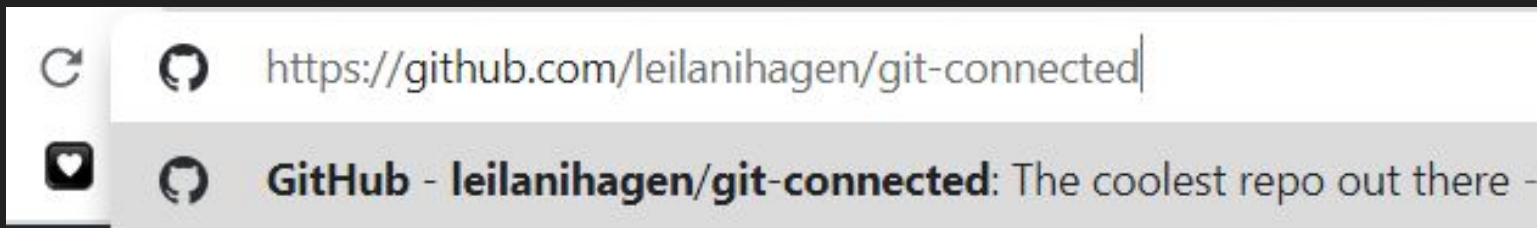


Git-Connected

Get our slides!

1.



Go to <https://github.com/leilanihagen/git-connected>

(No GitHub account necessary)

Get our slides!

2.

leilanihagen / git-connected

<> Code ! Issues 🔗 Pull requests ▶ Actions 📁 Projects 🛡 Security 📊 Insights

Join GitHub today
GitHub is home to over 50 million developers working together to host and review code, manage projects, and build software together.
[Sign up](#)


🔗 main ▾ 🌿 1 branch 🏷 0 tags [Go to file](#) [Code ▾](#)

leilanihagen Initial commit! 3090570 39 seconds ago ⌚ 1 commits

📄	Git-Connected.pdf	Initial commit!	39 seconds ago
📄	Git-Connected.pptx	Initial commit!	39 seconds ago


Get our slides!


3.


 main ▾

git-connected / Git-Connected.pdf


Go to file ...

 leilanihagen Initial commit!



Latest commit 3090570 6 minutes ago  History

 1 contributor

3.24 MB



Download

File management



Navigation icons: Back, Forward, Up, Down, Home, Search. Address bar: STAT297 FINAL. Search bar: Search STAT297 FINAL.

Left sidebar: Favorites, OneDrive, This PC, Network.

Name	Date modified	Type	Size
final-draft	8/1/2017 3:48 PM	Microsoft Word D...	16 KB
final-draft2	8/3/2017 3:49 PM	Microsoft Word D...	12 KB
final-draft-final	8/3/2017 10:22 PM	Microsoft Word D...	13 KB
final-modified	8/8/2017 3:49 PM	Microsoft Word D...	13 KB
real-final-draft	8/9/2017 2:22 PM	Microsoft Word D...	13 KB
real-final-draft-v2	8/10/2017 12:33 PM	Microsoft Word D...	16 KB
final	8/11/2017 11:59 PM	Microsoft Word D...	72 KB

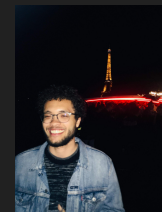
Git




- Distributed
- Fast
- Secure
- Offline




Installing git (Windows)




 **git** --fast-version-control


Git is a [free and open source](#) distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is [easy to learn](#) and has a [tiny footprint with lightning fast performance](#). It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like [cheap local branching](#), convenient [staging areas](#), and [multiple workflows](#).




**About**


The advantages of Git compared to other source control systems.

**Documentation**


Command reference pages, Pro Git book content, videos and other material.


**Downloads**





GUI clients and binary releases for all major platforms.

**Community**

Get involved! Bug reporting, mailing list, chat, development and more.

**Pro Git** by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).



 [Windows GUIs](#)  [Tarballs](#)
 [Mac Build](#)  [Source Code](#)

Downloads



Mac OS X



Windows

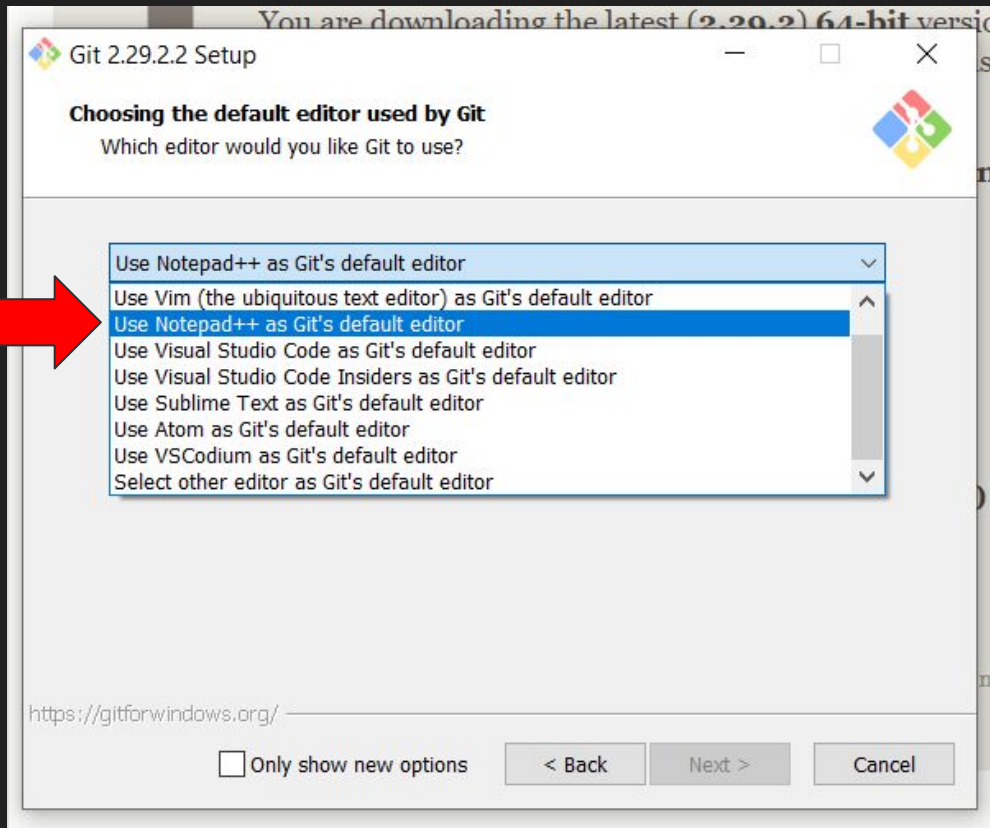


Linux/Unix

Older releases are available and the [Git source repository](#) is on GitHub.

