TESTNG FRAMEWORK

TESTNG FRAMEWORK

TestNG

Technical wellSkills - Yetenekler

- **Diller(Languages):**
  - JAVA, Lamda, xml, SQL, HTML, CSS, …
- **Frameworks:**
  - JUnit(gorduk), TestNG(Basliyoruz), Cucumber(Ogrenecez)
- **Design Pattern:**
  - Page Object Model(POM)
- **Version Control System:**
  - Git, Bitbucket(ucretli, kara ile entegresi daha guzel)
- **Front End Testing:**
  - Selenium WebDriver
- **API Testing:**
  - Postman for manual testing, Restful for automation
- **Backend/Database Testing:**
  - SQL, JDBC
- **Continuous Integration(CI), Continuous Deployment(CD):**
  - Jenkins
- **Remote Testing:**
  - Selenium Grid, Jenkins
- Mobile Testing:
  - Appium

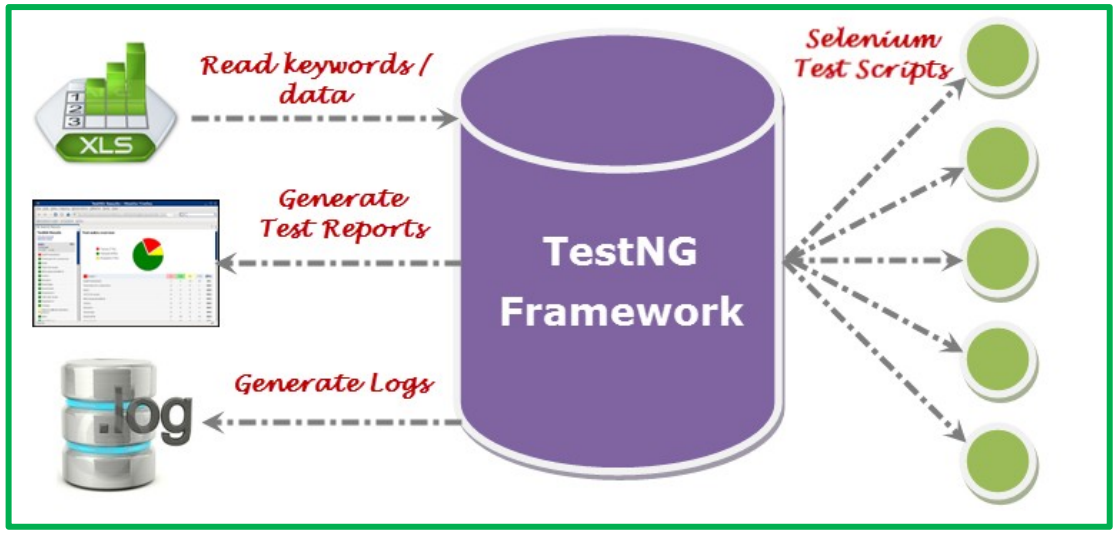
- **Arac/Gerecler(Tools)**
  - Selenium WebDriver
  - Maven
  - JUnit, **TestNG, Cucumber**
  - PostGreSQL
  - JDBC
  - IntelliJ, Eclipse, VS Code
  - JIRA
  - **Jmeter**
  - **Postman**
  - **Restful API**
  - **Jenkins**
  - **Selenium Grid**
  - Git, Github, **Bitbucket, AWC E2E**
  - **Appium**

Test NG

- Junit'in gelismis versiyonudur.
- İsmi Next Generation Test kelimelerinden türetilmiş, ve Cédric Beust tarafından oluşturulmuştur.
- Açık Kaynak kodludur.
- TestNG bir test kütüphanesidir.
- TestNG sadece JAVA ile calisir ve JDK 7 ve daha ust versiyonlari gereklidir
- TestNG ile ilgili dokumanlara asagidaki adresten ulasilabilir
- https://testng.org/doc/documentation-main.html

Test NG

- TestNG tester'lara daha fazla kontrol imkani verir ve testleri daha etkili yapmamizi saglar.
- Tester'lar TestNG'yi etkili bir framework tasarlamak ve test case'leri TestNG annotation'ları ile organize etmek için kullanırlar.
- Test caseleri siralama ozelligi (priority) ve test caselerin birbirine bagimliligi (dependsOnMethod) bize testleri organize etmekte yardim eder.
- Paralel ve Cross-Browser Test yapmamiza imkan tanir
- Kullanisli HTML veya xml raporlari olusturmaya yarar
- **Data Provider** ile DDT(**data driven testing**) imkani saglar



Test NG

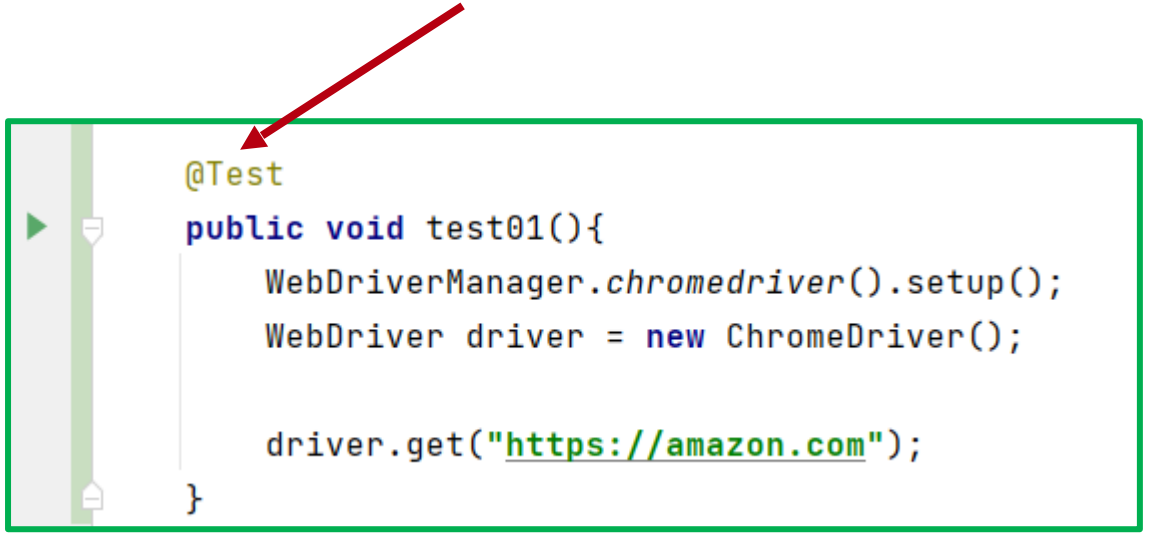
Nasil Yuklenir?

- Yeni bir maven projesi olustur : **B103TestNGProject**
- Java nin altında bir dosya olustur : **techproed**
  - tests -> **techproed'e saga tikla** : techproed.tests
  - utilities -> **techproed'e saga tikla** : techproed.utilities
  - pages-> **techproed'e saga tikla** : techproed.pages
- Dependency'leri pom.xml e ekle
  - Get the dependencies mvnrepository.com
  - selenium java-JUNITDE DE KULLANDIK
  - webdrivermanager-JUNITDE DE KULLANDIK
  - testNG-TESTNG FRAMEWORKU IIN KULLANACAZ

```xml
<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>
```

@Test annotation

- En cok kullanilan TestNG notasyonudur.
- Method'u test case olarak işaretler ve calistirir.
- Ayrica bir Main Method'a ihtiyac duymaz.



```
@Test
public void test01(){
    WebDriverManager.chromedriver().setup();
    WebDriver driver = new ChromeDriver();

    driver.get("https://amazon.com");
}
```

@BEFORE @AFTER

- `@BeforeSuite` `@AfterSuite` => runs before/ after all tests in this suite
- `@BeforeTest` `@AfterTest` => run before/after all the test methods after Test
- `@BeforeGroups` `@AfterGroups` =>run before/after any specific test group
- `@BeforeClass` `@AfterClass` =>run before/ after all the test methods in a test class
- `@BeforeMethod` `@AfterMethod` =>run before/ after each test method (SAME AS @Before method in the JUnit)

Test NG

@Before @After Annotations

@BeforeSuite: The annotated method will be run before all tests in this suite have run.
@AfterSuite: The annotated method will be run after all tests in this suite have run.
@BeforeTest: The annotated method will be run before any test method belonging to the classes inside the <test> tag is run.
@AfterTest: The annotated method will be run after all the test methods belonging to the classes inside the <test> tag have run.
@BeforeGroups: The list of groups that this configuration method will run before. This method is guaranteed to run shortly before the first test method that belongs to any of these groups is invoked.
@AfterGroups: The list of groups that this configuration method will run after. This method is guaranteed to run shortly after the last test method that belongs to any of these groups is invoked.
@BeforeClass: The annotated method will be run before the first test method in the current class is invoked.
@AfterClass: The annotated method will be run after all the test methods in the current class have been run.

NOT :
Bu notasyonlar kullanilinca her method icin ayri ayri calismaz, soylendigi sekilde sadece bir kere calisir.

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

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

Test NG

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

public class TestNGFirst {
    @BeforeMethod @BeforeClass @BeforeTest @BeforeSuite//use only one
    public void setUp(){
        System.out.println("Contains what ever you need before executing a test" +
                "//such as driver set up, waits, maximize, and, go to main URL etc.");
    }
    @Test
    public void test1() { System.out.println("Test case 1"); }

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

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

Test NG

Priority

- TestNG (default) olarak @Test methodlarını alfabetik sıraya göre run eder. (Yukardan asagi degil!)

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

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

- priority annotation Testlere öncelik vermek için kullanılır. Kucuk olan Numara daha once calisir

- priority yazmayan Test method'u varsa priority= 0 kabul edilir, siralama buna gore yapilir

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

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

@IGNORE

- TestNG metotu atlamak

```java
public class TestNGOrdering {
    @Test(priority = 2)
    public void a(){
        System.out.println("This is test 1");
    }
    @Ignore//shipping this test
    @Test(priority = 4)
    public void b(){
        System.out.println("This is test 2");
    }
    @Test(priority = 1-4)
    public void c(){
        System.out.println("This is test 3");
    }
}
```

```
TestNGOrdering
Tests passed: 2 of 2 tests – 14 ms
Default Suite 14 ms
mymavenpr 14 ms
TestNGO 14 ms
c 14 ms
a 0 ms
/Library/Java/JavaVirtualMachines/jdk-11.0.1.
This is test 3
This is test 1
```

Test NG

dependsOnMethods

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

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

TestNGDependencies › searchTest()

TestNGDependencies

Tests failed: 1, passed: 2, ignored: 1 of 4 tests – 52 ms

mymaven 52 ms
TestNG 52 ms
hom 39 ms
login 9 ms
resul 4 ms
searc 0 ms

/Library/Java/JavaVirtualMachines/jdk-11.0.1.jdk/Contents/Home/bin/jav

This is home page test

Test NG

Test Otomasyonun temel noktalarından biri Assertions'dır.

- **Her bir test case icin bir Assertion veya Verification kullanmalıyız.**

- **TestNG ile iki cesit Asertion yapmak mumkundur.**

  1. **) Junit'te kullandigimiz sekilde Assert Class'indan method'lar kullanarak yapmak.**

  2. **) Junit'te olmayan, TestNG'ye ozel olarak kullanabilecegimiz SoftAssert Class'indan method'lar kullanarak yapmak.**

```
public class C01 {

    WebDriver driver;

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

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

Test NG

Assertions

1.HARD ASSERT

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

JUnit'ten bildigimiz uzere en çok kullandığımız 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 yürütmez (stops execution).

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

Test NG

Assertions

2. SOFT ASSERT (VERIFICATION)

SoftAssert doğrulama (verification) olarak da bilinir.

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

Test method'unun istedigimiz bolumde yapilan tum testleri raporlar

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

SoftAssert class'indaki method'lari kullanmak icin kullanabilmemiz için object olusturmamiz gerekir.

Test NG

Soft Assert

- 1.Adım: SoftAssert objesi olusturalim

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

- 2.Adım: Istedigimiz sayida verification'lari yapalim

```
softAssert.assertTrue( );
softAssert.assertFalse( ); ….
```

- 3.Adım: SoftAssert'in durumu raporlamasini isteyelim

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

Test NG

Soft Assert vs Hard Assert

- **Ortak ozellikleri**

  SoftAssert ve HardAssert method'ları TestNG'den gelmektedir.

  Kullanma amaçları farklı olsa da method'lar aynıdır.

- **Farklari**

  •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.

Test NG

Soft Assert Class Work

Yeni bir Class Olusturun : C03_SoftAssert

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

PAGE OBJECT MODEL INTRO

Page Object Model

- Klasorler
  - tests
  - pages
  - utilities
- Classlar
  - Reusable Classes
- Dosyalar
  - Configuration files
  - Rapor files

Test NG / POM

Java'dan Hatirlamamiz Gerekenler

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

A- inheritance (Miras)

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

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

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

```java
public class Okul {
    String okulIsmi="Yildiz Koleji";
    static int ogrenciSayisi=120;

    public void okulMethod(){
        System.out.println("Yildiz Koleji");
    }
}
```

extends

```java
public class Ogrenci extends Okul {

    public void ogrenciMethod(){
        System.out.println(okulIsmi);
        okulMethod();

        System.out.println(ogrenciSayisi);
    }
}
```

Test NG / POM

Java'dan Hatirlamamiz Gerekenler

B- Object olusturarak

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

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

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

```java
public class Okul {
    String okulIsmi="Yildiz Koleji";
    static int ogrenciSayisi=120;

    public void okulMethod(){
        System.out.println("Yildiz Koleji");
    }
}
```

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

        System.out.println(Okul.ogrenciSayisi);
    }
}
```

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

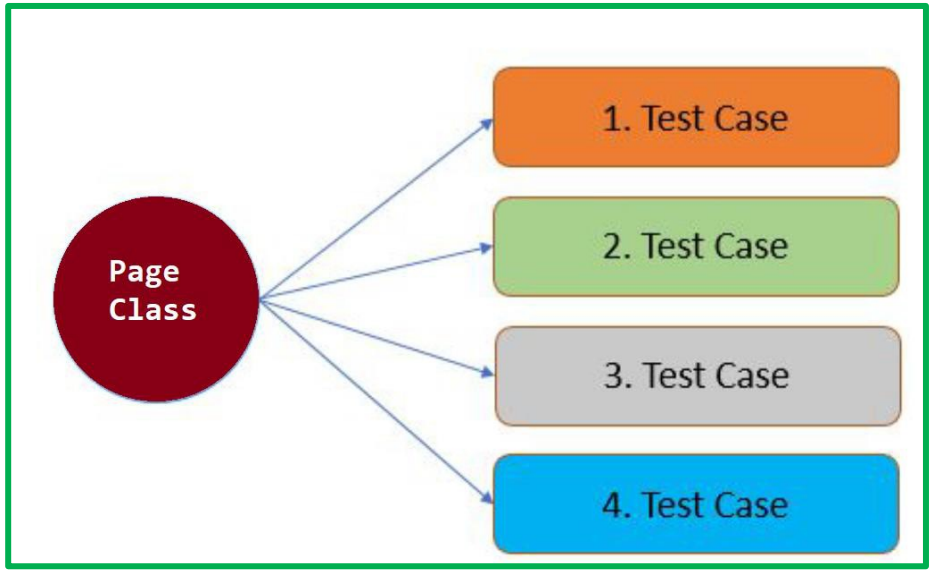
Page Object Model

Framework design

- Bir sirkette test framework'u olusturdugumuzda kullanici adi, sifresi, gidilecek web adresi gibi test datalari tum testler icin gecerlidir. Ayrica surekli kullanmamiz gereken variable ve method'lar olacaktir.

