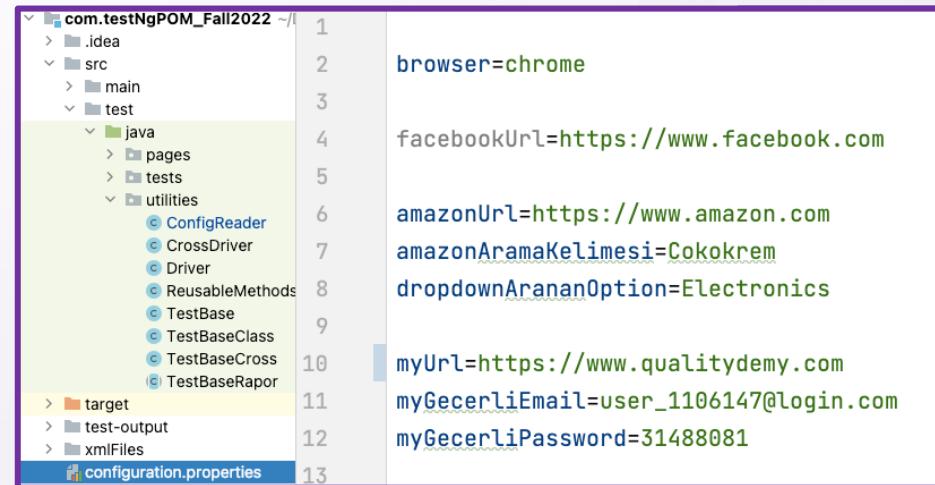
## POM Properties File

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

New => File => isim : configuration.properties

Dosya olustururken bizim icin onemli olan ismi degil, uzantisidir (.properties)

Bu uzanti sayesinde Java kutuphanesinden Properties Class'ini kullanabilir, olusturacagimiz obje yardimiyla configuration.properties dosyasindaki key-value ikililerine ulasabiliriz



The screenshot shows a Java project structure in a code editor. The project root is 'com.testNgPOM\_Fall2022'. It contains '.idea', 'src' (with 'main' and 'test' subfolders), 'java' (with 'pages', 'tests', and 'utilities' subfolders, containing classes like ConfigReader, CrossDriver, Driver, ReusableMethods, TestBase, TestBaseClass, TestBaseCross, and TestBaseRapor), 'target', 'test-output', and 'xmlFiles'. A 'configuration.properties' file is located at the bottom of the 'src/test/java' folder. The file content is as follows:

```
browser=chrome
facebookUrl=https://www.facebook.com
amazonUrl=https://www.amazon.com
amazonAramaKelimesi=Cokokrem
dropdownArananOption=Electronics
myUrl=https://www.qualitydemy.com
myGecerliEmail=user_1106147@login.com
myGecerliPassword=31488081
```

properties file test dataları saklar.

Bu dosyayı kullanmanın amacı, kodu sabit(hard coded) değil, dinamik yapmaktır. Bu dosya sayesinde tüm kullanıcılar verilere kolayca ulaşabilir ve update edebilir.

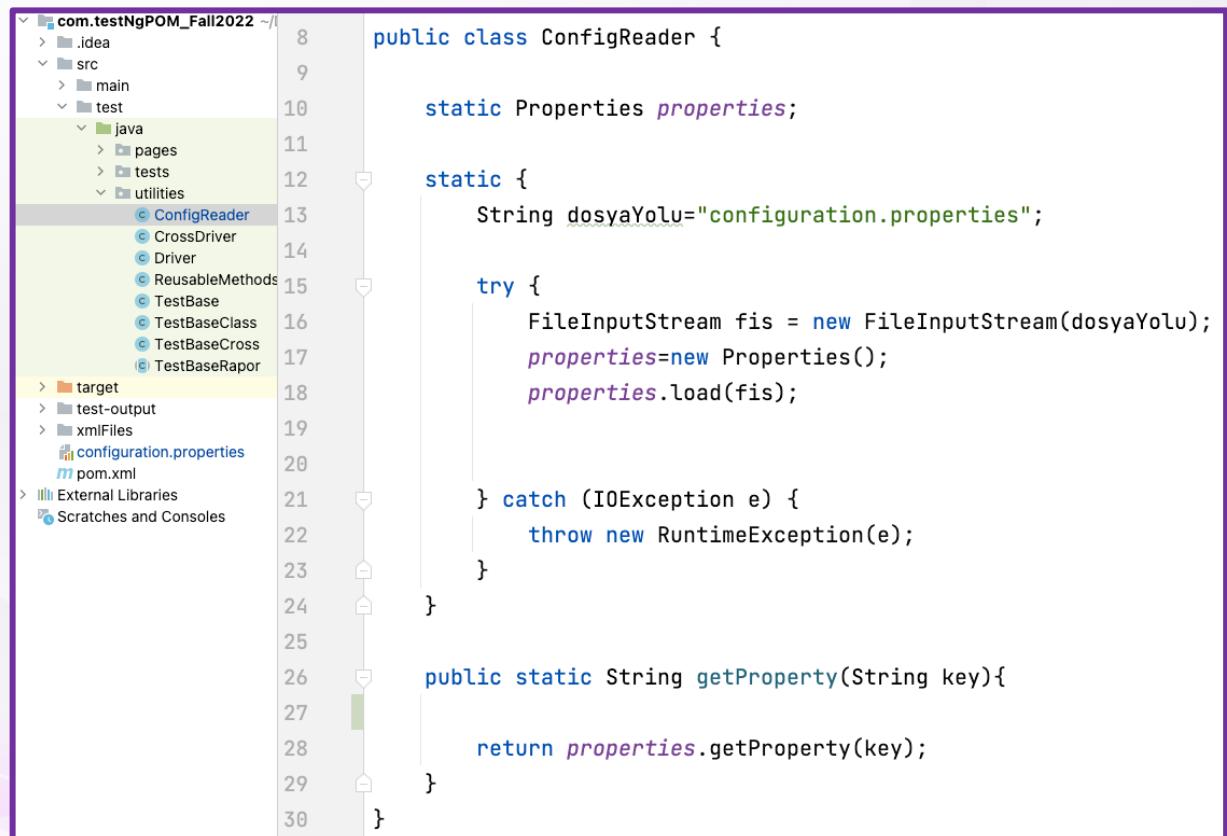
## POM ConfigurationReader

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

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

Method static oldugundan method icerisinden cagiracagimiz variable da static olmalidir.

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



The screenshot shows a file structure for a Java project named 'com.testNgPOM\_Fall2022'. The 'src/main/test/java' directory contains several files: ConfigReader.java, CrossDriver.java, Driver.java, ReusableMethods.java, TestBase.java, TestBaseClass.java, TestBaseCross.java, and TestBaseReport.java. The 'src/main/test/resources' directory contains 'configuration.properties' and 'pom.xml'. The 'Scratches and Consoles' tab is also visible. The code for 'ConfigReader.java' is displayed:

```
public class ConfigReader {
    static Properties properties;
    static {
        String dosyaYolu="configuration.properties";
        try {
            FileInputStream fis = new FileInputStream(dosyaYolu);
            properties=new Properties();
            properties.load(fis);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
    public static String getProperty(String key){
        return properties.getProperty(key);
    }
}
```

# TestNG

## POM ConfigurationReader

### Test Class

```
public class TestClass {
    @Test
    public void test1(){
        String url = ConfigReader.getProperty("amazon_url")
    }
}
```

- 1 ) Test method'undan "amazon\_url" key'e ait value'yu kullanmak istedigimizde
- 2 ) ConfigReader Class'indan getProperty( ) method'unu kullaniriz.
- 3 ) getProperty( ) method'u configuration.properties dosyasina gidip istedigimiz key'e ait value'yu bize dondurur.

### ConfigurationReader Class

```
public class ConfigReader {
    private static Properties properties;

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

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

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

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

### Configuration.properties file

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

## Page Object Model

### PositiveTest

- 1- <https://www.testotomasyonu.com/> anasayfasina gidin
- 2- account linkine basin
- 3- Kullanici email'i olarak gecerli email girin
- 4- Kullanici sifresi olarak gecerli password girin
- 5- Login butonuna basarak login olun
- 6- Basarili olarak giriş yapılabilidigini test edin

**Login Now**

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

[Don't have an Account? Sign Up.](#)



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-13

TestNG Framework

TestNG Assertions

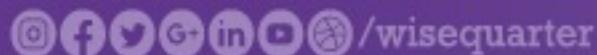
Xml File Kullanımı



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# TestNG

## Page Object Model

### Negative Test

1- <https://www.testotomasyonu.com/> anasayfasina gidin

2- account linkine basin

3- 3 farkli test method'u olusturun.

- gecerli email, gecersiz password
- gecersiz email, gecerli password
- gecersiz email, gecersiz password.

4- Login butonuna basarak login olmayi deneyin

5- Basarili olarak giris yapilamadigini test edin

**Login Now**

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

Don't have an Account? [Sign Up.](#)

# TestNG

## Page Object Model

Image	Product Name	Quantity	Total Price
	DELL Core i3 11th Gen - (8 GB/256 GB SSD/32 GB EMMC Storage/Ubuntu)  Model: DELL220 Unit Price: \$399.00	<input type="button" value="1"/> <input type="button" value="^"/> <input type="button" value="▼"/> <input type="button" value="C"/>	\$399
<input type="button" value="Continue Shopping"/>			

Urun sepeti testi

- 1- <https://www.testotomasyonu.com/> anasayfasina gidin
- 2- phone icin arama yapin
- 3- Listelenen sonuclardan ilkini tiklayin
- 4- urun ismini kaydedin ve sepete ekleyin
- 5- your cart linkine tiklayin
- 6- kaydettiginiz urun ismi ile sepetteki urun isminin ayni oldugunu test edin
- 7- sayfayı kapatın



## POM Driver Class

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

### Basic Driver Class

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

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

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

## Tum Browserlar icin Kullanim

Driver Classımızı tüm tarayıcılar(browser) için geliştireceğiz.

Driver objesi create etmeden önce farklı tarayıcı koşullarını kontrol etmek için switch statement kullanıyoruz.

Driver Class'i singleton pattern'e uygun dizayn ederek tüm projede farklı driver üretilmesinin onune geceriz

Sonra TestBasede yaptığımız gibi "wait" koyabiliriz.

Ardından driver'i kapatmak için method oluşturuyoruz.

Şu andan itibaren TestBase Classını değil Driver Classını kullanacağız.

# TestNG

## POM Driver Class

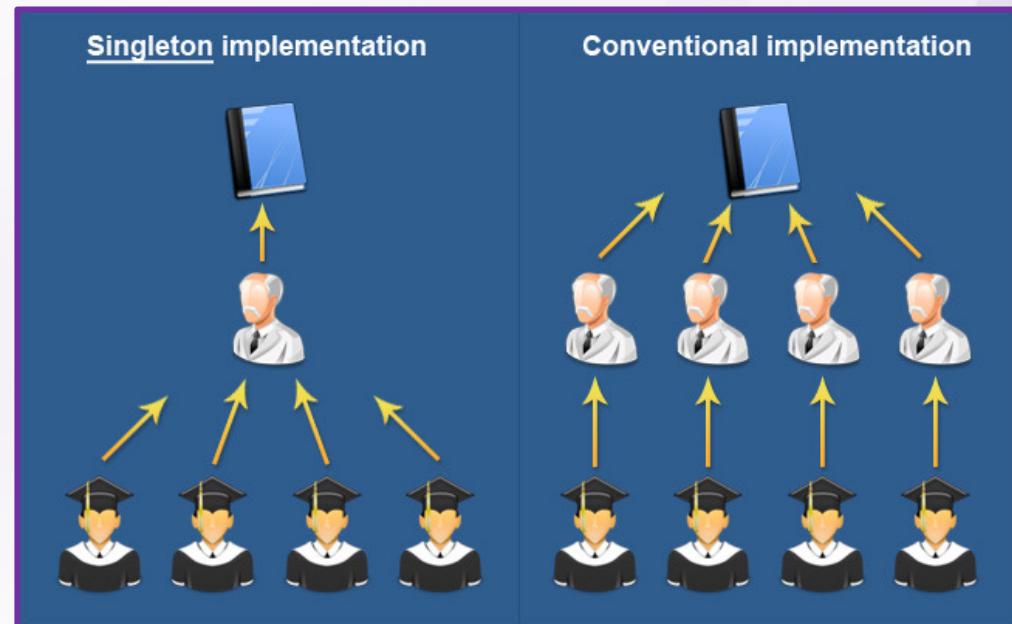
```
public class Driver {  
    private Driver(){  
    }  
    private static WebDriver driver;  
    public static WebDriver getDriver() {  
        if (driver == null) {  
            switch (ConfigReader.getProperty("browser")) {  
                case "chrome":  
                    WebDriverManager.chromedriver().setup();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

