Git Cheat Sheet

Create

From existing data
```
cd ~/my_project_dir
git init
git add .
```
From existing repo
```
git clone ~/existing/repo ~/new/repo
git clone you@host.org:dir/project.git
```
default protocol is **ssh**

Browse

Files changed in working directory
```
git status
```
Changes to tracked files
```
git diff
```
Changes between ID1 and ID2
```
git diff <ID1> <ID2>
```
History of changes
```
git log --decorate --graph
gitk
```
Who changed what and when in a file
```
git blame <file>
```
A commit identified by ID (commit hash)
```
git show <ID>
```
A specific file from a specific ID
```
git diff <ID>:<FILE>
```
Search for patterns
```
git grep <pattern> [path]
```

Useful tips

Get help
```
git help [command]
git help (git|tutorial|glossary)
```
Switch branch w/o touching the working tree
```
git symbolic-ref HEAD refs/heads/<branch>
git read-tree <branch>
```
Create empty/unconnected branch
```
git symbolic-ref HEAD refs/heads/<branch>
rm .git/index
git clean -fdx
<do work>
git add your files
git commit -m 'Initial commit'
```
Short graphical log
```
git log –oneline --graph
```
Delete remote branch and locally
```
git push --delete <origin> <branch>
git branch -d <branch>
```
Information about remotes
```
git remote –v show [origin]
```

Change

Using your favorite editor / IDE

Revert

Return to the last committed state
```
git checkout -f | git reset --hard
```
you cannot undo a hard reset

Revert the last commit
```
git revert HEAD
```
Creates a new commit

Revert specific commit
```
git revert $id
```
Creates a new commit

Fix the last commit
```
git commit -a --amend
```
after editing the broken files
(if you haven't pushed)

Checkout the ID version of a file
```
git checkout <ID> <file>
```

Branch

List all branches
```
git branch -vv [-r]
```
Switch to the BRANCH branch
```
git checkout <BRANCH>
```
Merge branch B1 into branch B2
```
git checkout <B2>
git merge <B1>
```
Create branch based on HEAD
```
git branch <BRANCH>
```
Create branch based on another
```
git branch <new> <base>
```
Delete a branch
```
git branch -d <branch>
```

Resolve merge conflicts

View merge conflicts
```
git diff
```
After resolving conflicts, merge with
```
git add <CONFLICTING_FILE>
git commit
```
Or use 3-way merge tool (like meld or vimdiff)
```
git mergetool
```

Update

Fetch latest changes from origin
```
git fetch
```
this does not merge them

Pull latest changes from origin
```
git pull
```
does a fetch followed by a merge

Apply a patch that someone sent you
```
git am -3 patch.mbox
```
In case of conflict, resolve the conflict and
```
git am --resolve
```

Commit

```
git (add|rm) <files>; git commit -v
```
Commit all local changes
```
git commit –a -v
```

Publish

Prepare a patch for other developers
```
git format-patch origin
```
Push changes to origin
```
git push [origin] [branch]
```
add -u to set default source for future pulls/pushes

Make a version or milestone
```
git tag –a <version_name>
```

Configuration

```
git config [--global]
```
global is stored in ~/.gitconfig

user
```
user.name $name
user.email $email
```
color
```
color.ui auto
```
add branch names to output of git log
```
log.decorate short
```
github
```
github.user $user
github.token $token
```

windows
```
core.autocrlf true
```

https://github.com/eddyp/gitcheatsheet