

# Git ve Github

Mehmet Kutay  
Akpinar

## Git Terminal

ls : tüm dokümları gösterir.

pwd : print working directory. Güncel bulunduğu klasör.

cd : change directory. cd KlasörAdı  
o klasöre girer. cd .. case sensitive  
↓

clear : ekranı temizler. Bir önceki yere gider

mkdir : make directory. Klasör oluşturulur.  
mkdir KlasörAdı.

touch : folder içinde uzantılı dosya oluşturulur.  
touch dosya.txt

rm : remove . remove KlasörAdı . uzantı.  
Uzantılı dosya kaldırır.

rm -rf : klasörü siler. Remove folder .

git : Dökümantasyon gösterir.

## Git'e Kullanıcı Adı ve Mail Kaydetme

git config --global user.name "İslam"

git config --global user.name } Show

git config --global user.email "email.."

commit : Kaydetme

branch : İkiye ayırma

git add : commitlenede önce yapılır.

git/reposting : git iceren genel program

git status : gitin güncel durumunu gösterir.

git init : folder gitler. Bu klasör artık repository olur.

⚠️ Herhangi bir yerde git init çalıştırmadan önce git status çalıştır ve kontrol et. 2 defa git init olmasın.

ls -la : gizli dosyalar da gösterir.

`git add DosyaAdı` → ilerde repoya commit etmek igin.

`git commit` → commitler.

`git log` ⇒ Tüm commitleri ve kim tarafından ne zaman yapıldığını gösterir.

`git add .` → Tüm dosyalar ekler.

⚠ `gitignore` oluştur. içinde ignore olacak dosyayı yaz. Repoda asla bu dosya olmaz.

Normal add. kullanın bile dmaz.

## BRANCH

Ana branch: Master.

`Head`: git içerisinde neredeyiz. Gerelde son commit. ↳ güncel olarak

`git branch`: yeni branch. `git branch X`  
↓ `X` branchi oluştur.  
Sadece bu tür branchleri gösterir.

`git switch X`: `X` branchine geçer.

git switch master : ana branche girer.

git merge : branch ile master birlestirir.

git merge X : X ile master birlestirir.

---

Fast forwarding : Aktigim branchte birseyler  
yaptim ama masterda birsey yapmadim.

git restore DosyaAdi : DosyaAdi son commit  
de durumdaysa ona dondurur.

stash : ( git stash ) : yaptigim degisiklikleri  
indexlemeden ( git add kullanmadan ) ve committeneden  
etiketlemek.

git stash pop : stashden onceki durum geri getinir.

git stash list : guzel stash listesini verir.

git stash apply stash @{0} : 0'inci indexdeki  
stashi geri dondurur.

---

## Geri Dönme Checkout

---

git checkout commit\_id : 0 commit durumunu gider

Detached head : Head son committe degil. Basika yani

gösterir.

Ya son commit'e geri döneriz  
ya da branch agmanız lazym.

**git reset commit\_id**: O commit'e döner AMA sonrası  
commitleri siler. Ama logda commit  
değişikleri kalsın. Silinmez.

**git reset --hard commit\_id**: O commit'e döner. Ama  
bu sefer logdaki commit değişiklerini  
de siler.

**git revert commitId**: O commit revert edilir, Geri alınır ama  
hala branch'te kalır.  
Değişiklikler silinir.

## Diff

**git diff HEAD**: En sonki commit'e göre değişiklikler  
gösterir.

**git diff commit\_id1 commit\_id2**: İki commit arasıındaki  
farkı gösterir.

**git diff branchname1 branchname2**: İki branch arasıındaki  
farkı österir.

## Rebase

Benim masterdaki tim commitleri sıralayıp ardından kendi branchindeki commitleri sırasıyla koymak.



⚠️ Sıkıntı oluşturabilir.

Githubda paylaştıkları sonra rebase edip tekrar paylaşınca tim herkes etkileşir.

Hem log temizlenerek hem de commit tarihi düzelenek için kullanılır.

## Github Repo Oluşturma

`git push origin master`: Master branchı obak push

`git branch -r`: remote branchlerini gösterir.

`git fetch origin master`: Githubda değişiklik yapıldığa berim pc'indeki programda değişiklik yapılmaması için kullanılır.

! Remote branchlerə gələşken git checkout  
kullanır.

git pull = git fetch + git merge.

git clone https : Adəsleki projeyi bilgisayara  
indirir.

---

Son