## Singleton Pattern (Tekli Kullanım)

Herhangi bir Java classından obje kullanımını sınırlayabiliriz. Buna Singleton pattern(tekli kullanım) denir.

Singleton pattern, class'ı tek bir instance ile kısıtlayan bir software dizayn kalıbıdır.

Proje genelinde farklı objeler oluşturulmadan tek bir instance gereksinimimiz olduğunda Singleton class kullanırız.



Sadece bir obje oluşturmak ve buna her ihtiyaç duyduğumuzda kullanmamız performans ve bellek kullanımı için de faydalıdır.

# TestNG

## Assertions

Test otomasyonunun temel amacı Test Case'lerde belirlenen her bir adimin test edilmesidir.

TestNG testleri yaparken bize daha fazla tercih imkani verir.

- Junit'deki gibi ilk failed olan assertion'da calismayi durdurmak
- Biz raporla diyene kadar tum testleri gerceklestirip, raporla deyince kac testin FAILED oldugunu raporlayip, sonra calismayi durdurmak.

```
public class C01 {  
  
    WebDriver driver;  
  
    @Test  
    public void amazonTest(){  
        WebDriverManager.chromedriver().setup();  
        driver=new ChromeDriver();  
        driver.get("https://www.amazon.com");  
        Assert.assertTrue(driver.getTitle().contains("amazon"));  
    }  
    @Test (dependsOnMethods = "amazonTest")  
    public void amazonAnasayfaTest(){  
  
        SoftAssert softAssert =new SoftAssert();  
        WebElement aramaKutusu=driver.findElement(By.id("twotabsearchtextbox"));  
        aramaKutusu.sendKeys( ...keysToSend: "java"+ Keys.ENTER);  
        WebElement ilkUrun=driver.findElement(By.xpath("//span[@class='a-size-base-plus a-color-base a-text-normal'][1]"));  
        softAssert.assertTrue(ilkUrun.getText().contains("java"), message: "ilk urun java icermiyor");  
        softAssert.assertAll();  
    }  
}
```

# TestNG

## Hard Assert

JUnit'te Öğrendiğimiz Assertion ile aynidir. TestNG'de soft assertion da oldugundan, ayristirmak icin bu isim kullanilmistir.

JUnit'ten bildigimiz uzere kullanabilecegimiz 3 cesit hard assertion turu vardir

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

Hard assertion herhangi bir assertion FAILED olursa, test method'nun calismasini durdurur ve kalan kodlari yürütmmez (stops execution).

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

# TestNG

## Soft Assert (Verification)

SoftAssert doğrulama (verification) olarak da bilinir.

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

Test method'unun istedigimiz bolumde  
yapılan tüm testleri raporlar

Eğer assertion'lardan FAILED olan varsa  
raporlama yapılan satırdan sonrasını  
calistirmaz.

SoftAssert class'ındaki method'lari kullanmak icin kullanabilmemiz için object  
oluşturmamız gereklidir.



# TestNG

## Soft Assert (Verification)

1

SoftAssert objesi olusturalim

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

2

Istedigimiz sayida verification'lari yapalim

```
// 3- sifre bosluk icermemeli
softAssert.assertFalse(sifre.contains(" "), message: "Sifre bosluk icermemeli");
// 4- uzunlugu en az 8 karakter olmali
softAssert.assertTrue(condition: sifre.length()>=8, message: "uzunluk en az 8 karakter olmali");
```

3

SoftAssert'in durumu raporlamasini isteyelim

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

## Soft Assert vs Hard Assert

### Ortak Ozellikler :

SoftAssert ve HardAssert method'ları TestNG'den gelmektedir.

Kullanma amaçları farklı olsa da method'lar aynıdır.

### Farklar :

Hard Assertion fail olursa test method'larının execute etmesi durur. Ve FAILED olarak tanımlanır.

Eğer softAssert FAIL olursa test method'ları execute etmeye devam eder. assertAll dedigimizde FAILED olan assertion varsa execution durur.



## Soft Assert (Verification)

1. testotomasyonu anasayfaya gidin
2. Title'in Test icerdigini doğrulayin
3. url'in <https://www.testotomasyonu.com> oldugunu doğrulayin
4. arama kutusunun kullanilabilir durumda oldugunu doğrulayin
5. belirlenmis aranacak kelimeyi aratip urun bulundugunu doğrulayin
6. Nutella aratip, urun bulunamadigini doğrulayin
7. Sayfayı kapatın

## Soft Assert (Verification)

1. "http://zero.webappsecurity.com/" Adresine gidin
2. webbappsecurity ana sayafaya gittiginizi doğrulayın
3. Sign in butonuna basin
4. Login kutusuna "username" yazın
5. Password kutusuna "password" yazın
6. Sign in tusuna basin
7. Back tusuna basin
8. Giriş yapılabildigini doğrulayın
9. Online banking menusunu tiklayın
10. Pay Bills sayfasına gidin
11. "Purchase Foreign Currency" tusuna basin
12. Currency dropdown menusunun erisilebilir olduğunu doğrulayın
13. "Currency" dropdown menusünden Eurozone'u seçin
14. "Eurozone (euro)" secildigini doğrulayın
15. Dropdown menude 16 option bulundugunu doğrulayın.
16. Dropdown menude "Canada (dollar)" bulunduğu doğrulayın
17. Sayfayı kapatın

1- Page Object Model : Testlerimizi daha kolay ve düzenli olarak hazırlamamız ve calistirmamız icin olusturulmus bir modeldir.

Framework icin uretilmis benzer modeler olmakla birlikte en guncel olan ve cok kullanilan model oldugu icin POM'i ogrendik

2- POM dosya yapisi :

- Pages : Test yapacagimiz web page'ler icin Pages package'in altında bir class olusturuyoruz. Bu class'larda mutlaka yapmamız gereken sey driver'i olusturdugumuz Driver clasından alip PageFactory.initElements ile ilk deger atamasi yapmaktir. Sonrasinda web sayfamizda kullanacagimiz WebElementlerin tamamini public olarak olusturmak ve @FindBy notasyonu ile locate etmektir. Eger istersek login gibi bazi adimlari yapacak methodlari da bu class'da olusturabiliriz.

Test clasimizdan Page sayfasindaki variable ve method'lara obje olusturup erisim saglariz.

- **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 güncelleleyebiliriz.

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

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

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

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

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

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

Ayrica her driver cagirdigimizda yeni driver 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



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-14

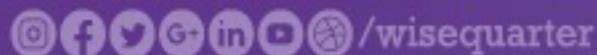
Xml File Kullanimi  
Parallel Testing  
TestNG Reports



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# TestNG

## Xml Files

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

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

Biz de framework'umuzdeki belirli testleri veya tüm testleri otomatik olarak calistirmak icin xml dosyaları kullanırız

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

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

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

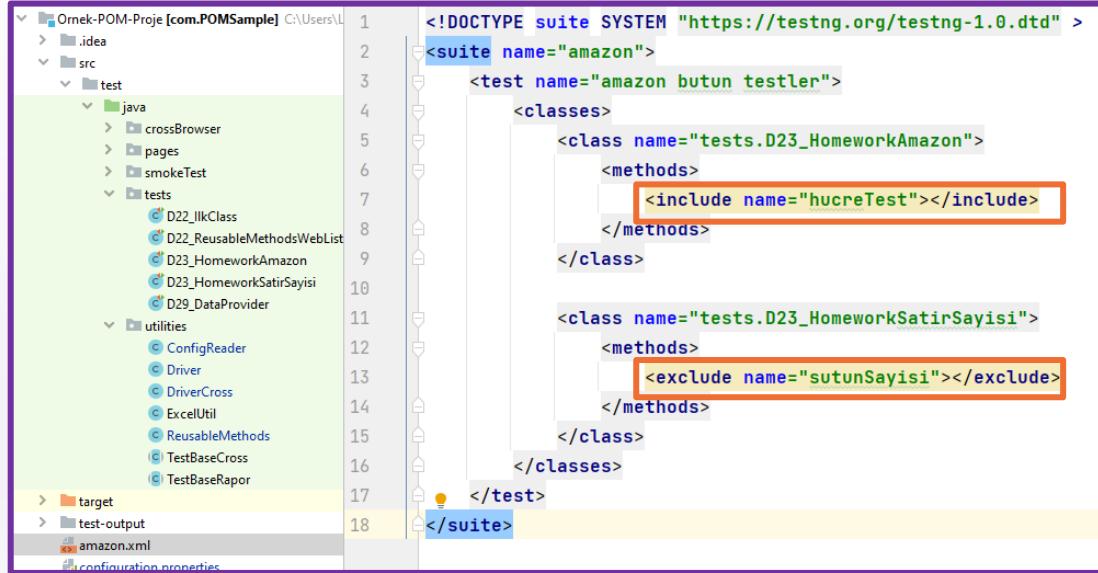
TestNG xml ile ilgili tüm dokumantasyon için :  
<https://testng.org/doc/documentation-main.html#testng-xml>

# TestNG

## Xml Files

Testng framework'de xml dosyasi kullanma nedenlerinden biri, belirli suitleri, testleri, package lari, classları veya method lari çalıştırmaktadır.