- Daha kullanisli bir Framework olusturmak icin temel hedefimiz, tekrar tekrar yaptigimiz islemleri ve testlerimizde kullandigimiz Test Data'larini onceden hazirladigimiz dosyalarda tutmaktir.

- Bu sekilde testlerimizde ihtiyac duydugumuzda bu verilere kolayca ulasabilir veya test datalari ile ilgili bir degisiklik yapmamiz gerektiginde sadece kaynak dosyadan bir degeri degistirerek tum test case'leri guncellemis oluruz.



Test NG

Page Object Model

Framework design

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

  Bu nedenle,

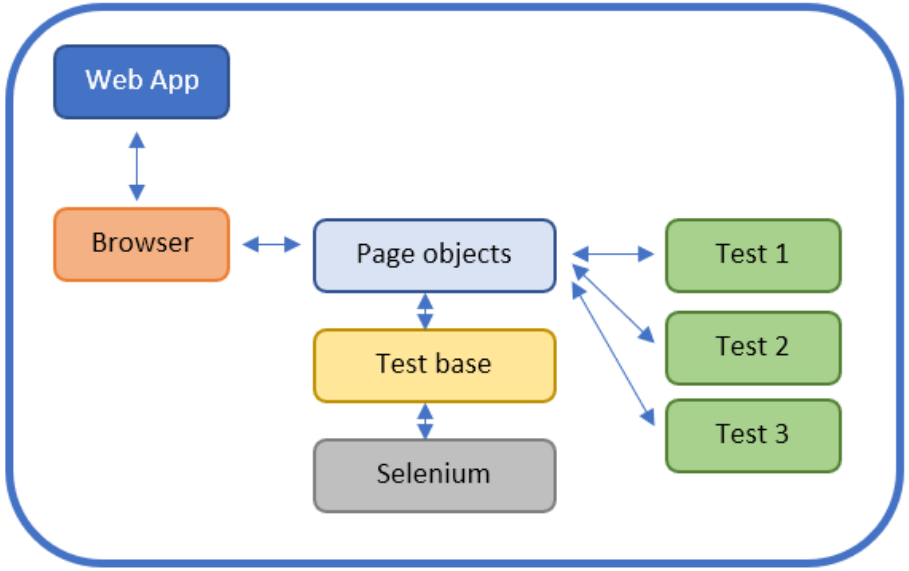
  Kolay bakim yapilabilir(maintainable),

  yeniden kullanılabilir(reusable),

  daha hızlı(faster),

  anlaşılabilir(understandable)

  daha iyi bir framework dizaynina ihtiyacımız vardir.



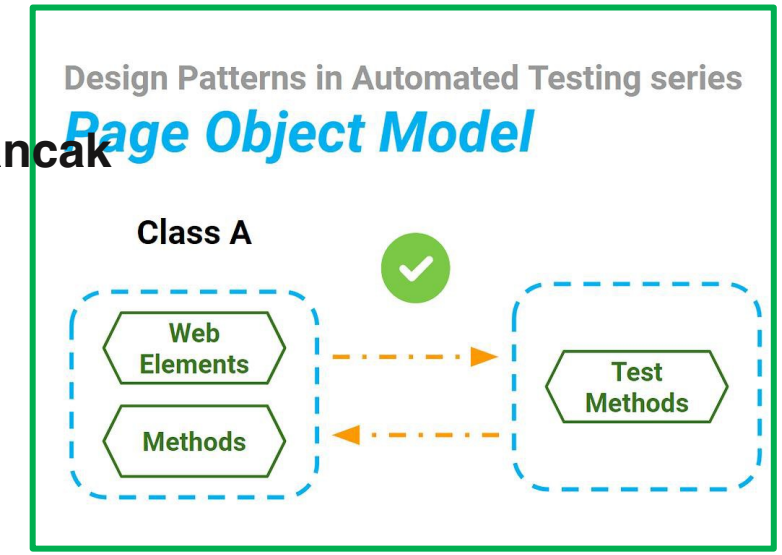
- Page object model ile, sayfaya özgü elementleri veya methodlari page class icinde tutar, ve bunları gerçek test classlarından uzak tutarız.
- POM ile ihtiyacimiz olan class uyelerini sadece bir kez create edip birçok kez kullanabiliriz.

Test NG

Page Object Model

Framework design

- Framework un verimliliğini artırmak için core Java ve Selenium konseptini kullanarak temel olarak page classları ve test classları oluşturacağız.
- Bütün şirketler page object model dizaynini kullanmayabilir, ancak herkes bunu bilir ve daha da popüler hale gelmektedir.
- Daha iyi bir tasarım, testin yürütme süresini daha hızlı hale getirir.
- Bir uygulamanın(application) işlevselliği değiştiğinde, kodu düzeltmek için framework kontrol edilmesini ve gerekli duzeltmelerin yapilmasini kolaylastirir.

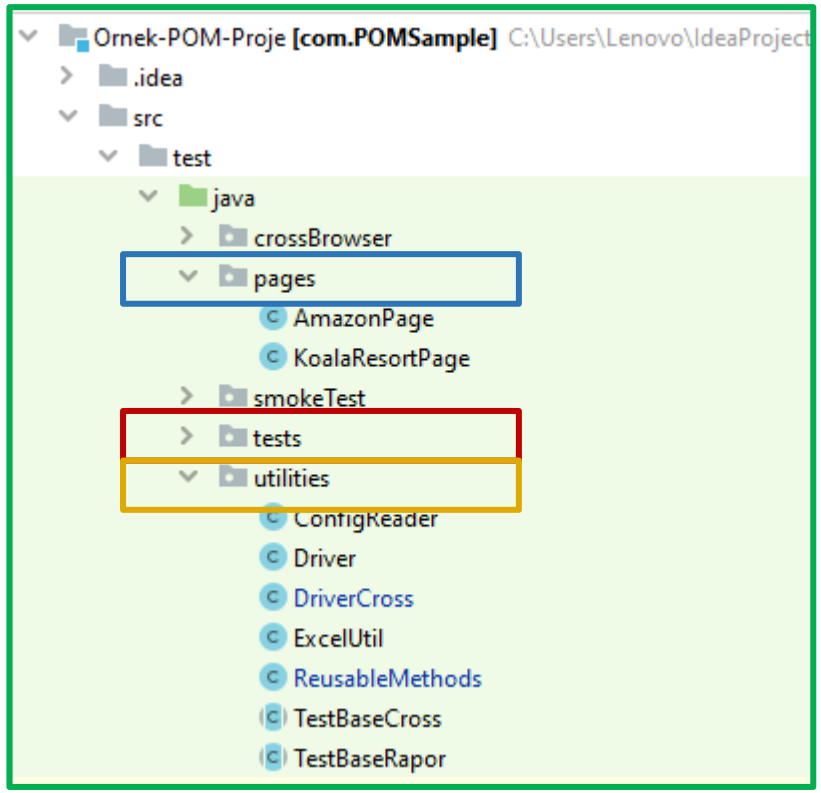


- Page Object Design daha bağımsız test senaryoları oluşturmamıza yardımcı olur, böylece test komut dosyalarında(script) hata ayıklamak daha kolay olacaktır.

Test NG

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 girisi yapmayacagiz

- **Pages :** Test yapacagimiz sayfalardaki Web Elementlerini locate etmek ve temel methodlari olusturmak icin kullanilir.

- **Utilities :** Driver,TestBase ve ConfigurationReader class'larini icerir

Test NG

PAGE CLASS

```java
public class KoalaResortPage {

    public KoalaResortPage(){

        PageFactory.initElements(Driver.getDriver(), page: this);
    }

    @FindBy(linkText = "Log in")
    public WebElement ilkLoginLinki;

    @FindBy(id = "UserName")
    public WebElement kullaniciAdiTextBox;


    @FindBy(id = "Password")
    public WebElement passwordTexBox;

    @FindBy(id = "btnSubmit")
    public WebElement loginButonu;


    @FindBy(xpath = "//*[text()='Try again please']")
    public WebElement girisYapilamadiElementi;

}
```

Page Object Model

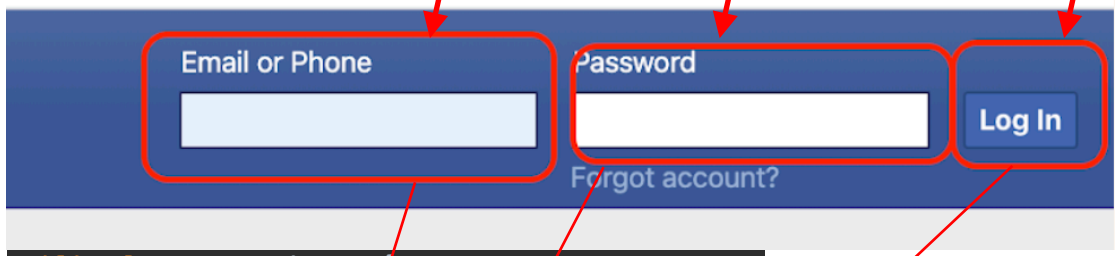
Framework design

TEST CLASS

```java
    @Test
    public void positiveLoginTest() {
        // http://qa-environment.koalaresorthotels.com adresine git
        Driver.getDriver().get(ConfigReader.getProperty("kr_url"));

        //login butonuna bas
        KoalaResortPage koalaResortPage = new KoalaResortPage();
        koalaResortPage.ilkLoginLinki.click();

        //test data username: manager ,
        koalaResortPage.kullaniciAdiTextBox.sendKeys(ConfigReader.getProperty("kr_valid_username"));

        //test data password : Manager1!
        koalaResortPage.passwordTexBox.sendKeys(ConfigReader.getProperty("kr_valid_password"));

        //Degerleri girildiginde sayfaya basarili sekilde girilebildigini test et
        koalaResortPage.loginButonu.click();
        Assert.assertEquals(Driver.getDriver().getCurrentUrl(), ConfigReader.getProperty("kr_basarili_giris_url"));

        Driver.closeDriver();
    }
}
```

Sample Test Case 1

1.PAGE ELEMENTS: EMAİL, PASSWORD, LOGİN BUTTON



- **Page Class:**
  - Create page elements and/or major methods
- **Test Class:**
  - Call page objects
  - Create test cases and assertion

PAGE CLASS

```
public class FaceLoginPage {

    WebDriver driver;
    public FaceLoginPage(WebDriver driver) {//We need th
        this.driver=driver;//to connect the page elemen
        PageFactory.initElements(driver, page: this);//
    }

    @FindBy(id = "email")
    public WebElement username;

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

    @FindBy(id = "u_0_b")
    public WebElement loginButton;

    public void login(String userid,String pass){
        username.sendKeys( ...charSequences: userid);
        password.sendKeys( ...charSequences: pass);
        loginButton.click();

    }
```

Face Log In Page ELement

Face Log In Page Method

TEST CLASS

2.USE REUSABLE PAGE OBJECTS AND WRITE TEST CASES

```
public class LogIn_PageObjectModel extends TestBase {

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

    }
```

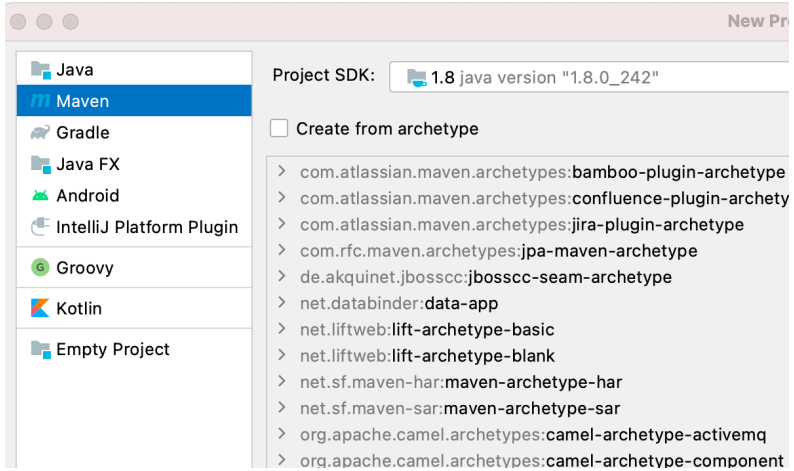
PAGE OBJECT MODEL

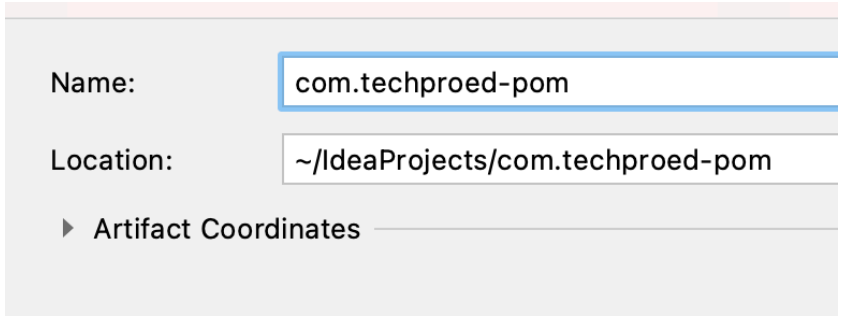
Page Object Model

SIFIRDAN PAGE OBJEST MODEL NASIL OLUSTURULUR?

