

git config

<code>git config --global user.name <"name"></code>	postavljanje korisničkog imena
<code>git config --global user.email <"email"></code>	postavljanje korisničkog emaila
<code>git config --global core.editor "vim --nofork"</code>	postavljanje Vim-a kao editora
<code>git config --global core.editor "atom --wait"</code>	postavljanje Atom-a kao editora
<code>git config --global core.editor "code --wait"</code>	postavljanje VS Code-a kao editora

git init

<code>git init</code>	inicijalizacija trenutnog direktorija kao repozitorija
-----------------------	--

git clone

<code>git clone <URL></code>	kloniranje repozitorija
------------------------------------	-------------------------

git remote

<code>git remote [-v]</code>	[detaljni] prikaz <i>remote</i> -ova
<code>git remote add upstream <URL></code>	dodavanje <i>upstream remote</i> -a (za <i>fetch</i> -anje <i>update</i> -ova na originalnom repozitoriju)
<code>git remote add origin <URL></code>	dodavanje <i>upstream remote</i> -a (za <i>push</i> -anje <i>update</i> -ova na <i>forked</i> repozitorij)
<code>git remote rename <old> <"new"></code>	preimenovanje <i>remote</i> -a
<code>git remote remove <name></code>	uklanjanje <i>remote</i> -a

git fetch

<code>git fetch <remote></code>	<i>update</i> -a stanje svih <i>branch</i> -eva <i>remote</i> repozitorija
<code>git fetch <remote> <remote-branch></code>	<i>update</i> -a stanje pojedinačnog <i>branch</i> -a <i>remote</i> repozitorija

git pull

- `git pull origin <remote-branch> = git fetch origin <remote-branch> + git merge <remote>/<branch>`

<code>git pull <remote> <remote-branch></code>	<i>update-a stanje pojedinačnog branch-a remote repozitorija i sukladno update-a lokalno stanje</i>
--	---

git push

<code>git push <remote> <local-branch></code>	<i>push-anje lokalnog branch-a na remote istog naziva</i>
<code>git push --tags <remote> <local-branch></code>	<i>push-a tagove na remote repozitorij (tag-ovi se neće push-ati automatski)</i>

git status

<code>git status</code>	<i>status repozitorija (branch, untracked, unstaged, staged...)</i>
-------------------------	---

git log

<code>git log</code>	<i>puni ispis svih commit-a na repozitoriju</i>
<code>git log --oneline</code>	<i>jednolinijski ispis svih commit-a na repozitoriju</i>

git add

<code>git add <file></code>	<i>dodavanje datoteka u staging area</i>
-----------------------------------	--

git commit

<code>git commit</code>	<i>commit staged datoteka (naknadni unos commit poruke)</i>
<code>git commit -m <message></code>	<i>commit staged datoteka uz poruku</i>
<code>git commit -a -m <message></code>	<i>dodavanje svog sadržaja radnog direktorija u staging area i commit uz poruku</i>
<code>git commit --amend</code>	<i>redo posljednjeg commit-a ili uređivanje poruke posljednjeg commit-a</i>

git branch

<code>git branch -l [-v]</code>	[detaljni] prikaz lokalnih <i>branch</i> -eva
<code>git branch -r [-v]</code>	[detaljni] prikaz remote <i>branch</i> -eva
<code>git branch -a [-v]</code>	[detaljni] prikaz svih <i>branch</i> -eva
<code>git branch <branch-name></code>	kreiranje novog <i>branch</i> -a
<code>git branch -m <branch-name></code>	preimenovanje aktivnog <i>branch</i> -a (koristiti <code>-M</code> za nasilno)
<code>git branch -d <branch-name></code>	brisanje <i>branch</i> -a (koristiti <code>-D</code> za nasilno)

git switch

<code>git switch <branch-name></code>	prebacivanje s trenutnog na drugi <i>branch</i> (ako se koristi ime <i>remote branch</i> -a, kreira lokalnu kopiju)
<code>git switch -</code>	prebacivanje u prethodno posjećeni <i>branch</i>
<code>git switch -c <branch-name></code>	kreiranje novog <i>branch</i> -a s imenom name s trenutnog <i>HEAD</i> -a i prebacivanje u njega

git merge

- *merge*-ani *branch* se ne briše

<code>git merge <branch-name></code>	<i>merge</i> -anje <i>branch</i> -a u aktivni <i>branch</i>
--	---

git rebase

- rebase funkcijski može služiti kao zamjena za *merge*
 - reorganizira/kopira *commit* s aktualnog *branch*-a na vrh master *branch*-a

<code>git rebase main</code>	reorganizacija/kopiranje <i>commit</i> -a s aktualnog <i>branch</i> -a na vrh main <i>branch</i> -a
<code>git rebase -i <commit-hash></code>	ulazak u interaktivno uređivanje prethodnih <i>commit</i> -a, počevši od <i>commit-hash</i> -a

git diff

<code>git diff <file></code>	prikazivanje <i>unstaged</i> promjena datoteke
<code>git diff --staged <file></code>	prikazivanje <i>staged</i> promjena datoteke
<code>git diff HEAD <file></code>	prikazivanje svih promjena datoteke od posljednjeg <i>commit</i> -a
<code>git diff <old-commit-hash> <new-commit-hash></code>	prikazivanje promjena između dva <i>commit</i> -a
<code>git diff <branch-name-1> <branch-name-2></code>	prikazivanje promjena između "vrhova" <i>branch</i> -eva
<code>git diff <branch-name>@{0} <branch-name>@{1}</code>	prikazuje promjene između stanja na <i>branch</i> -u (vidi <code>git reflog show</code>)

git stash

- *stash* je globalan, nije vezan za *branch*
 - moguće je aplicirati *stashed* promjene s jednog na drugi *branch*