Belirli testleri, package lari, classları veya method'lari dahil edebilir (include) veya hariç (exclude) tutabiliriz. Bu da bize esneklik (flexibility) kazandırır.



The screenshot shows a Java project structure in an IDE. The project root is 'Ornek-POM-Proje [com.POMSample]'. It contains 'src' and 'test' directories. 'src' has 'java', 'resources', and 'main' sub-directories. 'test' has 'java', 'resources', and 'main' sub-directories. 'test/java' contains several test classes like D22\_IlkClass, D22\_ReusableMethodsWebList, etc. 'test/resources' contains utility classes like ConfigReader, Driver, etc. 'test/main' contains XML files like amazon.xml and configuration properties. To the right, the 'amazon.xml' file is open in a code editor. It defines a suite named 'amazon' containing a test named 'amazon butun testler'. This test includes classes: 'tests.D23\_HomeworkAmazon' which has methods including 'hucreTest'. Another class 'tests.D23\_HomeworkSatirSayisi' is defined with methods excluding 'sutunSayisi'. The XML code is highlighted with syntax coloring.

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

Sadece birkaç basit yapılandırma ile TestNG.xml dosyasını kullanarak belirli test senaryolarını yürütebiliriz.

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



# TestNG

## Xml Files

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

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

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

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



# TestNG

## Xml Files

### Istenen Package'lari Calistirma

Yeni bir xml dosyasi olusturalim :

belirliPackageCalistirma

Smoke Test package'indaki tum testleri calistiralim

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

# TestNG

## Xml Files

### Istenen Class'lari Calistirma

Yeni bir xml dosyasi olusturalim :

belirliClasslariCalistirma

<package> attribute yerine <classes> ve sonra <class> attribute kullanarak istediginiz class'lari calistirin

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

# TestNG

## Xml Files

### Istenen Method'ları Çalıştırma

Yeni bir xml dosyası oluşturalım : belirliMethodlarıCalistirma

<classes> attribute altında <class> ve <methods> attribute'lerini ve içinde <include>, <exclude> attribute'lerini kullanalım

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

## Istenen Gruplari Calistirma

Yeni bir xml dosyasi olusturalim : belirliGruplariCalistirma

Group calistirmak icin hem grup adini tanimlamak hem de nerede arayacagimizi belirtmek zorundayiz

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



# TestNG

## Xml Files

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

## 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

## TestNG Paralel Testing

Coklu calistirma, Parallel calistirma ve Cross Browser calistirma farkli farkli islemlerdir.

Mesela 5 testi sirali olarak ama topluca calistirirsak → sirali coklu calistirma

5 testi ayni anda calismaya baslayan iki driver'la calistirirsam

→ parallel calistirma

5 testi sirali olarak calistirip, ilk ucunu chrome, son ikisini firefox'da calistirirsam

→ sirali cross browser

5 testin ucunu chrome, ikisini firefox ile calistirip, chrome ve firefox'u ayni zamanda calistirirsam

→ parallel cross browsing test olur



## TestNG Parallel Testing

### Classes

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



# TestNG

## TestNG Parallel Testing

### Packages

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

# 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>4.0.9</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

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

### Testimize rapor oluşturma adımları

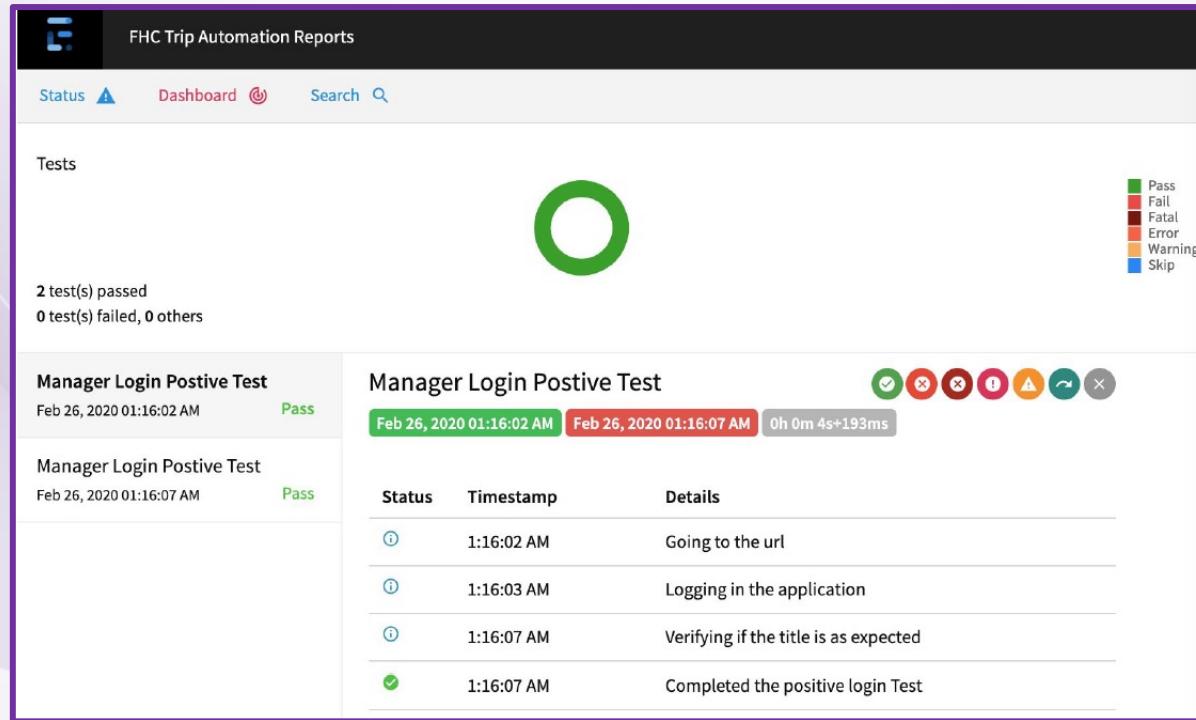
- 1- Test class'ini extends ile TestBaseRapor Class'ına child yapalım
- 2- *extentTest=extentReports.createTest("Test ismi", "Tanim");* rapor olusturalım
- 3- Gerekli/istedigimiz satirlara extentTest.info("Acıklama") ekleyelim
- 4- Assert olan satırda açıklamayı extentTest.pass ile yapabiliriz

# 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 🔍

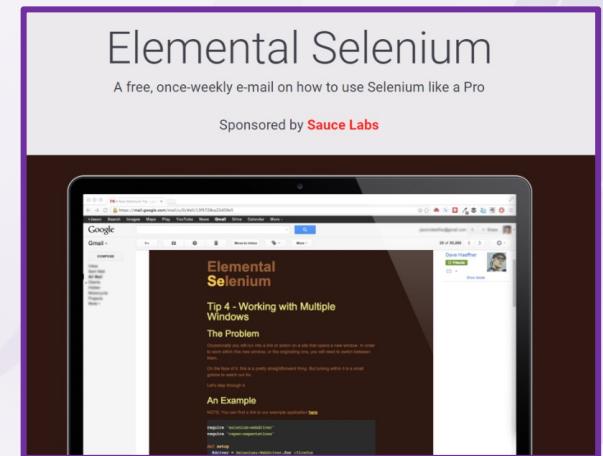
Tests

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 01:16:02 AM 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



Elemental Selenium

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

Sponsored by Sauce Labs

Tip 4 - Working with Multiple Windows

The Problem

Occasionally you will run into the situation where a file or action on a file that opens a new window. In order to interact with this file or action, or to register one, you will need to switch context.

For example, let's say you are trying to log in to a website. You click on the "Log In" button and a new window opens up asking for your password. You will need to switch contexts to interact with this new window.

An Example

NOTE: You can find a link in our example application here



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-15

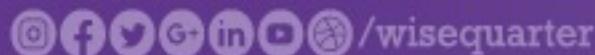
@DataProvider  
Cross Browser Tests



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)



# TestNG

## @DataProvider

@DataProvider bir TestNG annotation'ıdır, dolayısıyla sadece TestNG ile kullanılır. DDT (Data Driven Test) konseptinde teste veri sağlamak için kullanılır.

```
public class C01_DataProvider {
    @DataProvider
    public static Object[][] AranacakKelimeProvider() {

        Object[][] arananKelimeler= {{ "Nutella"}, {"Java"}, {"Apple"}, {"Samsung"}, {"TV"}};
        return arananKelimeler;
    }

    @Test(dataProvider = "AranacakKelimeProvider")
    public void aramaTesti(String aranacakKelime){
        // Amazon anasayfaya gidin
        Driver.getDriver().get(ConfigReader.getProperty("amazonUrl"));
        // Nutella, Java, Apple, Samsung, TV için arama yapın
        AmazonPage amazonPage=new AmazonPage();
        amazonPage.aramaKutusu.sendKeys( ...keysToSend: aranacakKelime + Keys.ENTER);
        // Arama sonuçlarının aranın kelime içerdigini test edin
        String actualSonucYazisi=amazonPage.aramaSonucElementi.getText();
        ReusableMethods.bekle( saniye: 3);
        Assert.assertTrue(actualSonucYazisi.contains(aranacakKelime));
    }
}
```

Cucumber'da gorecegimiz Scenario Outline ile ayni islev sahiptir



# TestNG

## @DataProvider

The screenshot shows a web browser window with the URL 'testotomasyonu.com' in the address bar. The page title is 'YouTube testotomasyonu'. A dark blue header bar contains the text 'Sign Up And 15% special discount on your first order' and a 'Sign Up Now' button. Below the header is the 'TEST OTOMASYON' logo, which includes a checkmark icon and the text 'TEST OTOMASYON'. To the right of the logo are a search bar with the placeholder 'Search Product,Category...', a 'Select Category' dropdown menu, and a teal-colored search button with a magnifying glass icon. At the bottom of the page, there is a horizontal navigation bar with categories: Electronics, Men Fashion, Women Fashion, Shoes, Furniture, Travel, and Kids Wear.

