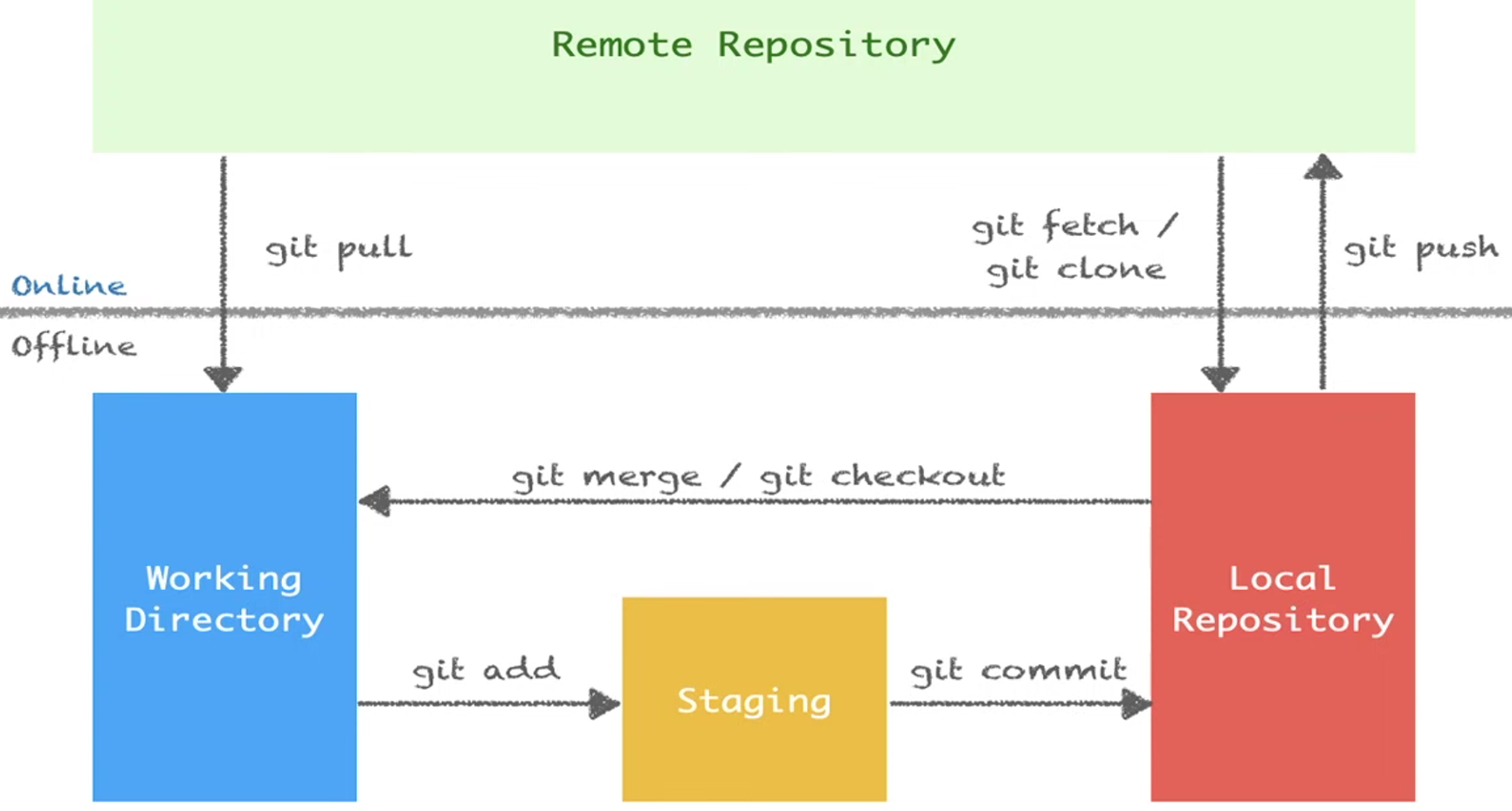
# Git



Git bir versiyon kontrol sistemi, github da depomuz.

Git’le depolar oluşturuyoruz. Burada projemizi depoluyoruz. Yaptığımız projeleri belirli versiyonlarda tutuyor. Kim neyi ne zaman yapmış, hnagi sürüm şu an yayında, kimler hangi uygulamayı geliştiriyor,..

Github git deposunu yedeklediğimiz/yüklediğimiz bulut platformu

Uygulama geliştirirken git’i kullanıyoruz, yüklemek istersek Github gibi platformalara yüklüyoruz.

