Bu yazımda git versiyon kontrol sisteminin Branches(Dallar) konusunu anlatacağım ...

Git versiyon kontrol sisteminde dalları oluşturmak farklı kişiler ile aynı proje üzerinde farklı konular da çalışmanızı , görevlendirme imkanını verir.Dallar projenizin birer kopyasıdır.Bu dallar birleştirilmediği (merge işlemi) sürece birbirlerini etkilemezler.Projede herkesin işi bittiğinde “merge”(birleştirme) işlemi yapılır.Git ana dal olan “master” dalı ile başlar , “master” dalının kopyası alınarak projemizi farklı dallara ayırabiliriz.Şimdi bu işlemi örnekleyelim ;

Sırası ile istediğiniz dizinde bir klasör oluşturup bu klasörü git deposu haline getirelim ve uzak sunucumuzu ayarlayalım.Daha sonra klasörde 4 txt dosyası oluşturalım.

```
mkdir branches-example
cd branches-example
git init
git remote add origin https://github.com/serkankaya/git-cli-test
touch dosya{1..4}.txt
```

Şimdi bu işlemi index e ekleyip commitleyelim

```
git add .
git commit -m "first commit"
```

Şimdi ise elimizde kaç dal var buna bakalım ;

```
git branch
* master
//uzak sunucumuz ayarlanmış ve uzak sunucudaki dalları görmek istersek
aşağıdaki komutu yazalım ;
git branch --all
```

Görüldüğü gibi sadece git ile birlikte gelen “master” dalı bulunmakta … Şimdi farklı bir arkadaşın projemizde çalışmasını istediğimizi düşünelim ama aynı anda bizde çalışacağız.Bizim master dalında çalıştığımızı diğer arkadaşın farklı bir dal da çalışacağını varsayalım.İlk önce ortamı hazırlayıp bir dal daha oluşturalım.Bu dala da youtube videolarından git cli öğrenmeme katkısı olanlardan ***Barış Aslan*** ismini verelim

```
git branch barisaslan
```

Şimdi dalları kontrol edelim ;

```
git branch
* master
  barisaslan
```

Gördüğümüz üzere “master” ve yeni oluşturduğumuz “barisaslan” dalını görüyoruz.”master” dalının başında ki * işareti bulunduğumuz dalı göstermekte …

Şimdi “barisaslan” dalına geçelim

```
git chechout barisaslan
Switched to branch 'barisaslan'
```

Şimdi tekrar dal kontrolü yapalım ;

```
git branch
  master
* barisaslan
```

Şimdi * işareti “barisaslan” dalında yani bu daldayız.Bu adımdan sorna farklı bir dosya daha oluşturalım ve commit edelim …

```
touch isimler.txt
git commit -m "barisaslan commit"
```

diff komutu ile bulunduğumuz dal ile diğer dal ile arasında ki farkları görelim ;

```
git diff master
diff --git a/isimler.txt b/isimler.txt
new file mode 100644
```

Gördüğünüz üzere “isimler.txt” dosyamızın yeni bir dosya oluşturulduğunu göstermekte … bu adımdan sonra diğer dal olan “master” dalına geçelim ve dosyalarımızı listeleyelim ;

```
git chechout master
ls
dosya1.txt  dosya2.txt  dosya3.txt  dosya4.txt
```

Gördüğünüz üzere master dalındayız dosyalarımızı listelediğimizde “isimler.txt” dosyasını göremiyoruz.Bunun nedeni biz “barisaslan” dalında bu dosyayı oluşturmuştuk.Bu dala geçip dosyayı kontrol etmeyi de size bırakıyorum …

Projemizde arkadaşımız ile çalıştık ve projemiz bitti varsayalım.Eee şimdi iki dal var ve iki kişi de farklı işleri bitirdiler.Sıra geldi dalları birleştirmeye yani namı değer **merge** işlemini yapmaya … bunun için öncelikle “master” dalına geçiyoruz … zaten şu an master dalındayız …

```
git merge barisaslan
```

Bu işlem ile dal birleştirme işlemimizi de yaptık sonuçları görelim ;

```
```

```
Updating 59f3ef0..d8646ea
Fast-forward
 isimler.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 isimler.txt
```

Görüldüğü üzere “isimler.txt” dosyası iki dal arasında ki tek fark idi …”master” dalındayız ve dosyalarımızı kontrol edelim ;

```
ls
dosya1.txt  dosya2.txt  dosya3.txt  dosya4.txt  isimler.txt
```

Birleştirme ile “isimler.txt” dosyasını master dalında birleştirdik …

isterseniz dosyalarınızı daha önce gördüğümüz üzere uzak depoya atabilirsiniz …

Umarım faydalı olmuştur

Serkan Kaya
Full Stack Java Developer