1. testotomasyonu anasayfaya gidin
2. Asagida verilen urunler icin arama yapin,
3. Arama sonucunda her bir urun icin sonuc bulunabildigini test edin
4. Aranacak urunler : phone, java, dress, apple, Nutella, cokokrem, baby

# TestNG

## @DataProvider

**Login Now**

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

Don't have an Account? [Sign Up.](#)

1. testotomasyonu anasayfaya gidin
2. Account linkine tiklayin
3. Asagida verilen kullanici adi ve sifreleri ile giris yapilamadigini test edin
4. banu@gmail.com 898989  
sedat@yahoo.com 989887  
orkong@mail.com 122334  
fatih@hotmail.com 454545  
arzu@senmail.com 676767  
mehmet@mynet.com 878987

# 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. Herbir adımı ezbere bilmek zorunda değiliz ama mantığı anlamak ve bunu sözlü olarak ifade edebilmek zorundayız



## Cross Browser Testing

1. Crossbrowser testi icin yeni bir driver class'i olusturacagiz : DriverCross
  - getDriver methoduna parametre 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 .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

## 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 düzenleniyoruz

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>

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>
```

### c) Testng Kutuphanesi

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

### d) Java Faker Kutuphanesi

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



# TestNG

## Ornek Proje Olusturma

7) Pom Dosyasini oluşturma işlemimiz bitti. Kutuphaneler kırmızı renkte olabilir. Sağ tarafta Maven yazan sekmede Reload oklarina tiklayip beklediğimiz de hata gitmiş oluyor

8) Kullanicinin gordugu arayuzde test ederiz. (UI)

Kullanicagimiz paketleri uygun isimlerde test-Java bolumun içerisinde oluşturuyoruz

9) Java ya sag tiklariz new package – com (paketin ismi ) yazariz

10) com package'ina sag tiklariz – new package – abc (paketin ismi) yazariz

11) artik projemiz com. abc

Bu package'in altina frameworkumuzun package'larini yolosturuyoruz. Bunlar

A-pages

B-smokeTest

C-tests

D-utilities

## 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

İsmi configuration.properties yaziyoruz

Bu dosyanin içine Data'larimizi key=value seklinde yaziyoruz

kr\_url= <https://www.qualitydemy.com/>

kr\_valid\_username=user\_1106147@login.com

kr\_valid\_password=31488081

### 14) ConfigReader Class'l olusturma

utilities package inin altinda ConfigReader Classi olusturuyoruz. Bu class configuration.properties deki dosyalarimizi okumak için bir araci

## Ornek Proje Olusturma

## 15) ConfigReader Classinda :

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

Bu objeyi sadece bu class ta kullanacagim icin private yapmamiz önerilir

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

```
public class ConfigReader {  
  
    static Properties properties;  
  
    static {  
        String dosyaYolu="configuration.properties";  
  
        try {  
            FileInputStream fis = new FileInputStream(dosyaYolu);  
            properties=new Properties();  
            properties.load(fis);  
  
        } catch (IOException e) {  
            throw new RuntimeException(e);  
        }  
    }  
  
    public static String getProperty(String key){  
  
        return properties.getProperty(key);  
    }  
}
```

## Ornek Proje Olusturma

## 16) Driver class'ini düzenleniyoruz

Singleton class : object olusturulmasi kontrol altina alınan (genelde izin verilmeyen) classdir. Bunun icin baska classlarda Driver clasindan obje uretmemizi saglayan default constructor'i gorunur sekilde yapip 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 WebDriver driver;  
    public static WebDriver getDriver(){  
        if(driver==null) { // method ilk cagrildiginda driver degeri atanma  
            // sonraki calistirmalarda degeri atanmis oldugu  
            String browser= ConfigReader.getProperty("browser");  
            switch (browser){  
                case "chrome" :  
                    WebDriverManager.chromedriver().setup();  
                    driver = new ChromeDriver();  
                    break;  
                case "firefox":  
                    WebDriverManager.firefoxdriver().setup();  
                    driver=new FirefoxDriver();  
                    break;  
                case "safari" :  
                    WebDriverManager.safaridriver().setup();  
                    driver=new SafariDriver();  
                    break;  
                default:  
                    WebDriverManager.chromedriver().setup();  
                    driver = new ChromeDriver();  
            } }  
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(15));  
        driver.manage().window().maximize();  
        return driver;  
    }
```

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

- b) Ardinda Locate islemlerimizin tamamini yaziyoruz bu sayfaya

```
@FindBy(xpath = "//a[text()='Log in']")  
public WebElement loginLinki;  
  
@FindBy(xpath = "//input[@id='login-email']")  
public WebElement emailKutusu;  
  
@FindBy(xpath = "//input[@id='login-password']")  
public WebElement passwordKutusu;  
  
@FindBy(xpath = "//button[text()='Login']")  
public WebElement loginButonu;  
  
@FindBy(linkText = "My courses")  
public WebElement coursesLinki;
```

## 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

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.

Su ana kadar JUnit ve TestNG ile HTML elementleri otomasyon ile nasıl kullanabilecegimizi gorduk, Cucumber derslerinde framework'e odaklanacagiz.

```
Feature: US1006 Dogru kullanici adi ve password ile pozitif login testi

Scenario: TC12 Kullanici mycoursedemy sitesine giris yapabilmeli

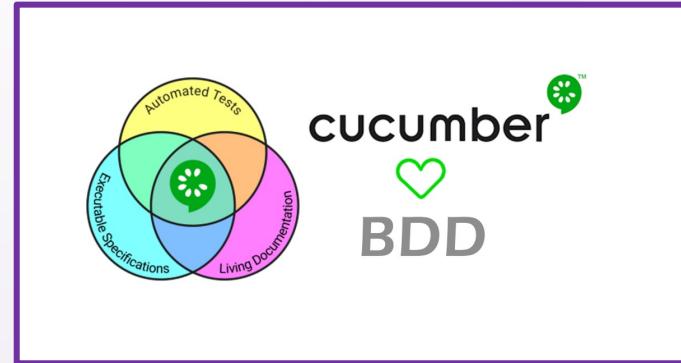
    Given Kullanici "myUrl" anasayfaya gider
    Then myCourse anasayfa login linkine tiklar
    And myCourse kullanici adi olarak "myGecerliEmail" girer
    And myCourse password olarak "myGecerliPassword" girer
    And myCourse login butonuna basar
    Then myCourse giris yapabildigini test eder
    And Sayfayı kapatır
```

Agile methodolojisinde, insanlar uygulamanın işlevsellliğini geliştirmek için birlikte çalışmak zorundadır. Cucumber development team'deki herkesin test case'leri anlayabilmesini saglar.

BDD (behaviour driven development) (Davranış güdümlü geliştirme)- ilk olarak behavior(davranis) veya functionalitileri yazıyorsunuz (Epic=Feature, Story, AC, etc), daha sonra development and testing baslıyor.

BDD'de behaviour'lar başarısız olduğunda kod başarısız olur..

Anlasılabilir Gherkin Language nedeniyle BDD development için Cucumber harika bir uygulamadır.



Gherkin: Projede her bir davranış için .feature uzantılı bir Gherkin dosyası oluşturulur. Bu feature dosyasına ilgili özelliğin farklı durumlardaki davranışları tanımlanır.

**Given** anahtar kelimesi ile ön koşul yani başlangıç durumu tanımlanır,

**When, And** anahtar kelimeleri ile olayı

**Then** anahtar kelimesi ile de sonuç tanımlanır.

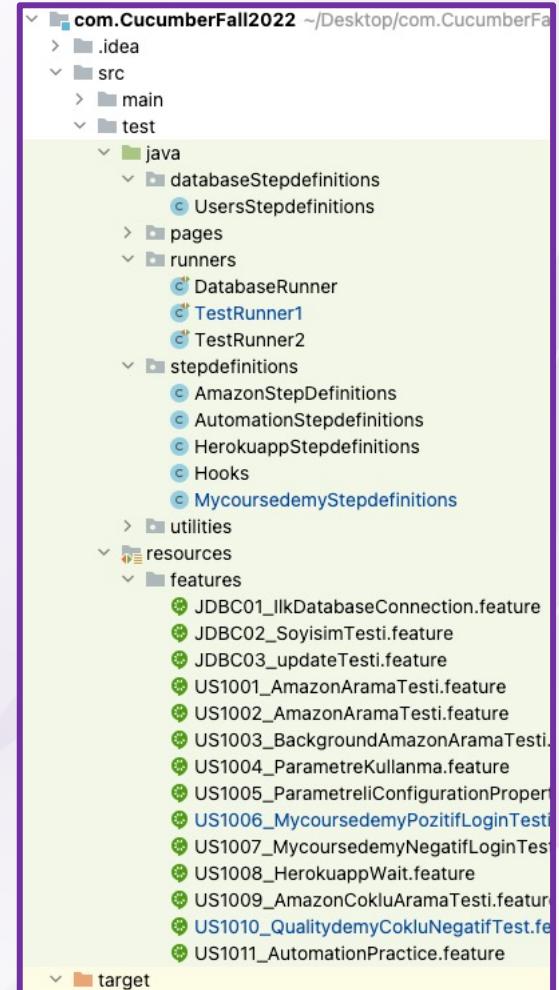
**Given-When-And-Then** adımları sadece okuyanlar için farkeder Java için hepsi birdir.