<https://git-scm.com/downloads>

Installing git (Windows)



Notepad++ is much easier to use than Vim. In Vim you cannot use your mouse/cursor.

<https://git-scm.com/downloads/win>

Installing git (Windows)



Use Notepad++ as Git's default editor

(NEW!) [Notepad++](#) is a popular GUI editor that can be used by Git.

Downloads

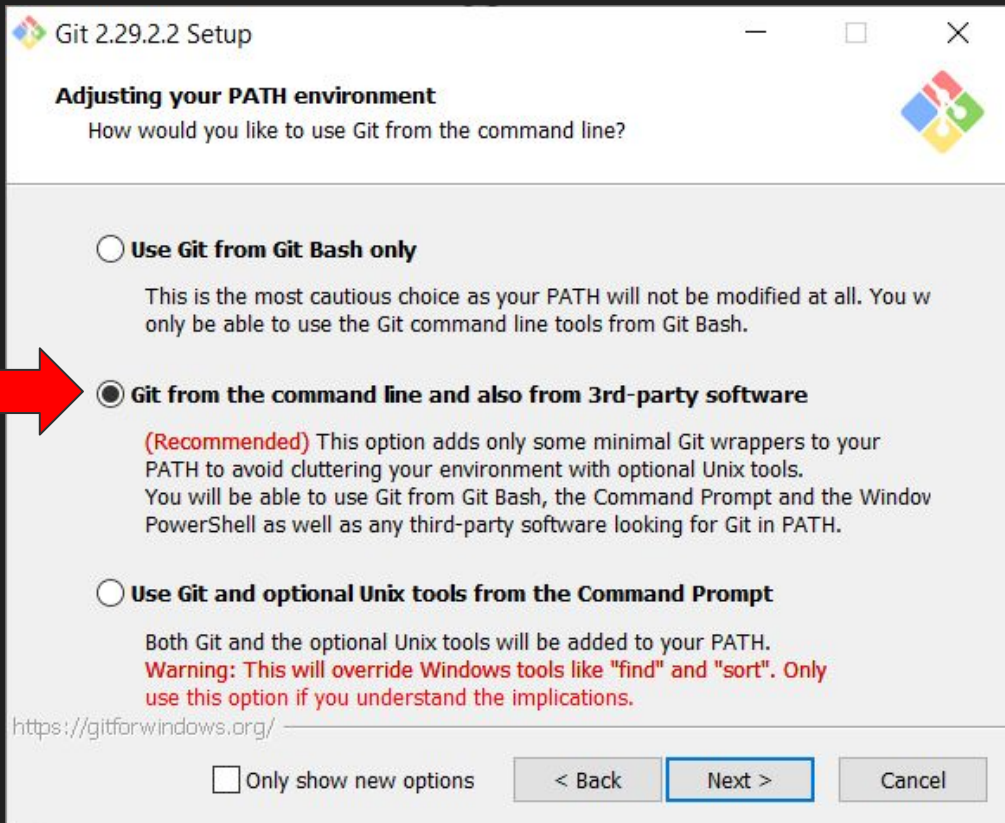
➔ Notepad++ 7.9.1: pour Samuel Paty

Download 64-bit x64

- [Installer](#) | [GPG Signature](#)

<https://git-scm.com/downloads/win>

Installing git (Windows)



This option allows you to use git from git bash, Windows cmd, PowerShell and other places!

<https://git-scm.com/downloads/win>



Installing Git - Mac OSX

If you have Xcode Command Line Tools on your mac, you already have git.

Type '**git --version**' and press enter

```
Leila — -bash — 80x24
[(base) /Users/Leila $ git --version]
```

If you see similar output, a git version number, then you already have git, and you can skip the next steps!

```
Leila — -bash — 80x24
[(base) /Users/Leila $ git --version
git version 2.20.1 (Apple Git-117)
(base) /Users/Leila $
```



Installing Git - Mac OSX - Install a Disk Image

Navigate to: <https://sourceforge.net/projects/git-osx-installer/>

Click on the link to download the latest disk image file (it will be at the top of the list)

Project Activity

Released [/git-2.27.0-intel-universal-mavericks.dmg](#)



4 months ago

Installing Git - Mac OSX - Check Installation



Check to make sure your install is successful with the following command:

Type '**git --version**' and press enter

```
Leila — -bash — 80x24
(base) /Users/Leila $ git --version
```

If you see similar output, congrats, you have git!

```
Leila — -bash — 80x24
(base) /Users/Leila $ git --version
git version 2.20.1 (Apple Git-117)
(base) /Users/Leila $
```

Issues setting your PATH? Get help [here](#)



Installing Git - Linux - Debian-based distros

If you have a Debian based Linux distribution, use the following command:

Debian ditros:

Ubuntu, Linux Mint, MX Linux, Deepin Linux, Kali Linux, PureOS.

Type '**sudo apt install git-all**' and press enter

leilani@leilani-MacBookPro: ~

File Edit View Search Terminal Help

```
leilani@leilani-MacBookPro:~$ sudo apt install git-all
```



Installing Git - Linux - RPM-based distros

If you have an RPM-based Linux distribution, use the following command:

RPM distros: Fedora, RHEL, CentOS.

Type **'sudo dnf install git-all'** and press enter

```
:~$ sudo dnf install git-all
```



Installing Git - Linux - Check Installation

Type '**git --version**' and press enter

If you see a version number, your install was successful!

