git - the simple guide

just a simple guide for getting started with git. no deep shit ;)

Tweet 4,747

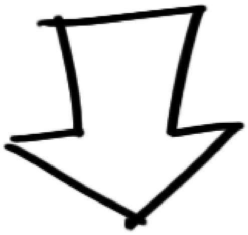
by Roger Dudler

credits to @tfnico, @fhd and Namics

ı deutsch, español, français, indonesian, italiano, nederlands, polski, português, русск

□□□□□, 日本語, 中文, 한국어 Vietnamese

please report issues on github



setup

Download git for OSX

Download git for Windows

Download git for Linux

create a new repository

create a new directory, open it and perform a

```
git init
```

to create a new git repository.

checkout a repository

create a working copy of a local repository by running the command

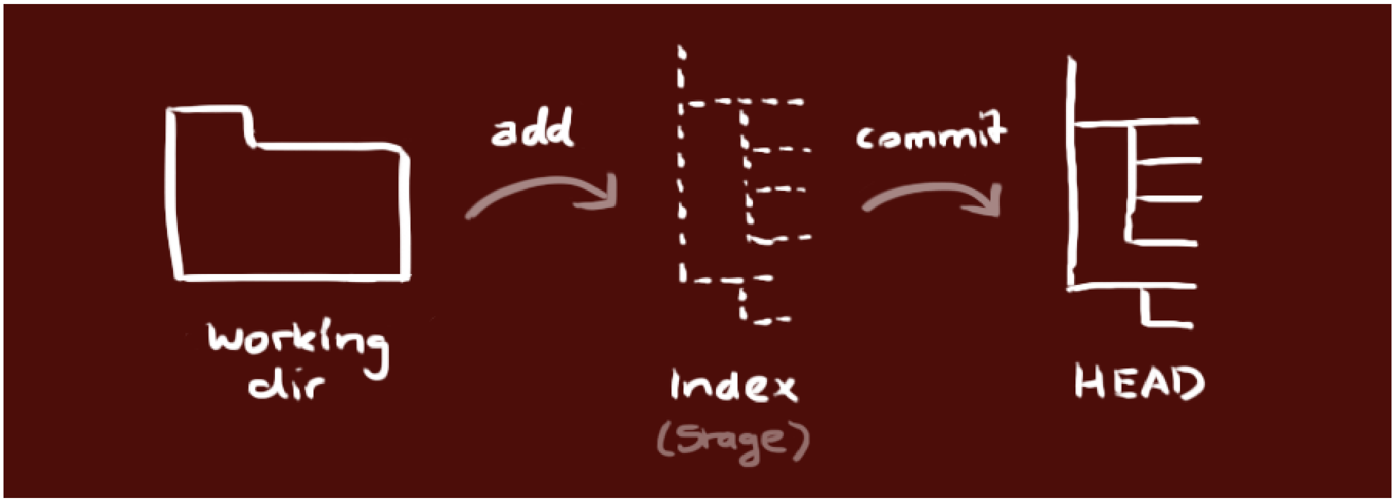
```
git clone /path/to/repository
```

when using a remote server, your command will be

```
git clone username@host:/path/to/repository
```

workflow

your local repository consists of three "trees" maintained by git. the first one is your `Working Directory` which holds the actual files. the second one is the `Index` which acts as a staging area and finally the `HEAD` which points to the last commit you've made.



add & commit

You can propose changes (add it to the **Index**) using

```
git add <filename>
```

```
git add *
```

This is the first step in the basic git workflow. To actually commit these changes use

```
git commit -m "Commit message"
```

Now the file is committed to the **HEAD**, but not in your remote repository yet.

pushing changes

Your changes are now in the **HEAD** of your local working copy. To send those changes to your remote repository, execute

```
git push origin master
```

Change *master* to whatever branch you want to push your changes to.