1. DEPENDENCIES

- **Create a new project:com.techproed**
- **Add dependencies**

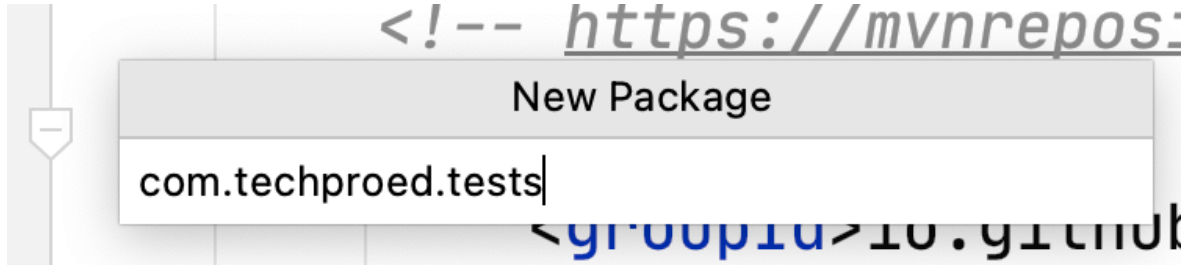


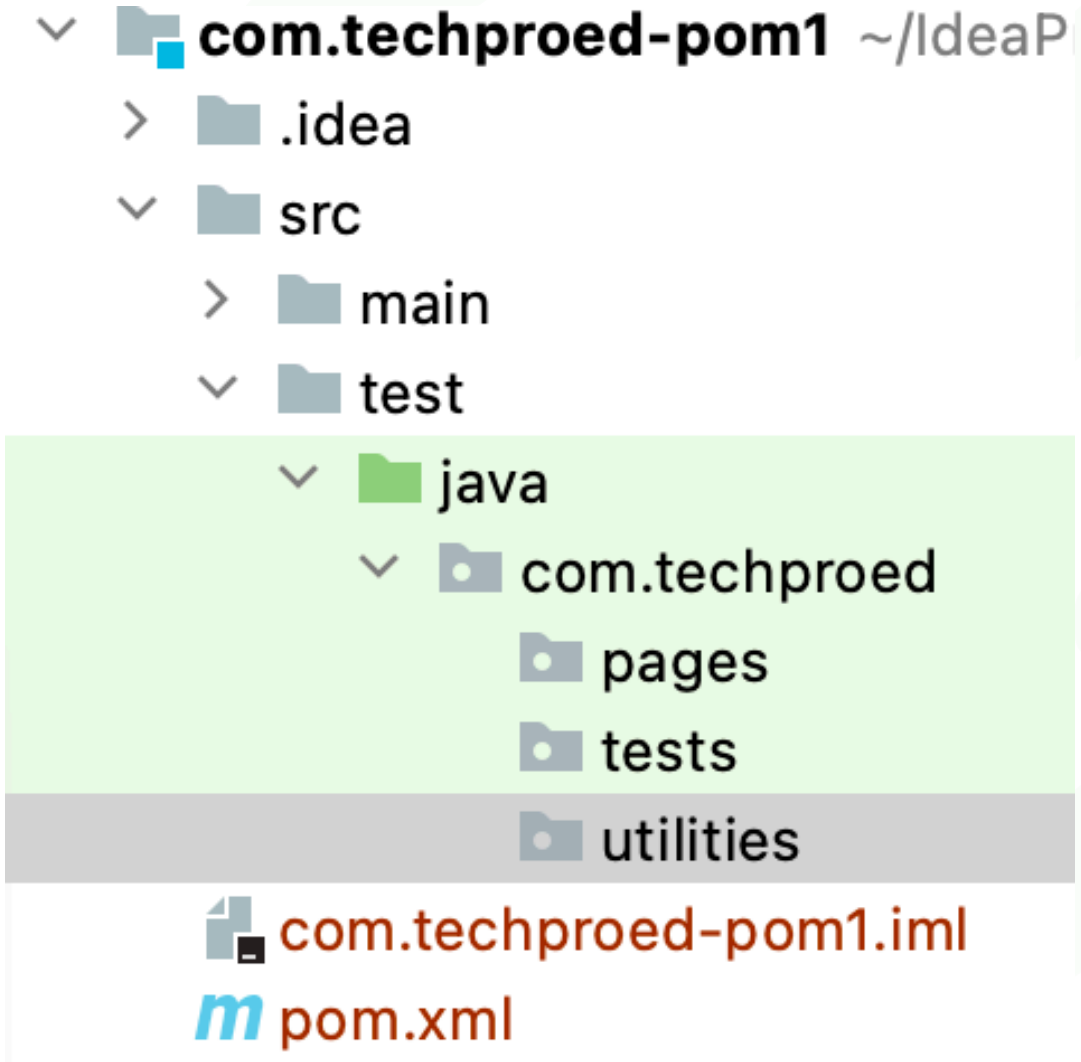



```
<dependencies>
    <!-- https://mvnrepository.com/artifact/org.seleniumhq.selenium/selenium-java
-->
    <dependency>
        <groupId>org.seleniumhq.selenium</groupId>
        <artifactId>selenium-java</artifactId>
        <version>3.141.59</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/io.github.bonigarcia/webdrivermanager
-->
    <dependency>
        <groupId>io.github.bonigarcia</groupId>
        <artifactId>webdrivermanager</artifactId>
        <version>4.4.3</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>
</dependencies>
```

2. Klasorler

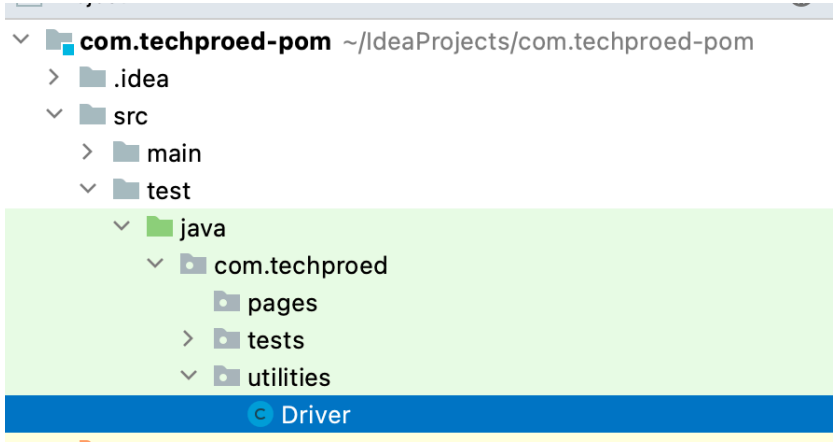
- **pages, tests, utilities**





3. Driver Class

- **Driver**



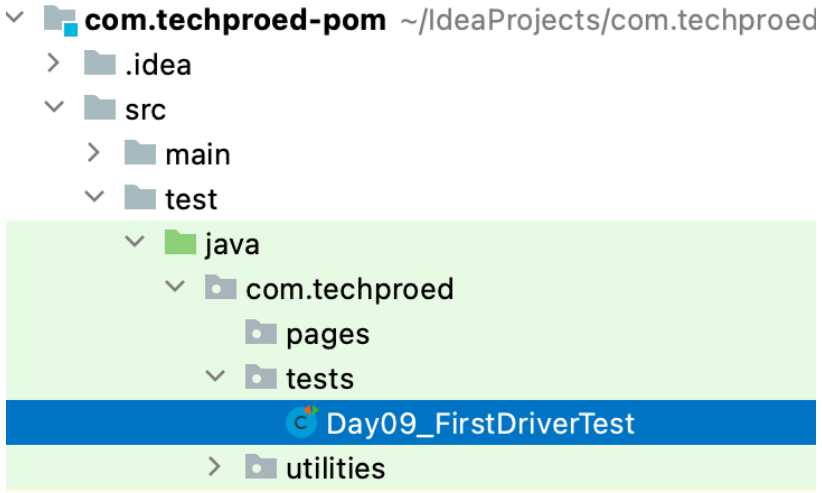

```java
package com.techproed.utilities;

import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import java.util.concurrent.TimeUnit;

public class Driver {
    private static WebDriver driver;

    public static WebDriver getDriver(){
        if(driver==null) {
                    WebDriverManager.chromedriver().setup();
                    driver = new ChromeDriver();
        }
        driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);
        driver.manage().window().maximize();
        return driver;
    }

    public static void closeDriver(){
        if (driver!=null) {//if driver is pointing anywhere
            driver.quit();//quit when I call closeDriver
            driver=null;//make the driver null so when we call getDriver, we
can open the driver again
        }
    }
}
```

3. Driver Class

- Driver class utilies classda oluşturulacak.
- Tüm tarayıcılar(browser) için geliştireceğiz.
- Driver objesi public method yardımıyla tum sınıflardan ayni sekilde ulasilabilecek. Bu sekilde **singleton pattern**'e uygun dizayn edilerek tum projede farkli driverlar uretilmesinin onune gecilecek.
- **Tum classlarda ayni tek bir objenin kullanılması icin oluşturulan class a `Singleton design pattern` denir**
- **Driver objesi static olacakdir**
- **`Soru :`** static I framework unde nerelerde kullandin?
- **`Answer :`** In my **Driver** class, our driver is static so It can be shared across the framework(or so driver instance can be used globally in the framework).

Driver Class Test

- Create FirstDriverTest class
- Go to amazon page
- Verify the title includes amazon
- Check if Driver class is working




```java
package com.techproed.tests;

import com.techproed.utilities.Driver;
import org.testng.annotations.Test;

public class FirstDriverTest {
    @Test
    public void firstDriverTest(){
        //driver -> Driver.getDriver()
        //driver.get("https://www.amazon.com");
        Driver.getDriver().get("https://www.amazon.com");
//
Driver.getDriver().get(ConfigReader.getProperty("qa_environment"));
    }
}
```

Driver Class

Basic Driver Class

```
public class Driver {

    static WebDriver driver;//IQ:Why WebDriver static

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

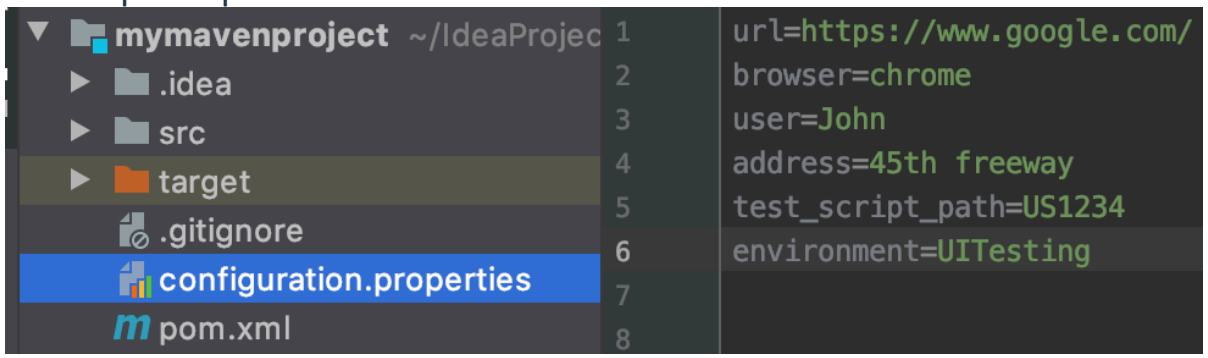
Test Class(driver->Driver.getDriver())

driver.get("url") —> Driver.getDriver().get("url")

```
public class SingletonDriver {

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

4. Config Properties Dosyasi

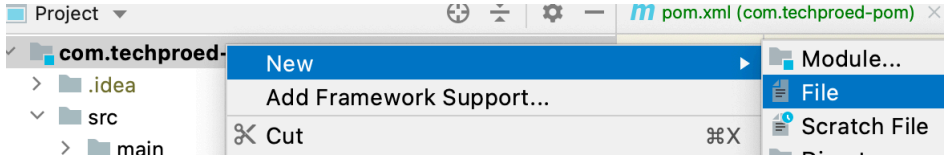
- Properties dosyalari **`.properties`** uzantili dosyalardır
- Bu dosyalar **temel test datalarının depolamak icin kullanilir: url, username, password, browser(tarayıcı),..**
- Test datalari bu dosyadan bir JAVA class ile (**ConfigReader**) alinir ve test caselerde dinamik olarak kullanılır
- Ornegin url=https://www.techproeducation.com ise,
  - **`driver.get("www.techproeducation.com"); yerine driver.get(url);kullanılır`**
- Bu şekilde **key, value degerleri** in config dosyalarinda depolanip test caselerde dinamik sekilde kullanılır
  - **KEY = VALUE**
  - url=https://www.techproeducation.com
  - browser=chrome
  - username=manager
  - password=pass
  - name=Ali

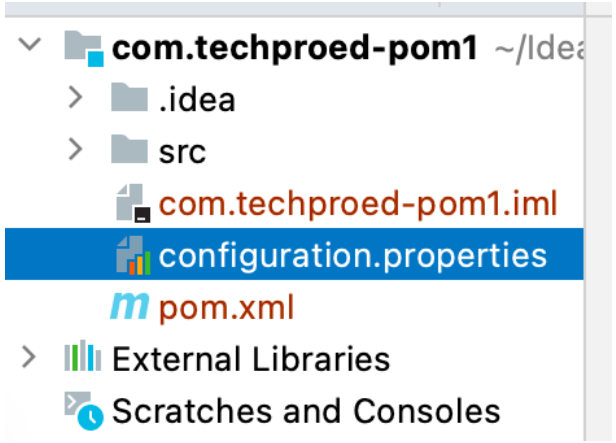


- Properties dosyalari developerlar tarafinda data çekmek icin kullanılır. IT dünyasında popülerdir
- Genelde proje seviyesinde oluşturulur. Projeye sag tıkla->**New->File ->** **`configuration.properties`**

Config Properties Dosyasi

- Proje seviyesinde .properties dosyası olustur
- configuration.properties
- Data lari ekle url,id,...