Git Bash First time set up



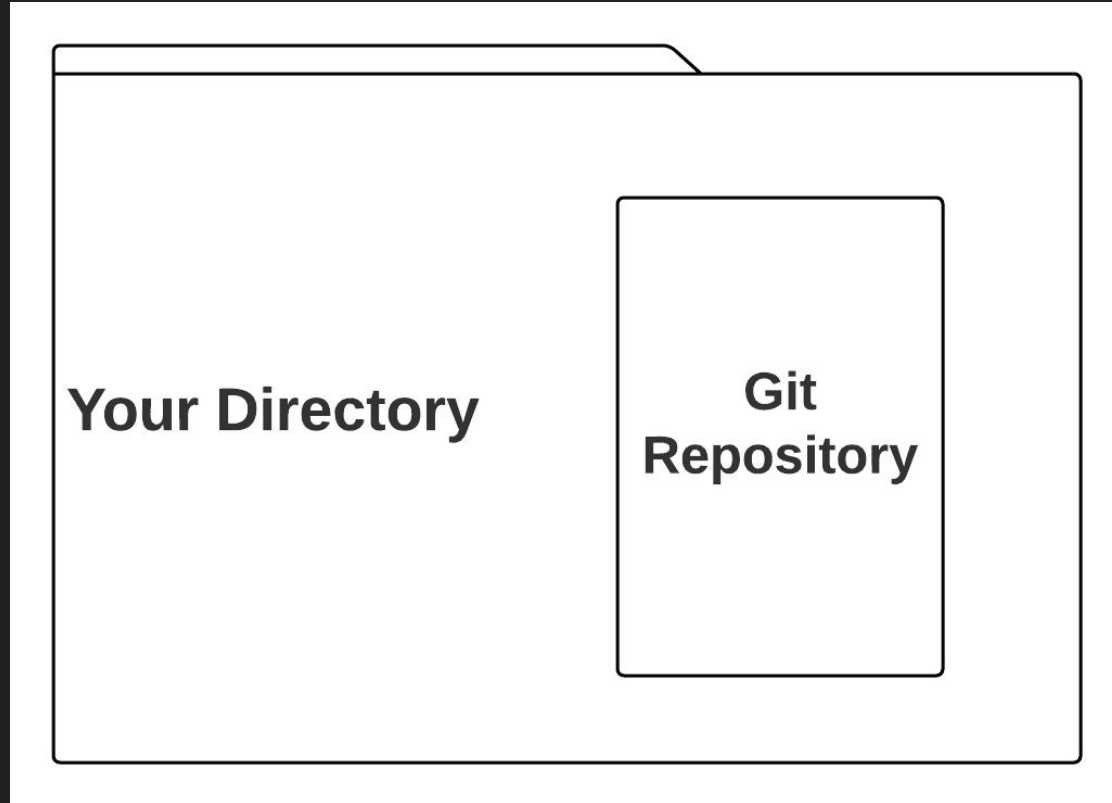
- Git Bash
- User Settings
 - `Git config --global user-name <user-name>`
 - `Git config --global user-email <your-email>`
- Terminal commands
 - `Mkdir <directory-name>`
- Initializing a Repository
 - `Git init`

Initializing a Local Repository



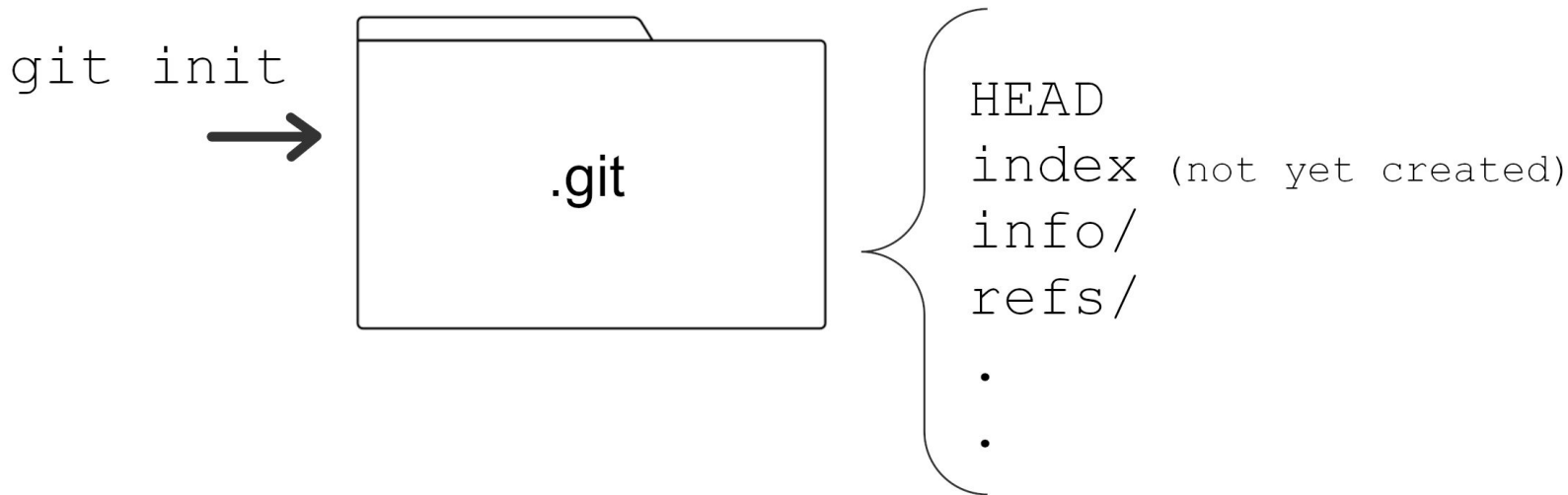
```
michael@Michael123Laptop MINGW64 ~/Desktop/Projects/git_demo
$ git init
Initialized empty Git repository in C:/Users/micha/Desktop/Projects/git_demo/.git/
```

So... what is a repository, really? Directory?



What makes the repository?

When you run `git init`, git creates the repository which is a hidden folder in your working directory. This folder contains important files and folders that Git uses to operate.

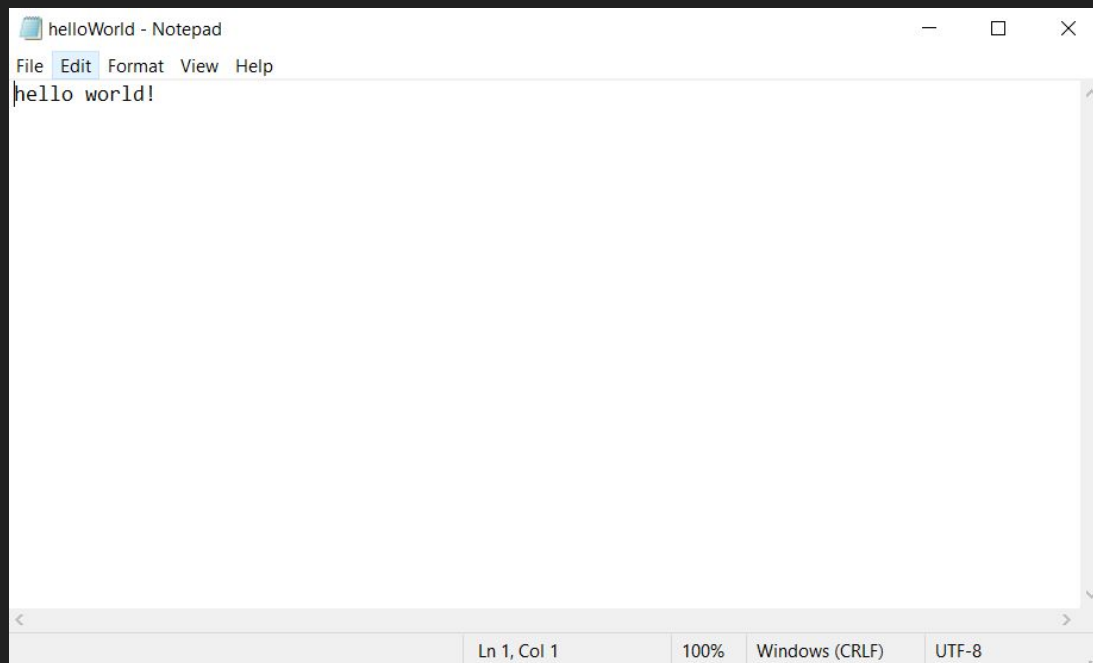


creating & editing files through Git bash



- Creating Files
 - `touch <filename>`
- Editing Files From command line
 - `start <filename>`

```
michael@Michael123Laptop MINGW64 ~/Desktop/Projects/git_demo (master)  
$ start HelloWorld.txt
```