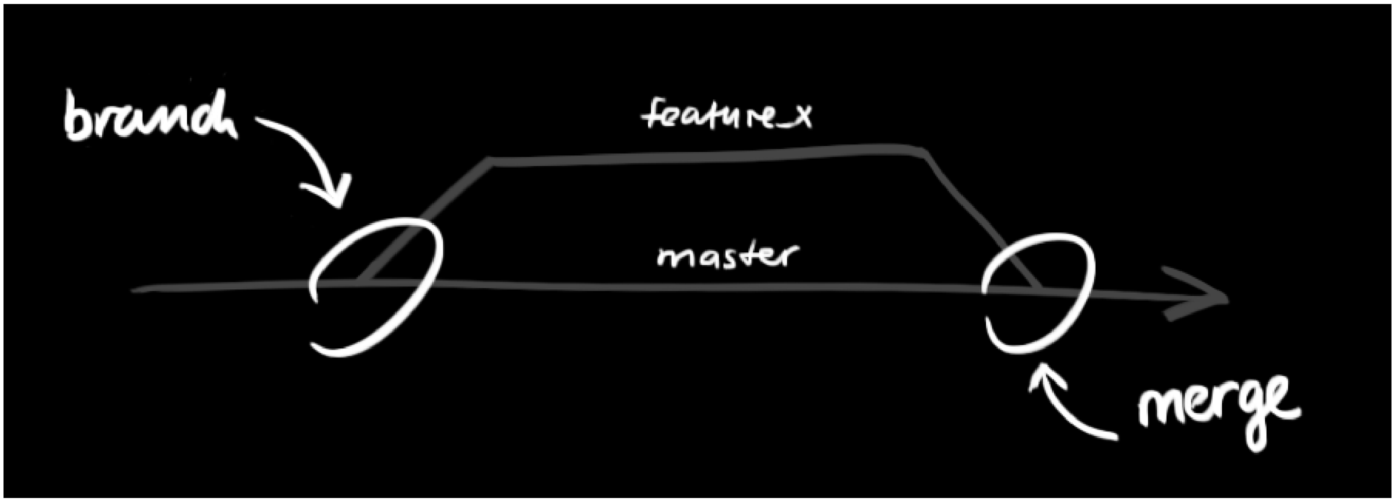
If you have not cloned an existing repository and want to connect your repository to a remote server, you need to add it with

```
git remote add origin <server>
```

Now you are able to push your changes to the selected remote server

branching

Branches are used to develop features isolated from each other. The *master* branch is the "default" branch when you create a repository. Use other branches for development and merge them back to the master branch upon completion.



create a new branch named "feature_x" and switch to it using

```
git checkout -b feature_x
```

switch back to master

```
git checkout master
```

and delete the branch again

```
git branch -d feature_x
```

a branch is *not available to others* unless you push the branch to your remote repository

```
git push origin <branch>
```

update & merge

to update your local repository to the newest commit, execute

```
git pull
```

in your working directory to *fetch* and *merge* remote changes.

to merge another branch into your active branch (e.g. master), use

```
git merge <branch>
```

in both cases git tries to auto-merge changes. Unfortunately, this is not always possible and results in *conflicts*. You are responsible to merge those *conflicts* manually by editing the files shown by git. After changing, you need to mark them as merged with

```
git add <filename>
```

before merging changes, you can also preview them by using

```
git diff <source_branch> <target_branch>
```

tagging

it's recommended to create tags for software releases. this is a known concept, which also exists in SVN. You can create a new tag named *1.0.0* by executing

```
git tag 1.0.0 1b2e1d63ff
```

the *1b2e1d63ff* stands for the first 10 characters of the commit id you want to reference with your tag. You can get the commit id by looking at the...

log

in its simplest form, you can study repository history using.. `git log`

You can add a lot of parameters to make the log look like what you want.

To see only the commits of a certain author:

```
git log --author=bob
```

To see a very compressed log where each commit is one line:

```
git log --pretty=oneline
```

Or maybe you want to see an ASCII art tree of all the branches, decorated with the names of tags and branches:

```
git log --graph --oneline --decorate --all
```

See only which files have changed:

```
git log --name-status
```

These are just a few of the possible parameters you can use. For more, see `git log --help`

replace local changes

In case you did something wrong, which for sure never happens ;), you can replace local changes using the command

```
git checkout -- <filename>
```

this replaces the changes in your working tree with the last content in HEAD. Changes already added to the index, as well as new files, will be kept.

