**Selenyum Grid**

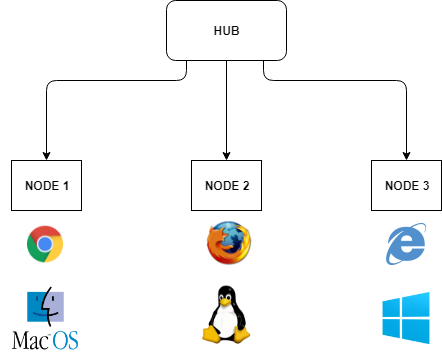
Selenium-Grid testlerimizi paralel olarak farklı ortamlarda, farklı tarayıcılarda ve farklı sayılarda koşmamıza imkân tanıyan bir sunucudur. Yani aynı anda farklı tarayıcılarda ve işletim sistemlerinde birden fazla test çalıştırabiliriz.

Kullanmamız için sebeplerimiz;

Ölçeklendirerek farklı işletim sistemlerinin farklı versiyonları olan tarayıcılarda koşabilir. Zaman Tasarrufu sağlar. Sunucuyu, ortamı çok hızlı bir şekilde ayağa kaldırır. Testleri paralel olarak çalıştırmak için birden fazla makine kullanarak testleri koşması hızlandırılabilir.

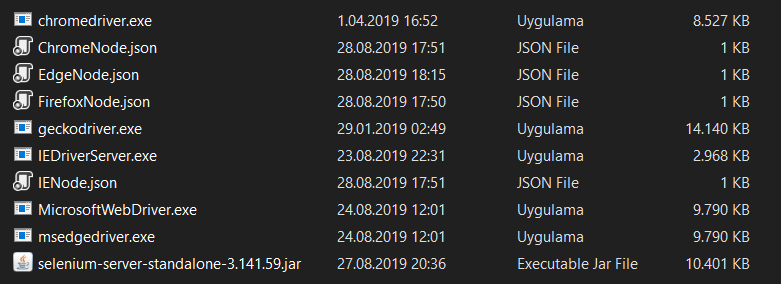
Selenium-Grid Hub ve Node’dan oluşur. Sunucu üzerinde sadece bir Hub ve birden fazla Node olabilir.

Hub: Testin yürütülmesi gereken, istenilen tarayıcıyı, tarayıcı versiyonu ve işletim sistemi hakkında ki bilgileri alır ve testleri bunları sağlayan destekleyen makineye yani node’a yönlendirir. Bir örnekle Hub’ı bir müdür ve Nodeları çalışan olarak düşünelim.  
Yapılacak bir iş olduğunu ve müdürün bu iş için uygun olan çalışanı bulup elindeki bu işi yönlendirdiğini düşünelim. Hub ve Node arasındaki ilişki bu ilişkiye benzer.Bir işçi(Node) seçildikten sonra Selenium test komutları önce Hub(müdür)’a ve Hub da atanmış olan Node makineye yönlendirir. Node tarayıcıyı başlatır ve aldığı Selenium test komutlarını koşar.

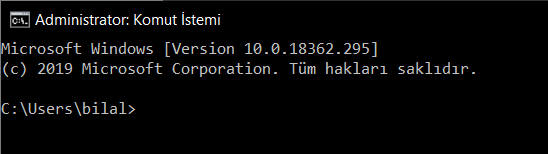
****

**Selenyum Grid Çalıştırılması**

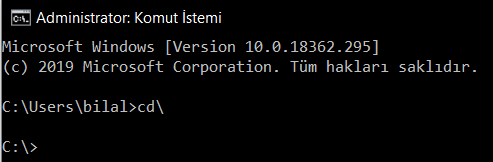
1. İlk olarak kurulum için kullanacağımız selenium-server-standalone-3.141.59.jar, web driverlar, ve node’ lar için json dosyalarını bilgisayarımız Yerel Disk C sürücüsüne kopyalıyoruz.