Staging Files

- Git Status
- Git Add
 - `git add <filename>`
 - `git add .`





Type **'git status'** and press enter

```
michael@Michael123Laptop MINGW64 ~/Desktop/Projects/git_demo (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    hello-world.txt
    winter-todo.txt

nothing added to commit but untracked files present (use "git add" to track)
```




```
michael@Michael123Laptop MINGW64 ~/Desktop/Projects/git_demo (master)
$ git status
On branch master

No commits yet

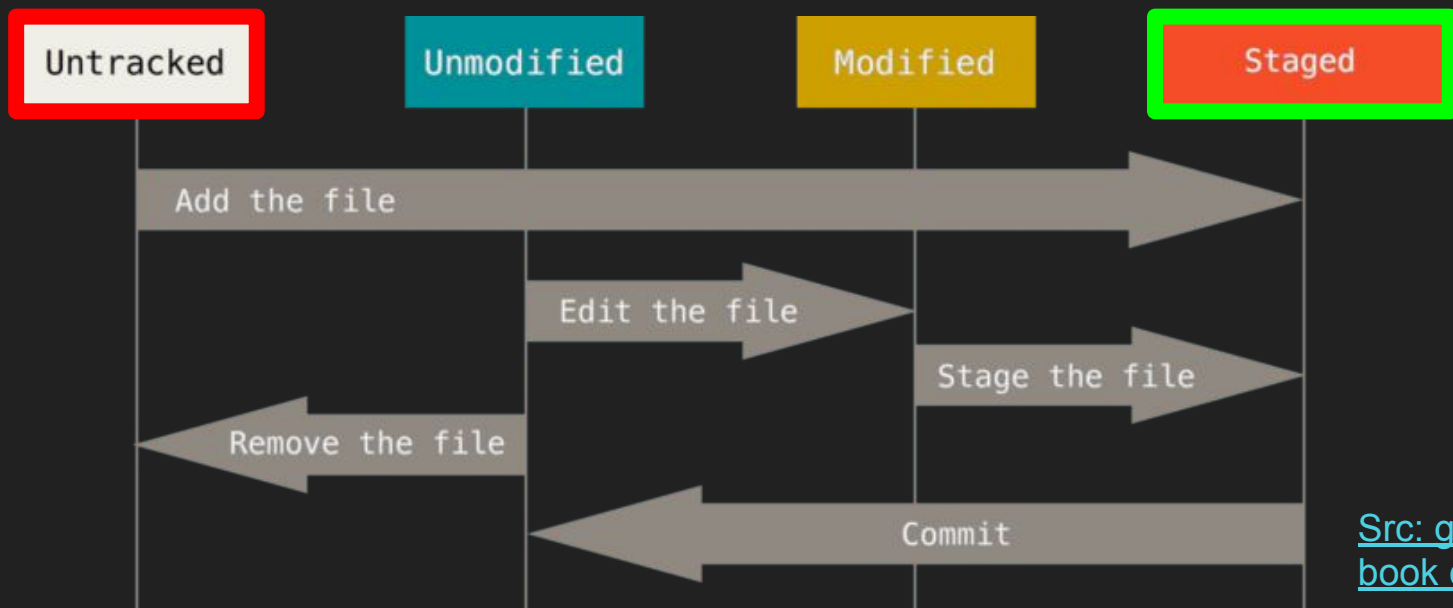
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   hello-world.txt
        new file:   winter-todo.txt
```

File Staging

The lifecycle of your files in a git repository will follow this flow.

Tracked = included in the repository's previous commits

Staged = added to the current index, ready to be committed in the next commit



[Src: git-scm book chapter 2.2](#)



What is a commit?



Commit "snapshot"



Committing



Prepare a **payload** for pushing any changes into remote repository. The commit contains logs file (what have you changed in your code, files).

Command:

```
$ git commit -m "[your message go here]"
```

Note that every commit require message, -m stand for message.



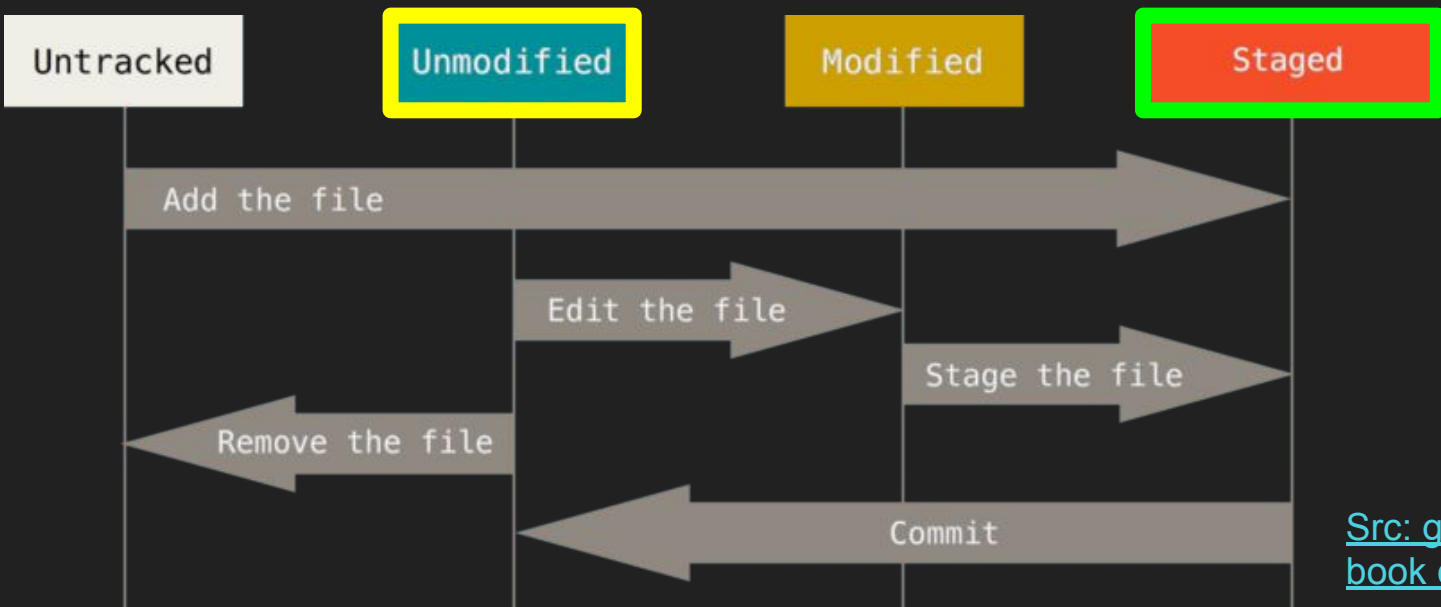
```
michael@Michael123Laptop MINGW64 ~/Desktop/Projects/git_demo (master)
$ git commit -m"initial commit"
[master (root-commit) 7b95eb3] initial commit
2 files changed, 1 insertion(+)
create mode 100644 helloworld.txt
create mode 100644 history_of_the_world.txt
```

File Tracking and Staging

The lifecycle of your files in a git repository will follow this flow.

