**Veri Sağlayıcı ve TestNG XML:**

Yazılım oluştururken, her zaman farklı bir veri kümesiyle farklı şekilde çalışmasını isteriz. Aynı yazılım parçasını test etmek söz konusu olduğunda, onu tek bir veri kümesiyle test etmek haksızlık olamaz. Burada yine, sistemimizin desteklemesi beklenen tüm kombinasyonları aldığını doğrulamamız gerekiyor. Bunun için test skiplerimizi parametreleştirmemiz gerekiyor.

**Selenyumda parametrelendirme**

Selenium'da parametrelendirme, çalışma zamanında uygulamaya birden çok veri iletmek için test komut dosyalarını parametrelendirme işlemidir. Farklı değerler kullanarak test senaryolarını birden çok kez otomatik olarak çalıştıran bir yürütme stratejisidir. Test komut dosyalarının parametrelendirilmesiyle elde edilen kavrama Veriye Dayalı Test adı verilir.

**TestNG'de Parametrelendirme Türü**

Parametreleştirmeyi daha net hale getirmek için, Selenium Webdriver - TestNG için en popüler çerçevede parametrelendirme seçeneklerini gözden geçireceğiz .

* TestNG'de parametreleştirmeyi başarmanın iki yolu vardır

1. Parametreler ek açıklaması ve TestNG XML dosyası yardımıyla.

@Parameters({“name”,”searchKey”})

1. DataProvider ek açıklaması yardımıyla.

@DataProvider(name=”SearchProvider)

Testng.xml'deki parametreler paket veya test düzeyinde olabilir

DataProvider'dan gelen parametre, parametre olarak Method ve ITestContext alabilir.

**TestNG.xml Nedir?**

TestNG.xml dosyası, testlerimizi düzenlememize yardımcı olan bir yapılandırma dosyasıdır. Test uzmanlarının birden çok test sınıfı oluşturmasına ve işlemesine, test süitlerini ve testleri tanımlamasına olanak tanır. Tüm test senaryolarını bir araya getirip tek bir XML dosyası altında çalıştırarak testlerin yürütülmesini kontrol ederek bir test uzmanının işini kolaylaştırır. Bu güzel bir konsept olmadan TestNG'de çalışmak zordur.

**TestNG.xml'nin Avantajları**

TestNG.xml dosyasının başlıca avantajları şunlardır:

1. Test yöntemlerinin paralel yürütülmesini sağlar.
2. Bir test yönteminin başka bir test yöntemine bağımlı olmasına izin verir.
3. Test yöntemlerimizi önceliklendirmemize yardımcı olur.
4. Test yöntemlerinin test grupları halinde gruplandırılmasına izin verir.
5. @Parameters ek açıklamasını kullanarak test senaryolarının parametreleştirilmesini destekler.
6. @DataProvider ek açıklamasını kullanarak Veriye dayalı testlerde yardımcı olur.
7. Beklenen sonuçların gerçek sonuçlarla doğrulanmasına yardımcı olan farklı iddia türleri vardır.
8. Test özetimizin daha iyi ve net bir şekilde anlaşılması için farklı türde HTML raporları, Kapsam raporları vb. vardır.
9. Günlük oluşturmaya yardımcı olan dinleyicileri vardır.

**TestNG.xml Dosyası Oluşturma Adımları**

TestNG'de, birden çok test sınıfını işlemek için TestNG.xml dosyasını oluşturmamız gerekir. Test çalıştırmamızı yapılandırmalı, test bağımlılığını ayarlamalı, sınıfları, test yöntemlerini, paketleri, testleri vb. Dahil etmemiz veya hariç tutmamız ve XML dosyasında da önceliği belirlememiz gerekir.