Repository veya repo, projelerinizin dosyalarının depolandığı bir dizindir. GitHub'ın alanında veya bilgisayarınızdaki yerel bir depoda bulunabilir. Dosyalar, fotoğraflar, sesler veya projenize alakalı her şeyi repository'inizde depolayabilirsiniz.

**Terminal Kodları**

**ls**: klasör içindekileri listeler

**ls** *name* 🡪 bu dizindeki name dosyasının içindekileri gösterir.

**ls -a**: gizli dosyaları da gösterir

**dir**: listeleme, renksiz

**ls --help**: ls ile kullanılacak komutları görme

Genelde tek harfli komutlarda tek tire, çok harfli komutlarda çift tire kullanılıyor. (ls -a, ls –all)

**clear**: ekranı temizler.

**cd:** change directory. terminal penceresinde bulunduğunuz klasörü değiştirmenizi sağlar. (cd Desktop)

**cd ..** : bulunduğunuz dizinin bir üst dizinine gitmek için kullanılır

(/home/kullanici/adim/dosya 🡪 /home/kullanici/adim giderek dizini değiştirir)

**mkdir** *name*: make directory. klasör oluşturur.

**touch** *name*: dosya oluşturur. Html, txt,..

cd *dos*: bikaç harf yazıp tab tuşuna basarsan dosya ismini tamamlar.

cd *d*: yazıp iki kere taba basarsan d harfi ile başlayanları gösterir.

Yukarı ok tuşu: daha önce kullandığın komutlar arasında geçiş yapabilirsin

**cat** <buraya yazdığını geri döndürür.> çıkmak için: ctrl+d (end of file), ctrl+c (tüm uygulamalarda geçerli sonlandırma işlemi), ctrl+z (durduruyor)

cat >file.txt 🡪 cat içine yazdıklarımı yoksa oluşturarak file.txt dosyasına yönlendir. Onun içinde ne varsa boşverip üzerine bunları yaz

cat file.txt 🡪 txt içindekileri göster

cat file.txt>file2.txt 🡪 file içindekileri file2’ye yaz

cat file.txt>>file2.txt 🡪 var olanları tutup file2’nin içine sona ilave olarak yazıyor

cat >>file.txt 🡪 file içine sona ilave ederek yaz

cat file.txt file2.txt > file3.txt 🡪 file3 oluşturup içine ilk olarak file sonra da file2 dekiler yazdı

**rm** file3.txt 🡪 silme

**cp** file2.txt file3.txt 🡪 copy. file2’dekileri kopyalayıp file3 oluştur/yaz

“cat >>” sadece içeriği kopyalarken “cp” dosyayı tüm özellikleriyle kopyalıyor

**rmdir** *dosya* 🡪 boş dosya silme

**rm** **-r** *name* 🡪 name adlı dosyayı silme

rm -r dosya/ 🡪 dosyayı içindekilerle slime (r: recursive)

sudo rm -r / 🡪 tüm bilgisayara format atarsın, dikkat!

pwd 🡪 içinde bulunduğun dizini göster

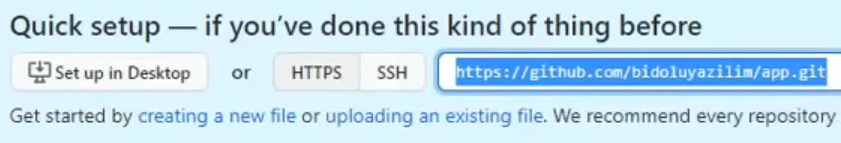
hostname 🡪 bilgisayarın ismini göster

ps 🡪 çalışan uygulamaları göster

ps aux 🡪 tüm çalışan uygulamaları göster

ps aux | grep bash 🡪 bash içerenleri göster

Shift + Insert 🡪 yapıştır



git remote add origin *https://github.com/toygar77/new.git* 🡪 projelerde kullanılan isim genellikle origin.

Github’daki Bir Projeyi Yüklemek:

**git clone** *url* 🡪 githubtaki repository’i bilgisayarımıza klonlama

Local’de Proje Oluşturmak:

**git init** git-repo 🡪 git deposu oluşturma

Working Directory to Github:

**git add .** 🡪 klasördeki tüm değişiklikleri ekler.

**git add** ./metin\_belgem.txt 🡪 Git’e dosyayı takip etmesini söylemek için. Bekleme alanına alıyor.

**git commit -m** “Updated main.py”

**git push -u origin main** 🡪 yerel depomuzdaki değişiklikleri github’taki **main** branch’ine yüklemiş olduk. push: yükleme, u: upstream. (Buradaki mail adresinin github’taki mail adresiyle eşleşmesi gerekiyor. Mail adreslerini eşleştirerek bağdaştırıyor.)