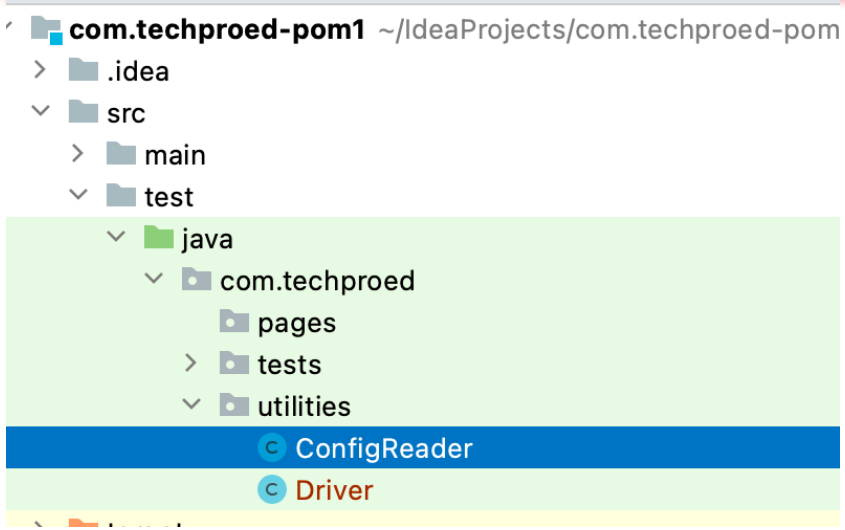





```
#Add major Test Data on properties file
qa_environment=https://qa-environment.com
amazon_url=https://www.amazon.com
manager_username=manager123
manager_password=Manager2!!!
browser=chrome
username=john
password=1234
test_email=testtid@gmail.com
test_username=manager
test_phone_number=3442134582
incorrect_username=fakeusername
incorrect_password=fakepass
dblogincreds=asdg!fa
#getProperty(qa_environment)  ->https://qa-environment.com
#getProperty(browser) ->chrome
#getProperty(username) ->john
#driver.get("https://qa-environment.com")  ->
driver.get(getProperty(qa_environment))
#driver.sendKeys("john") -> driver.sendKeys(getProperty(username))
```

5. Config Reader class

- ConfigReader java sinifini Utilities klasorunde ekle




```java
package com.techproed.utilities;
import java.io.FileInputStream;
import java.util.Properties;

public class ConfigReader {
    //This class reads the configuration.properties file
    //Create Properties instance
    private static Properties properties;
    static {
        //path of the configuration file
        String path="configuration.properties";
        try {
            //Opening configuration.properties file using FileInputStream
            FileInputStream fileInputStream = new FileInputStream(path);
            properties = new Properties();
            properties.load(fileInputStream);
            //close the file
            fileInputStream.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
    //This method will get the key from properties file,
    //And return the value as String
    //We create this method to read the file
    public static String getProperty(String key){
        String value=properties.getProperty(key);
        return value;
    }
    //TEST IF LOGIC WORKS
//    public static void main(String[] args) {
//        System.out.println(properties.getProperty("qa_environment"));
//    }
}
```

Ilk Properties dosyası Testi

- In FirstConfigTest
- Get url from config.properties file
- Get the title from config.properties file

```
package com.techproed.tests;

import com.techproed.utilities.ConfigReader;
import com.techproed.utilities.Driver;
import org.testng.annotations.Test;

public class Day11_FirstConfigTest {

    @Test
    public void firstConfigTest(){
//        go to app_url
//        Driver.getDriver().get("http://www.carettahotel.com/");
//        ConfigReader.getProperty("app_url")   ===>>>> http://www.carettahotel.com/
        Driver.getDriver().get(ConfigReader.getProperty("app_url"));

//Assert the title equals : Caretta Hotel - Home
        String actualTitle = Driver.getDriver().getTitle();
        String expectedTitle = ConfigReader.getProperty("app_title");
        Assert.assertEquals(actualTitle,expectedTitle);
    }
}
```

Data Akisi

Configuration.properties file

LogIn_PageObjectModel.java | ConfigR

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

ConfigurationReader Class

```
import java.io.FileInputStream;
import java.util.Properties;

public class ConfigurationReader {

    //We write the config properties one and keep using it.
    private static Properties properties;//We use Properties class to handle properties i
    //I am making static because I want it to run every time I call. I wan tit to belong
    static {//to initilize we use static block.
        String path="configuration.properties";//path of the config properties file
        try {
            FileInputStream file=new FileInputStream(path);//To open an external file we
            properties=new Properties();//initiliazing the configurationFile
            properties.load(file);// Loading the file
            file.close();//closing the file after loading. This is not mandatory
        } catch (Exception e) {
            //System.out.println("Path doesn't exist");// we can send a message if file d
            e.printStackTrace();
        }
    }

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

    //Testing if I can read from the configuration.properties file. Key and Value pairs
//    public static void main(String[] args) {
//        String a=properties.getProperty("url");
//        System.out.println(a);
//    }

}
```

Test Class

LogIn_PageObjec
LogIn_Traditional
LoginPage
SelectableWithA
TestBaseDemo
TestNGAnnotatic
TestNGBeforeAf
TestNGDepende
TestNGFirst
TestNGOrdering
WebTables

```
30          fhcLoginPage.managerUsername.sendKeys( ...charSeq
31          fhcLoginPage.managerpassword.sendKeys( ...charSeq
32          fhcLoginPage.FHCloginButton.click();
33
34      }
35
36      @Test
37      public void configPropertiesExample(){
38          String user= ConfigReader.getProperty("user");
39          System.out.println(user);
40      }
41
```

LogIn_PageObjectModel › configPropertiesExample()

LogIn_TraditionalWay.LoginTest | LogIn_PageObjectModel.configProperties...

Tests passed: 1 of 1 test – 4 s 914 ms

Defaul 4 s 914 ms
myn 4 s 914 ms
L 4 s 914 ms
con 2 ms

John

Datalarin dinamik olması neden onemlidir?

- 500 test casemin olsun ve her birinde URL e gidilsin: Driver.getDriver().get("https://www.techproeducation.com")
- Sonra tum testleri UAT environment de test etmek isteyelim. Bu URL :https://www.uat-techproeducation.com
- Bu testlerin o URL de test edilmesi icin ne yapılması gerekir?
- Then how do you fix the problem?
  - URL dinamik degilse : Tum test classlara git, URL I tek tek register
  - URL dinamik ise(properties file): url = https://www.techproeducation.com olan satiri url = https://www.uat-techproeducation.com olarak değiştirmek yeterlidir

6. Dinamik Driver class

- We will develop Driver class for all browsers.
- So Before we create driver object we use switch statements to check different browser conditions.
- Put the waits in Driver as we put in TestBase.
- Then close the driver.
- From now on, we don't need to use TestBase class.
- We are using the Driver class
- After Switch, If you see error, Then set language level to 7

```
"/
f(driver==null) {
   //ConfigReader.getProperty("browser") ==>chrome
   switch (ConfigReader.getProperty("browser")) {
       case "chrome":
           WebDriverManage
           driver = new Ch
```

Incompatible types. Found: 'java.lang.String', required: 'byte

Set language level to 7 - Diamonds, ARM, multi-catch etc. ⌥⇧⏎

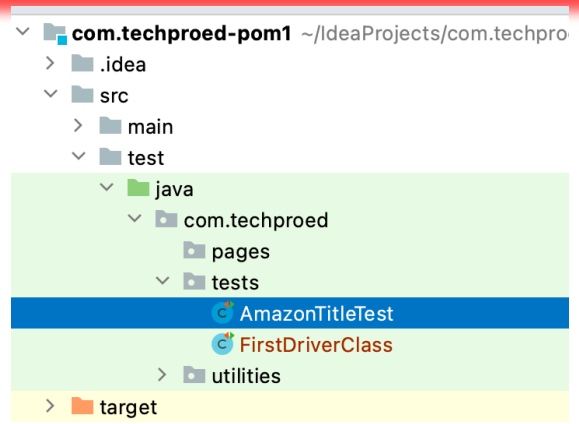
com.techproed.utilities.ConfigReader

```
package com.techproed.utilities;

import io.github.bonigarcia.wdm.WebDriverManager;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.firefox.FirefoxDriver;
import java.util.concurrent.TimeUnit;

public class 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 "chrome-headless":
                    WebDriverManager.chromedriver().setup();
                    driver = new ChromeDriver(new ChromeOptions().setHeadless(true));
                    break;
            }}
        driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);
        driver.manage().window().maximize();
        return driver;
    }//getDriver ends here
    public static void closeDriver(){
        if (driver!=null) {
            driver.quit();
            driver=null;
        }}}
```

```
browser=chrome
```

6. Dinamik Driver class Test

- Create a new class: TechproedTitleTest
- When user goes to teachproed(get the test data from config file)
- Then verify title includes 'Techpro Education'

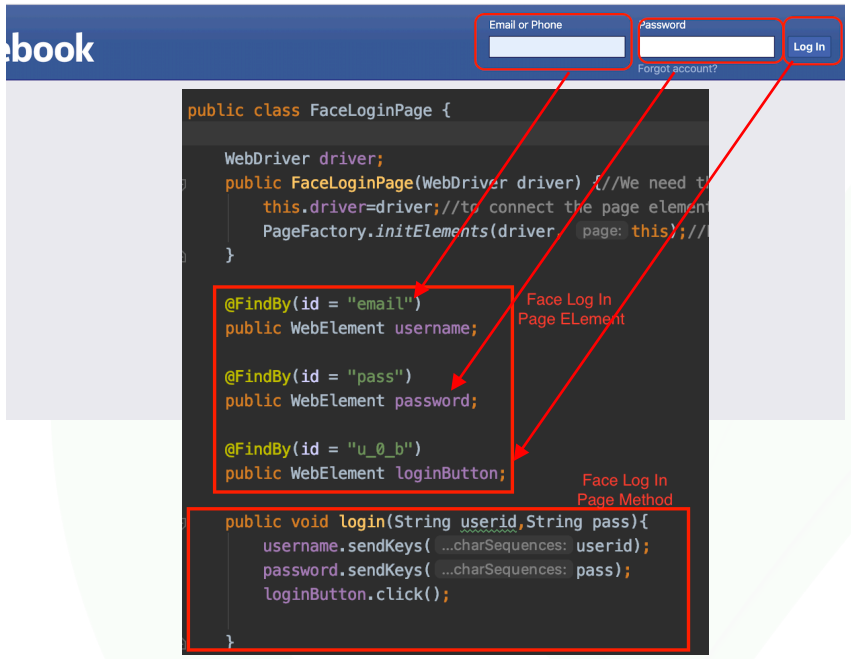



```java
package com.techproed.tests;

import com.techproed.utilities.ConfigReader;
import com.techproed.utilities.Driver;
import org.testng.Assert;
import org.testng.annotations.Test;

public class AmazonTitleTest {
    @Test
    public void amazonTitleTest(){
        Driver.getDriver().get(ConfigReader.getProperty("amazon_url"));
        String amazonTitle=Driver.getDriver().getTitle();
        Assert.assertTrue(amazonTitle.contains("Amazon"));
    }
}
```

Traditional Way | POM

```java
public class LogIn_TraditionalWay extends TestBase {

    @Test
    public void LoginTest(){
        driver.get("https://www.facebook.com/");
        driver.findElement(By.id("email")).sendKeys( ...charSequences: "username");
        driver.findElement(By.id("pass")).sendKeys( ...charSequences: "password");
        driver.findElement(By.xpath("locator of the login button")).click();
        Assert.assertTrue(driver.findElement(By.xpath("log in message")).isDisplayed());
    }
}
```

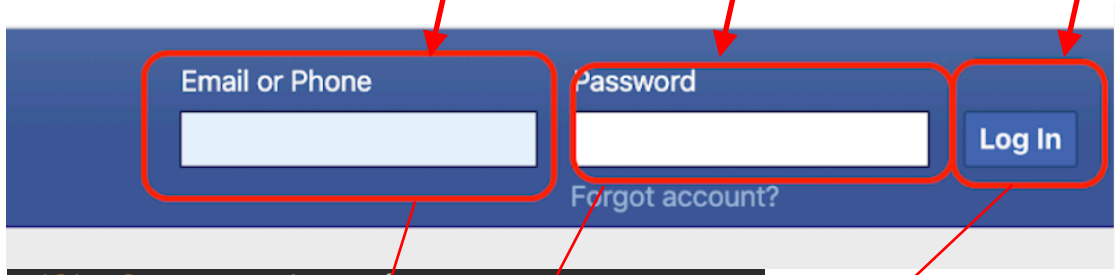
- **PAGE OBJECT MODEL**
- **Page Class:**
  - Sayfa elementlerini locate etmek icin kullanılır
- **Test Class:**
  - Page Classdan sayfa elementlerini cagrilir
  - Test Case lerde bu elementler kullanılır




```java
public class LogIn_PageObjectModel extends TestBase {

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

    }
```

Ornek POM Test Case

1.PAGE ELEMENTS: EMAİL, PASSWORD, LOGİN BUTTON



- **Page Class:**
  - Sayfa elementlerini locate etmek icin kullanılır
- **Test Class:**
  - Page Classdan sayfa elementlerini cagrilir
  - Test Case lerde bu elementler kullanılır

PAGE CLASS

```
public class FaceLoginPage {

    WebDriver driver;
    public FaceLoginPage(WebDriver driver) {//We need th
        this.driver=driver;//to connect the page elemen
        PageFactory.initElements(driver, page: this);//
    }

    @FindBy(id = "email")
    public WebElement username;

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

    @FindBy(id = "u_0_b")
    public WebElement loginButton;

    public void login(String userid,String pass){
        username.sendKeys( ...charSequences: userid);
        password.sendKeys( ...charSequences: pass);
        loginButton.click();

    }
```

Face Log In Page ELement

Face Log In Page Method

TEST CLASS

2.USE REUSABLE PAGE OBJECTS AND WRITE TEST CASES

```
public class LogIn_PageObjectModel extends TestBase {

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

    }
```

Test Case

- https://opensource-demo.orangehrmlive.com/web/index.php/auth/login

- Page Class : OpenSourcePage
  - kullaniciAdi, kullaniciSifre, submitButton elementlerini bul
- Test Class : OpenSourceLogin
  - Page objesi olustur
  - Login Testini basarili oldugunu test et

@FINDBY ANNOTATION

- **@FindBy** web elementlerin page classlarda locate etmek icin kullanilir
- Bu objectler test siniflarinda tekrar tekrar cagrilip kullanilabilir

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

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

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

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

}
```

PAGE FACTORY

- **PageFactory** page factory design icin kullanılır
- **PageFactory.initElements(driver,this);**
- **PageFactory, initElements** stastatic methodunu ile class elementlerini instantiate etmek icin kulanilir.

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

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

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

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

}
```

Test Case

- https://testcenter.techproeducation.com/index.php?page=form-authentication
- Page object Model kullanarak sayfaya giriş yapildigini test edin
- Sayfadan cikis yap ve cikis yapildigini test et

techproed

SuperSecretPassword

CONFIG READER:

techpro_test_url = https://testcenter.techproeducation.com/index.php?page=form-authentication

techpro_test_username = techproed

techpro_test_password=SuperSecretPassword

PAGES:

TechproLoginPage

- userName
- password
- submitButton

TechproHomePage

- homeHeader
- homeLogoutButton

TEST:

- Class: TechproLoginTest
- Metot : loginTest()

Smoke Test

Smoke Test ve Regression Tests

SMOKE TEST

- Smoke Test nedir? : Uygulamanin stabil durumda oldugunu, ve onemli özelliklerin calisir oldugunu test edilir
  - Login
  - Checkin
  - Checkout
  - Add to card
  - Make payment
  - Sign out
  - ……
- Ne sıklıkla yapılır?
  - Her sabah 8 am civarı
- Ne kadar surer?
  - 15-20 dakika
- Smoke Test I nasıl yaparsın?
  - Smoke test suite klasorumuz var. Bu klasordeki test leri hergun calistiririz.
  - Aslinda jenkins her sabah 8 am de smoke testleri calistirir ve raporları takıma email atar. Testerlar bu raporu inceler ve fail varsa takıma email atar.
  - Virtual Machine(VM-Sanal Bilgisayar) test caselerin calismasi icin kullanilabilir.
- Smoke test suite de kac test case iniz var
  - 18 tane
- Tum smoke test caseleriniz automate edilmismidir? Manual de varmidir?
  - Tum onemli test casesler automate edilmiştir.
- Hangi testlerin smoke test e eklenecegine Kim karar verir?
  - Test Lead. Sr. Automation Testers. QA. En kidemli tester.

