

Fundamentos de programación y scripting
Erick Estuardo Dávila Hernández
Carné: 999011545

Git

¿Qué es Git?

Es una herramienta de control de versiones que se utiliza para dar seguimiento de los cambios realizados principalmente en archivos de texto (ideal para código de programación). Facilita la inspección, recuperación y combinación de las modificaciones realizadas gracias a sus puntos de control (commits) registrados en una bitácora (log). Además cuenta con varias características que agilizan el trabajo colaborativo (accounts, branches, pull requests, etc).

Control de versiones con Git

Para gestionar las versiones utilizando Git existen diversos modelos, en el caso del desarrollo de software el más recomendado es el desarrollo basado en trunks. Es una práctica en la que los desarrolladores fusionan pequeñas actualizaciones de forma frecuente a la rama principal. Se ha convertido en una práctica habitual entre los equipos de DevOps ya que simplifica las fases de fusión e integración, prácticas obligatorias de la CI y la CD. Permite a los desarrolladores crear ramas de corta duración con pequeños commits, a diferencia de otras estrategias de ramas de larga duración. A medida que la complejidad, el tamaño del código y del equipo crece, el desarrollo basado en trunks ayuda a mantener el flujo de publicación de la producción.

Estados de un archivo en Git

Los archivos pueden estar en tres estados diferentes:

Working directory

Es el estado en el que se encuentra el archivo cuando se están realizando cambios en él.

Staging area

Es el estado en el que se encuentra el archivo cuando se ha modificado y se ha agregado a la zona de preparación (al crear un nuevo archivo dentro del repositorio).

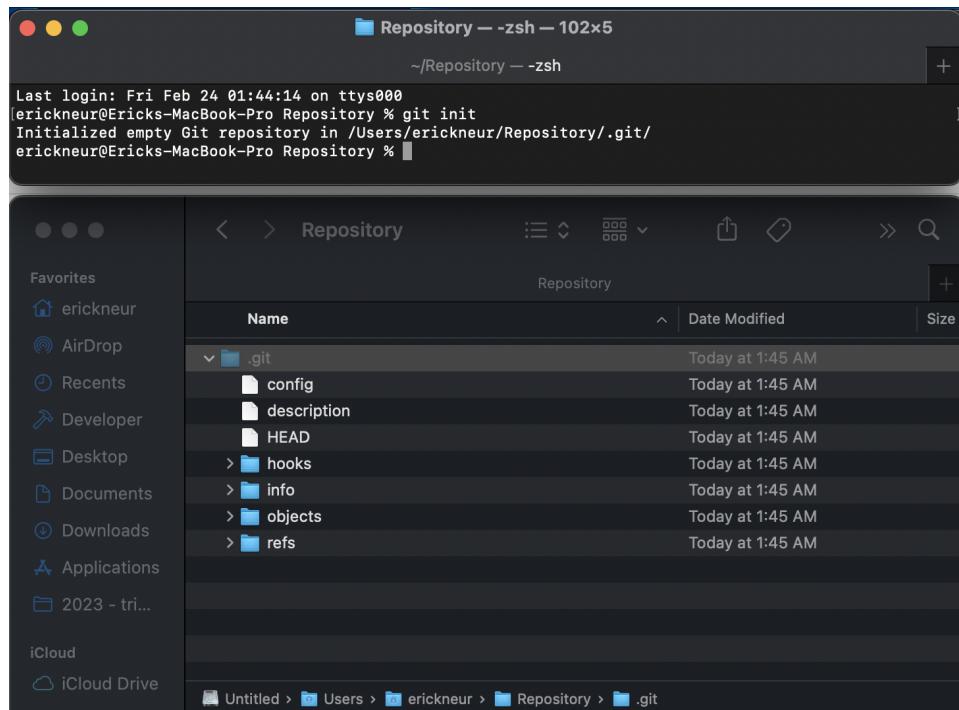
Repository

Es el estado en el que se encuentra el archivo cuando se ha confirmado en el repositorio.

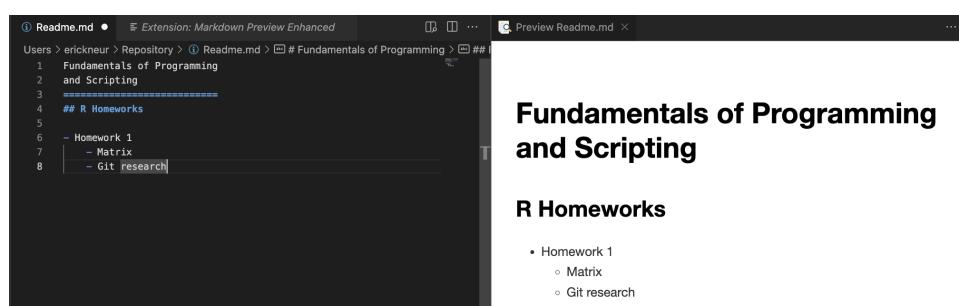
¿Cómo se configura un repositorio?