Github to Working Directory:

**git pull origin main**

**vim** main.py 🡪 dosyayı açıp içine yazma. Kaydedip çıkmak için esc ile insert modundan çıkıp “:wq” yaz. Kaydetmeden çıkmak için “:q!”

**git pull** 🡪 githubtaki dosyada manuel değişiklik yaptıysan git push’tan önce yapman gerekiyor

**git log** 🡪 geçmişte yaptığımız/takımımızın yaptığı değişiklikleri gösteriyor. **q**’ya basıp bu alandan çıkıyoruz.

**git log --oneline** 🡪 geçmişte yaptıklarımızı birer satır halinde özet olarak gösterir

**git commit --amend 🡪** İşlenmemiş bir commit varsa. Belki de daha önce yapılmış ancak işlenmemiş bir commit var. Bu durumda, önceki commit'i tamamlamak için git commit --amend komutunu kullanabilirsiniz. “On branch main nothing to commit” hatasının nedenlerinden biri

**git rm** test.txt 🡪 dosya silme. Sildikten sonra commit yap.

**İsim ve email konfigürasyonu**

**git config** --global user.name “Toygar”🡪 global yazmamızın amacı bundan sonra oluşturacağımız tüm git commitlerinde bunun kullanılmasını istememiz. Artık tüm git depolarında kullanılacak

**git config** --global user.email [toygar@gmail.com](mailto:toygar@gmail.com)

**git config** --**list** 🡪 yaptığın bu değişiklikleri en altta görebilirsin

**Move**

**git mv** deneme.txt src/ 🡪 deneme dosyasını src klasörüne atar.

**git mv** deneme.txt src/abc.txt 🡪 deneme dosyasını src klasörüne atıp ismini değiştirir.

**Alias**

**git config --global alias.***st* *status* 🡪 status yerine st de yazabilirim artık

**Hatayı geri alma:**

**git diff** 🡪 eklenen ve çıkarılan şeyleri gösteriyor

Hata working directory’de yapıldıysa:

**git checkout** -- *deneme.txt* 🡪 yapılan işlemi geri alma

Hata’yı add’lediysek / şu an staging’teysek:

**git reset HEAD** *deneme.txt* 🡪 dosyayı çalışma alanına (working directory) geri çekiyor

**git checkout** -- *deneme.txt* 🡪 yapılan işlemi geri alma

Hata’yı commit’lediysek / şu an local repository’deysek:

**git checkout** *previous\_commit\_id* -- **.** 🡪 git log’ta önceki bir işlemi geri getirmek için. Mesela bir dosya yanlışlıkla silindi. Silinmeden önceki commit id’sini alıp o ana geri dönüyorsun. Git log yapıp mouse ile id’yi tarayınca otomatik kopyalıyor.

Görünmez Yapma

**touch .gitignore** 🡪 dizinde txt oluşturur. Bu metin belgesinin içine yazdığın dosya isimleri track sisteminden kaybolur. Textin içine şunları yazarsak:

forbidden.txt 🡪 bu dosyayı izlemeyi bırakır.

forbid/\* 🡪 bu klasörü izlemeyi bırakır.

!forbid/qwe.bmp 🡪klasördeki bu dosyayı izlemeye devam eder.

**Branch**

Görme

**git branch** 🡪 var olan branchleri görme. Yıldızlı olan kullanımda

**git branch --all** 🡪 localdeki ve uzak sunucudaki branchleri gösterme

Yaratma

**git fetch** 🡪 github’ta yeni branch yarattıktan sonra bu kodla görmesini sağlıyoruz.

**git branch** *name* 🡪 localde branch yaratma

**git checkout** *name* 🡪 branch değiştirme

Birleştirme

**git merge** *name* 🡪 main branch’teyken bunu uygulayınca *name* branch’ini main’e atıyor. Eklemesi ve çıkarmasıyla birlikte.

Silme

**git fetch -p** 🡪 Github’da sildiğimiz branch’i uzak sunucudan siliyor. main’deyken yapabilirsin

**git branch -d** *name* 🡪 branch’i localden silme

132f1ba (HEAD -> main, origin/main, origin/HEAD) deneme2.txt added

**HEAD**, Git'te mevcut çalışma kopyasını ve üzerinde çalışılan commit'i işaret eden bir gösterge veya işaretçidir.

**(HEAD -> main, origin/main, origin/HEAD)**: Bu, HEAD'in şu anda **main** dalını işaret ettiğini ve aynı zamanda **origin/main** ve **origin/HEAD** ile aynı konumda olduğunu gösterir.