GİTHUB KOMUTLARI

**İçindekiler**

1. **SENKRONİZASYON DEĞİŞİMLERİ 2**
2. **KAYIT PARÇALARI 3**
3. **TARİHLERİ GÖZDEN GEÇİRME 3-4**
4. **DEĞİŞİM YAPMA İŞLEMLERİ 4-5**
5. **GRUP DEĞİŞİMLERİ 5**
6. **KONFİGÜRASYON AYARLARI 6**
7. **RESETLEME AYARLARI 6**
8. **ÖRNEK UYGULAMA YÜKLEME 7**

**1.Senkronizasyon Değişimleri**

1.1 GitHub Pull Komutu

Yer imi geçmişini indirir ve değişiklikleri içerir. Yerel deponuza indirir.

Çalışma Şekli: git pull

* 1. GitHub Push Komutu

Local de yapılan değişiklikleri hedefe aktarmak için kullanılır.

Çalışma Şekli: git push origin <gonderilecek\_dosya>

Örneğin master gidecek ise “git push origin master”

* 1. GitHub Merge Komutu

Localde ve hedefteki dallanmaları birleştirme işlemi için kullanılır.

Çalışma Şekli: git merge <dal(branch)>veya<bookmark>

* 1. GitHub Fetch Komutu

Repodaki tüm yer imi geçmişini indirir.

Çalışma Şekli: git fetch <bookmark>

**2.KAYIT PARÇALARI**

2.1 GitHub Stash Komutu

Değiştirilen tüm izlenen dosyaları geçici olarak saklar.

Çalışma Şekli: git stash

2.2 GitHub Stash Pop Komutu

En son saklanan dosyaları geri yükler.

Çalışma Şekli: git stash pop

2.3 GitHub Stash List Komutu

Tüm saklanan dosyaları listeler.

Çalışma Şekli: git stash list

2.4 GitHub Stash Drop Komutu

En son yapılan değişiklikleri atar(siler).

Çalışma Şekli: git stash drop

**3.TARİHLERİ GÖZDEN GEÇİRME**

3.1 GitHub Log Komutu

Şuanki dallanmanın versiyon tarihlerini listeler.

Çalışma Şekli: git log

3.2 GitHub Log –follow[file]

Seçilen dosyanın versiyon tarihi ile beraber yeniden adlandırmayı içerir.

Çalışma Şekli: git log --follow [file]

3.3 GitHub Diff [first-branch]...[second-branch]

2 dallanma arasındaki içerik farklarını gösterir.

Çalışma Şekli: git diff [first-branch]...[second-branch]

3.4 GitHub Show komutu

Özel yorum değişim içeriklerini ve metadata cıktısı verir.

Çalışma Şekli: git show [commit]

**4. DEĞİŞİM YAPMA İŞLEMLERİ**

4.1. GitHub Status Komutu

Değişen veya yeni commit olmus dosyaları listeler.

Çalışma Şekli: git status

4.2. Github Diff Komutu

Henüz hazırlanmamış, sunulmamış dosya farklarını gösterir.

Çalışma Şekli: git diff

4.3 GitHub Add Komutları

Sürüm hazırlığı için dosyayı anlık görüntü

Çalışma Şekli: git add [file]

4.4 GitHub diff - -staged

Hazırlama ve son dosya sürümü arasındaki farkları gösterir.

Çalışma Şekli: git diff --staged

4.5 GitHub reset [file]

Dosyayı sabitler ama içeriği korur.

Çalışma Şekli: git reset [file]

4.6 GitHub Commit –m “ text”

Text içerisindeki yazıyı commitler.

Çalışma Şekli: git commit -m "text commit"

**5.GRUP DEĞİŞİMLERİ**

5.1 GitHub Branch Komutu

Şuanki repo da bütün lokal dallanmaları listeler.

Çalışma Şekli: git branch

5.2 GitHub Branch [branch-name]

Yeni dallanma oluşur.

Çalışma Şekli: git branch [branch-name]

5.3 GitHub Checout [branch-name]

Belirtilen diziye geçer ve çalışma dizinini gösterir.

Çalışma Şekli: git checkout [branch-name]

5.4 GitHub Merge [branch]

Belirtilen şubenin geçmişini mevcut şubeyle birleştirir.

Çalışma Şekli: git merge [branch]

5.5 GitHub branch –d [branch-name]

Seçilen dallanmayı siler.

Çalışma Şekli: git branch -d [branch-name]

**6.KONFİGÜRASYON AYARLARI**

6.1 GitHub Config name ve email ayarları

Name: hangi kullanıcı adına gönderileceğini belirttiğimiz adres

Email: kullanıcı adının kayıtlı olduğunu belirttiğimiz mail adresi

Çalışma Şekli:git config - - global user.name “name”

git config - - global user.email “email”

**7.RESETLEME AYARLARI**

7.1 GitHub Resetleme Komutu

Tüm değişiklikleri yerel olarak koruyup tahahütleri geri alır.

Çalışma Şekli: git reset [commit]

7.2 GitHub Reset Hard

Tüm geçmişi iptal eder ve belirtilen tahahütle(commitleri) yapılan değişiklikleri geri alır.

Çalışma Şekli: git reset --hard [commit]

ÖRNEK UYGULAMA

Git Bash’i açtıktan sonra aşağıdaki adımları teker teker izliyoruz.

1.Adım olarak Girmek istediğimiz dosya içerisine cd komutu ile giriyoruz.

🡺cd Ornek\_Dosyam

2. adım olarak github üzerinde olusturdugumuz reponun echo kısmını alıyoruz.

🡺echo "# Ornek\_Dosyam" >> README.md

3.Adım olarak config işlemi gerçekleştiriyoruz.

🡺git config - -global user.name "username"

🡺git config - -global user.email "email@email.com"

4.adım olarak local depo olusturuyoruz.

🡺git init

5.adım olarak dosya içerisine Readme Dosyasını local depoya eklıyoruz.

🡺 git add Readme.md

6.Adım local dosyanın içerisindeki dosyaları ekleme işlemi yapıyoruz. Fakat burda işlem bitmiyor.

🡺 git add .

7.Adım commit ile ilk yorumu yapıyoruz. İlk ekleme de genel olarak “first commit” olarak geçirilir.

🡺 git commit –m “first commit”

.

8.adım github üzerinden new repo olusuturulan yerın adresını alıyoruz ve dosyaları remote(gönderme) işlemi yapıyoruz.

🡺 git remote add origin <https://github.com/isakordis/Ornek_Dosyam.git>

9. adım son olarak dosyaları itme işlemi yapıyoruz.

🡺git push –u origin master