< Learn These Shortcuts (/app/dojos/git)

Git Cheat Sheet

Git Basics

git init	Initialize a repository
git status	Show status of working tree
git add file.txt	Start tracking file.txt
git add main.txt	Stage modified file main.txt
git diff	Show what's changed but not yet staged
git commit	Commit changes
git commit -a	Stage files and commit
git mv main.txt file.txt	Rename main.txt to file.txt
git fetch develop	Pull data from remote 'develop' without merging
git pull origin develop	Fetch and merge branch 'develop' from origin
git clone url	Create local copy of remote repository at 'url'

Branching

git branch	Show current branches
git push origin master	Push master branch to origin server
git branch -v	Show last commit on all branches
git checkout master	Switch to branch 'master'
git branch feature1	Create new branch called 'feature1'
git checkout -b feature2	Create branch 'feature2' and switch to it
git branch -d mybranch	Delete branch 'mybranch'
git branch --merged	Show branches already merged into current branch
git branch --no-merged	Show branches not yet merged into current branch
git branch -D fix	Force delete branch 'fix' that is not yet merged
git push origin feature1	Push local branch 'feature1' to origin
git push staging develop:master	Push develop branch to remote staging master
git checkout -b fix1 origin/fix1	Create local branch 'fix1' based off origin branch
git checkout --track origin/fix2	Create tracking branch 'fix2' based off origin
git push origin :fix2	Delete remote branch 'fix2' from origin

Merging / Rebasing

git mergetool	Use graphical merge tool
git commit	Finalize merge after resolving conflicts
git merge feature1	Merge branch 'feature1' with current branch
git add file.txt	Mark file.txt as resolved after merge

git rebase develop	Rebase changes made on current branch over develop
git rebase master develop	Rebase master onto develop without checking it out
git rebase --onto master 1a 1b	Rebase master onto branch 1b made from branch 1a

Remotes

git remote	Show remote servers you have configured
git remote -v	Show remote servers with URL displayed
git remote add myurl url	Add remote server 'url' with shortname 'myurl'
git remote rename server1 server2	Rename remote 'server1' to 'server2'
git remote rm server1	Remove remote 'server1'
git remote show origin	Show info about remote origin

Commit Logs

git log	Show commit logs
git log -p -2	Show last two commits with diffs
git log --stat	Show commit logs with stats
git log --pretty=oneline	Show commit logs one per line
git log --graph	Show commit logs with ascii graph
git log --since=1.week	Show commit log for the last week
git blame -L 10,15 file.rb	Show prev commits for each lines 10-15 of file.rb

Undo / Change History

git rm --cached main.txt	Remove main.txt from staging but keep in working
git commit --amend	Replace last commit with whats in staging
git checkout -- file.txt	Discard changes to file.txt
git reset HEAD file.txt	Unstage file.txt
git commit --amend	Modify last commit message
git rebase -i HEAD~3	Make changes to the last 3 commits

Using Tags

git tag	Show available tags
git tag -a v3.0	Create annotated tag 'v3.0'
git show v3.0	Show info for tag v3.0
git tag -s v3.0	Create signed tag v3.0
git tag v2.1-lw	Create lightweight tag v2.1
git tag -v v3.0	Verify signed tag v3.0
git tag -a v2.2 8feb	Tag previous commit '8feb' as v2.2
git push origin v2.2	Push tag v2.2 to origin
git push origin --tags	Push all local tags to origin

Using Stashes

git stash	Stash changes without committing

git stash list	Show stores stashes
git stash apply	Reapply most recent stash
git stash apply stash@2	Reapply stash 2
git stash apply --index	Reapply stashed changes along with staged changes
git stash drop stash@{2}	Drop stash 2
git stash pop	Apply most recent stash and drop from stack
git stash branch mybranch	Create branch 'mybranch' from stash
git stash clear	Delete all stashes
git diff --staged	Show what's staged but not yet committed
git diff --check	Check for whitespace errors before committing

Using Bisect

git bisect start	Start binary search of commits to find bad commit
git bisect bad	Mark current commit as broken during bisect
git bisect good v2.2	Mark v2.2 as last known good commit during bisect
git bisect good	Mark current commit as good during bisect
git bisect reset	Reset HEAD when finished with bisect
git bisect run test.sh	Run 'test.sh' on each commit during bisect

< Learn These Shortcuts (/app/dojos/git)

Blog (/blog) About (/about) Privacy Policy (/privacy) Terms of Service (/terms) Pricing (/app/pricing)

(https://www.facebook.com/shortcutfoo) (https://www.twitter.com/shortcutfoo)