Cucumber (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





Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-16

### Cucumber



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



1. Create Project: File => New => Project => Select maven => click next
2. Name: team105Cucumber => finish
3. Add Dependencies =>
  - Selenium-java,
  - webdrivermanager,
  - cucumber java,
  - cucumber junit
4. Click Maven => click “Enable auto-reload after any changes” (Reload)

## Proje Olusturma

5. 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)

6- Utilities paketi altinda Driver ve ConfigReader Class'larini olusturalim

7- Projeye sag click yapip configuration.properties dosyasi olusturalim

8- test paketi altinda yeni bir klasor olusturalim : resources

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

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

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

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



## Ilk Cucumber Testi

Yeni bir feature file olusturun ve 3 farkli Scenario ile asagidaki testleri yapin

Given kullanici TestOtomasyon sayfasina gider

And phone icin arama yapar

Then aradigi urunun bulundugunu test eder

Given kullanici TestOtomasyon sayfasina gider

And shoes icin arama yapar

Then aradigi urunun bulundugunu test eder

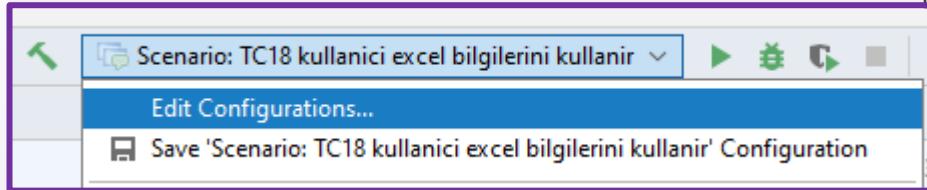
Given kullanici TestOtomasyon sayfasina gider

And dress icin arama yapar

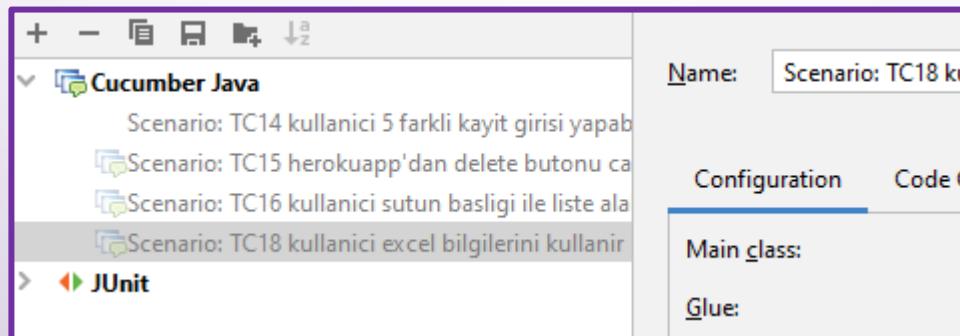
Then aradigi urunun bulundugunu test eder

# Cucumber

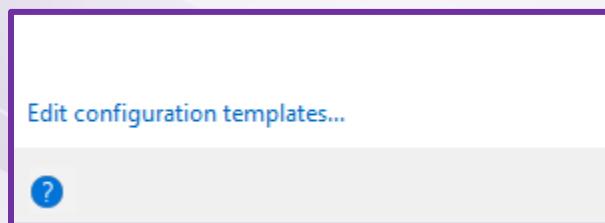
## Build Ayarları



1- Edit Configurations secin



2- Sol bolumdeki CucumberJava ve altindakileri silin.



3- Alt kisimdan Edit configuration menusunu acin

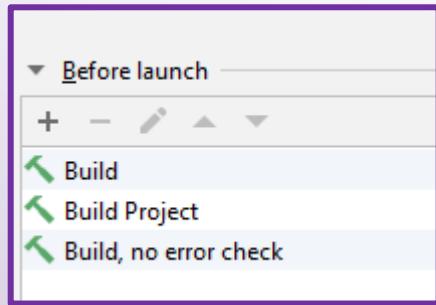


# Cucumber

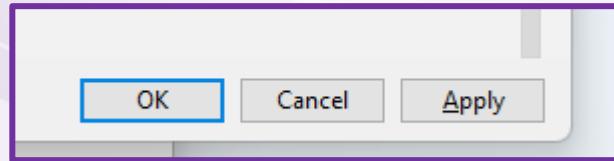
## Build Ayarları



4- Listededen Cucumber Java'yi secin



5- Altaki Before Launch bolumunde 3 maddenin oldugunu kontrol edin, yoksa ekleyin



6- degisiklikleri kayit etmek icin Apply'a basin



# Cucumber

## Background

Feature: US1001 amazon arama

@amazon @nutella

Scenario: TC01 amazon nutella arama

```
When kullanıcı amazon sayfasına gider
And nutella için arama yapar
Then sonucun nutella icerdigini test eder
And sayfayı kapatır
```

@amazon @java

Scenario: TC02 amazon java arama

```
When kullanıcı amazon sayfasına gider
And java için arama yapar
Then sonucun java icerdigini test eder
And sayfayı kapatır
```

@amazon @ipad

Scenario: TC03 amazon ipad arama

```
When kullanıcı amazon sayfasına gider
And ipad için arama yapar
Then sonucun ipad icerdigini test eder
And sayfayı kapatır
```

Farklı senaryoların başında ortak adımlarımız varsa:

1. Feature file in basına Background oluşturun.
2. Bu ortak adımları Background altına yazın.

Feature: US1002 amazon background ile arama

Background: amazon sayfasına gitme

When kullanıcı amazon sayfasına gider

Scenario: TC04 amazon nutella arama

And nutella için arama yapar

Then sonucun nutella icerdigini test eder

And sayfayı kapatır

@wip

Scenario: TC05 amazon java arama

And java için arama yapar

Then sonucun java icerdigini test eder

And sayfayı kapatır

Scenario: TC06 amazon ipad arama

And ipad için arama yapar

Then sonucun ipad icerdigini test eder

And sayfayı kapatır

3. Background, Feature file'daki her Scenario'dan önce çalışır

4. Duplication olmadigindan emin olun.  
Background un altındaki adımı yazdıktan sonra senaryolardan silin.

# Cucumber

## Runner Class

Cucumber'da Runner class'i bos bir class'tir. Runner class'ini farkli kilan ve TestNG'deki xml dosyalari gibi calismasini saglayan 2 adet notasyon mevcuttur

```
@RunWith(Cucumber.class)
@CucumberOptions(
    features = "src/test/resources/features" ,
    glue = "stepdefinitions" ,
    tags= "@wip",
    dryRun = false
)

public class Runner {
```

`@RunWith` notasyonu projemize cucumber junit dependency eklememizin sebebidir bu sayede runner class'larimiz cucumber ile calisir

`@CucumberOptions` notasyonu ile istedigimiz ozellikleri Runner class'ina ekleyebiliriz. Raporlama gibi ekstra option'lari da ileride ekleyecegiz

Ancak oncelikli gorevi features dosyalari ile stepdefinitions'da bulunan Java method'larini ilisklendirmektir

`dryRun` : iki deger alabilir

`true` : secilirse, verilen tag ile işaretli olan Feature veya Scenario'daki eksik stepdefinitions'lari bulup ilgili method'lari olusturur. Hic bir sekilde testimizi calistirmaz. Eksik adim yoksa test passed olarak işaretler

`false` : secilirse, verilen tag ile işaretlenen Feature veya Scenario'lari calistirir

`tags` : features classoru içerisinde yazilan tag(lari) aratip buldugu tum feature veya scenario'lari calistirir



# Cucumber

## @tags

Tag'ları onceden belirledigimiz senaryoları (scenario) çalıştırılmak için kullanırız.

Tag'ları senaryolarımızı gruplandırmak için de kullanabiliriz (smoke test, regression test, vs.)

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

    dryRun= false           // dryRun=true dedigimizde test
```

**Feature:** US1001 amazon page search

@amazon @search @apple

**Scenario:** TC01 amazon arama testi

```
Given kullanıcı amazon sayfasına gider
And "apple" için arama yapar
Then sonuçların "apple" içerdigini test eder
Then sayfayı kapatır
```



# Cucumber

## Feature File'i Parametre ile Kullanma

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

```
Scenario: TC01 iphone aramasi testi
Given kullanici amazon sayfasina gider
And iphone icin arama yapar
Then sonuclarin iphone icerdigini test eder
```

```
Scenario: TC02 tea pot aramasi testi
Given kullanici amazon sayfasina gider
And tea pot icin arama yapar
Then sonuclarin tea pot icerdigini test eder
```

```
Scenario: TC03 flower aramasi testi
Given kullanici amazon sayfasina gider
And flower icin arama yapar
Then sonuclarin flower icerdigini test eder
Then sayfayı kapatır
```

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

```
@rapor2
```

```
Scenario: TC07 istenen kelimenin oldugunu test etme
Given kullanici "amazonUrl" sayfasina gider
And "iphone" icin arama yapar
Then sonucun "iphone" icerdigini test eder
And sayfayı kapatır
```

Kodlarımızı parametreli olarak yazdıktan sonra sadece " " içindeki değeri değiştirerek test datalarını feature file dan kontrol edebiliriz.

Kodlarımızı parametreli olarak yazmak framework'u daha dinamik hale getirir(kodumuz artık hard coded degildir diyebiliriz).



# Cucumber

## Parametre ile Pozitif login testi

Login Now

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

Don't have an Account? [Sign Up.](#)

```
Given kullanici "toUrl" anasayfaya gider
Then account butonuna basar
And email olarak "toGecerliEmail" girer
And password olarak "toGecerliPassword" girer
Then signIn butonuna basar
And basarili giris yapilabildigini test eder
And 3 saniye bekler
And sayfayı kapatır
```

# Cucumber

## Parametre ile Negatif login testi

Testotomasyonu giriş sayfasında yanlış bilgilerle giriş yapılamayacağını 3 farklı Scenario ile test edin

- 1- Gecersiz email – gecerli password
- 2- Gecerli email – gecersiz password
- 3- Gecersiz email – gecersiz password

```
Given kullanıcı testotomasyonu anasayfaya gider
Then account butonuna basar
And email olarak "toGecersizEmail" girer
And password olarak "toGecerliPassword" girer
Then signIn butonuna basar
And sisteme giriş yapamadığını test eder
And 3 saniye bekler
And sayfayı kapatır
```

**Login Now**

Enter your email address and password to access Test Otomasyon account

Email Address \*

Password \*

**Sign In**

Don't have an Account? [Sign Up.](#)

# Cucumber

## Facebook Kayıt Testi

