

# A primer on using git and GitHub

@thomasvochten

# About me



## Thomas Vochten

Technology Evangelist       
#Microsoft365 #Azure #CommunityRocks

@thomasvochten

<https://thomasvochten.com>

[mail@thomasvochten.com](mailto:mail@thomasvochten.com)



**Microsoft®**  
Most Valuable  
Professional









**BIWUG**



The image features two puppets. The puppet on the left is designed to look like Albert Einstein, with wild white hair, a white mustache, and a dark jacket. It has a weary or skeptical expression, with its eyes looking slightly to the side. The puppet on the right is a larger, more comical character with a large, bulbous nose, thick eyebrows, and a wide, open-mouthed smile. It is also wearing a dark jacket. They are positioned in front of a background that appears to be a red curtain or a wall with vertical wooden slats. The overall lighting is somewhat dim, giving the scene a slightly somber or intimate feel.

Why exactly would I care?

# Rings a bell?

 Script\_todo\_Stuff.ps1  
 Script\_todo\_Stuff\_v2.ps1  
 Script\_todo\_Stuff\_v2-CustomerA.ps1  
 Script\_todo\_Stuff\_v3.ps1  
 Script\_todo\_Stuff\_v4.ps1  
 Script\_todo\_Stuff\_v4-CustomerB.ps1  
 Script\_todo\_Stuff\_v5.psm1  
 Script\_todo\_Stuff\_v5\_prod.psm1



Git is a free and open-source  
distributed version control system

# Key concepts

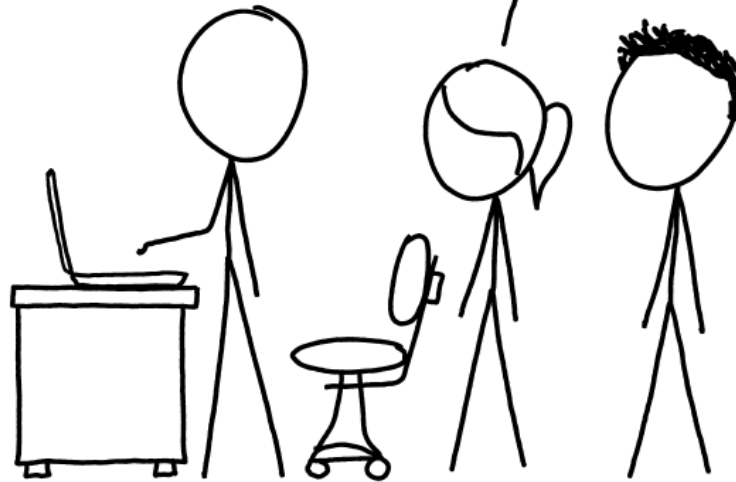
- Distributed
- Branching & merging
- Staging area



THIS IS GIT. IT TRACKS COLLABORATIVE WORK  
ON PROJECTS THROUGH A BEAUTIFUL  
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL  
COMMANDS AND TYPE THEM TO SYNC UP.  
IF YOU GET ERRORS, SAVE YOUR WORK  
ELSEWHERE, DELETE THE PROJECT,  
AND DOWNLOAD A FRESH COPY.



<https://ohshitgit.com/>

# Oh shit, git!

Git is hard: screwing up is easy, and figuring out how to fix your mistakes is fucking impossible. Git documentation has this chicken and egg problem where you can't search for how to get yourself out of a mess, unless you *already know the name of the thing you need to know about* in order to fix your problem.

So here are some bad situations I've gotten myself into, and how I eventually got myself out of them *in plain english*.\*

## Oh shit, I did something terribly wrong, please tell me git has a magic time machine!?!

```
git reflog
# you will see a list of every thing you've done in git, across all br
# each one has an index HEAD@{index}
# find the one before you broke everything
git reset HEAD@{index}
# magic time machine
```

You can use this to get back stuff you accidentally deleted, or just to remove some stuff you tried that broke the repo, or to recover after a bad merge, or just to go back to a time when things actually worked. I use `reflog` A LOT. Mega hat tip to the many many many many many people who suggested adding it!

## Oh shit, I committed and immediately realized I need to make one small change!



# Dangit, git!

Git is hard: screwing up is easy, and figuring out how to fix your mistakes is nigh on impossible. Git documentation has this chicken and egg problem where you can't search for how to get yourself out of a mess, unless you *already know the name of the thing you need to know about* in order to fix your problem.

So here are some bad situations I've gotten myself into, and how I eventually got myself out of them *in plain english*.\*

## Dangit, I did something terribly wrong, please tell me git has a magic time machine!?!

```
git reflog
# you will see a list of every thing you've done in git, across all br
# each one has an index HEAD@{index}
# find the one before you broke everything
git reset HEAD@{index}
# magic time machine
```

You can use this to get back stuff you accidentally deleted, or just to remove some stuff you tried that broke the repo, or to recover after a bad merge, or just to go back to a time when things actually worked. I use `reflog` A LOT. Mega hat tip to the many many many many many people who suggested adding it!