Para configurar un repositorio Git, se pueden seguir los siguientes pasos utilizando una terminal de comandos:

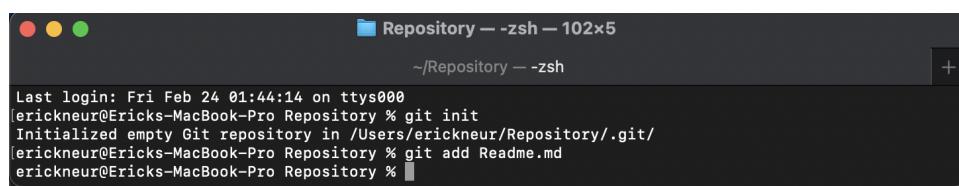
1. Crear un nuevo directorio y ejecutar el comando `git init` el cuál agrega una carpeta oculta llamada `.git` con todos los componentes necesarios para el control de versiones.



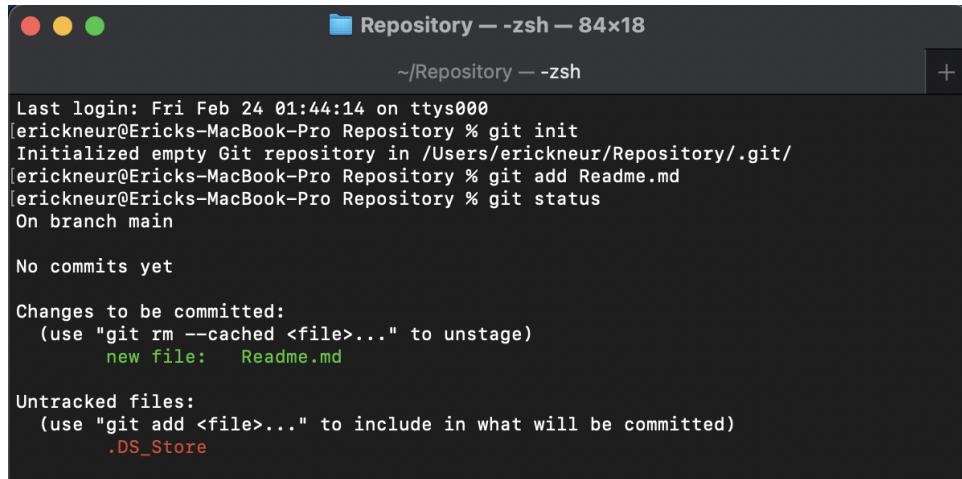
2. Crear un archivo README.md en la carpeta del repositorio, el cuál contendrá la descripción del mismo utilizando el formato Markdown.



3. Agregar el archivo al Staging area utilizando el comando `git add`.



4. Revisa que el estado de los archivos sea el correcto con el comando `git status`.



```
Repository — zsh — 84x18
~/Repository — zsh

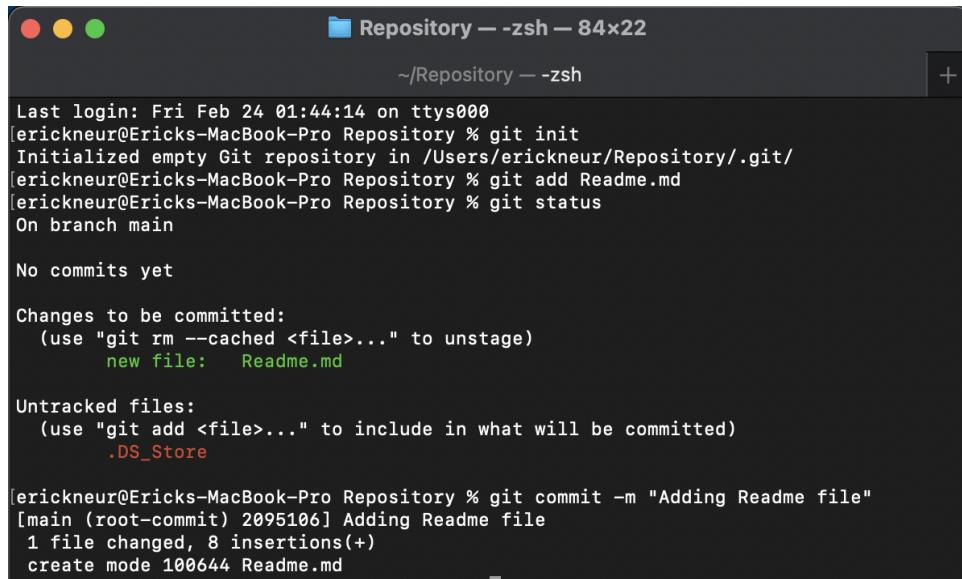
Last login: Fri Feb 24 01:44:14 on ttys000
[erickneur@Ericks-MacBook-Pro Repository % git init
Initialized empty Git repository in /Users/erickneur/Repository/.git/
[erickneur@Ericks-MacBook-Pro Repository % git add Readme.md
[erickneur@Ericks-MacBook-Pro Repository % git status
On branch main

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)
  new file:   Readme.md

Untracked files:
(use "git add <file>..." to include in what will be committed)
  .DS_Store
```