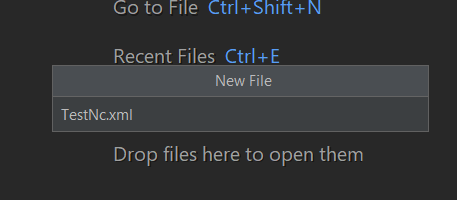
**Aşağıdaki adımları kullanarak Testng.xml dosyasını oluşturalım.**

**Aşama 1:** Proje klasörüne sağ tıklayın, Yeni'ye gidin ve aşağıdaki resimde gösterildiği gibi 'Dosya' yı seçin.

metin, ekran görüntüsü, ekran, siyah içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Adım 2:** Dosya adını şu şekilde ekleyin: **'Testng.xml'** Aşağıdaki resimde gösterildiği gibi ve enter yapın.



**Aşama 3:** Şimdi aşağıdaki XML kodunu testng.xml dosyanıza ekleyebilirsiniz. Gereksinimlere göre Test paketi adınızı ve Test adını seçebilirsiniz.

**Gerekli bilgileri sağlayan gönderi, testng.xml dosyası aşağıdaki gibi görünür:**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yukarıdaki XML dosyasında, etiketlerin sırasını doğru ve doğru bir şekilde görebilirsiniz. **Suite => Test Sınıfları => Sınıf.**

Burada, Süit adı “istenen class'lari calistirma”*>*

Test adı*”classlar”>*

XML dosyasında Suite ve Test'e herhangi bir isim verebiliriz. Ancak, Paket adınız ve Test Vakası adınızın bir birleşimi olan sınıflar etiketine doğru adı vermeliyiz.

**Paket adı “tests” ve test senaryosu adları şöyledir:**

<class name="tests.day16.C03\_Priority"/>  
<class name="tests.day17.C03\_SoftAssert"/>  
<class name="tests.day18.C03\_ConfigReader"/>

**Aşama 4:** Xml dosyasını çalıştıralım. Önce ctrl+s yapıp kaydedelim, daha sonra TestNG xml dosyasına sağ tıklayarak testi çalıştırın.

metin, ekran görüntüsü, ekran, siyah içeren bir resim

Açıklama otomatik olarak oluşturuldu

* Testng.xml dosyası çalıştıktan sonra sonuçları konsolda görebiliriz.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

XML, projemize isteklerimizi iletmek için hazırladığımız bir haberleşme dosyası diyebiliriz.Biz XML dosyamız sayesinde test class'ımızı açmadan run tuşuna basmadan istediğimiz test methodunu istediğimiz classı ve istediğimiz package'i çalıştırabilir ve package içinde hangi class'ın yada hangi test methodunun çalışmasını hangisinin çalışmamasını belirtebiliriz. HTML kodlarındaki gibi tag'ların açılıp kapanışına dikkat etmemiz gerekir. XML dosyamızı oluştururken ilk önce <!DOCTYPE komutu ile başlamamız gerekiyor. Bunu TestNG sayfasından da alabiliriz. Sonrasında suite tag'ı ile başlayıp hangi testi çalıştıracaksak ona göre .xml formatını yazmamız gerekir.

**Belirli Testi Çalıştırma**

➢ XML dosya oluştururken hiyerarşi (büyükten küçüğe sıralama) önemlidir. Her zaman suite ile başlayıp hangi seviyede test çalıştırmak istersek o seviyeye kadar sıralı olarak kademeleri yazmalıyız.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Belirli İstenen Package’leri Çalıştırma**

➢<tests> altında <packages> denilerek istenen pagkage seçilir.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Belirli İstenen Method’ları Çalıştırma**

➢ Eğer çalıştıracağımız class’lar farklı hiyerarşilere ait ise yine suite ile başlarız, sonra ayrışma kademesinden itibaren farklı hiyerarşi kümeleri oluştururuz.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Belirli İstenen Grupları Çalıştırma**

➢ Grup çalıştırmak için hem grup adını tanımlamak hem de nerede arayacağımızı belirtmek zorundayız.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

XML dosyasında belirli methodları gruplayıp çalıştırmak istersek öncelikle gruplandırmak istediğimiz methodları belirtmemiz gerekir, @Test notasyonundan sonra parantez açıp grup ismi belirtmemiz gerekiyor. .xml dosyasında test tag'ından sonra groups tag'ı altında run tag'ı içine gruplandırdığımız methodların grup ismi belirtmemiz gerekiyor ve son olarak grupladığımız methodlara hangi package altında olduğunu belirtmemiz gerekir.