```
Given kullanici "faceUrl" anasayfaya gider
Then facebook cookies kabul eder
And facebook'da yeni hesap olustur butonuna basar
And 1 saniye bekler
When Faker class'indan bilgilerle facebook kayıt
formunu doldurur
Then 2 saniye bekler
And facebook kayıt ol tusuna basar
Then 2 saniye bekler
And sayfayı kapatır
```

**Kaydol**

Hızlı ve kolaydır.

Adın  Soyadın

Cep telefonu numarası veya e-posta

Yeni şifre

Doğum Tarihi  Ara  2023

Cinsiyet  Kadın  Erkek  Özel

Hizmetimizi kullanan kişiler senin iletişim bilgilerini Facebook'a yüklemiş olabilir. [Daha fazla bilgi al.](#)

Kaydol



# Cucumber

## Saucedemo Alisveris Testi

QTY	Description
1	<b>Sauce Labs Backpack</b> carry.allTheThings() with the sleek, streamlined Sly Pack that melds  \$29.99

Given kullanici "sauceUrl" anasayfaya gider

And 2 saniye bekler

Then saucedemo username kutusuna "standard\_user" yazar

And saucedemo password kutusuna "secret\_sauce" yazar

And 2 saniye bekler

Then saucedemo login tusuna basar

And ilk urunun ismini kaydeder ve bu urunun sayfasina gider

When saucedemo add to Cart butonuna basar

Then saucedemo alisveris sepetine tiklar

And 2 saniye bekler

And sectigi urunun basarili olarak sepete eklendigini test eder

And 2 saniye bekler

And sayfayı kapatir



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-17

### Cucumber



The future at your fingertips

+1 912 888 1630

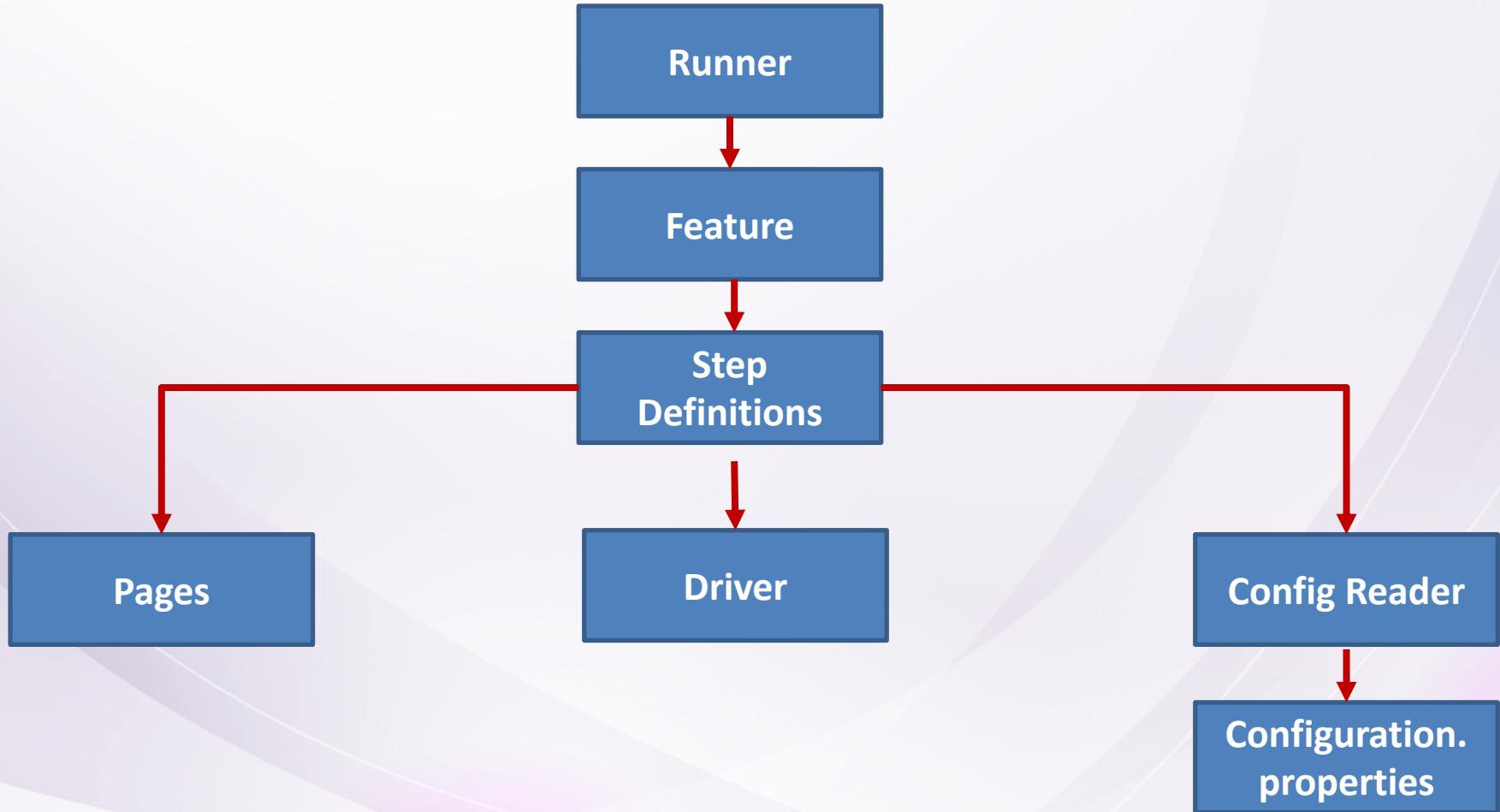
[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



# Cucumber

Calisma Semasi





# Cucumber

## Herokuapp add/remove Elements Testi

the-internet.herokuapp.com/add\_remove\_elements/

YouTube testotomasyonu

### Add/Remove Elements

Add Element

Given kullanici "heroUrl" anasayfaya gider  
When kullanici Add Element butonuna basar  
And 2 saniye bekler  
And Delete butonu'nun gorunur oldugunu test eder  
Then Delete tusuna basar  
And Add Remove Elements yazisinin gorunur oldugunu test eder  
And 2 saniye bekler  
And sayfayı kapatır



# Cucumber

## Scenario Outline

```
Given kullanici "toUrl" anasayfaya gider
Then arama kutusuna "<aranacakKelimeler>" yazip ENTER tusuna basar
And arama sonucunda urun bulunabildigini test eder
Then 1 saniye bekler
And sayfayı kapatır
```

Examples:

aranacakKelimeler
iphone
java
samsung
apple
nutella

Scenario Outline: aynı teste birden fazla datayı kullanmamızı sağlar

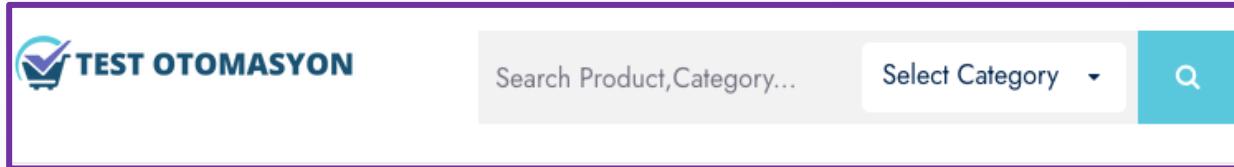
Bir liste kullanmak istediğimiz değeri “<value>” şeklinde yazarız

Daha sonra testin sonuna Examples: yazıp ilk satır olarak | value| yazarız ve altına kullanmak istediğimiz değerleri ekleriz. (| phone|, | java| ... gibi)



# Cucumber

## Testotomasyonu Coklu Arama Testi



Given kullanici "toUrl" anasayfaya gider  
Then arama kutusuna "<aranacakKelimeler>" yazip ENTER tusuna basar  
And arama sonucunda urun bulunabildigini test eder  
Then 1 saniye bekler  
And sayfayı kapatır

Examples:

| aranacakKelimeler  
| phone  
| java  
| samsung  
| apple  
| nutella



```
Given kullanici "toUrl" anasayfaya gider
Then account butonuna basar
When email olarak "<verilenEmail>" girer
And password olarak "<verilenPassword>" girer
Then signIn butonuna basar
And sisteme giris yapamadigini test eder
And 1 saniye bekler
Then sayfayı kapatır
```

Examples:

verilenEmail	verilenPassword
toGecerliEmail	toGecersizPassword
toGecersizEmail	toGecerliPassword
toGecersizEmail	toGecersizPassword
toGecersizEmail2	toGecersizPassword2
toGecersizEmail3	toGecersizPassword3



```
Given kullanici "toUrl" anasayfaya gider
Then account butonuna basar
When email olarak listede verilen "<verilenEmail>" girer
And password olarak listede verilen "<verilenPassword>" girer
Then signIn butonuna basar
And sisteme giris yapamadigini test eder
And 1 saniye bekler
Then sayfayı kapatır
```

Examples:

<i>verilenEmail</i>	<i>verilenPassword</i>
esra@fmail.com	1234555
gamze@gmail.com	mesela123
zehra@zmail.com	asdf4321
ayca@mail.com	6543241tre65
<u>gulnur@gmail.com</u>	123ert678



Given kullanici "toUrl" anasayfaya gider

Then urun excelindeki "<istenenSatir>" daki urunun min. miktarini ve urun ismini kaydeder

And urun ismini testotomasyonu sayfasinda arar ve sonuc sayisini kaydeder

And bulunan urun sayisinin "<istenenSatir>" da verilen min. miktardan fazla oldugunu test eder

And sayfayı kapatır

Examples:

|istenenSatir|

|2|  
|3|  
|4|  
|5|  
|6|  
|7|  
|8|  
|9|

# Cucumber

## Scenario Outline Homework

Name	Position	Office	Start date	Salary
Airi Satou	Accountant	Tokyo	2008-11-28	\$162,700
Angelica Ramos	Chief Executive Officer (CEO)	London	2009-10-09	\$1,200,000

New

Edit

Delete

Search:

Farkli datalarla tabloya 5 kayit ekleyip, asagidaki testi yapin.

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

# Cucumber

## Homework

### Add/Remove Elements

Add Element

Powered by Elemental Selenium

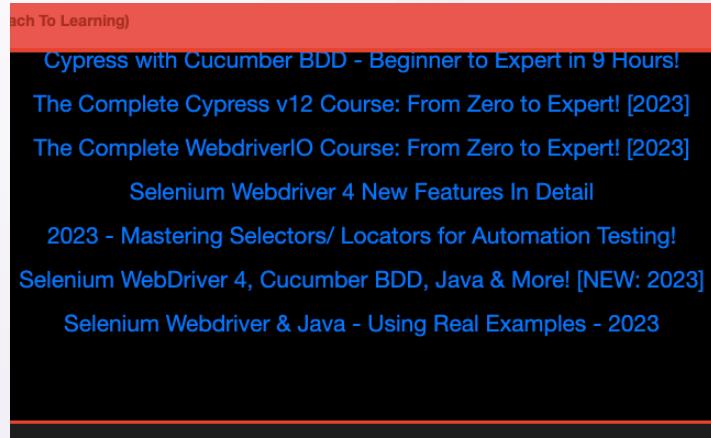
[https://the-internet.herokuapp.com/add\\_remove\\_elements/](https://the-internet.herokuapp.com/add_remove_elements/) adresine gidin

- 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

# Cucumber

## Homework

- 1."<https://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



The screenshot shows a search results page for 'Cypress with Cucumber BDD'. The results include:

- Cypress with Cucumber BDD - Beginner to Expert in 9 Hours!
- The Complete Cypress v12 Course: From Zero to Expert! [2023]
- The Complete WebDriver Course: From Zero to Expert! [2023]
- Selenium Webdriver 4 New Features In Detail
- 2023 - Mastering Selectors/ Locators for Automation Testing!
- Selenium WebDriver 4, Cucumber BDD, Java & More! [NEW: 2023]
- Selenium Webdriver & Java - Using Real Examples - 2023

**CONTACT US**

Want to practice your skills? BDD also referred to as 'Behaviour Driven Development' is a great way to test and simulate different user interactions. Click the button to access the challenge to simulate user(s) inputting different types of data into the form using an email address in the incorrect 'Mandatory' format?

**LOGIN PORTAL**



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-18

### Cucumber



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter

# Cucumber

## Scenario Outline Homework

All	Group A	Group B	Group M	Group T
Company	Group	Prev Close (Rs)		
Ujjivan Financial	A	963.3		
Quess Corp	A	438.4		
Power Finance Co	A	162.1		
Bharat Petroleum	A	928.7		
Hind. Petrol	A	215.6		
Varun Beverages Ltd.	A	628.8		
NCC	A	84.3		
Blue Dart Express	A	91		
Kansai Nerolac Paint	A	492.4		
Max Financial Servic	A	59.9		
IDFC L	A	461.5		
Dewan Housing	A	951.2		

<http://demo.guru99.com/test/web-table-element.php> sayfasina gidip asagidaki gorevi tamamlayin

Given user web sayfasinda

And “Istelenen Baslik”, sutunundaki tum degerleri yazdirir

# Cucumber

## HTML Reports

Cucumber raporları, şirketlerin Cucumber kullanmasının ana nedenlerinden biridir.

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

Html rapor almak için runner classına eklenti(plugin) eklememiz yeterlidir.

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

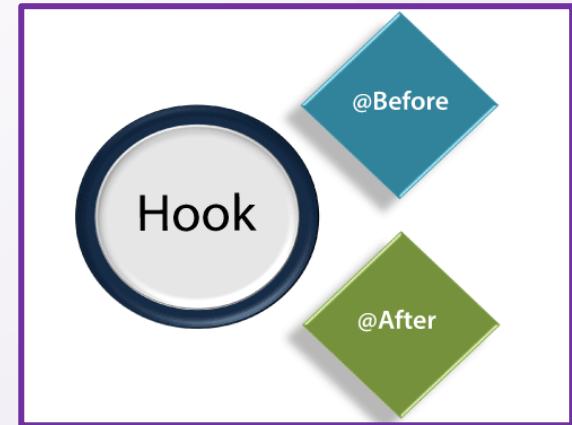
# Cucumber

## Hooks ve Screenshot 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();  
}
```