5. Confirmar los cambios utilizando el comando `git commit`.



```
Repository — zsh — 84x22
~/Repository — zsh

Last login: Fri Feb 24 01:44:14 on ttys000
[erickneur@Ericks-MacBook-Pro Repository % git init
Initialized empty Git repository in /Users/erickneur/Repository/.git/
[erickneur@Ericks-MacBook-Pro Repository % git add Readme.md
[erickneur@Ericks-MacBook-Pro Repository % git status
On branch main

No commits yet

Changes to be committed:
(use "git rm --cached <file>..." to unstage)
  new file:   Readme.md

Untracked files:
(use "git add <file>..." to include in what will be committed)
  .DS_Store

[erickneur@Ericks-MacBook-Pro Repository % git commit -m "Adding Readme file"
[main (root-commit) 2095106] Adding Readme file
  1 file changed, 8 insertions(+)
  create mode 100644 Readme.md
```

6. Crear un nuevo repositorio en Github o en el servicio de alojamiento que se prefiera y copiar la URL generada en formato HTTPS.

The image shows two screenshots of the GitHub interface. The top screenshot is a 'Create a new repository' form. It has fields for 'Owner' (set to 'Erickneur'), 'Repository name' (set to 'Repository'), and a 'Description (optional)' field which is empty. Below these are options for 'Public' or 'Private' repository status, both of which are selected. Under 'Initialize this repository with:', there is a checkbox for 'Add a README file' which is unchecked. There are also sections for '.gitignore' template (set to 'None') and 'Choose a license' (set to 'None'). A note at the bottom says 'You are creating a public repository in your personal account.' At the bottom is a large green 'Create repository' button. The bottom screenshot shows the resulting repository page for 'Erickneur / Repository'. The page title is 'Erickneur / Repository (Public)'. Below the title is a 'Quick setup' section with instructions for cloning the repository via HTTPS, SSH, or command line. It also includes sections for creating a new repository on the command line and pushing an existing repository from the command line.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Repository template
Start your repository with a template repository's contents.

No template ▾

Owner * Erickneur / **Repository name *** Repository ✓

Great repository names! Your new repository will be created as [Repository-.github.io](#) about [miniature-waffle](#)?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file
This is where you can write a long description for your project. [Learn more](#).

Add .gitignore
Choose which files not to track from a list of templates. [Learn more](#).

.gitignore template: None ▾

Choose a license
A license tells others what they can and can't do with your code. [Learn more](#).

License: None ▾

ⓘ You are creating a public repository in your personal account.

Create repository

github.com/Erickneur/Repository

Erickneur / Repository (Public)

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH https://github.com/Erickneur/Repository.git

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a `README`, `LICENSE`, and `.gitignore`.