REGRESSION TEST

- Regression Test Nedir? : Tum ana, major, onemli fonksiyonların test edildiği testlerdir
- Regression kapsamlı bir testdir:
  - Smoke test+(muster profili ile yapilsin)
  - Diger müşteri hizmetleri, admin, tech support
  - Farkli odeme sitemleri(PayPal, amex, visa, Mastercard, WU, BTC,…)
- Ne sıklıkla yapılır?
  - Production bug fix lerden sonra
  - Major(ana) releaselerden once
  - Biz 6 ayda bir major release yapariz ve bu releaselerden once regression test yapılır
- Ne kadar surer?
  - 6-7 saat surer
  - Bazi regression test casele automate edilmemiş ise o zaman manual test gerekir. Bu durum test suresi uzar
- Regression suite de kac test case iniz var
  - 400 den fazla test case var
- Tum regression test caseleriniz automate edilmismidir? Manual de varmidir?
  - Takim olarak hedefimiz tum regression test caselerinin automate edilmesidir. Fakat, bu çok mumkul olmuyor. Genelde yuzde 80 den fazla oranda automate ediyoruz
- Hangi testlerin regregression test e eklenecegine Kim karar verir?
  - Test Lead. Sr. Automation Testers. QA. En kidemli tester.
- Otomate edilemeyen bir durum betirmisiniz?
  - Dogrulama gerekdiren storylerde automate edemiyoruz. Cunki doğrulama telefondaki mobil uygulara geliyor
  - Asiri guvenlikli durumlar.
  - Bazi teknik user storyler sadece developerlarla alakalı olabiliyor. Testerlarin onların kullandigi environmentlara accesleri olmayabiliyor. Dolayisiyla bu grubu durumlarda biz automate edemiyoruz

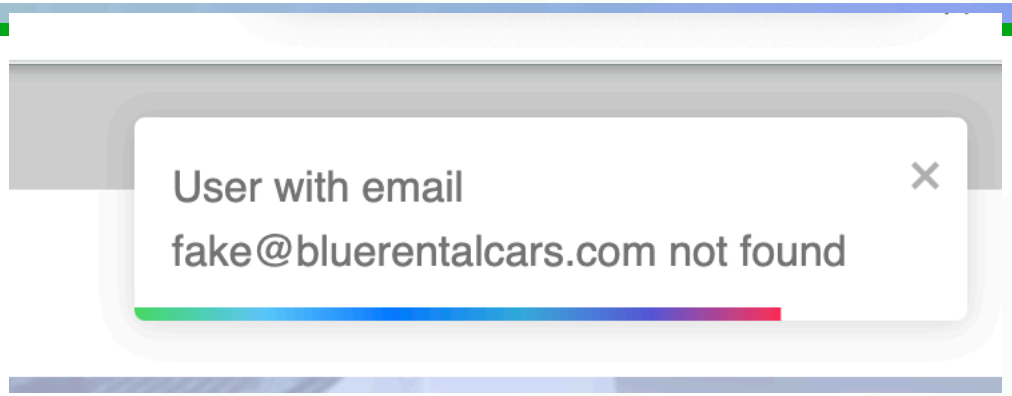
Project

- Blue Rental Car

Project

- **Name:** US100201_Admin_Login
- **Description:**
  - Admin bilgileriyle giriş
- **Acceptance Criteria:**
  - Admin olarak, uygulamaya giriş yapabilmeliyim
  - https://www.bluerentalcars.com/
  - Admin email: jack@gmail.com
  - Admin password: 12345

Project



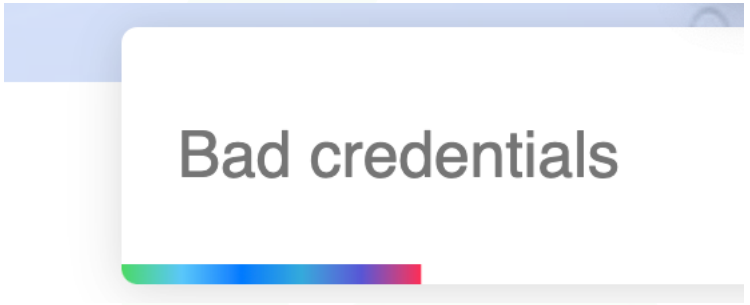
- **Name:**
  - US100208_Negative_Login
- **Description:**
  - Kullanimda olmayan kullanıcı adi ve şifre ile giriş yapilamamali
- **Acceptance Criteria**

  - Customer email: fake@bluerentalcars.com
  - Customer password: fakepass

  - Error:
    - **User with email fake@bluerentalcars.com not found**

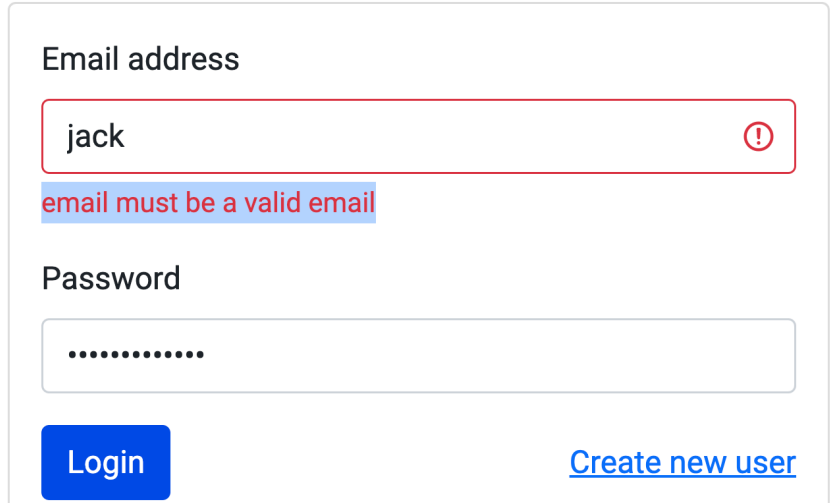
User Story 3

- **Name:**
  - US100402_Negative_Login
- **Description:**
  - Kullanimda olmayan kullanıcı adi ve şifre ile giriş yapılamamalı
- **Acceptance Criteria:**
  - Kullanici **dogru email** ve **yanlis sifre** girildiginde, hata mesajini alınmalı
    - **Hata Mesaji:**
      - Bad credentials
  - Test Data:
    - Customer email: jack@gmail.com
    - Customer password: fakepass



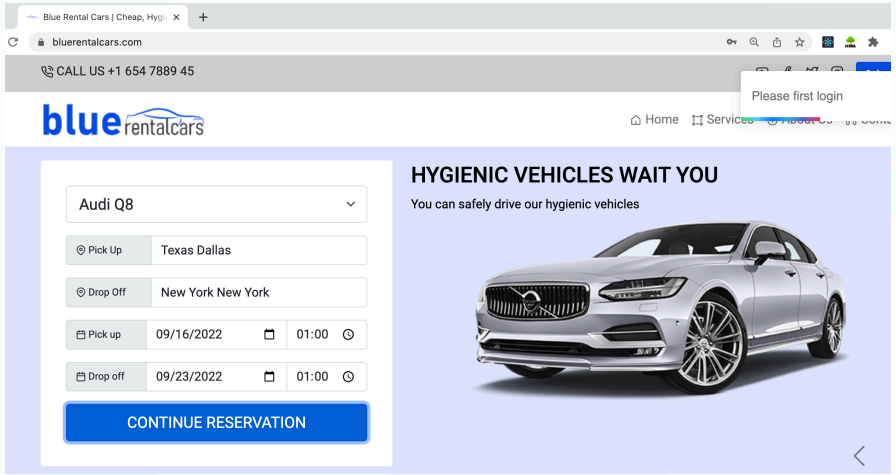
User Story 4

- Name:
  - US101122_Negative_Login
- Description:
  - Geçerli email uzantısı olmadan kullanıcı girişi yapılamamalı
- Acceptance Criteria:
  - Kullanici **geçersiz email uzantısı yazdiginda**, hata mesajini almalı
    - Error Message: **email must be a valid email**
  - Dogru email uzantisi girildiğinde hata mesajı alınmamalı
    - https://email-verify.my-addr.com/list-of-most-popular-email-domains.php



User Story 5

- Name:
  - US101201_Negative_Login
- Description:
  - Geçerli giriş yapmadan rezervasyon yapamamalı
- Acceptance Criteria:
  - Kullanici arac bilgilerini girip CONTINUE RESERVATION butonuna tikladiginda
  - Ve giris yapilmadiginda
  - Hata mesaji almali : **Please first login**
  - Giris yapildiginda hata mesaji alınmamalı



EXCEL AUTOMATION

Excel-Automation

- Excel Automation
  - **Apache poi kutuphanesi** microsoft dosyalarini otomate etmek icin kullanılır
  - Workbook > Sheet > Row > Cell
- ExcelUtils
  - Reusable metotumuz var ve test caselerde mümkün olduğunda bu classdaki metotları kullanırız
    - Datalari excelden bu metotlarla çekilir
    - Sonra gelen datalar test caselerde kullanılır

Test Data

- Test datalari nerden gelir?
  - BA
  - Test Lead
  - Tech Lead/Team Lead/Dev Lead
  - Manual Tester
  - Developer
- Test datalarini test caselerde nasıl kullanırsın?
  - Dinamic olarak alirim. Datalar dışardaki bazı dosyalardan gelir :
    - External Files
    - Config.properties
    - Excel
    - Json
    - Xml
    - Database
    - API
    - Faker

Excel-Automation

- Apache poi dependencylerini pom.xml e ekleyelim
- ExcelUtil classi utilities e ekleyelim

Excel-Automation

- Data Driven Test icin excel kullanılır
- Birden fazla datayı test caselerde ard arda kullanma işlemine Data Driven Testing denir
- Ornegin, farklı kullanıcı giris bilgilerini excel de depolayıp ordan çekebilirim
  - **admin bilgileri, manager bilgileri, customer service bilgileri,…**

Test Case:

- Login fonksiyonunu farkli kullanıcı bilgileri ile test et
- Test Data URL : https://www.bluerentalcars.com/
- Kullanici bilgileri : Excel dosyasında

TEST STEPS:

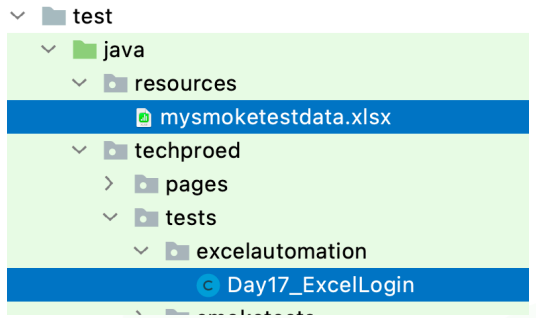
- Test Data
  - **Url I config dosyasına ekle**
  - **Excel dosyasını resources klasörüne ekle**
- Test case I excel automation klasorunde yaz
  - **Java class : LoginExcel**
  - **Methods : adminLoginTest**

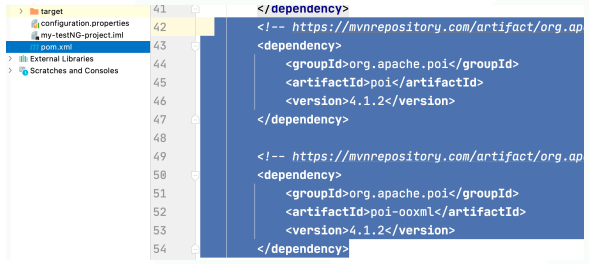
Excel

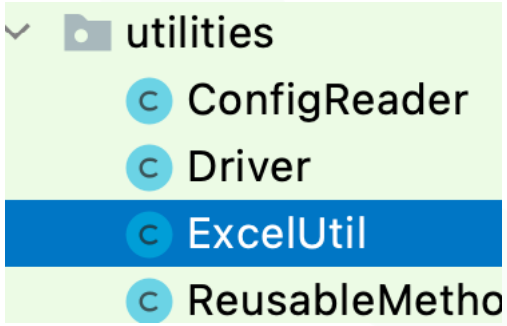
- ExcelUtil class I excelden datalari kolaylıkla çekmek icin kullanıyorum.
- Excel dosyasında istenilen datalar saklanabilir. Kullanıcı dataları, reservasyon dataları, odeme bilgileri…
- Excel den data cekme islemi DDT testi esnasinda yapilir. Bircok data ya ihtiyac duyulduğunda yapılır
- Test datalarini arttirinca test case sayisi artar, test data sayisi azaltirsak test case says da azalir. Yani test case sayisi test data sina bagli olarak degisir.
- Excel I test datalarini korunakli olarak depolamak icin de kullanılır. Bu sekilde datalar kaybolmaz.

Excel Setup

- Poi dependency olustur
- Java nin altinda Resources klasoru olustur
- Excel dosyasını oraya ekle
- ExcelUtil util classi olustur
- Excelautomation klasoru olustur
- AYARLAR BITTI. TEST CASE OLUSTURABILIRIZ
- Tests klasorunun altında **excelautomation** klasörü oluşturalım
- **Day23_ExcelLogin**







Data Provider

Data Provider

- DataProvider, TestNG nin bir ozelligidir
- Test Caselere data göndermek icin kullanilir
- Data Provider 2 boyutlu Array return eder
- 2 tane parametre alabilir. Parametre zorunlu degildir
  - **name** : yeni isimlendirme icin kullanilir
  - **parallel** : paralel test icin kullanilir

```
@DataProvider
public Object[][] urunler(){
    String urunListesi[][]={
            {"tesla"},
            {"bmw"},
            {"mercedes"},
            {"honda"},
            {"toyota"}
    };
    return urunListesi;
}
```

```
@Test(dataProvider = "urunler")
public void googleArama(String data){
    Driver.getDriver().get("https://www.google.com");
    Driver.getDriver().findElement(By.name("q")).sendKeys( ...keysToSend: data+ Keys.ENTER);
    Assert.assertTrue(Driver.getDriver().getTitle().contains(data));
    Driver.closeDriver();
}
```

Data Provider

```
//    DATA PROVIDER METHOD
    @DataProvider(name = "my_smoke_data",parallel = true)
    public Object[][] customerData(){
//    TEST DATA
        Object [][] customerCredentials = {
                {"sam.walker@bluerentalcars.com","c!fas_art"},
                {"kate.brown@bluerentalcars.com","tad1$Fas"},
                {"raj.khan@bluerentalcars.com","v7Hg_va^"},
                {"pam.raymond@bluerentalcars.com","Nga^g6!"}
        };
        return customerCredentials;
    }


//    TEST METHOD 1
    @Test(dataProvider = "my_smoke_data")
    public void dataProviderTest1(String username,String password){
        System.out.println("USERNAME : "+username+" | PASSWORD : "+password);
    }
```

Data Provider

- 3 tane test case olusturalim ve data providerin kullanilma durumlarını farklı sekillerde görelim

- **Package: dataprovider**
- **Class :DataProvider1**
- Datalari data provider metotundan test metoduna aktar
- **Test Data:**

{"sam.walker@bluerentalcars.com","c!fas_art"},

{"kate.brown@bluerentalcars.com","tad1$Fas"},

{"raj.khan@bluerentalcars.com","v7Hg_va^"},

{"pam.raymond@bluerentalcars.com","Nga^g6!"}

- **Package: dataprovider**
- **Class :DataProvider2**
- Datalari data provider metotundan test metoduna aktar ve login test caseinde bu datalari kullan
- **Test Data:**

{"sam.walker@bluerentalcars.com","c!fas_art"},

{"kate.brown@bluerentalcars.com","tad1$Fas"},

{"raj.khan@bluerentalcars.com","v7Hg_va^"},

{"pam.raymond@bluerentalcars.com","Nga^g6!"}

- **Package: dataprovider**
- **Class :DataProvider3**
- Datalari excelden data provider metotu ile test metoduna aktar ve login test caseinde bu datalari kullan
- **Test Data:**

{"sam.walker@bluerentalcars.com","c!fas_art"},

{"kate.brown@bluerentalcars.com","tad1$Fas"},

{"raj.khan@bluerentalcars.com","v7Hg_va^"},

{"pam.raymond@bluerentalcars.com","Nga^g6!"}

TestNG xml files

Test NG

XML File Olusturma

XML, hem insanların hem de makinelerin okuyabileceği belgeleri kodlamak için oluşturulan bir biçimlendirme dilidir.

