**Git Bash İstifadəsi üzrə Giriş**

Git Bash, Git əmrlərini istifadə edərək versiya nəzarətini idarə etmək üçün bir mühitdir. Burada bəzi əsas əmrləri və onların istifadəsini izah edəcəyəm.

**Əsas Bash Əmrləri**

1. **mkdir** - Yeni qovluq yaratmaq üçün istifadə edilir. Məsələn, mkdir yeni\_qovluq.
2. **cd** - Mövcud qovluqları dəyişdirmək üçün istifadə olunur. Məsələn, cd yeni\_qovluq.
3. **touch** - Yeni fayl yaratmaq üçün istifadə edilir. Məsələn, touch test.txt.
4. **echo "test" > test.txt** - Fayla mətn yazmaq üçün istifadə edilir. Bu əmrlə "test" mətnini test.txt faylına yazırsınız.

**Git Konfiqurasiyası**

1. **git config --global user.name "imanov ulvi"** - Git istifadəçi adını təyin etmək üçün istifadə olunur.
2. **git config --global user.email "imanov ulvi"** - Git istifadəçi e-poçt ünvanını təyin etmək üçün istifadə olunur.
3. **git config --list** - Mövcud konfiqurasiyaları göstərir.

**Git Əməliyyatları**

1. **git init** - Mövcud qovluğu Git repozitoriyası halına gətirir.
2. **git status** - İşçi qovluqda dəyişiklikləri göstərir. Lokal saxlanca əlavə edilməyən və ya əlavə edilərək commit edilməyən faylları göstərir.
3. **git add test.txt** - Faylı lokal saxlanca əlavə etmək üçün istifadə olunur.
4. **git add .** - Bütün faylları lokal saxlanca əlavə edir.
5. **git commit -m "test commit yazildi"** - Lokal saxlanca əlavə edilmiş faylları commit edir.

**Commitlərə Baxış**

1. **git log** - Bütün commitləri göstərir.
2. **git log --oneline** - Commitləri qısa formada, tək sətirdə göstərir.

**Versiyaya Qayıtmaq**

1. **git checkout <commit-id> -- .** - Hər hansı versiyaya qayıtmaq üçün istifadə olunur.Lakin muveqqeti branca kecir deyisiklik yeniden master branchda qayidanda file qayidir.
2. **git reset --hard <commit-id>** Əgər dəyişiklikləri geri qaytarmaq istəyirsinizsə və onları birdəfəlik silmək niyyətindəsinizsə, git reset əmri daha münasibdir.

**Dəyişikliklərə Baxış**

1. **git diff** - Fayllarda edilən dəyişiklikləri göstərir.
2. **git diff test1.txt** - Yalnız test1.txt faylındakı dəyişiklikləri göstərir.

**Dəyişiklikləri Geri Qaytarmaq**

1. **git checkout -- test2.txt** - Faylı dəyişdirib lokal saxlanca göndərməmişdən əvvəl əvvəlki vəziyyətinə qaytarmaq üçün istifadə olunur.
2. **git reset HEAD** - Lokal saxlanca əlavə edilən faylı oradan çıxarır, sonra isə git checkout -- test2.txt əmri ilə faylı əvvəlki vəziyyətinə qaytarmaq mümkündür.

**Stash**

1. **git stash**: Dəyişikliklərinizi müvəqqəti olaraq saxlamaq üçün istifadə olunur.
2. **git stash list**: Bütün saxlanmış stash-lərin siyahısını göstərir.
3. **git stash apply**: Son saxlanmış stash-i geri gətirir, lakin onu siyahıdan silmir.
4. **git stash pop**: Son saxlanmış stash-i geri gətirir və onu siyahıdan silir.
5. **git stash drop stash@{0}**: Müəyyən bir stash-i siyahıdan silir. Əgər git stash list ilə müəyyən bir stash-i seçmək istəyirsinizsə, onun ID-sini istifadə etməlisiniz.
6. **git stash clear:** Bütün saxlanmış stash-ləri təmizləyir.

**Fayl və Qovluq Silmək**

1. **git rm test.txt** - Git vasitəsilə hər hansı bir faylı silmək üçün istifadə olunur.
2. **git rm -r test/** - Qovluğu silmək üçün istifadə edilir.

**Git Əmrlərinə Qısa Adlar**

1. **git config --global alias.st status** - Hər hansı Git əmrlərinə qısa ad vermək üçün istifadə olunur. Bu əmrlə git st yazaraq git status əmrini işlədə bilərsiniz.

**GitHub İlə Bağlantı**

1. **git remote add origin https://github.com/imanovulvi/test.git** - Uzaq sunucuya (GitHub) bağlantı qurmaq üçün istifadə olunur.
2. **git remote** - Mövcud sunucu bağlantılarını göstərir.
3. **git push -u origin master** - Master branchını origin bağlantısı ilə uzaq sunucuya göndərmək üçün istifadə olunur.
4. **git pull origin master** - Yenilənmiş faylları proyektə əlavə etmək üçün istifadə edilir.

**.gitignore Faylı** .gitignore faylı Git-in izləməməsini istədiyiniz faylları və qovluqları göstərir. Məsələn:

bash

Copy code

ig.txt

git/\*

!git/ig2.txt

Bu nümunədə git qovluğu daxilində olan bütün fayllar izlənməyəcək, lakin ig2.txt faylı izlənəcək.

**Branch (Şaxə) İdarəsi**

1. **git branch** - Lokaldakı şaxələri göstərir.
2. **git branch dev** - Yeni dev adında şaxə yaradır.
3. **git branch --all** - GitHub və lokaldakı bütün şaxələri göstərir.
4. **git fetch** - GitHub-dakı dəyişiklikləri gostərir (yeni branc,file ve s) lokala gətirir amma lokala əlavə etmir. Əlavə etmək üçün pull əmrindən istifadə olunur.
5. **git checkout dev** - Şaxəni dəyişdirmək üçün istifadə edilir.
6. **git merge dev** - Master şaxəsində olarkən dev şaxəsini master ilə birləşdirmək üçün istifadə olunur.
7. **git fetch -p** - GitHub-dakı silinmiş şaxələri lokaldan da silmək üçün istifadə edilir.
8. **git branch -D test** - test adında olan şaxəni silmək üçün istifadə edilir.
9. **git clone** [**https://github.com/user/repository.git**](https://github.com/user/repository.git) git oz komputerine kopyalamaq ucun