<code>git stash save</code>	spremanje <i>unstaged</i> i <i>staged</i> promjena u <i>stash</i>
<code>git stash list</code>	listanje <i>stashed</i> promjena
<code>git stash pop [stash@<number>]</code>	apliciranje <i>stashed</i> promjena na radni direktorij (cut iz <i>stash</i> -a)
<code>git stash apply [stash@<number>]</code>	apliciranje <i>stashed</i> promjena na radni direktorij (copy iz <i>stash</i> -a)
<code>git stash drop [stash@<number>]</code>	brisanje <i>stashed</i> promjene
<code>git stash clear</code>	brisanje svih <i>stashed</i> promjena iz liste

git checkout

<code>git checkout <branch-name></code>	prebacivanje s trenutnog <i>HEAD</i> -a na drugi <i>branch</i> (funkcionira i za <i>remote branch</i> -eve)
<code>git checkout -b <branch-name></code>	kreiranje novog <i>branch</i> -a s imenom <i>name</i> s trenutnog <i>HEAD</i> -a i prebacivanje u njega
<code>git checkout --track <remote>/<branch></code>	kreira lokalnu kopiju <i>remote branch</i> -a istog naziva (vidi <code>git switch</code>)
<code>git checkout <commit-hash></code>	prebacivanje <i>HEAD</i> -a na neki drugi <i>commit</i>
<code>git checkout <tag></code>	prebacivanje <i>HEAD</i> -a na neki drugi <i>commit</i> označen <i>tag</i> -om
<code>git checkout HEAD <file></code>	poništanje svih promjena do posljednjeg <i>commit</i> -a za datoteku
<code>git checkout HEAD@{<number>}</code>	premještanje u poziciju <i>HEAD</i> -a prije <i>number</i> koraka (vidi <code>git reflog show</code>)
<code>git checkout <branch-name>@{<number>}</code>	premještanje u stanje <i>branch</i> -a prije <i>number</i> koraka (vidi <code>git reflog show</code>)

git tag

- *tag* je dodatan pointer na *commit*
 - *tag*-ovi su jedinstveni
- uglavnom služe za *semantic versioning*

<code>git tag -l ["<regex-search>"]</code>	ispisuje sve <i>tag</i> -ove na aktualnom <i>branch</i> -u [koji zadovoljavaju <i>regex</i> pretragu]
<code>git tag [-a] <"tag-name"></code>	tagira aktualni <i>HEAD/commit</i> s [<i>annotated</i>] <i>tag</i> -om
<code>git tag [-a] <"tag-name"> <commit-hash></code>	tagira <i>commit</i> s [<i>annotated</i>] <i>tag</i> -om
<code>git tag -f <"tag-name"> <commit-hash></code>	"nasilno" tagira <i>commit</i> s već iskorištenim <i>tag</i> -om (<i>tag</i> sa starog <i>commit</i> -a nestaje)
<code>git tag -d <tag-name></code>	briše <i>tag</i>

git restore

- nikad ne uklanja *commit log*

<code>git restore --staged <file></code>	prebacivanje <i>staged</i> promjena u <i>unstaged</i>
<code>git restore <file></code>	poništanje <i>unstaged</i> promjena do posljednjeg <i>commit</i> -a
<code>git restore --source=<commit-hash> <file></code>	poništanje promjena po uzoru na <i>commit</i> (<i>log</i> ostaje isti, promjene odlaze u <i>unstaged</i>)

git reset

- uvijek uklanja *commit log*
- *unstaged* i *staged* promjene se mogu prebaciti na drugi branch

<code>git reset --soft <commit-hash></code>	ponišćavanje promjena po uzoru na <i>commit</i> (<i>log</i> se mijenja, promjene odlaze u <i>staged</i>)
<code>git reset [--mixed] <commit-hash></code>	ponišćavanje promjena po uzoru na <i>commit</i> (<i>log</i> se mijenja, promjene odlaze u <i>unstaged</i>)
<code>git reset --hard <commit-hash></code>	ponišćavanje promjena po uzoru na <i>commit</i> (<i>log</i> se mijenja, promjene se <i>commit</i> -aju)

git revert

- nikad ne uklanja *commit log*

<code>git revert <commit-hash></code>	ponišćavanje promjena od <i>commit</i> -a i dalje (<i>log</i> ostaje isti, promjene odlaze u <i>unstaged-conflicted</i>)
---	--

git reflog

- *reflog*-ovi su lokalni i imaju "rok trajanja"
- služe za poništavanje `git reset` i sl.

<code>git reflog show HEAD</code>	prikazuje povijest "kretanja" <i>HEAD</i> -a povijest radnji na <i>branch</i> -u
<code>git reflog show <branch-name></code>	prikazuje povijest stanja na <i>branch</i> -u
<code>git reflog show HEAD@2.day.ago</code>	prikazuje povijest "kretanja" <i>HEAD</i> -a do prije 2 dana
<code>git reflog show <branch-name>@2.day.ago</code>	prikazuje povijest stanja na <i>branch</i> -u do prije 2 dana