Veri saklamak, okumak, ve farklı işletim sistemleri arasında veri transfer etmek için kullanılan .xml formatına sahip dosyalarda saklanır.

Biz de framework'umuzdeki belirli testleri veya tum testleri otomatik olarak calistirmak icin xml dosyalari kullaniriz

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

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

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

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

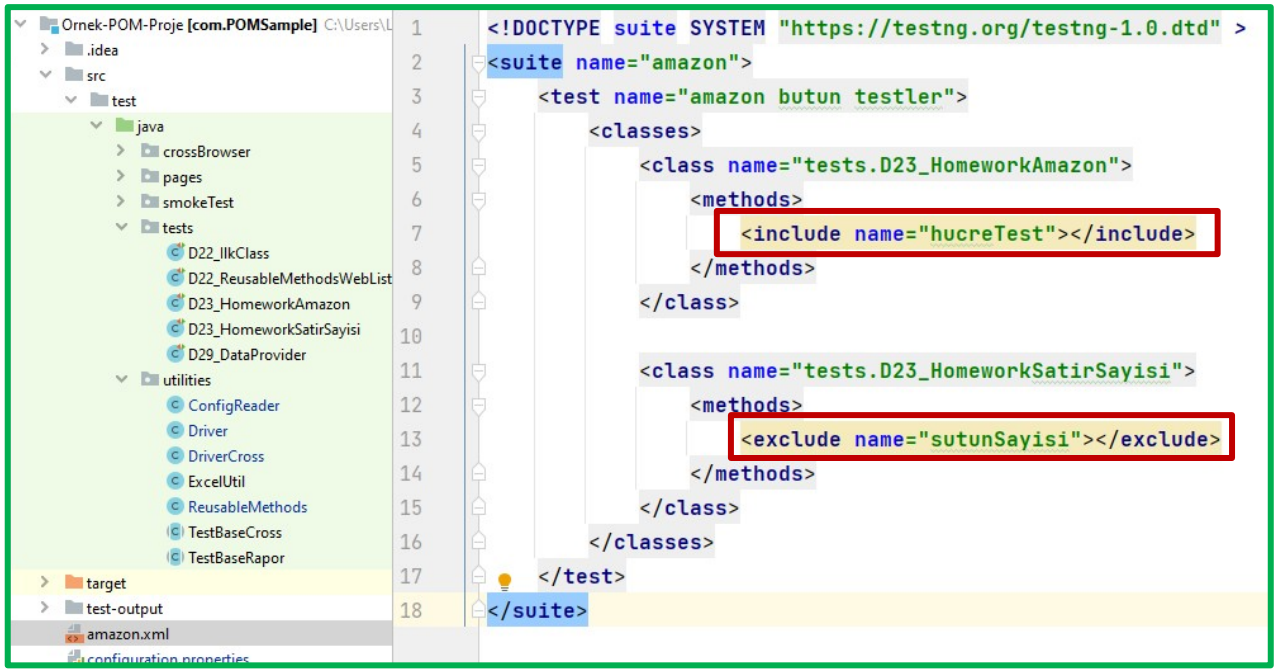
Test NG

XML File Olusturma

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

- Belirli testleri, package lari, classları veya method'lari dahil edebilir (include) veya hariç (exclude) tutabiliriz. Bu da bize testleri calistirmada esneklik (flexiblity) kazandirir.

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



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

Test NG

Belirli Test'ler Nasıl Calıştırılır?

- 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

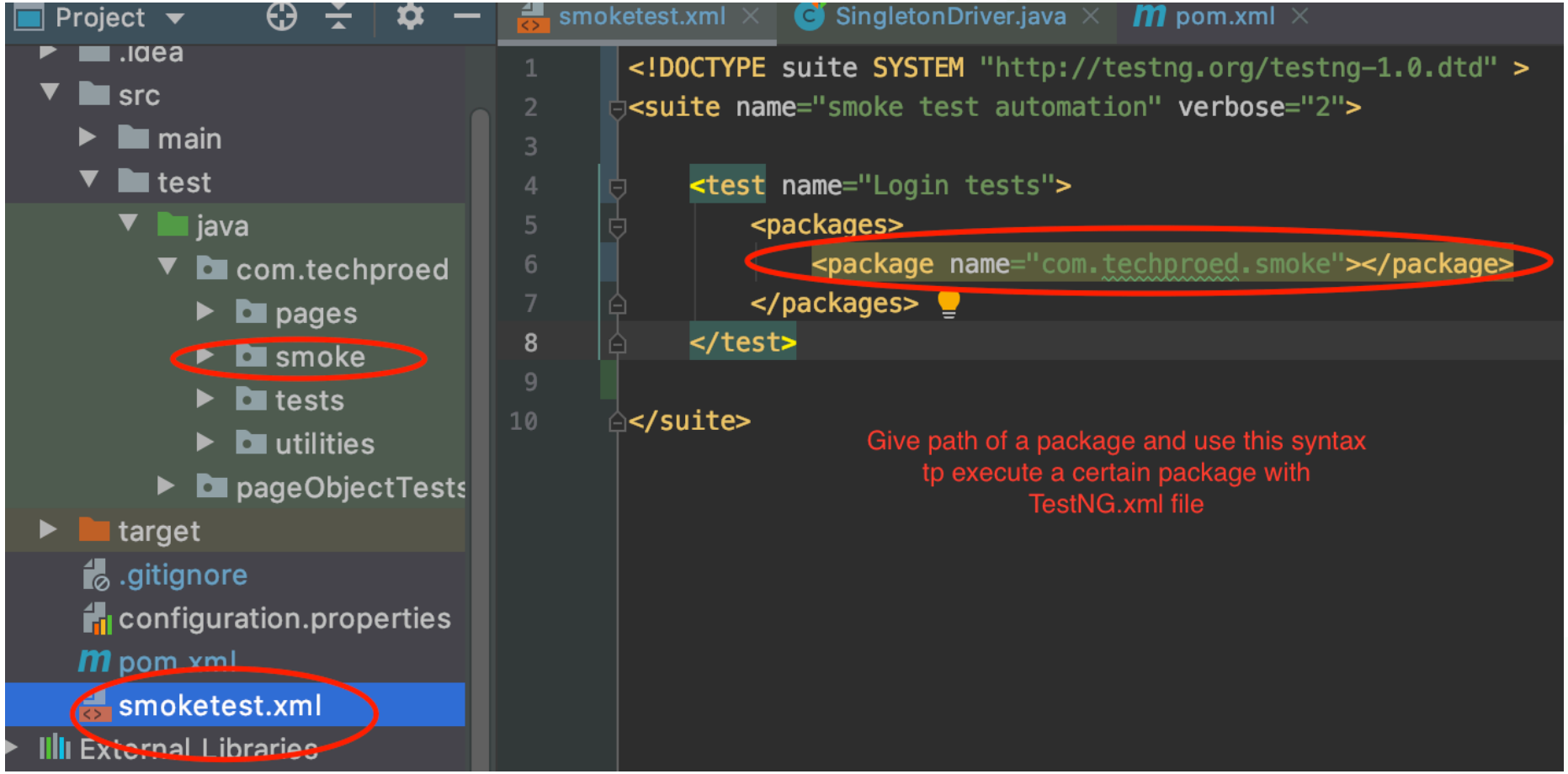
```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="coklu class calistirma">
    <test name="coklu calistirma testi">
        <classes>
            <class name="tests.D23_HomeworkSatirSayisi"></class>
            <class name="tests.D23_HomeworkAmazon"></class>
        </classes>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

Klasorleri run etme

=



Test NG

XML File Class Work Istenen Package'lari Calistirma

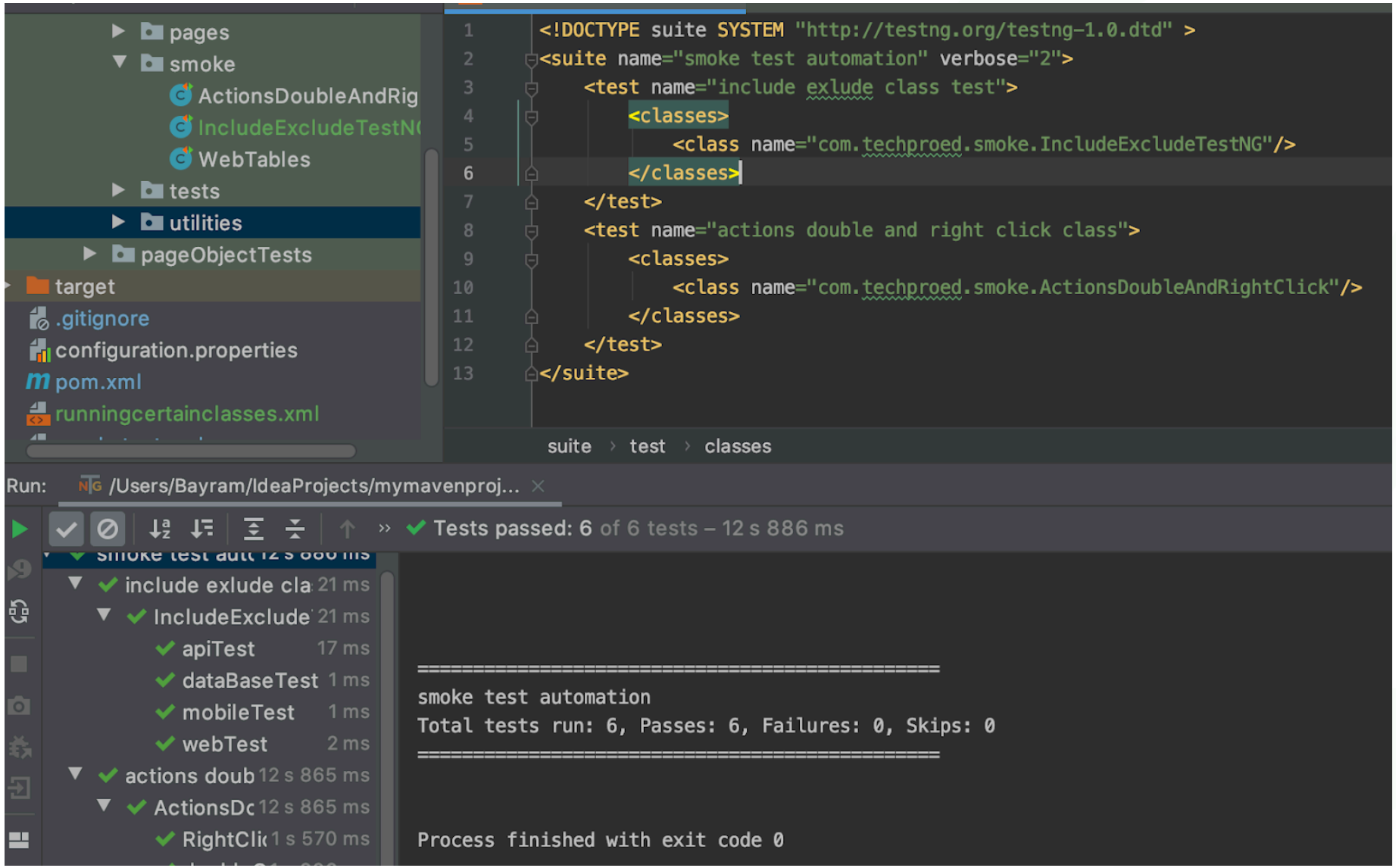
- Yeni bir xml dosyasi olusturalim :

  smoketest.xml

- Smoke Test package'indaki tum testleri calistiralim

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="Blue Rental Car Suite" verbose="1">
    <test name="Smoke Test">
        <packages>
            <package name="techproed.tests.dataprovider"></package>
            <package name="techproed.tests.smoketests"></package>
        </packages>
    </test>
</suite>
```

Classlari run etme



Test NG

XML File Class Work Istenen Class'lari Calistirma

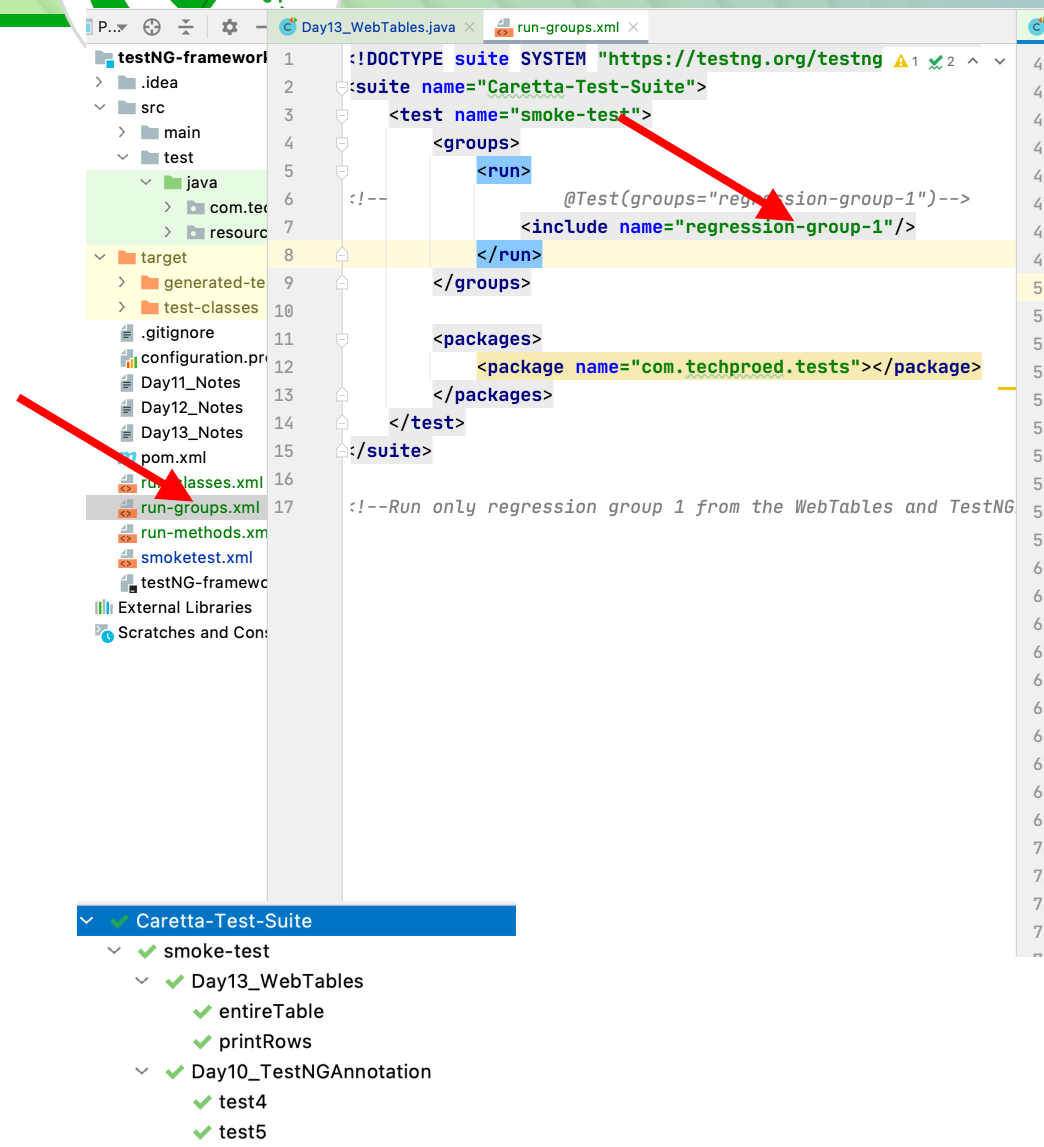
- Yeni bir xml dosyasi olusturalim :

