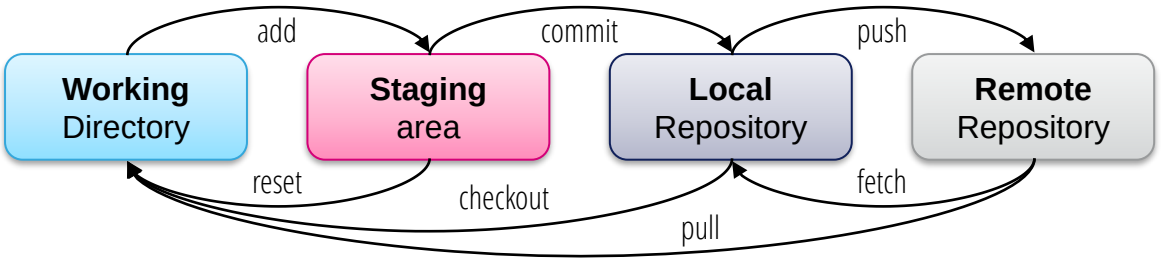
git essential commands



Create repository

Init or create <dir> as Git repo (defaults to '.')
$ git init [<dir>]

Clone an existing repo
$ git clone <repo_url> [<local_repo_name>]

Make changes

List actions, changed & new files in local repo
$ git status

Show diffs on tracked files in working dir
$ git diff [<file>]

Show staged diffs that will be commited
$ git diff --staged [<file>]

Stage given file(s)
$ git add <file> [<file> ...]

Stage all updates in directories
$ git add <dir> [<dir> ...]
$ git add .

Stage all updates in working directory
$ git add -A

Commit staged changes to local repo
$ git commit [-m "<commit_message>"]

Update current commit
$ git commit --amend

Explore history

Show all changes applied in <commit>
$ git show <commit>

Show current branch (entire with --all) history
$ git log [--all] [--graph] [--oneline]

Revert changes

Remove any local change in <file>
$ git restore <file> [<file> ...]

Unstage <file>, keeping changes
$ git restore --staged <file> [<file> ...]

Revert to <commit>, keeping changes
$ git reset <commit>

Revert to <commit>, loosing changes
$ git reset --hard <commit>

Synchronize

Push local changes to <remote>
$ git push <remote> <branch>

Get changes in <remote> (no merge)
$ git fetch <remote>

Get changes in <remote> & merge
$ git pull <remote> <branch>

Get all available remotes with URLs
$ git remote -v

Add a new remote repo
$ git remote add <name> <url>

Branches

master default branch name
origin default remote name
HEAD current point

Create <branch> at HEAD
$ git branch <branch>

Switch to <branch>
$ git checkout <branch>

Create and switch to <branch> at HEAD
$ git checkout -b <branch>

List all branches
$ git branch -a

Delete a local branch
$ git branch -D <branch>

Delete a remote branch
$ git push origin --delete <branch>

Merge <branch> into current branch
$ git merge <branch>

Rebase current branch over <branch>
$ git rebase <branch>

Rebase current branch interactively
$ git rebase -i <branch>

git additional commands

Stashing

Stash modifications (-u includes untracked)
$ git stash push [-u] [-m <message>]

Apply & remove stash (if no conflict)
$ git stash pop

Tagging

Create a <tag> on current <commit>
$ git tag -a <tag> [-m "<tag_message>"]

List tags present in repository
$ git tag -l

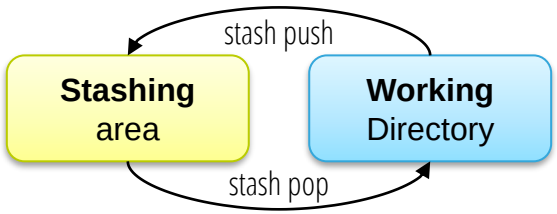
Delete <tag> in local repo
$ git tag -d <tag>

Push <tag> to remote repo
$ git push <remote> <tag>

Push all local tags to remote repo
$ git push <remote> --tags

Cherry picking

Apply all changes introduced by <commit>
$ git cherry-pick <commit>



life.augmented

Resolve merge conflicts

Get information on status & commands
$ git status

Cancel merge
$ git merge --abort

Continue once conflicts resolved
$ git add <conflict_file>
$git merge --continue

Resolve rebase conflicts

Get information on status & commands
$ git status

Cancel rebase
$ git rebase --abort

Skip replay of the current commit in rebase
$ git rebase --skip

Continue once conflicts resolved
$ git add <conflict_file>
$ git rebase --continue

Patching (no commit)

Export changes between <c1> and <c2>
$ git diff <c1> <c2> > output.patch

Apply patch
$ git apply output.patch

Patching (with commit)

Generate patch for last 2 commits
$ git format-patch ~2

Apply patch
$ git am file.patch

Debugging

Who did changes, where & when ?
$ git blame <file>

Find <pattern> in source code
$ git grep <pattern>

Find <pattern> in source code
$ git grep <pattern>

Misc

Get help on a git <command>
$ git <command> --help

Create aliases
$ git config --global alias.<cmd> "<aliased_command>"
$ git config --global alias.graph "log --all --graph --oneline"

Use **.gitignore** to avoid management of files or directories