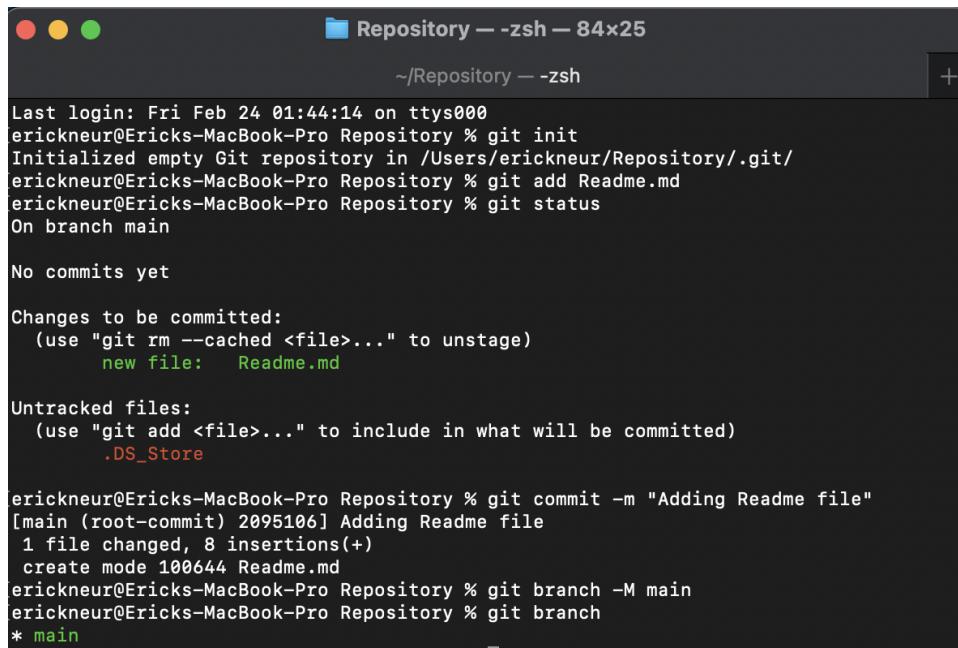
...or create a new repository on the command line

```
echo "# Repository" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/Erickneur/Repository.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/Erickneur/Repository.git
git branch -M main
git push -u origin main
```

7. Crear la rama principal del repositorio con el comando `git branch`.



```
Last login: Fri Feb 24 01:44:14 on ttys000
erickneur@Ericks-MacBook-Pro Repository % git init
Initialized empty Git repository in /Users/erickneur/Repository/.git/
erickneur@Ericks-MacBook-Pro Repository % git add Readme.md
erickneur@Ericks-MacBook-Pro Repository % git status
On branch main

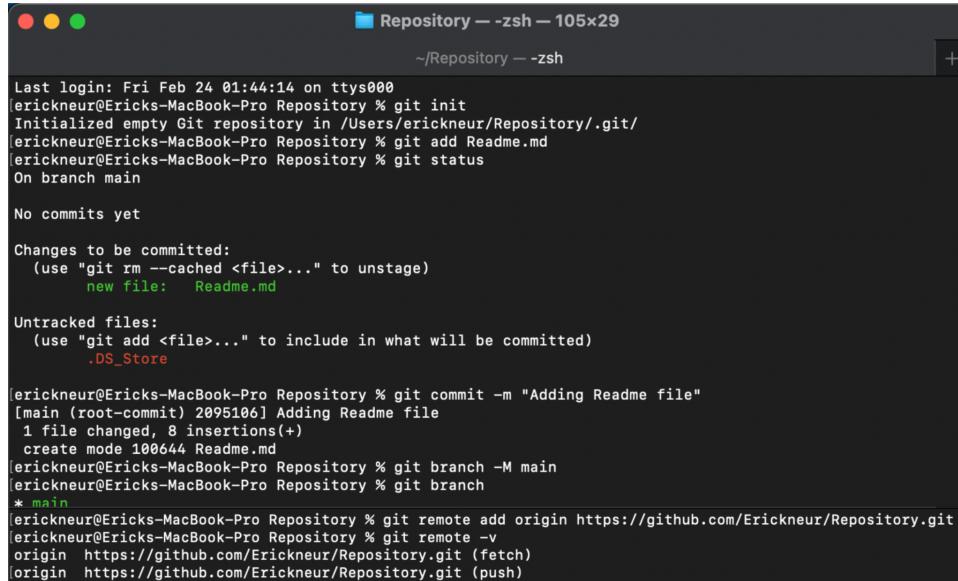
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Readme.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .DS_Store

erickneur@Ericks-MacBook-Pro Repository % git commit -m "Adding Readme file"
[main (root-commit) 2095106] Adding Readme file
  1 file changed, 8 insertions(+)
   create mode 100644 Readme.md
erickneur@Ericks-MacBook-Pro Repository % git branch -M main
erickneur@Ericks-MacBook-Pro Repository % git branch
* main
```

