Sayfa 73

UZMAN

1. Maven nedir?

- Runner sınıfımı çağıran ve bağımlılıklarımı yöneten POM xml dosyası olarak adlandırılan bir oluşturma aracı ve komut istemi aracı.
- Maven, bir derleme otomasyon aracı veya bir proje yönetim aracıdır. Maven ile tüm kitaplıkları içe aktarabilir ve ayrıca

  proje yapıları. Maven'de birçok dahili şablonumuz var. Bu şablonlara arketip denir. Bir Maven temelde

  uygulamalarımızı derlemek için kullanılan bir araç.
- Komut İstemi mvn arketipi; oluşturmak
  - o Proje oluşturur
- Bir # seçin ve enter tuşuna basın
- Bir # seçin ve enter tuşuna basın
- Grup kimliği; com.nameOfProject (genellikle com.example.foo gibi ters çevrilmiş bir alan adı)
- ArtifactID; testmavenproject
  - o Sürüm girişi
  - o Paket girişi
  - o Y; giriş

2. Neden Maven? Projenizi etkili bir şekilde geliştirmenize nasıl yardımcı olur?

- Proje yapısını veya dağıtım, temizleme, paketleme, kavanoz ve çok daha fazlası gibi uygulamaları geliştirmeye ve yönetmeye yardımcı olur

  Java tabanlı proje için özellikler.
- Başka bir deyişle, bir Java aracıdır. Örnek bir proje veya iskelet proje oluşturmak istiyorsanız Maven'i kullanabilirsiniz. O bir

  otomatik inşa aracı. Maven, akıllı başlangıçlar ürettiği ve zeka varsaydığı basitliğe odaklandı

  varsayılanlar. Ayrıca, Uygulama Yaşam Döngüsü Yönetimindeki derlemeye yönelik aşamaları da kapsar, yani test etme, devreye alma, derlemeler

  yönetimi ve sürüm oluşturma.
- Projenin çok hızlı bir şekilde kurulmasına **yardımcı olur** ve build.xml gibi karmaşık derleme dosyalarından kaçınır. Maven, POM.xml gibi dosyaları gerektirir;

  yalnızca Maven'in amacına hizmet eder. POM.xml, Java Projenizin belirleyebileceği bir bağımlılıklar koleksiyonudur.

  Maven'e ve daha sonra Maven hepsini internetten indirecek ve daha sonra bir depoda, yani yerel

  depo, merkezi depo ve uzak depo.

3. Maven Artifact nedir?

- Artefakt, bir Maven deposuna dağıtılan ve genellikle bir JAR olan bir dosyadır.
- Bir Maven derlemesi, derlenmiş bir JAR ve bir "kaynaklar" JAR gibi bir veya daha fazla yapı üretir.
- Her yapının bir grup kimliği (genellikle com.example.foo gibi ters çevrilmiş bir etki alanı adı), bir yapay kimliği (yalnızca bir ad) ve bir

  sürüm dizesi. Üçü birlikte eseri benzersiz bir şekilde tanımlar. Misal:

```
<groupId> org.seleniumhq.selenium </groupId>
<artifactId> seleniumjava </artifactId>
<version> 3.11.0 </version>
```

- Bir projenin bağımlılıkları yapay olarak belirtilir.

4. Bana maven yaşam döngüsünü açıklayın?

- Komutlar yalnızca belirli **pom xml** dosyasının bulunduğu dizinde çalışabilir
- 3 yerleşik derleme yaşam döngüsü
  - o Varsayılan → Proje dağıtımınızı yönetir
  - o Temiz → Proje temizliğini yönetir
  - o Site → Projenin site belgelerinin oluşturulmasını yönetir

Sayfa 74

5. Bir derleme yaşam döngüsü aşamalardan oluşur

- Doğrula → Projenin doğru olduğunu ve gerekli tüm bilgilerin mevcut olduğunu onaylayın
- Derleme → Projenin kaynak kodunu çalıştır (hata olup olmadığını kontrol etme, yoksa → derleme başarısı)
  → Hedef klasör oluşturulur ve Raporlar burada saklanır
- Ölçek
  - o Derlenen kaynak kodunu uygun bir birim test çerçevesi kullanarak test edin.
  - o Kodun paketlenmesini veya konuşlandırılmasını gerektirmemelidir
  - o Mvn D (VariableName) = testname → Parametreye göre özel testler çalıştırın
- Paket → Derlenen kodu alın ve JAR gibi dağıtılmış bir biçimde paketleyin
- Doğrula → Kalite kriterlerinin karşılandığından emin olmak için entegrasyon testlerinin sonuçları üzerinde her türlü kontrolü çalıştırır
- Yükle → Bağımlılık olarak kullanmak için paketi yerel depoya yükleyin
- Dağıt → Derleme ortamında Bitti, son paketi diğer geliştiricilerle paylaşmak için uzak depoya kopyalar
  ve projeler

6. maven projesini tutulma projesine nasıl dönüştürürsünüz?