**Tüm Testleri Çalıştırma**

➢Bu kısımda tüm projedeki package'lerin içindeki tüm testler çalışır.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Testleri Paralel Olarak Çalıştırma**

1. Xml dosyası oluşturulur.  
2. Koşulacak paralel ortamlar ayrı ayrı aşağıdaki şekilde test olarak eklenir.  
3. Gerekli parametreler gönderilir.  
4. Thread-count test kadar arttırılır.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Burada 2 ayrı browser da çalıştık.  
1.class'i bir browser 2. class'i 2.browser gibi sırayla çalıştırıyor. Paralel çalıştırmanın seviyesini burada classes olarak belirledik fakat büyük projelerde birbirine bağlı çalışan class'lar varsa bunu dikkate almak lazım o zaman class seviyesinde farklı browser'larda çalışmak sorun olabilir.

**TestNG'de Veri Sağlayıcı**

TestNG'deki Veri Sağlayıcı, bir kullanıcının karmaşık parametreleri iletmesi gerektiğinde kullanılan bir yöntemdir. Karmaşık parametrelerin Java'dan oluşturulması gerekir, örneğin karmaşık nesneler, özellik dosyalarındaki nesneler veya bir veritabanından veri sağlayıcı yöntemi ile aktarılabilir. Yöntem, @DataProvider tarafından açıklanır ve bir nesne dizisi döndürür.

**Dataprovider kullanan parametreler**

@Parameters ek açıklaması kolaydır, ancak birden fazla veri kümesiyle test etmek için Veri Sağlayıcı kullanmamız gerekir. Binlerce web formunu test çerçevemizi kullanarak doldurmak için, bize tek bir yürütme akışında çok büyük bir veri kümesi verebilecek farklı bir metodolojiye ihtiyacımız var. Bu veri odaklı konsept, TestNG'de @DataProvider ek açıklamasıyla elde edilir.

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

metin, ekran görüntüsü, ekran, televizyon içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yalnızca bir "ad" niteliği vardır. Name özniteliğini belirtmezseniz, DataProvider'ın adı, karşılık gelen yöntem adıyla aynı olacaktır.

Veri sağlayıcı, test yöntemine iki boyutlu bir JAVA nesnesi döndürür ve test yöntemi, M \* N türü nesne dizisinde M kez çağırır. Örneğin, DataProvider 2 \* 3 nesneden oluşan bir dizi döndürürse, karşılık gelen test olayı, her seferinde 3 parametre ile 2 kez çağrılır.

**Özet**:

* **Veriye Dayalı Test** oluşturmak için **parametrelendirme** gereklidir.
* TestNG destek parametre, iki çeşit kullanılarak **parametre + TestNG.xml @** kullanılarak **@DataProvider**’ın **Parametre + TestNG.xml @** parametreleri paketi düzeyinde ve test seviyesinde yerleştirilebilir. Eğer aynı parametre adı her iki yerde de bildirilir; test seviyesi parametresi, uygun seviye parametresine göre tercih edilir.
* @ Parametre + TestNG.xml kullanarak bir seferde yalnızca bir değer ayarlanabilir, ancak @DataProvider **bir 2d Object dizisi** döndürür.
* DataProvider farklı sınıfta mevcutsa, test yönteminin bulunduğu sınıf, **DataProvider statik yöntem** olmalıdır.
* **DataProvider** tarafından desteklenen iki parametre vardır: **Method** ve **ITestContext.**

**Kaynak:**

* <https://keytorc.com/blog/testng-nedir-nasil-kullanilir/>
* <https://tr.csstricks.net/8222949-dataprovider-and-testng-xml-parameterization-in-seleniumexample>
* <https://techproeducation.com/>

Zeliha Öznük