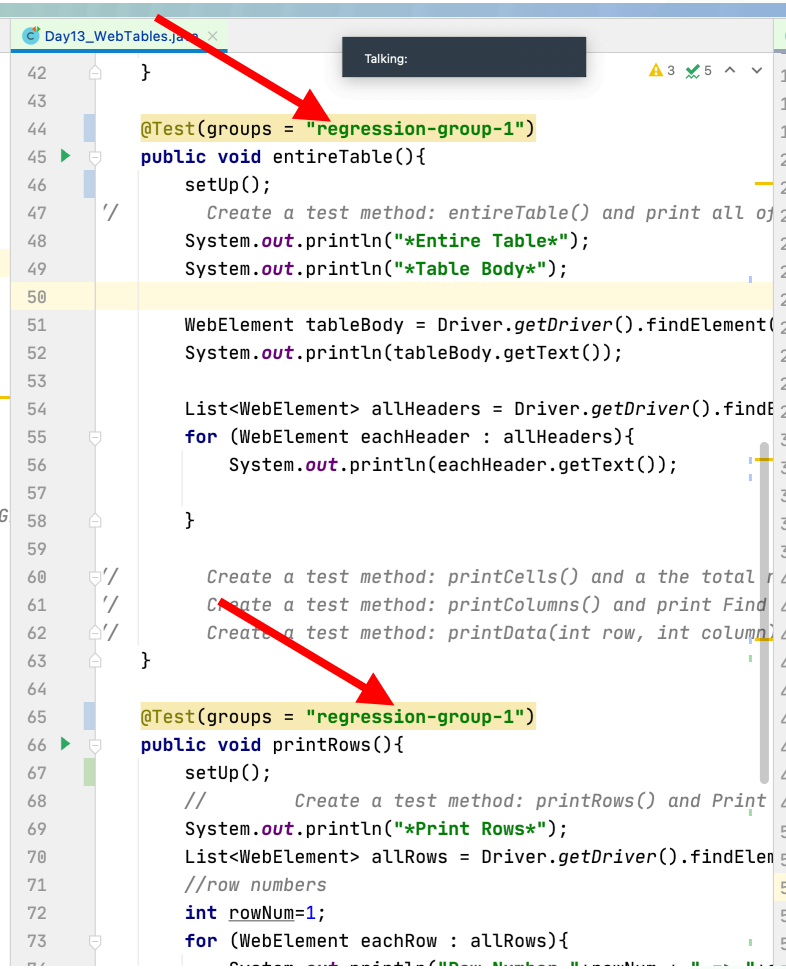
  belirliclasslar.xml

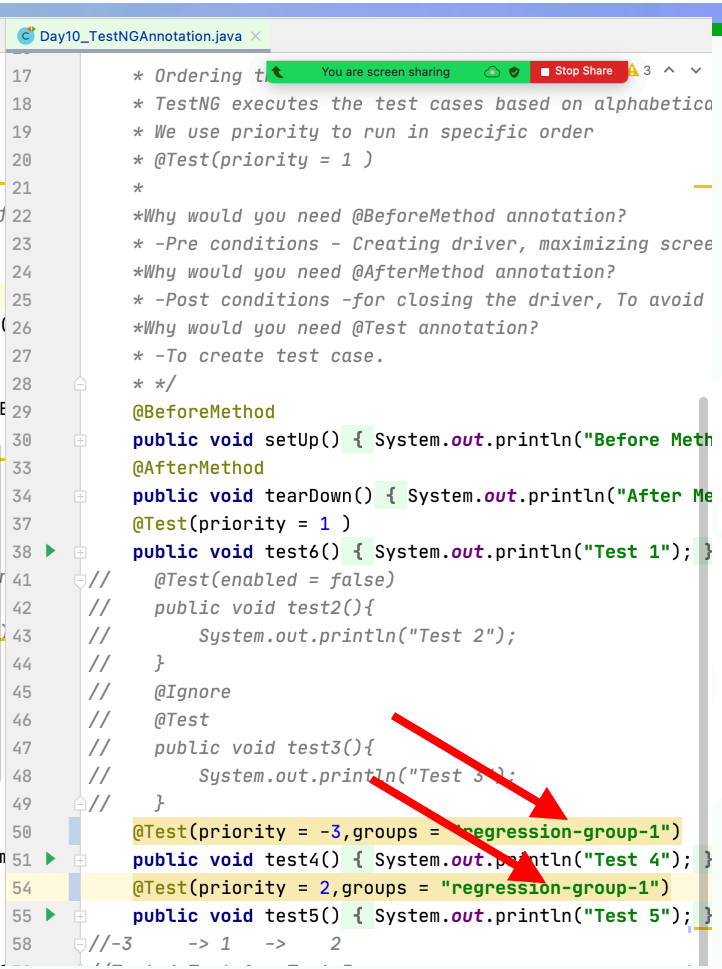
- <package> attribute yerine <classes> ve sonra <class> attribute kullanarak istediginiz class'lari calistirin

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="Suite1" verbose="1" >
    <test name="Positive Login Tests" >
        <classes>
            <class name="techproed.tests.smoketests.Day22_PositiveLoginTest"/>
        </classes>
    </test>
    <test name="Negative Login Tests">
        <classes>
            <class name="techproed.tests.smoketests.Day22_NegativeLoginTest"/>
        </classes>
    </test>
</suite>
```

Gruplari run etme







Test NG

XML File Class Work Istenen Gruplari Calistirma

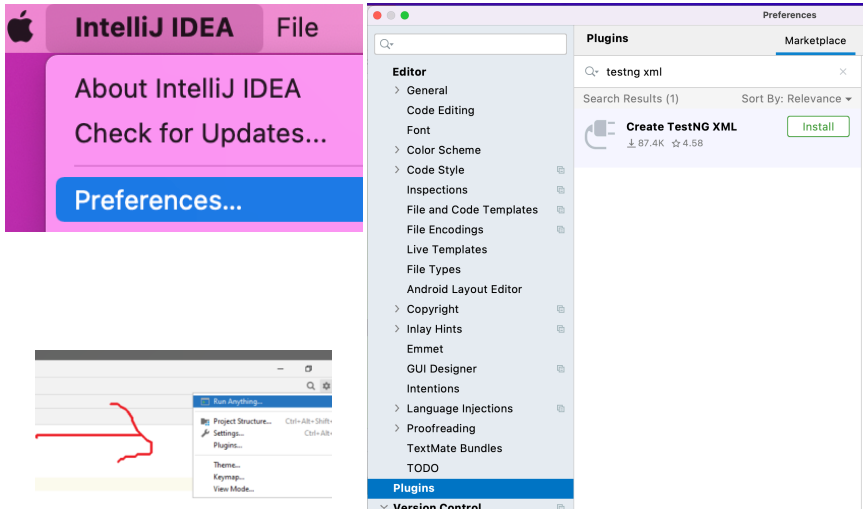
- Yeni bir xml dosyasi olusturalim : belirligruplar.xml
- Group calistirmak icin hem grup adini tanimlamak hem de nerede arayacagimizi belirtmek zorundayiz

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="Regression Suite">
    <test name="Regression1">
        <classes>
            <class name = “techproed.tests.Day16_Annotations”>
               <methods>
                 <include name="test1"/>
               </methods>
            </class>
        </groups>
    </test>
</suite>
```

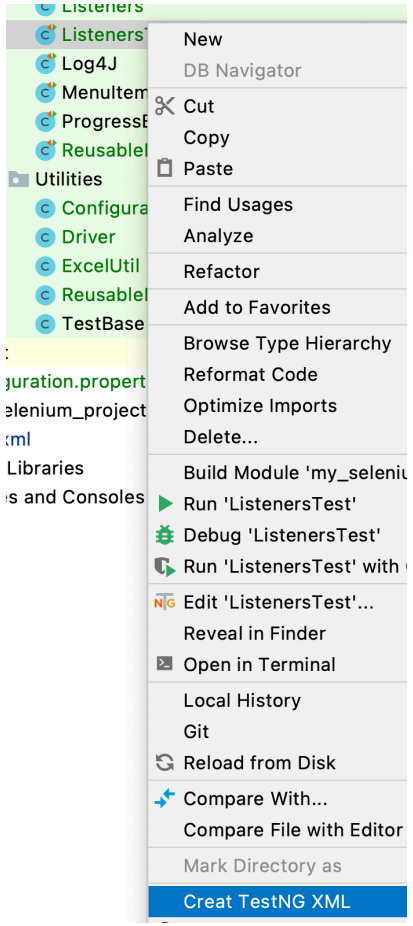
Test NG xml plugin

TestNG file lar TestNG xml plugin i ile de kolaylikla olusturulabilir.
Bu özellik icin TestNG xml plugin in intelliJ e yüklenmesi gerekir

1

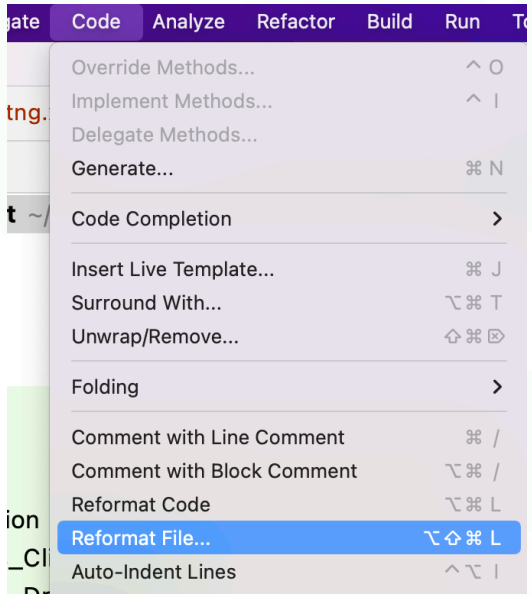


2



3

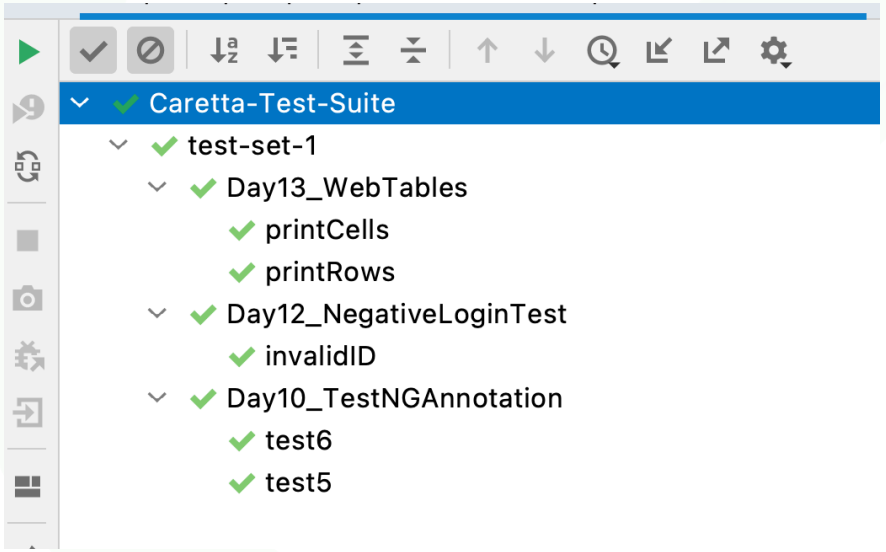
4



Test NG xml plugin

- 1. Java sinifina saga tikla
- 2. Create testing xml file
- 3. File ismini değiştir : belirlimetotlar.xml
- 4. Code > Reformat code (CONTROL + ALT + L) - (COMMAND +OPTION + L)
- 5. İstenilen Duzenlemeleri yap

Metotlari run etme



Test NG

XML File Class Work Istenen Method'lari

Calistirma

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

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >
<suite name="Regression Suite">
    <test name="Regression1">
        <groups>
            <run>
                <include name="regression-tests" />
            </run>
        </groups>
    </test>

<!--    TUM PROJEDEKI KLASORLERI ARA VE regression-tests isimli @Test caseleri calistir-->
    <packages>
        <package name="techproed.*"></package>
    </packages>
</suite>
```

Test NG

XML File Class Work Tum Testleri Calistirma

➢ **Yeni bir xml dosyasi olusturalim : regressiontest.xml**

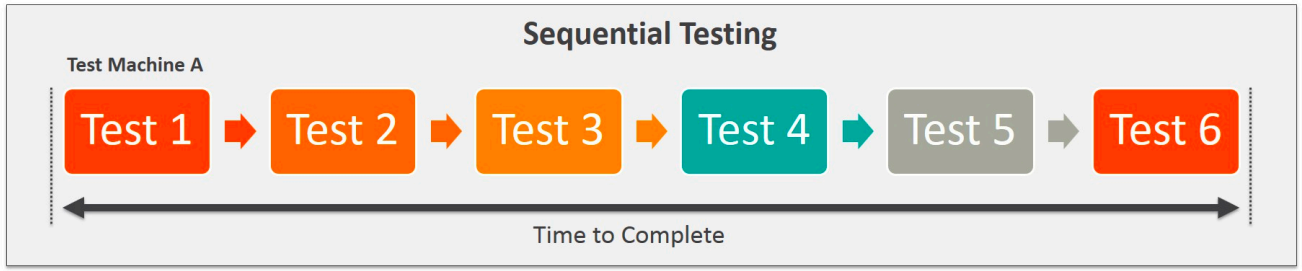
```
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >
   <suite name="minor regression" verbose="2">
            <test name="Regression1">
                     <packages>
                        <package name=“techproed.*"></package>
                     </packages>
             </test>
  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