## Dangit, I committed and immediately realized I need to make one small change!

# What about GitHub then?

- Hosted git
- Acquired by Microsoft in 2018



## Some alternatives



Bitbucket



Gitlab

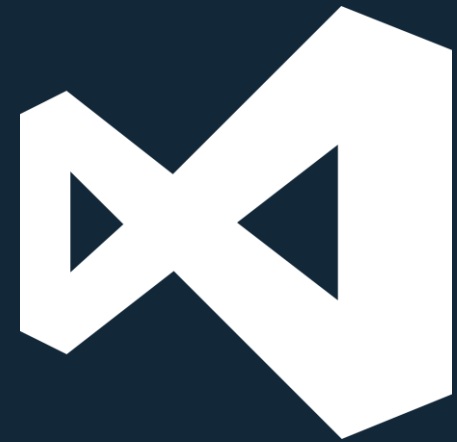


Azure Repos

Every geek loves tools!



Windows Terminal



Visual Studio Code

# Useful VSCode extensions



GitLens



Git History



GitHub Pull Requests & Issues

# GitHub CLI

```
$ gh issue list
```

```
Showing 4 of 4 issues in cli/cli
```

```
#16 Improving interactions with protected branches  
#14 PR commands on a detached head  
#13 Support for GitHub Enterprise (enhancement)  
#8 Add an easier upgrade command (bug)
```

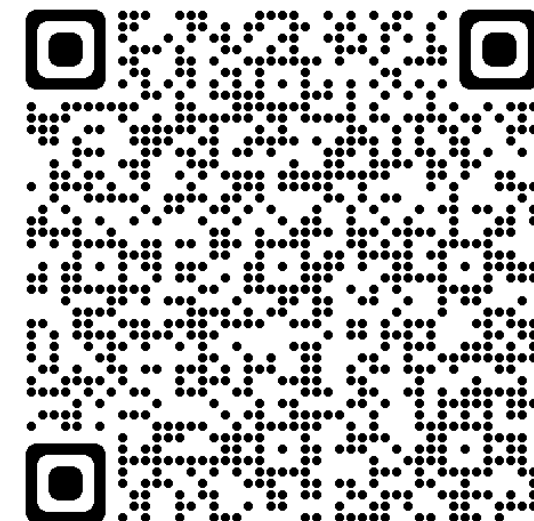
```
PowerShell
ThomasVochten@SB00K  ▸ winget-pkgs  ▸ OBSProject.OBSStudio.27.2.3  ~1  07:25:26  PS  90%
> dir

Directory: C:\Dev\winget-pkgs

Mode                LastWriteTime         Length Name
----                -
d-----          03/03/2022         09:28          .github
d-----          03/03/2022         09:28          .vscode
d-----          03/03/2022         09:28      DevOpsPipelineDefinitions
d-----          03/03/2022         09:28          doc
d-----          03/03/2022         09:28      manifests
d-----          03/03/2022         09:28      schemas
d-----          03/03/2022         09:28      Tools
-a----          03/03/2022         09:28         152 .editorconfig
-a----          03/03/2022         09:28          16 .gitattributes
-a----          03/03/2022         09:28       5985 .gitignore
-a----          03/03/2022         09:28       5656 AUTHORIZING_MANIFESTS.md
```

Upgrade your command prompt

```
-a----          03/03/2022         09:28       2789 SECURITY.md
-a----          03/03/2022         09:28        415 THIRD_PARTY.md
-a----          03/03/2022         09:28       2825 Troubleshoot.md
```



# Demo

Getting started with git & GitHub

@thomasvochten

# Common git & GitHub lingo

- Checking in your changes (**committing**)
- Download a remote repo to your pc (**cloning**)
- Copy your changes to a remote repository (**pushing**)
- Getting changes from the remote repository (**pulling**)
- Creating a separate area to work on stuff (**branching**)
- Integrating those changes back in (**merging**)
- Creating your own version of existing code on GitHub (**forking**)
- Asking to merge your work into the main copy (**pull request**)




# Basic git workflow

- You clone or create a repo
- You work in your local copy of the repo



- You add & commit changes to your local repo
- You push to the remote repo (e.g. GitHub) when you're ready
- You pull from the remote repo regularly to stay up to date

# Open-source contribution workflow

- Fork the repo you want to contribute to
- Create a branch for your change
- Make & test your changes
- Commit your changes to your local repo
- Push your changes to your remote repo
- Create a pull request
- Wait for the pull request to be accepted 
- Remove your fork

# GitHub Skills



<https://github.com/skills>

git docs: <https://git-scm.com/doc>

@thomasvochten



# Takeaways

- Learning by doing is key to becoming comfortable with git
- Don't be afraid of the command line
- Customize and extend your toolbox, it's not just eye candy
- Commit, push & pull as often as you can!
- Contribute to open source - Sharing is caring 💖

@thomasvochten

In case of fire



1. git commit
2. git push
3. leave building

Thank you

@thomasvochten

<https://thomasvochten.com>

[mail@thomasvochten.com](mailto:mail@thomasvochten.com)