

Git Basics

\$ git init

This command creates an empty Git repository

\$ git config

Get and set repository or global options

\$ git add

Add file contents to the index

\$ git commit -m "<message>"

Stores the current contents of the index in a new commit along with a log message from the user describing the changes.

\$ git status

Displays paths that have differences between the index file and the current HEAD commit

\$ git log

Show commit logs

\$ git diff

Show changes between commits, commit and working tree, etc

Remote

\$ git remote -v

List the remotes added to current repository

\$ git remote add <name> <url>

Add a new remote

git log

\$ git log -<number>

Limit no of commits by <number>

\$ git log --oneline

Limit each commit to only a single line

\$ git log -p

Display full diff for each commit.

\$ git log --stat

List files were added or updated with the number of lines added or removed.

\$ git log --author="<pattern>"

Search for commits by a particular author.

\$ git log --grep="<pattern>"

Search for commits with a commit message that matches <pattern>

\$ git log <since>..<until>

Show commits that occur between <since> and <until> . Args can be a commit ID, branch name, HEAD , or any other kind of revision reference.

\$ git log -- <file>

Only display commits that have the specified file.

\$ git log --graph --decorate

--graph flag draws a text based graph of commits on left side of commit messages. --decorate adds names of branches or tags of commits shown.



Configuration

```
$ git config --global user.name "[name]"
```

Sets the name you want attached to your commit transactions

```
$ git config --global user.email "[email address]"
```

Sets the email you want attached to your commit transactions

```
$ git config --global color.ui auto
```

Enables helpful colorization of command line output

Branch

```
$ git branch
```

List, create, or delete branches

```
$ git branch <name>
```

create copy of current checked out branch

```
$ git checkout <branch>
```

Switch branches or restore working tree files

```
$ git merge <branch>
```

Join two or more development commits/branch together

```
$ git branch -D <branch>
```

delete a branch

```
$ git fetch
```

Download objects and refs from another repository or fetch a branch from another repository(specified in remote)

```
$ git push
```

Update remote refs along with associated objects or push a local branch to remote server

```
$ git pull
```

Fetch from and integrate with another repository or a local branch

Repositories

```
$ git init [project-name]
```

Creates a new local repository with the specified name.

```
$ git clone [url]
```

Downloads a project and its entire version history to your local server.

Redo Commits

```
$ git reset [commit]
```

Undoes all commits after [commit], preserving changes locally.

```
$ git reset --hard [commit]
```

Discards all history and changes back to the specified commit.

Save Fragments

```
*$ git stash
```

Temporarily stores all modified tracked files.

```
$ git stash pop
```

Restores the most recently stashed files.

```
$ git stash list
```

Lists all stashed changesets.

```
$ git stash drop
```

Discards the most recently stashed changeset.

C

By **rohanraja**
cheatography.com/rohanraja/

Not published yet.
Last updated 17th December, 2017.
Page 2 of 2.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
<http://crosswordcheats.com>