# Cucumber

## 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.

Ayrica maven-cucumber-reporting plugin yuklemek istersek pom.xml'e plugin ekleyebiliriz

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

# Cucumber

## Paralel Testing

Paralel testing: Birden fazla browser'in es zamanlı çalıştırılmasıdır.

Cucumber ile parallel test çalıştırmak testing'ye göre daha zordur.

Ancak raporlama ihtiyacı varsa TestNG'deki karmaşık raporlama prosesleri yerine cucumber'da parallel çalıştırmanın zorluguna katlanmak tercih edilebilir.

Paralel çalıştırılmak için birden fazla Runner Class'ına ihtiyacımız var

Cucumber'da, testleri paralel olarak çalıştırılabilmek için bazı eklentilere(plugin) ve yapılandırmalara da ihtiyacımız vardır.

Pom da yaptığımız ayarlamalardan sonra testleri Runner'dan değil Terminalden “mvn clean verify” kodunu yazarak çalıştıracağız

# Cucumber

## Paralel Testing

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

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



# 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. Gelismis rapor icin gereklidir

# Cucumber

## Paralel Testing

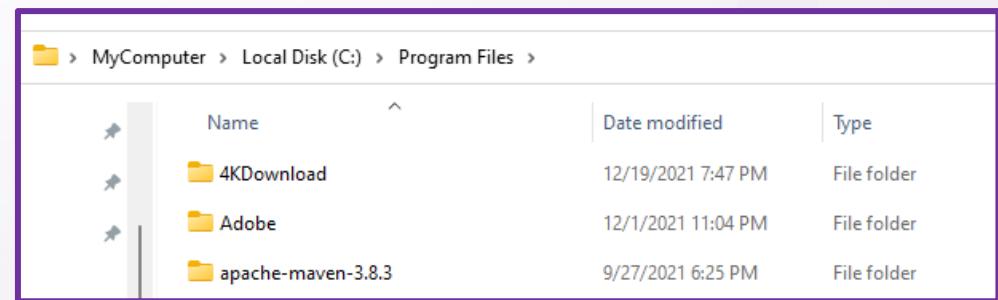
```
<plugin>
  <groupId>net.masterthought</groupId>
  <artifactId>maven-cucumber-reporting</artifactId>
  <version>5.0.0</version>
  <executions>
    <execution>
      <id>execution</id>
      <phase>verify</phase>
      <goals>
        <goal>generate</goal>
      </goals>
      <configuration>
        <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>
```

# Cucumber

## Maven Environment Ayarı

Bilgisayarına maven indirmeyenler terminal'den mvn clean verify ile calistiramazlar.

1- Bilgisayarınızda program Files clasorunde apache-maven olup olmadığını kontrol edin



2- maven.apache.org sitesinden bilgisayarınıza Binary zip archive dosyalarını indirin



Link	Checksums	Signature	
Binary tar.gz archive	apache-maven-3.8.6-bin.tar.gz	apache-maven-3.8.6-bin.tar.gz.sha512	apache-maven-3.8.6-bin.tar.gz.asc
Binary zip archive	apache-maven-3.8.6-bin.zip	apache-maven-3.8.6-bin.zip.sha512	apache-maven-3.8.6-bin.zip.asc
Source tar.gz archive	apache-maven-3.8.6-src.tar.gz	apache-maven-3.8.6-src.tar.gz.sha512	apache-maven-3.8.6-src.tar.gz.asc
Source zip archive	apache-maven-3.8.6-src.zip	apache-maven-3.8.6-src.zip.sha512	apache-maven-3.8.6-src.zip.asc

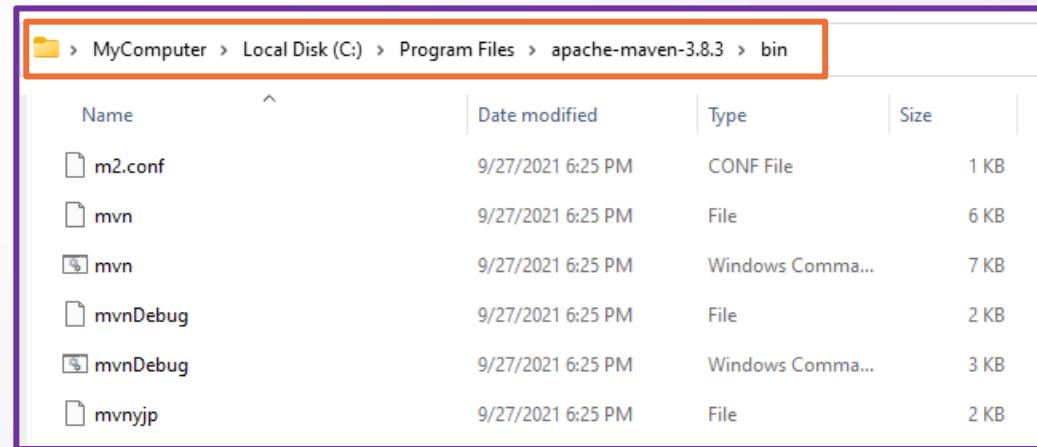
# Cucumber

## Maven Environment Ayarı

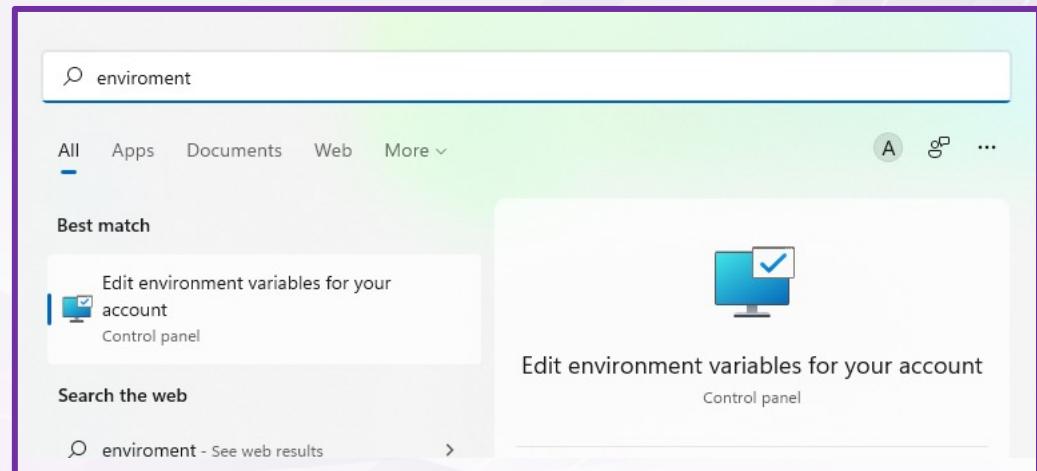
3- zip seklinde gelen dosyayı bir klasöre çıkartıp, bilgisayarınızda Program Files klasörüne kopyalayın

4- Maven klasörünün içerisinde bin klasörüne girip klasörün yolunu kopyalayın

5- Bilgisayarınızın arama kısmında environment yazıp Edit environment variables for your account menusune girin



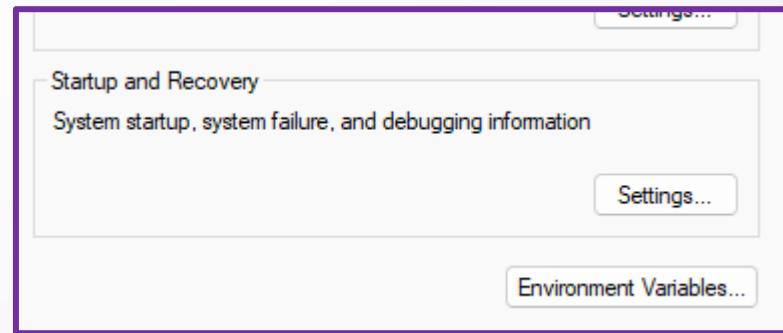
Name	Date modified	Type	Size
m2.conf	9/27/2021 6:25 PM	CONF File	1 KB
mvn	9/27/2021 6:25 PM	File	6 KB
mvn	9/27/2021 6:25 PM	Windows Comma...	7 KB
mvnDebug	9/27/2021 6:25 PM	File	2 KB
mvnDebug	9/27/2021 6:25 PM	Windows Comma...	3 KB
mvnyjp	9/27/2021 6:25 PM	File	2 KB



# Cucumber

## Maven Environment Ayarı

6- Acilan menuden Environment Variables'i tiklayin



7- Sistem variables bolumunde New butonuna basin ve kopyaladiginiz dosya yolunu buraya ekleyin.