Tracked = included in the repository's previous commits

Staged = added to the current index, ready to be committed in the next commit



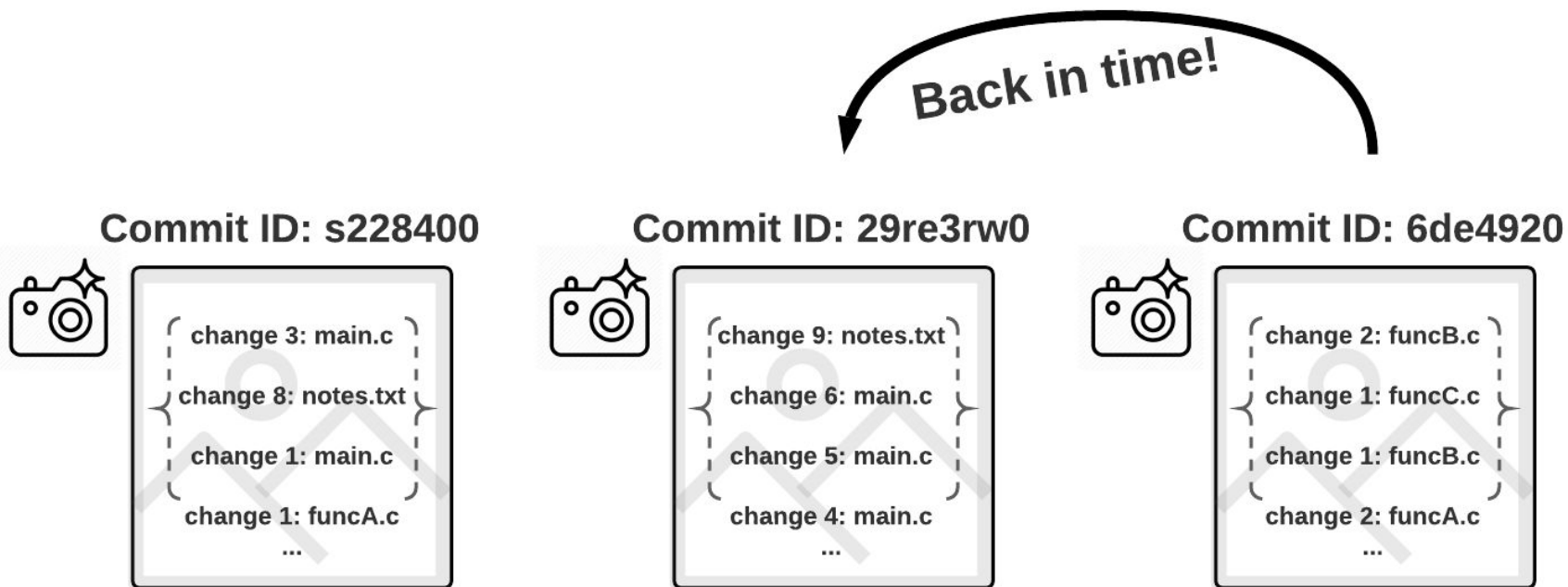
[Src: git-scm book chapter 2.2](#)





Undoing Things!

- No universal “undo” command
- Go back in time!





Undoing Things! - Git Log

We can use the git log command to view the git log - a saved list of recent commits made in your repository.

```
MINGW64:/c:/Users/hagen/git-connected

hagen@LAPTOP-VH8DI77G MINGW64 ~/git-connected (master)
$ git log
commit 5e09c34bea092d0222de1c2b043c1a9fdf381786 (HEAD -> master)
Author: Leilani Hagen <hagen.leilani@gmail.com>
Date: Thu Nov 19 10:17:03 2020 -0800
    Making my bucket list more realistic.
commit 6e34257bd52e7052654a31962a7818ca4320aaea
Author: Leilani Hagen <hagen.leilani@gmail.com>
Date: Thu Nov 19 10:16:14 2020 -0800
    Initial commit.
hagen@LAPTOP-VH8DI77G MINGW64 ~/git-connected (master)
$ |
```

Undoing Things! - Revert



Simplest and safest - commits the inverse of the commit you are reverting.



Read more about “undoing”: <https://www.atlassian.com/git/tutorials/undoing-changes>

Undoing Things! - Revert



To perform git revert:

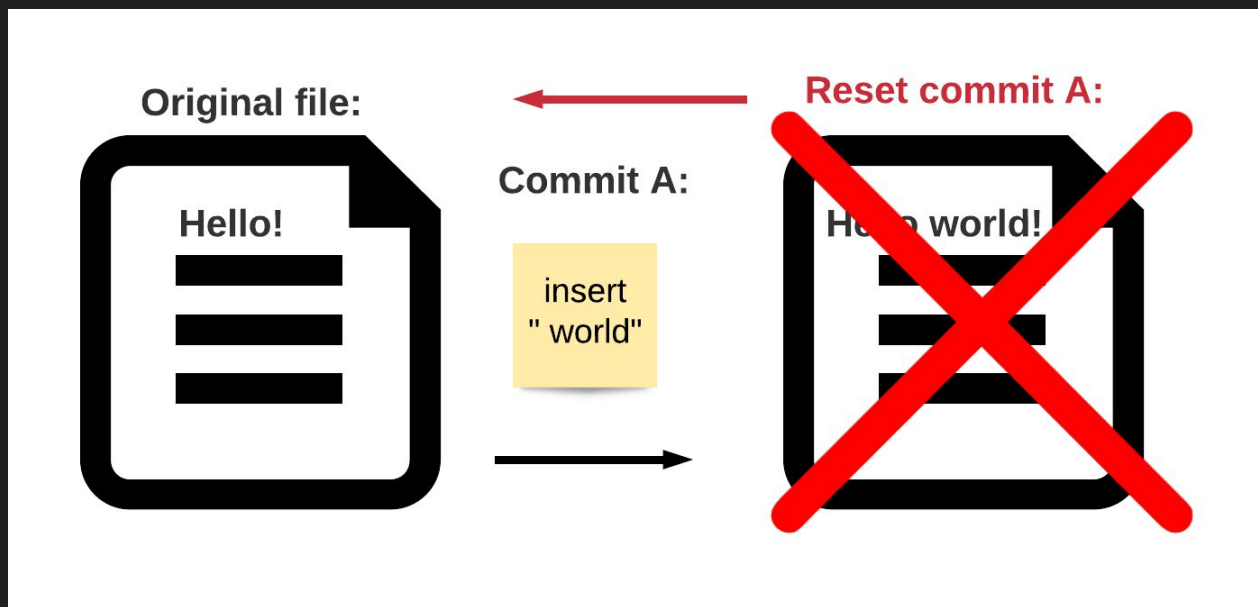
1. Copy the first 8 digits of the Commit ID from git log
2. Type '**git revert <commit ID>**'
3. Enter your revert commit message or accept the default

Read more about “undoing”: <https://www.atlassian.com/git/tutorials/undoing-changes>



Undoing Things! - Reset

Reset simply wipes git's history of a commit, making it as if the commit never happened and returning to the earlier commit



Read more about “undoing”: <https://www.atlassian.com/git/tutorials/undoing-changes>



Accidentally Committed too Early?