- Mvn tutulması

7. Java projeleri nasıl yapılır?

1. Klasörler / paketler oluşturun
2. Kitaplıklar / bağımlılıklar ekleyin
3. Sınıf dosyaları oluşturun
4. Derleyin
5. Testleri çalıştırın
6. Dağıtın

8. Bağımlılıklarınızı / kitaplıklarınızı nerede buluyorsunuz?

- Mvnrepository.com
- maven çalışmıyorsa projeyi güncelleyin
  - o Pom dosyanızın içinde bağımlılıklar olduğunda ve güncellemeyi kullandığınızda, maven JAR dosyalarını internetten çekecek ve
    projenize ekleyin

9. .m2 klasörü nedir?

- Jar dosyalarınızın / havuzlarınızın bilgisayarınızda kaydedildiği yer

10. POM xml dosyası nedir?

- Tüm projeyi yöneten bir dosya
- Bir maven komutunu çalıştırdığınızda, her şey pom.xml aracılığıyla yapılmalıdır

11. Araçların sürümleri?

- RestAssured 3.3.0 yayın tarihi: 2019-01-11

Sayfa 75

12. Log4j?

- Herhangi bir uygulama tarafından kullanılır
- Örnek: LOG4J2 → Apache'den
- Aktiviteyi kaydeder
- Dev, günlüklere bakacak, saate bakacak, IP adresine gidecek ve bir hata varsa neler olup bittiğini görecek
- Kaydediciler uygulamaların çok önemli bir parçasıdır ve gerçekleşen her adımı / olayı zaman damgasıyla birlikte tutar
- Normalde günlükler programlı olarak .log dosyasına yazılır
- Herhangi bir çerçeveye veya uygulamaya eklenebilecek hazır araçlar / kitaplıklar vardır.
- Java'da, en ünlü günlük kaydı kütüphanesi / çerçevesi apache'den LOG4J'dir.

13. Günlüklerin amacı?

- Uygulamayla ilgili olabilecek sorunları gidermemize yardımcı olun.
- Bazen uygulamada bir hata bulunduğunda, geliştiriciler öncelikle günlükleri kontrol ederler. Kullanıcının hangi adımlar olduğunu görmek için
  alma ve uygulama beklendiği gibi davranmadı.
- Günlükler, sorunun kaynağını bulmanıza yardımcı OLABİLİR (test açısından değil, uygulama açısından)

14. Test otomasyonunda günlüklerin rolü nedir?

• ● Test çalıştırmalarımızın durumunu görmek için konsol veya html raporuna bakıyoruz. Bir şey başarısız olursa, oradan buluruz.
• ● Çerçevemizde oturum açmayı uygularsak, otomasyon yürütme adımlarına bakmanın başka bir yolu olacak ve

testimiz başarısız olduğunda sorunu bulmamıza yardımcı olun

Sayfa 76

TESTNG & JUNIT

1. TestNG nedir?

• 500 test vakanız var → Her test vakası için bir Java Paketi ve 500 Sınıfı oluşturuyoruz

Müşteri sizden sadece 40 tanesini duman testi için çalıştırmanızı istedi → Blokları ve raporlama mekanizmasıyla Jasmine'de hallediyoruz.
• TestNG bir test çerçevesidir
• Merkezi kontrolör: farklı test senaryolarının çalıştırılmasını yönetin ve ardından raporlar, günlükler oluşturun
• Toplu yürütme: 100 test durumu ve bunları birer birer çalıştırın
• İsteğe bağlı yürütme: bazı test durumlarını atlayabiliriz

2. TestNG'deki iddialar nelerdir?

• Testi çalıştırdık ve başlık testi durumu başarısız oldu. Diğer test durumlarını etkilemeyeceği için betiğimizin durmasını istemiyoruz.
  o Kritik → dur / başarısız İddiası
    ▪ Bir boole bağımsız değişkeni ve String mesajı alır. Bir koşulun doğru olduğunu iddia eder. Değilse, bir Onay Hatası,

    verilen mesajla birlikte atılır.
  o Kritik değil → hata / devam SoftAssert
    ▪ Soft Assert, bir iddia başarısız olduğunda bir istisna atmaz ve sonraki adımla devam eder.

    ifade iddia.

3. JUnit ve TestNG arasındaki fark

• Ek açıklamalar; **JUnit:** @Test, @BeforeClass, @AfterClass, @Before, @After, @Ignore
TestNG : @Test, @BeforeTest, @BeforeClass, @BeforeSuite, @BeforeMethod,
@AfterTest, @AfterClass, @AfterSuite, @AfterMethod
• Her ikisi de bize yardımcı olmak için çerçeve test ediyor

otomasyon komut dosyalarını çalıştırma.
• TestNG, html raporu sağlar
• TestNG vardır @Hayalhanemersin

ek açıklama Salatalık ile aynı
Veriye Dayalı Senaryo Özeti
Test yapmak.
• TestNG'de paralel testler yapabiliriz,

ancak JUnit paralel olmayı desteklemiyor
testi, bu yüzden bunun için SauLab kullanıyoruz.
• TestNG destek grubu testi ancak JUnit

desteklemiyor