System variables	
Variable	Value
PROCESSOR_REVISION	3c03
PSModulePath	%ProgramFiles%\WindowsPowerShell\Modules;C:\Windows\system32\WindowsPowerShell\v1.0\Modules;
TEMP	C:\Windows\TEMP
TMP	C:\Windows\TEMP
USERNAME	SYSTEM
windir	C:\Windows

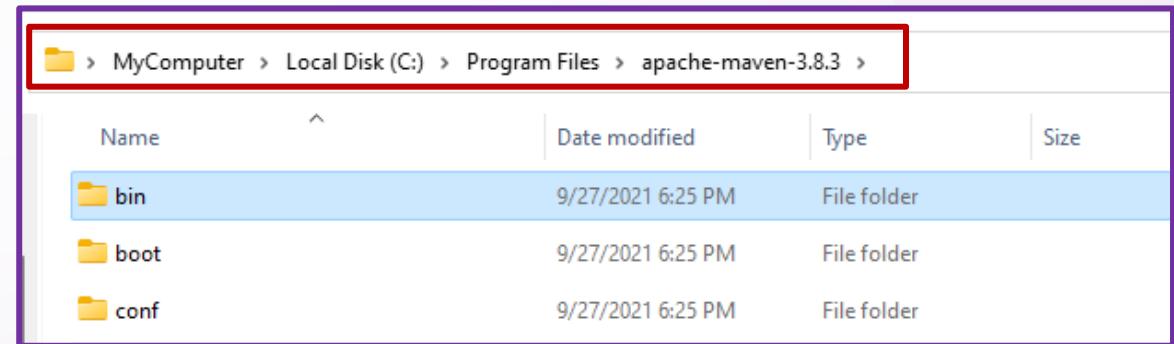
New... Edit... Delete

8- Ok diyerek menuleri kapatın

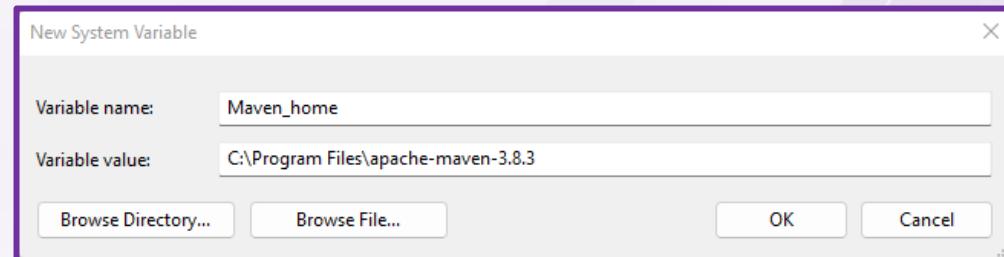
# Cucumber

## Maven Environment Ayari

9- Program Files klasorunde maven klasorune girip dosya yolunu kopyalayin



10- Yeniden Edit envoirement variables for your account menusune girip Environment Variables'i acin New butonuna basip kopyaladiginiz maven klasorunun yolunu yapistirin



11- Ok diyerek menuleri kapatin

12- Ayarlari yaptiktan sonra IntelliJ'yi kapatip acin



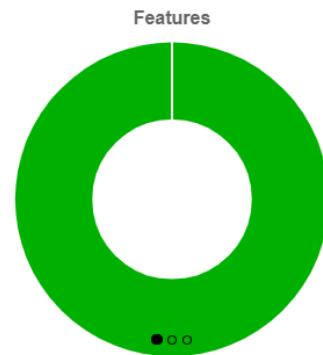
# Cucumber

## Paralel Testing

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

### Features Statistics

The following graphs show passing and failing statistics for features



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



# Cucumber

## 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%

# Cucumber

## 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.icinArاماYapar(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



Wise Quarter  
first class IT courses

# Selenium Team135

## Ders-19

### Cucumber



The future at your fingertips

+1 912 888 1630

[www.wisequarter.com](http://www.wisequarter.com)

/wisequarter



# Cucumber

## Genel Tekrar

The screenshot shows the homepage of AutomationExercise. At the top left is a yellow button labeled "Automation Exercise". At the top right are links for "Home" and "Products". The main title "AutomationExercise" is centered above the subtitle "Full-Fledged practice website for Automation Engineers".

1. <https://automationexercise.com/> sayfasina gidelim
2. Cucumber ile asagidaki testi yapalim
  - Given user web sayfasinda
  - And user sign up linkine tiklar
  - And user New user signUp bölümüne name ve email adresi girer
  - And signUp butonuna basar
  - Then cookies kabul eder
  - And user kisisel bilgilerini ve iletisim bilgilerini girer
  - And user Create Account butonuna basar
  - Then hesap olustugunu test edin



# Cucumber

## Genel Tekrar

All	Group A	Group B	Group M	Group T
Company	Group	Prev Close (Rs)		
Ujjivan Financial	A	963.3		
Quess Corp	A	438.4		
Power Finance Co	A	162.1		
Bharat Petroleum	A	928.7		
Hind. Petrol	A	215.6		
Varun Beverages Ltd.	A	628.8		
NCC	A	84.3		
Blue Dart Express	A	91		
Kansai Nerolac Paint	A	492.4		
Max Financial Services	A	59.9		
IDFC L	A	461.5		
Dewan Housing	A	951.2		

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

# Cucumber

## Genel Tekrar

All	Group A	Group B	Group M	Group T
Company	Group	Prev Close (Rs)		
Ujjivan Financial	A	963.3		
Quess Corp	A	438.4		
Power Finance Co	A	162.1		
Bharat Petroleum	A	928.7		
Hind. Petrol	A	215.6		
Varun Beverages Ltd.	A	628.8		
NCC	A	84.3		
Blue Dart Express	A	91		
Kansai Nerolac Paint	A	492.4		
Max Financial Servic	A	59.9		
IDFC L	A	461.5		
Dewan Housing	A	951.2		

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

# Cucumber

## Genel Tekrar

All	Group A	Group B	Group M	Group T
Company	Group	Prev Close (Rs)		
Ujjivan Financial	A	963.3		
Quess Corp	A	438.4		
Power Finance Co	A	162.1		
Bharat Petroleum	A	928.7		
Hind. Petrol	A	215.6		
Varun Beverages Ltd.	A	628.8		
NCC	A	84.3		
Blue Dart Express	A	91		
Kansai Nerolac Paint	A	492.4		
Max Financial Services	A	59.9		
IDFC L	A	461.5		
Dewan Housing	A	951.2		

1. <http://demo.guru99.com/test/web-table-element.php> sayfasına gidelim
2. Cucumber framework'de US1012\_Guru\_Web\_Tables altında

Scenario : TC\_18\_kullanici\_satir\_sutun\_degeri\_ile\_yazi\_alabilmeli olusturun ve asagidaki testi yapin

Given user web sayfasında

And “Istelenen Satır”, “Istelenen Sutun” daki yazıyı yazdırır

# Cucumber

## Genel Tekrar

All	Group A	Group B	Group M	Group T
Company	Group	Prev Close (Rs)		
Ujjivan Financial	A	963.3		
Quess Corp	A	438.4		
Power Finance Co	A	162.1		
Bharat Petroleum	A	928.7		
Hind. Petrol	A	215.6		
Varun Beverages Ltd.	A	628.8		
NCC	A	84.3		
Blue Dart Express	A	91		
Kansai Nerolac Paint	A	492.4		
Max Financial Servic	A	59.9		
IDFC L	A	461.5		
Dewan Housing	A	951.2		

1. <http://demo.guru99.com/test/web-table-element.php> sayfasina gidelim
  2. Cucumber framework'de US1012\_Guru\_Web\_Tables altinda  
Scenario : TC\_18\_kullanici\_sutun\_basligi\_ile\_liste\_alabilmeli olusturun  
ve asagidaki testi yapin
- Given user web sayfasinda
- And “Istenen Baslik”, sutunundaki tum degerleri yazdirir

# Cucumber

## Genel Tekrar

Baskentler excelini framework'e ekleyin ve asagidaki gorevleri yapin

Given kullanici baskentler exceline ulasir

Then 1.satir 2.hucredeki datayı yazdirir

And 1.satir 2.hucredeki datayı bir string degiskene kaydeder

And kaydedilen datanın "Başkent (İngilizce)" olduğunu test eder

Then baskenti Jakarta olan ulkenin Turkce isminin "Endonezya" olduğunu test eder

And excelde kayitli ulke sayisinin 190 olduğunu test eder

And excelde kullanılan fiziki satır sayisinin 191 olduğunu test eder

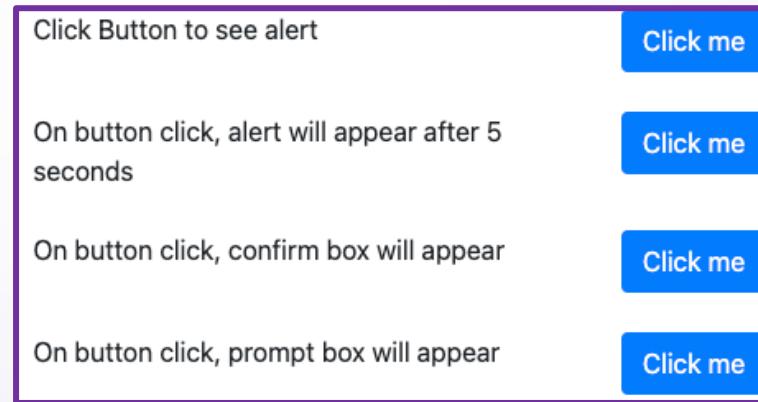
When Tüm bilgileri map olarak kaydedip

Then baskenti Jakarta olan ulkenin tüm bilgilerini yazdirir

And mapi kullanarak turkce ismi Hollanda olan bir ulke bulundugunu test eder

# Cucumber

## Genel Tekrar



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



# Cucumber

## Genel Tekrar

Will enable 5 seconds

Color Change

Visible After 5 Seconds

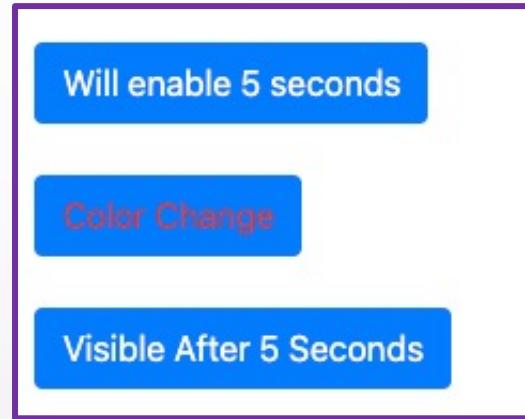
Yeni bir test methodu olusturun

- 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



# Cucumber

## Genel Tekrar



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



# Cucumber

## Genel Tekrar

### Add/Remove Elements

Add Element

[https://the-internet.herokuapp.com/add\\_remove\\_elements/](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