If you instead want to drop all your local changes and commits, fetch the latest history from the server and point your local master branch at it like this

```
git fetch origin
```

```
git reset --hard origin/master
```

useful hints

built-in git GUI

```
gitk
```

use colorful git output

```
git config color.ui true
```

show log on just one line per commit

```
git config format.pretty oneline
```

use interactive adding

```
git add -i
```

links & resources

graphical clients

GitX (L) (OSX, open source)
Tower (OSX)
Source Tree (OSX & Windows, free)
GitHub for Mac (OSX, free)
GitBox (OSX, App Store)

guides

Git Community Book
Pro Git
Think like a git
GitHub Help
A Visual Git Guide

get help

Git User Mailing List
#git on irc.freenode.net

comments

516 Comments git - the simple guide 1 Login ▾

♥ Recommend 136 Share Sort by Newest ▾

Join the discussion…

Mohamed Zaatar • 3 hours ago

nice guide but with workflows diagrams will be perfect.

^ | ˅ • Reply • Share ›

Sebastian • 2 days ago

Awesome!

^ | ˅ • Reply • Share ›

Md Imran Mondal • 3 days ago

love this guide, well explained

^ | ˅ • Reply • Share ›

Mbah-Mbole Sama • 3 days ago

great tutorial, short concise and elaborate. Thank you!

^ | ˅ • Reply • Share ›

Yarina • 4 days ago

Love this!

^ | ˅ • Reply • Share ›

schlumpfmarkus • 5 days ago

Super simple and to the point! Love the colors. Thank a lot!!

^ | ˅ • Reply • Share ›

Ralf Tenbrink • 5 days ago

I cannot get past step 2 (clone). I get permission denied (publickey).
Was hoping this would help me understand github but I cannot get my head around it. I better stick with SVN

^ | ˅ • Reply • Share ›

Nils Mundhenke • 6 days ago

This is the best thing I saw this year!
Thank you so much for this :)

^ | ˅ • Reply • Share ›

devang nathwani • 6 days ago

thank you. thank you.

1 ^ | ˅ • Reply • Share ›

PHORN Ya • 6 days ago

The best tutorial for beginner

1 • Reply • Share ›

Sree Kumar • 7 days ago

Really awesome !!!

Reply • Share ›

devesh • 7 days ago

how to merge change made in one branch to another

Reply • Share ›

sirslimedia devesh • 6 days ago

git merge <branch2> e.g git merge updates and you can check the diff with git diff branch1 branch2

Reply • Share ›

Rajat Pawar • 7 days ago

Awesome tutorial !

Reply • Share ›

Barnaby Jones • 10 days ago

A great but still quite complex tutorial. How do you know what server to connect to? Doesn't this happen automatically if you use Github? How does git automatically know your username and email? Where are these stored?

Reply • Share ›

Rajat Pawar Barnaby Jones • 7 days ago

This tutorial is intended for software engineers.

Reply • Share ›

Chytap Richfriend • 12 days ago

Thank you very much

Reply • Share ›

kuldipem • 13 days ago

nice quick tutorials....

Reply • Share ›

carlos enrique olivares rodrig • 13 days ago

Excelent very helpfull. I just migrate from subversion to git and this guide is exactly what i need. Thanks Roger & Nina

Reply • Share ›

Kaia Konsap • 14 days ago

I LOVE this page, thank you!

Reply • Share ›

Eduardo Adrian Perez • 14 days ago

Excelent !

Reply • Share ›

Dennis Okparaocha • 14 days ago

Just what I needed. thanks :)

Reply • Share ›

Nerunjakumar Soubaya • 14 days ago

Very useful and handy. Thanks. I think, command to delete a file and also how to get it back could you added here.

⌃ | ⌄ • Reply • Share ›

David Méndez Acuña • 15 days ago

Just great! Thanks a lot!

⌃ | ⌄ • Reply • Share ›

Ryan Knutson • 17 days ago

This is wonderful! I now understand git! Thank you!

⌃ | ⌄ • Reply • Share ›

Febriyant Abidin • 18 days ago

nice web guide dude

⌃ | ⌄ • Reply • Share ›

karthick • 19 days ago

Awsome post, I had one issue with pull files from server, I am created a new branch in gitlab and pushed some files to my new repo , and then i get logged in with ssh and get some other files from other server through wget method , the files are get stored in server , and files are showing in browser , but when i pull the files it not getting into my localrepo , every i remove and again clone the repo but still i am not able to get the files which is get with wget method . Can someone help me on this.