Just amend! Amend lets you add more staged changes to a previous commit

To amend to your last commit, type '**git commit --amend**'
and press enter



When Should I Commit? How Often? WHY?!

How often?

- Every time you reach a point in your project that you want to be able to go back to
- More frequent commits is best

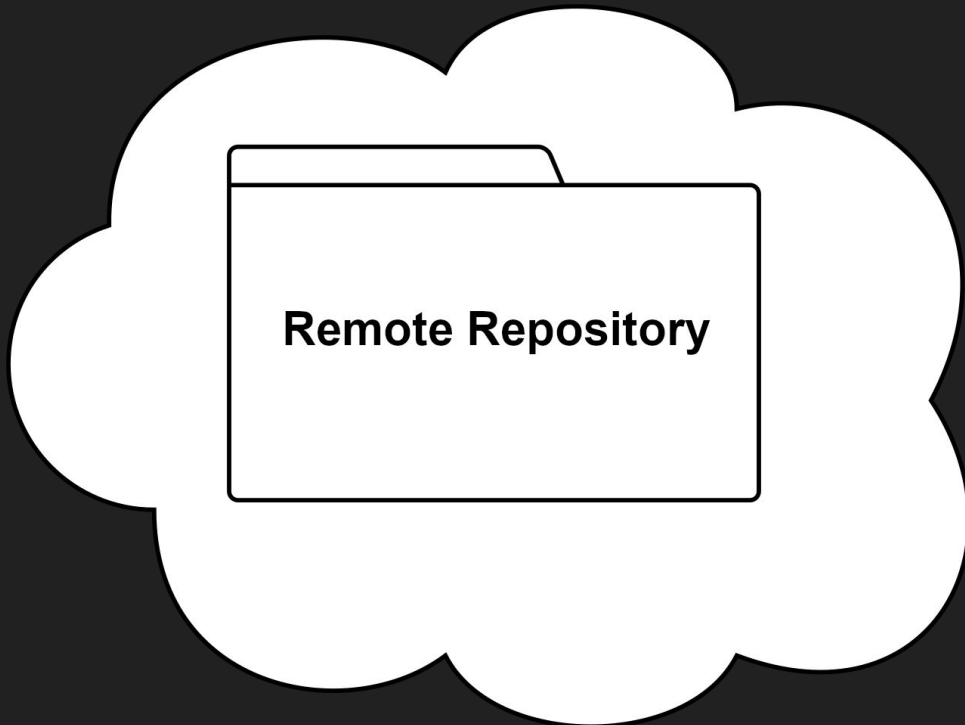
Why?!

- It saves your place in your project
- Documents the work in your project! Without you even thinking about it =)

Remote Repositories

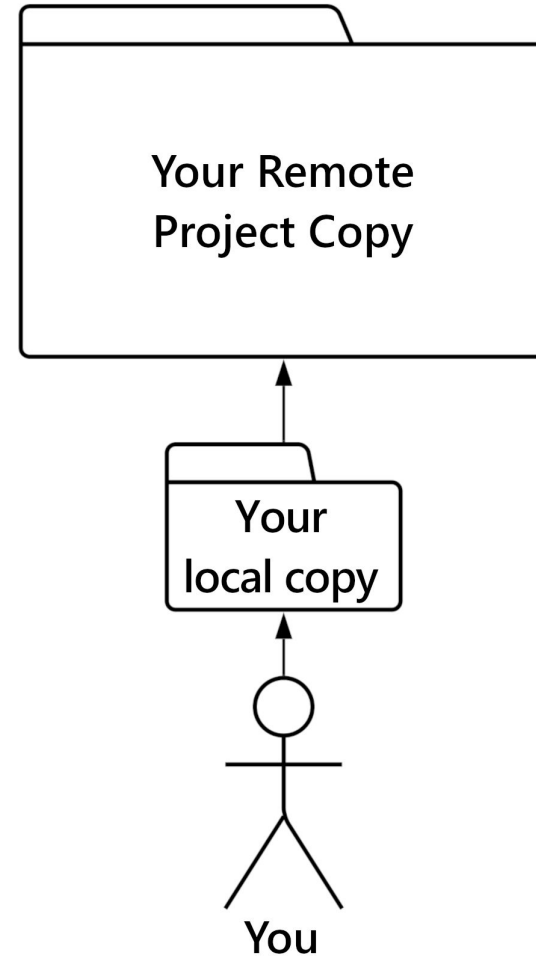


Remote repositories are git repositories, just like the one we've just created, that live on the cloud somewhere.