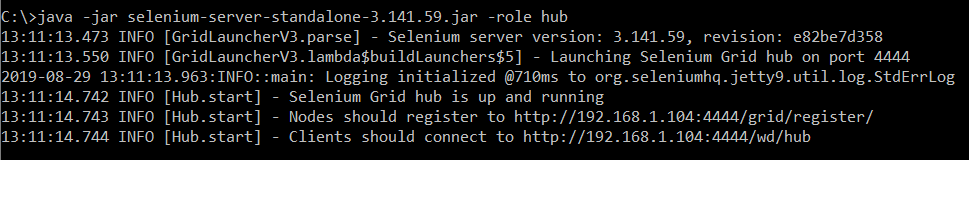
1. Belgeleri yukarıdaki gibi Yerel Disk C içine attıktan sonra Komut Satırını açıyoruz.



1. Komut satırı uzantımız varsayılan olarak C:\Users\KullanıcıAdı olarak açılır bizim dosyalarımız Yerel Disk C içinde olduğu için cd\ komutu ile C:\ dizinine geçiyoruz.

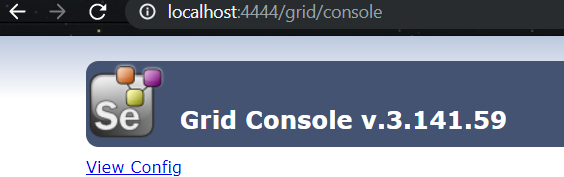


1. Görüntü yukarıdaki gibi olduktan sonra “java -jar selenium-server-standalone-3.141.59.jar -role hub” kodunu komut satırına yazarak Hub u oluşturalım.



Kodu yazdıktan sonra görüntü yukarıdaki görünecektir.

1. Tarayıcı üzerinde <http://localhost:4444/grid/console> adresine girerek Hub un çalışıp çalışmadığını da kontrol edebiliriz.



Görüntümüz yukarıdaki gibiyse sıkıntı yok demektir.

1. Hub’ umuz sıkıntısız şekilde çalışıyor şimdi nodeları çalıştırabiliriz. Açık olan komut satırını kapatmayıp yeni komut satırı açıyoruz. cd\ komutu ile Yerel Disk C dizinine geçiyoruz.

Chrome Driver Eklemek İçin;

java -Dwebdriver.chrome.driver="C:\chromedriver.exe" -jar selenium-server-standalone-3.141.59.jar -role node -nodeConfig ChromeNode.json -hub <http://localhost:4444/grid/register/>

Edge driver eklemek için;

java -Dwebdriver.edge.driver="C:\MicrosoftWebDriver.exe" -jar selenium-server-standalone-3.141.59.jar -role node -nodeConfig EdgeNode.json -hub <http://localhost:4444/grid/register/>

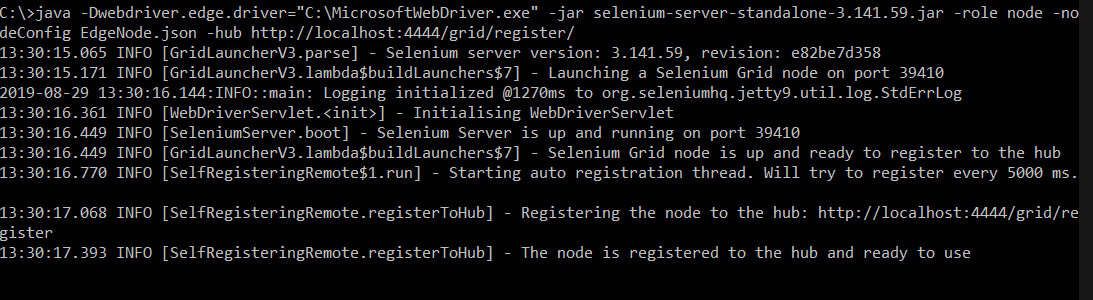
Firefox driver eklemek için;

java -Dwebdriver.firefox.driver="C:\geckodriver.exe" -jar selenium-server-standalone-3.141.59.jar -role node -nodeConfig FirefoxNode.json -hub <http://localhost:4444/grid/register/>

İnternet Explorer driver Eklemek için;