⌃ | ⌄ • Reply • Share ›

Arunvel Sriram • 21 days ago

Simply explained. Very useful. Thank you.

⌃ | ⌄ • Reply • Share ›

Sriram B • 22 days ago

Really awesome !!! Simple and Useful :)

⌃ | ⌄ • Reply • Share ›

WalkerHaleIV • 24 days ago

Note that the download link for Mac OS X is out of date, currently gives an obsolete version, and will break when Google Code shuts down in 2016. Instead go here: http://git-scm.com/download/ma...

⌃ | ⌄ • Reply • Share ›

ameya • 24 days ago

Would have been better if you would have should every steps on gitbash/github with example.

⌃ | ⌄ • Reply • Share ›

Anubha Kumari • 24 days ago

Excellent concept! I've featured it on Hackr.io's Git section as well - http://hackr.io/tutorials/git

⌃ | ⌄ • Reply • Share ›

girish • a month ago

great work guys..

⌃ | ⌄ • Reply • Share ›

JustWatching • a month ago

"overall, best straight-forward guide I've ever seen." ditto. excellent work.

⌃ | ⌄ • Reply • Share ›

faizalheesyam • a month ago

would be great if you can add the "config" step, right before creating a new repository.

overall, best straight-forward guide I've ever seen.

this is for real, no deep shit!

∧ | ∨ • Reply • Share ›

pasumarthi • a month ago

I clone one repository from the github.And in that repo they made some changes and how can i update my local repository with those commits

∧ | ∨ • Reply • Share ›

faizalheesyam ➦ pasumarthi • a month ago

i believe you just have to execute `git pull` (on the same branch, ex: master)

∧ | ∨ • Reply • Share ›

djshaqtus • a month ago

Much appreciated. Muchas Gracias!

∧ | ∨ • Reply • Share ›

Chris • a month ago

Really appreciated this post.

∧ | ∨ • Reply • Share ›

sc1512 • a month ago

This just got me out of deep shit. Many thanks.

∧ | ∨ • Reply • Share ›

Anca Barbu • a month ago

I'm always checking this guide whenever I start a new project :)
Very clear and well organized!

Ty

∧ | ∨ • Reply • Share ›

Boyong Lambert • a month ago

Great post. It just saved me. Courage!!! It is worth putting into a pdf format

1 ∧ | ∨ • Reply • Share ›

JACK • a month ago

Thanks for your guide, well structured and defintily no deep shit!

∧ | ∨ • Reply • Share ›

hit • a month ago

Awesome...

∧ | ∨ • Reply • Share ›

Kaustubh Page • a month ago

Awesome...

∧ | ∨ • Reply • Share ›

Greg Werner • a month ago

good post thanks

∧ | ∨ • Reply • Share ›

Sarath Annareddy • a month ago

These are essential commands. Thanks for sharing.

∧ | ∨ • Reply • Share ›

Mahendran Kathirvel • a month ago

effective.

effective.

Reply • Share ›

disqus_whBZ9upNZ6 • a month ago

I use this all the time. So simple and useful.... thanks!

Reply • Share ›

Хусейн Мирахмади • 2 months ago

Thanks for keeping it simple .

Reply • Share ›

Load more comments

ALSO ON **GIT - THE SIMPLE GUIDE**

WHAT'S THIS?

git - 簡単ガイド

1 comment • 2 years ago

Avat **Mormont** — Git is git. I personally use it.

git - la guida tascabile - niente di complicato!

33 comments • 2 years ago

Avat **Flavio** — Questa guida forse e' utile a chi git lo conosce gia'. Io sto cercando di imparare a usarlo, e non mi e' ...

git - petit guide - no deep shit!

17 comments • 2 years ago

Avat **SGH** — Très utile, juste ce qu'il faut pour GIt et pas besoin de lire un bouquin encore Merci

git - 간편 안내서 - 어렵지 않아요!

59 comments • 2 years ago

Avat 하늘 — 뭐라고 하는지 모르겠어요. 하아.. 이거 ㅓㄹ마ㅣㄴㄹ어ㅏㅣㅁ넝라ㅣㅓㅏㅣㅁ널아ㅣㅓ

Subscribe Add Disqus to your site Privacy