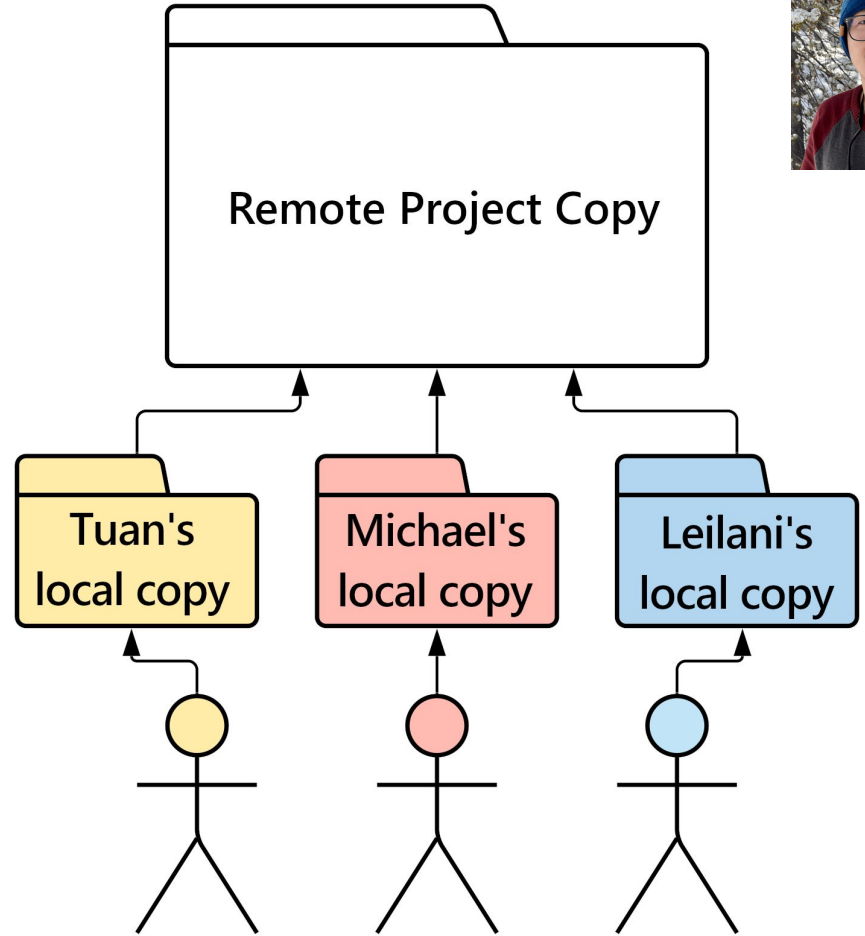
Remote Repositories - Individual

Remote repositories can be used as a way to back-up and version control your own files, without collaboration.



Remote Repositories - Collaboration

Remote repositories are often used as the central point of collaboration between team members or classmates.





SSH Access

In order work with remote repositories, we need to configure a public SSH key.

SSH = Secure Shell network protocol. In short, it lets your computer securely transfer files to your Git remote repository over an unsecured open network. SSH accomplishes this by using a pair of keys, one public key and one private key.

Open git bash

Command: \$ ssh-keygen

Src (read more about SSH keys): <https://www.atlassian.com/git/tutorials/git-ssh>



```
nhttu@g7-TuanNguyen MINGW64 ~
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/nhttu/.ssh/id_rsa):
/c/Users/nhttu/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/nhttu/.ssh/id_rsa.
Your public key has been saved in /c/Users/nhttu/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:WGeC [REDACTED] zp96c0 nhttu@g7-TuanNguyen
The key's randomart image is:
+---[RSA 3072]-----+
[REDACTED]
|
|  .+.o+ =o +
| =+.+ o.o +
| OB . E S
| @==
| *o..
| o
|
+-----[SHA256]-----+
nhttu@g7-TuanNguyen MINGW64 ~
```

Creating Your Remote Repository

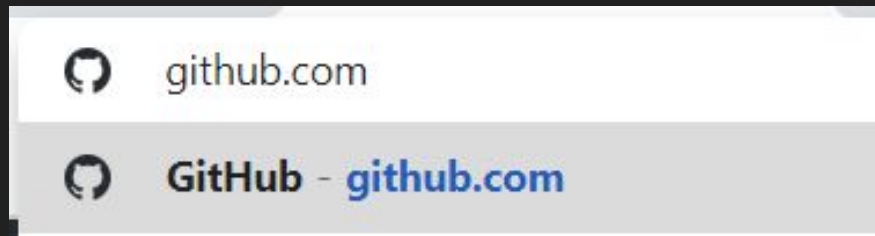


Once you have configured your SSH keys, you can create a remote and add it to your local repository's list of remotes. You can then use the remote repo to back up your local repository that you just created.

Creating Your Remote Repository



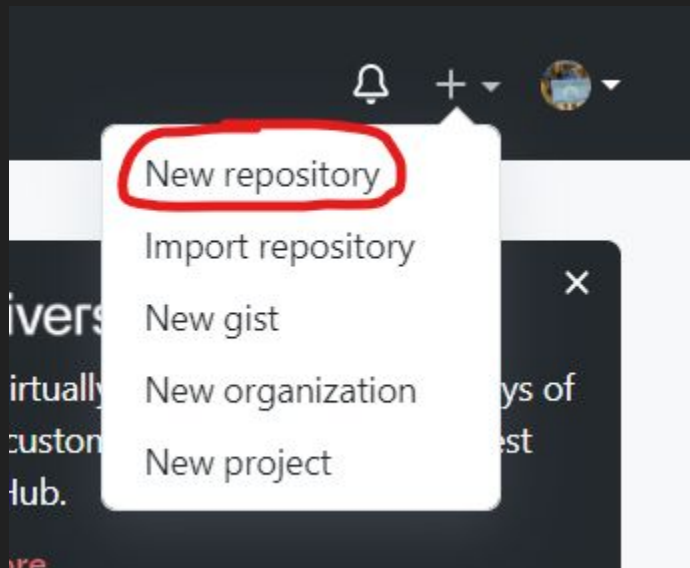
Navigate to [GitHub.com](https://github.com) and sign in or create an account.



Creating Your Remote Repository



Once you are signed in, find the '+' icon in the top right corner, click and choose 'New repository'





Creating Your Remote Repository - Name

Choose a name (NO SPACES, dashes are common convention), and add a description if you like.

Owner *



leilanihagen ▾

/

Repository name *

my-awesome-repo



Great repository names are short and memorable. Need inspiration? How about **legendary-octo-bassoon**?

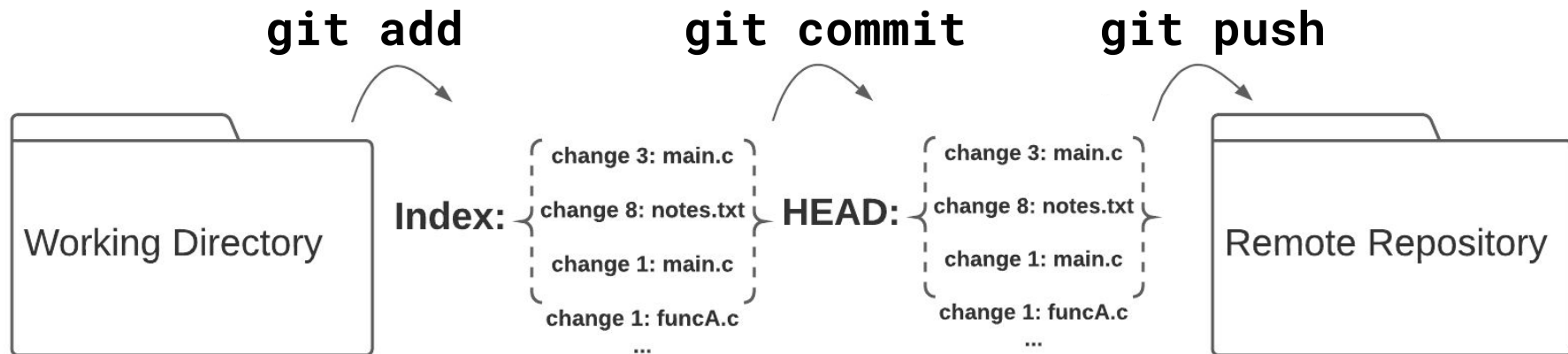
Description (optional)