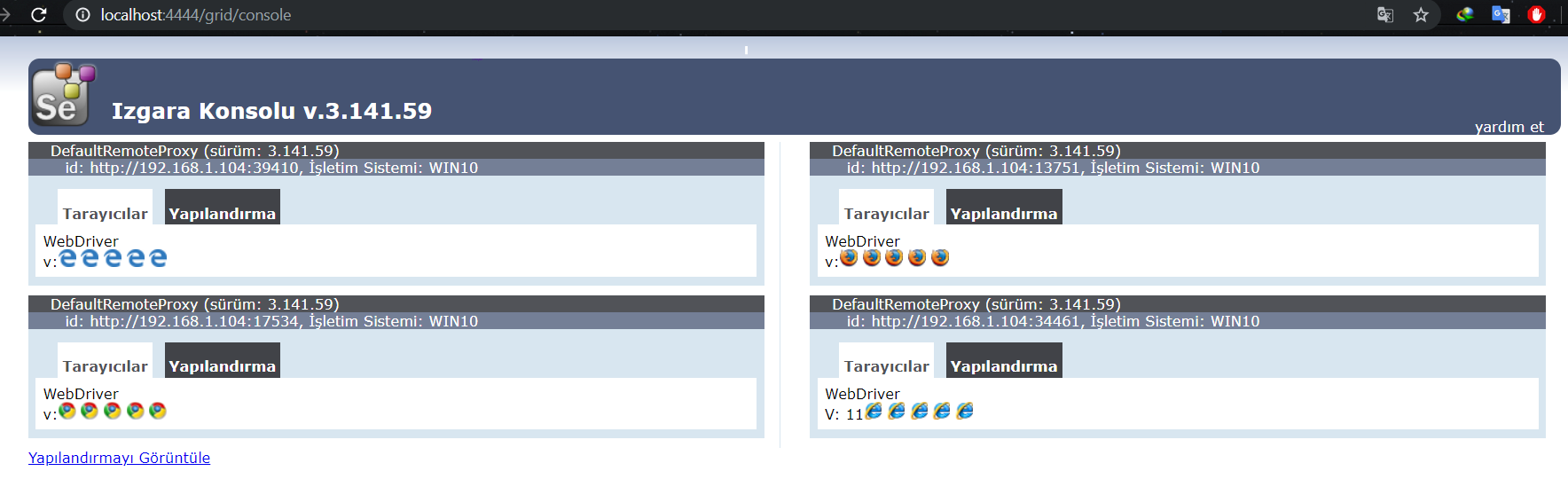
java -Dwebdriver.ie.driver="C:\IEDriverServer.exe" -jar selenium-server-standalone-3.141.59.jar -role node -nodeConfig IENode.json -hub http://localhost:4444/grid/register/

kodlarını komut satırına yazarak istediğimiz driveri node olarak ekleyebiliriz. Ben örnek olması açısından farklı komut satırları açarak bütün Driverleri node olarak ekledim.