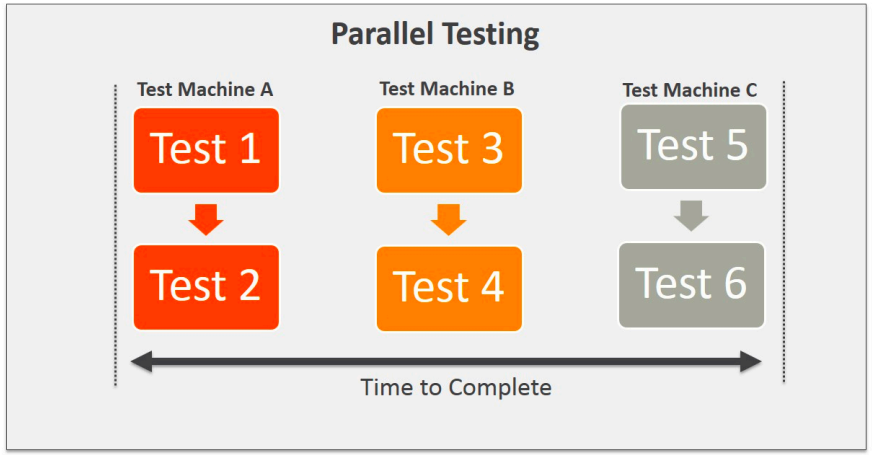
TestNG Parallel Testing

Test NG

- **Paralel test ayni anda birden fazla test case'i eszamanli olarak calistirmak demektir.**
- **Paralel test calisma suresini azaltir, dolayisiyla zaman kazanmak icin parallel test kullanilir.**

- **parallel :** paralel test icin kullanilir
- **thread-count :** Thread(code yada tarayici) sayisi





Test NG parallel testing

- **********
- TestNg'de paralel **testNG xml dosyasi** kullanilarak yapilir
- **xml file:**
  - **parallel :** paralel test icin kullanilir
  - **thread-count :** Thread(code yada tarayici) sayisi
- **********
- **Data provider** ile de ayni sekilde parallel test yapılabilir. Bu data kisitli bir yöntemdir.

```
@DataProvider(parallel = true)
```

- **xml file:**
  - **parallel :** paralel test icin kullanılır
  - **data-provider-thread-count :** Thread(code yada tarayici) sayisi

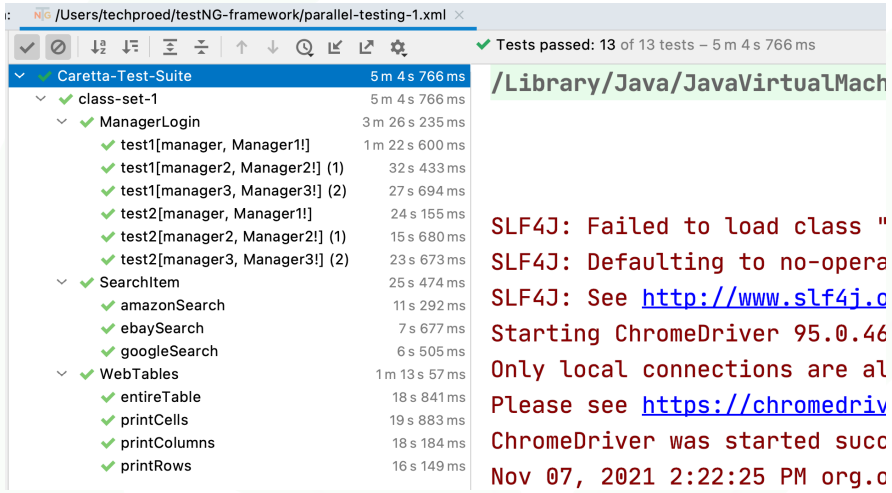
Test NG parallel testing

- Yeni bir klasör olustur
  - paralleltest
- 3 classes olustur:
  - ManagerLogin
  - SearchItem
  - WebTables
- Xml file I olustur : parallel-testing.xml

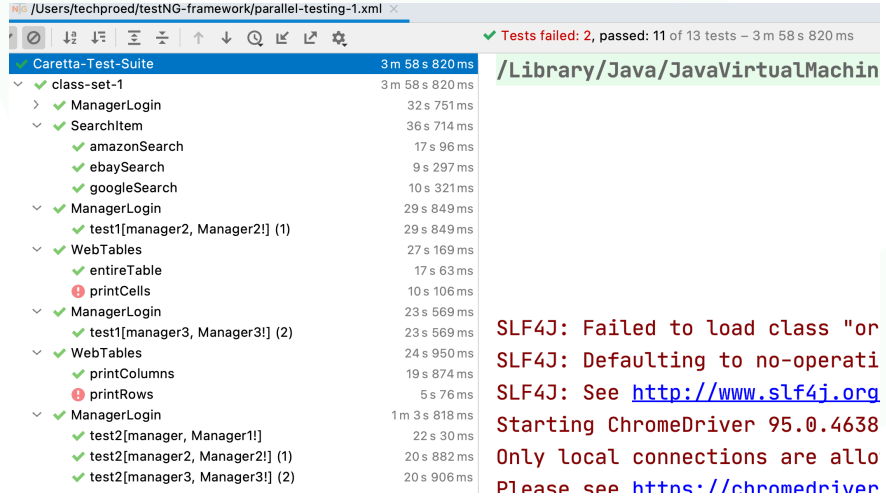
parallel-testing-1.xml

```
1  <!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
2  <suite name="Caretta-Test-Suite" parallel="classes" thread-count="2">
3
4      <test name="class-set-1">
5          <classes>
6              <class name="com.techproed.paralleltesting.ManagerLogin"></class>
7              <class name="com.techproed.paralleltesting.SearchItem"></class>
8              <class name="com.techproed.paralleltesting.WebTables"></class>
9          </classes>
10     </test>
11
12 </suite>
```

Sequential

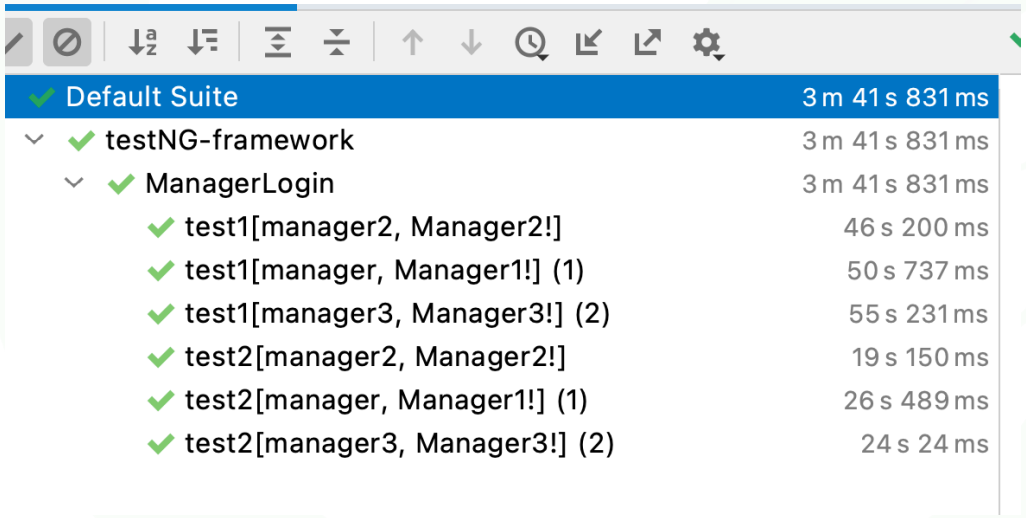


Parallel



Test NG parallel testing with Data Provider

```
//    parallel=false is the default value. If parallel=false, t
@DataProvider(name = "manager-profiles", parallel = true)
public Object [][] getData(){
    String [][] managerProfile= {
            {"manager","Manager1!"},
            {"manager2","Manager2!"},
            {"manager3","Manager3!"}
    };
    return managerProfile;
}
```



BATCH : 129 DT-NT

LESSON : **SELENIUM**

DATE : 06.05.2023

SUBJECT : **Listeners**

ZOOM GİRİŞLERİNİZİ LÜTFEN **LMS** SİSTEMİ ÜZERİNDEN YAPINIZ

TestNG Listeners

TestNG Listeners

https://testng.org/doc/documentation-main.html#testng-listeners

- **********Listeners NEDIR?**********
- TestNG Listeners (@Listeners) selenium kodlarini **"DINLEMEK"** icin kullanilir.
- Listenerslar yardımıyla test case **sonuclarini**, hata mesajlarını(**exceptions yada errors**), yada testlerin **baslama**, **bitme**, **gecme(PASS)**, veya **kalma(FAIL)** durumlarini dinleyebiliriz.
- Hatta sayfa geçişleri, button tıklama, data girme gibi daha detaylı kod dinlemeleri yapilabilir.

- **********Listeners NEDEN kullanilir?**********
- Test Automasyondaki PASS, FAIL, SKIP, ERROR durumlarında raporlamalar icin(Test Fail ettiginde ekran goruntusu al gibi)
- Loglama(mesaj yazdirma) islemleri icin. Ornegin **login** Test Case Fail olma durumunda "login test case Fail etti" yazdirilabilir

- **********

ITestListener:

- **`onStart(ITestContext arg0)`**
  - Tum Testlerden once tek bir sefer calisir (@BeforeSuite benzeri)
- **`onFinish(ITestContext arg0)`**
  - Tum Testlerden sonra tek bir sefer calisir (@AfterSuite benzeri)
- **`onTestFailure(ITestResult result)`**
  - Sadece FAIL eden testlerden hemen sonra calisir
- **`onTestSkipped(ITestResult arg0)`**
  - Ignore(Atlanan) edilen testlerden hemen sonra calisir
- **`onTestStart(ITestResult arg0)`**
  - Her bir Test den sonra birer sefer calisir (@AfterMethod benzeri)
- **`onTestSuccess(ITestResult arg0)`**
  - Sadece PASS eden testlerden hemen sonra calisir
- **`onTestFailedButWithinSuccessPercentage(ITestResult arg0)`**
  - Belirli bir test FAIL sonrasi calisir

Listeners

- 1. Utilities de **Listeners** class olustur.
- 2. Implement **ITestListeners interface**. Bu TestBase class gibidir.
- 3. **ITestListeners** metotlarını ekle.
- **********
- 4. tests package da, yeni bir package olustur : listeners
- 5. listeners package da yeni bir class olustur : ListenersTest1
- **********
- **NOT :** Test Classlari ile Listenerlari iliskilendirmenin 2 yolu vardır
    1. **@Listeners annotation :** `@Listeners(techproed.utilities.Listeners.class)`
    2. **Xml file :** `listeners taginin xml file da kullan`

```
<listeners>
    <listener class-name="techproed.utilities.Listeners"></listener>
</listeners>
```

Listeners Test 1

- 1.listeners paketi olustur
- 2. **ListenersTest1** classi olustur
- 3. 4 tane metot olustur: **test1(PASS), test2(FAIL), test3(SKIP), test4(NOSUCHELEMETNEXCEPTION)**
- 4. Test classi Listeners class ile iliskilendir : @Listeners(utilities.Listeners.class)

Listeners Test 2

- 1. listeners paketi olustur
- 2. **ListenersTest2** classi olustur
- 3. 4 tane metot olustur: **test1(PASS), test2(FAIL), test3(SKIP), test4(NOSUCHELEMETNEXCEPTION)**
- 4. Test classi Listeners class ile iliskilendir
  - xml file i olustur
  - Ismi değiştir: listeners1.xml
  - Code > Reformat code(CONTRL + ALT +L)
  - Listeners tagini suite taginden hemen sonra kullan

```
<listeners>
    <listener class-name="techproed.utilities.Listeners"></listener>
</listeners>
```

Failed Test caseleri TestNG de tekrar calistirma

1. Util Class olustur : **ListenersRetry**

```
import org.testng.IRetryAnalyzer;
import org.testng.ITestResult;
public class ListenersRetry implements IRetryAnalyzer {
    private int retryCount = 0;
    private static final int maxRetryCount = 1;
    @Override
    public boolean retry(ITestResult result) {
        if (retryCount < maxRetryCount) {
            retryCount++;
            return true;
        }
        return false;
    }
}
```

```
package utilities;
import org.testng.IRetryAnalyzer;
import org.testng.ITestResult;
public class ListenersRetry implements IRetryAnalyzer {
    private int retryCount = 0;
    private static final int maxRetryCount = 3;
    @Override
    public boolean retry(ITestResult result) {
        if (retryCount < maxRetryCount) {
            retryCount++;
            return true;
        }
        return false;
    }
}
```

2. Test sinifi olustur : ListenersTest3. Listeners Retry i iliskilendir : **@Test(retryAnalyzer = techproed.utilities.ListenersRetry.class)**

3. Sonuc : Eger herhangi bir test case FAIL olursa, bu test case 3 kez data run edilecektir.

Running Failed Test cases in TestNG

- 1. Failed test caseleri tekrar run etme islemini testNG xml file ile tum test caselere automatic olarak uygulayabiliriz.
  - Util class olustur : **ListenersRetryAnalyzer**
  - Implement **IAnnotationTransformer** interface
  - **transform** metotunu olustur
  - Metot da setRetryAnalyzer kullan: **annotation.setRetryAnalyzer(ListenersRetry.class)**
- 2. xml file da bu util classi test caselerle ilişkilerdir

```
<listeners>
    <listener class-name="techproed.utilities.ListenersRetryAnalyzer"></
listener>
</listeners>
```

- 3. Bu Test sinifini calistirdigimda, FAILE test caseler otomatik olarak calisacakdir

TestNG REPORTS

- Smoke test e saga tiklayip xml dosyasi olustur
- Run Et
- Projeye tıkla > Run > Edit configuration
- Xml dosyasını sec > listeners > + a tikla > reports lari ekle
- APPLY > OK
- XML i run et
- Projeyi yenile
- test-output dosyasında raporlar cikmis olacak
- **********
- Eclipse: test-output dosyası testng non xml raportlari otomatik olacak oluşur
- IntelliJ: bu dosyanın oluşması icin mini bir ayar gerekir
- ******TRABLE SHOOOTING***
- 1. Xml dosyası classa saga tiknayip create xml seklinde oluşturulmalı
- 2. Xml RUN edilmeli
- 3. proje>run edit configurations>listeners> rapolari ekle ve kaydet
- 4. RAPORTARIN CIKMASI ICIN BU ILISKILENDIRDIGIM XML DOSYASI RUN EDILMESI