The coolest repo out there

Local-Remote Workflow



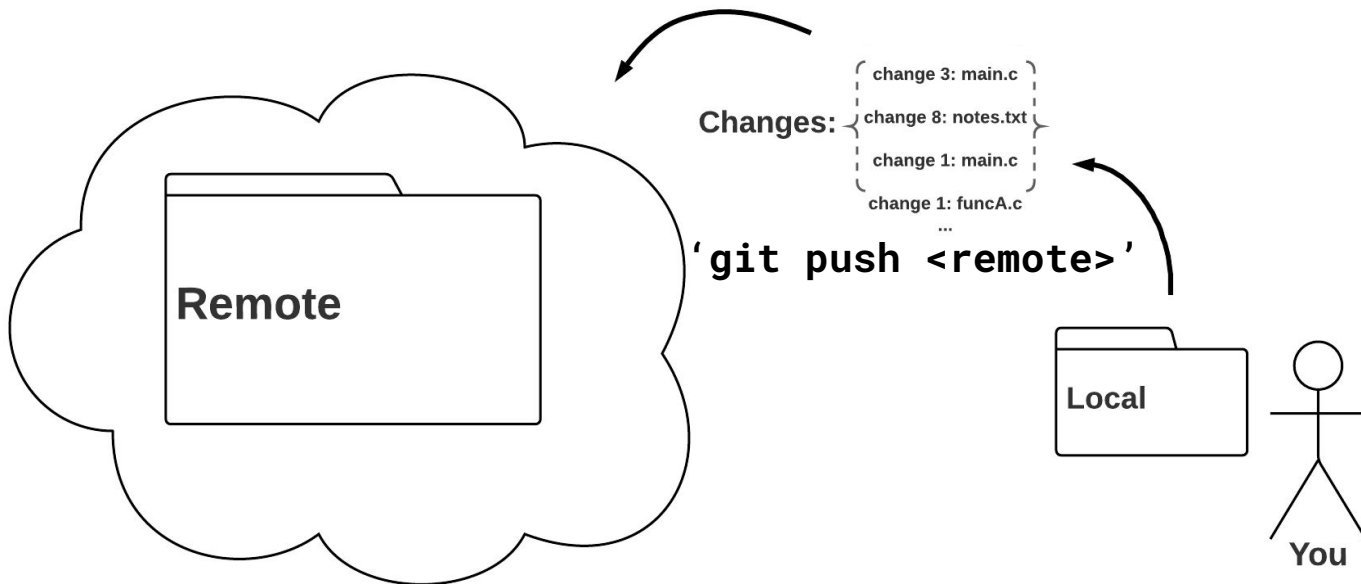
The Basic Local-Remote Git Workflow





Pushing Changes to a Remote Repository

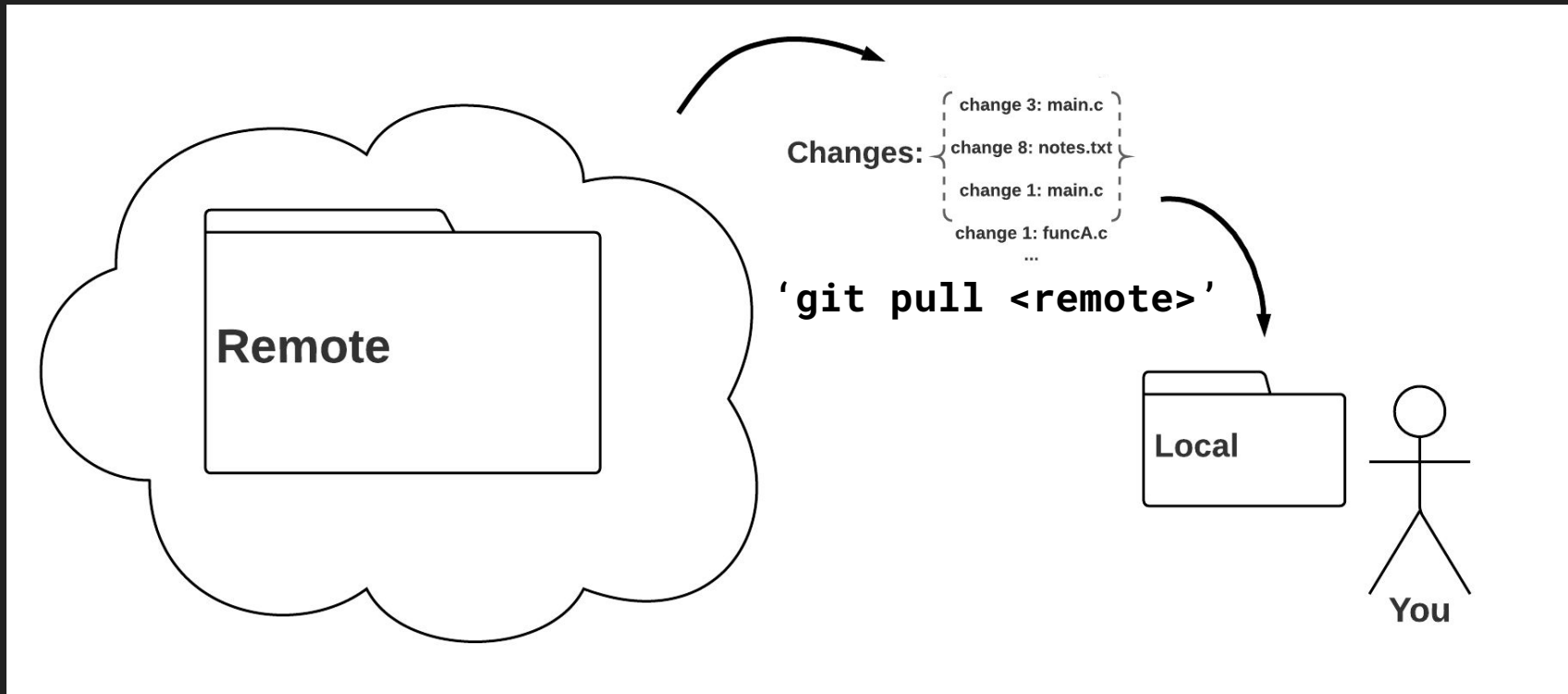
Pushing is how you transfer a commit (group of tracked changes) from your local repository to your remote repository.



Pulling Changes from a Remote Repository



Pulling is how you update your local repository to be up-to-date with the current state of the remote repository.





Q&A - No Tough Questions!

(Here's a Git cheat sheet to remember all of today's commands, plus more!):
<https://training.github.com/downloads/github-git-cheat-sheet/>