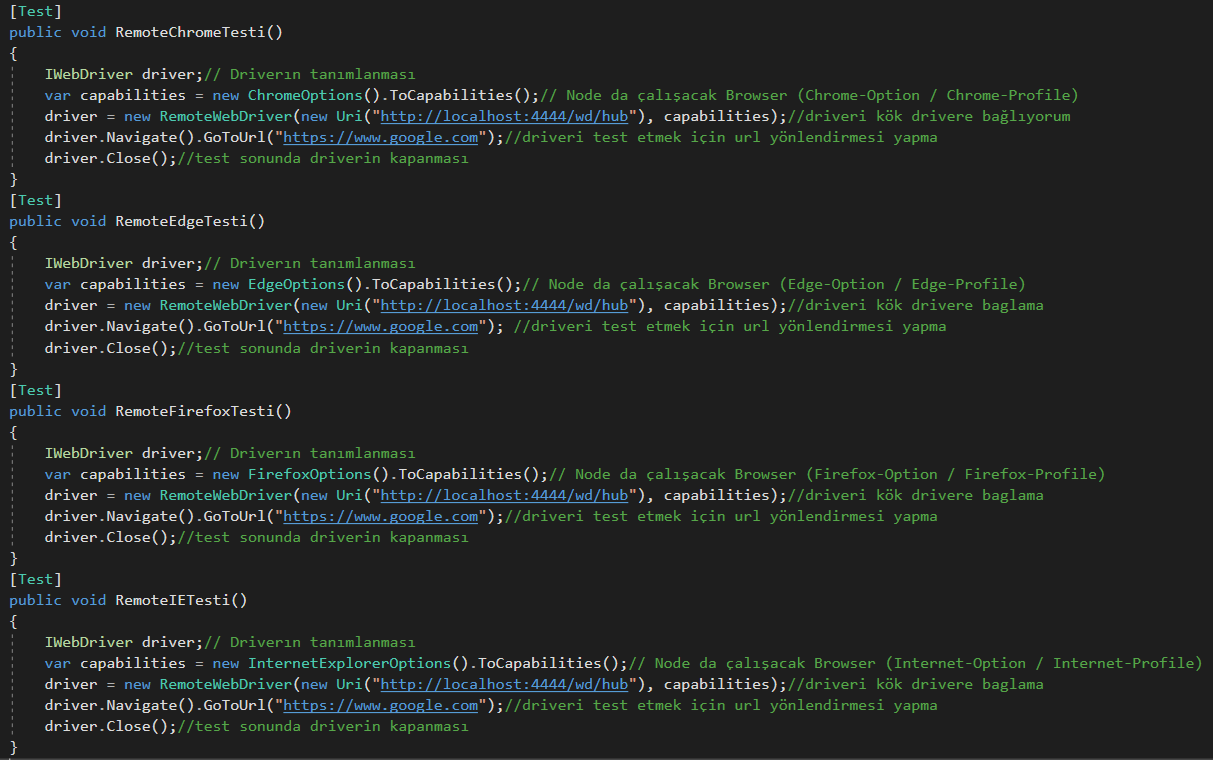


1. Ekledikten sonra komut satırının görüntüsü yukarıdaki gibi olmalıdır.

<http://localhost:4444/grid/console> adresinde ise eğer hepsini eklediyseniz görüntü aşağıdaki gibi olacaktır.



1. Node’ları da ekledikten sonra şimdi test etmeye başlayabiliriz. Açık olan komut satırlarını kapatmadan Visual studio üzerinde yeni bir Nunit test projesi açalım ve Nuget Packages kullanarak Selenium.Firefox.WebDriver, Selenium.WebDriver, Selenium.WebDriver.ChromeDriver, Selenium.WebDriver.IEDriver paketlerini yükleyelim.
2. Test Classımıza OpenQA.Selenium, OpenQA.Selenium.Remote, OpenQA.Selenium.Chrome, OpenQA.Selenium.IE, OpenQA.Selenium.Edge, OpenQA.Selenium.Firefox namespace lerini ekleyelim. Şimdi Testlerimizi Yazmaya başlayabiliriz.



1. Yukarıdaki fotoğrafta görüldüğü gibi her driver için bir test metodu açıp testleri gerçekleştirdik.
2. Kısaca bir driver için yazdığımız kodları açıklayayım;

- İlk olarak IWebDriver driver koduyla driverimizi oluşturuyoruz.

- var capabilities = new InternetExplorerOptions().ToCapabilities(); kodu ile Node da çalışacak Browser türünü tanımlıyoruz.

-driver = new RemoteWebDriver(new Uri("http://localhost:4444/wd/hub"), capabilities); koduyla driverimizi Hub’ a bağlıyoruz.

- driver.Navigate().GoToUrl("https://www.google.com"); koduyla driveri test etmek için url yönlendirmesi yapıyoruz.

1. Test leri Çalıştırdığımız zaman ilgili driverin açılıp Google.com sitesine yönlendirildiğini görüyoruz. Bunları yaparken bizim oluşturmuş olduğumuz node’ları bağlanarak işlemi gerçekleştirdiğini görüyoruz.