8. Conectar el repositorio local al repositorio remoto utilizando el comando `git remote`.



```
Last login: Fri Feb 24 01:44:14 on ttys000
erickneur@Ericks-MacBook-Pro Repository % git init
Initialized empty Git repository in /Users/erickneur/Repository/.git/
erickneur@Ericks-MacBook-Pro Repository % git add Readme.md
erickneur@Ericks-MacBook-Pro Repository % git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   Readme.md

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .DS_Store

erickneur@Ericks-MacBook-Pro Repository % git commit -m "Adding Readme file"
[main (root-commit) 2095106] Adding Readme file
  1 file changed, 8 insertions(+)
   create mode 100644 Readme.md
erickneur@Ericks-MacBook-Pro Repository % git branch -M main
erickneur@Ericks-MacBook-Pro Repository % git branch
* main
erickneur@Ericks-MacBook-Pro Repository % git remote add origin https://github.com/Erickneur/Repository.git
erickneur@Ericks-MacBook-Pro Repository % git remote -v
origin  https://github.com/Erickneur/Repository.git (fetch)
origin  https://github.com/Erickneur/Repository.git (push)
```

9. Cargar los cambios al repositorio remoto utilizando el comando `git push`.

```
Repository -- zsh -- 105x29
~/Repository -- zsh

Last login: Fri Feb 24 01:44:14 on ttys000
erickneur@Ericks-MacBook-Pro Repository % git init
Initialized empty Git repository in /Users/erickneur/Repository/.git/
erickneur@Ericks-MacBook-Pro Repository % git add Readme.md
erickneur@Ericks-MacBook-Pro Repository % git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:  Readme.md

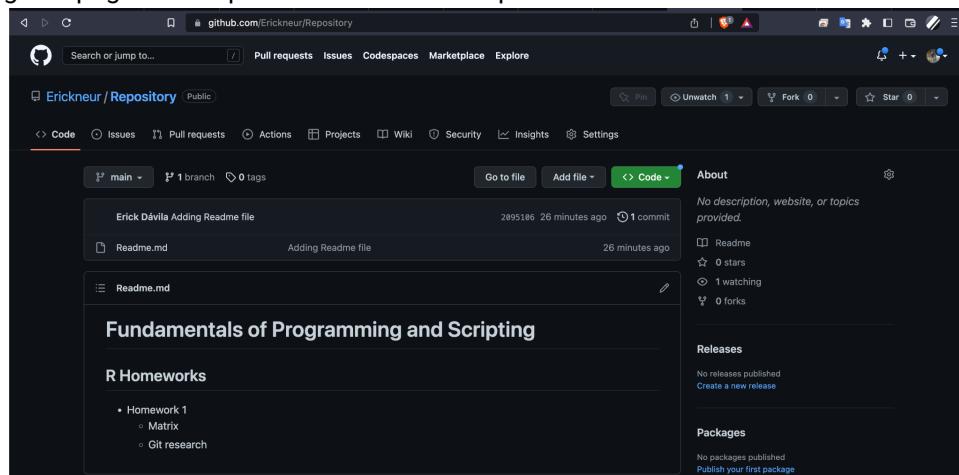
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .DS_Store

erickneur@Ericks-MacBook-Pro Repository % git commit -m "Adding Readme file"
[main (root-commit) 2095106] Adding Readme file
  1 file changed, 8 insertions(+)
   create mode 100644 Readme.md
erickneur@Ericks-MacBook-Pro Repository % git branch -M main
erickneur@Ericks-MacBook-Pro Repository % git branch
* main

erickneur@Ericks-MacBook-Pro Repository % git remote add origin https://github.com/Erickneur/Repository.git
erickneur@Ericks-MacBook-Pro Repository % git remote -v
origin  https://github.com/Erickneur/Repository.git (fetch)
origin  https://github.com/Erickneur/Repository.git (push)
erickneur@Ericks-MacBook-Pro Repository % git push -u origin main
git@github.com: Permission denied (publickey).
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
erickneur@Ericks-MacBook-Pro Repository % git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 322 bytes | 322.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Erickneur/Repository.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
erickneur@Ericks-MacBook-Pro Repository %
```

Al recargar la página del repositorio de GitHub se pueden observar los cambios realizados.



Comandos en GIT

Algunos comandos útiles en Git son:

Iniciar un área de trabajo

- clone: Clona un repositorio Git existente dentro de un directorio.
- Init: Inicializa un nuevo repositorio Git o reinicializa uno existente.

Trabajar sobre cambios existentes

- Add: Agrega archivos al área de preparación para ser confirmados.
- mv: Mueve o renombra un archivo, directorio o enlace simbólico:
- restore: Restaura el árbol de archivos de trabajo.
- rm: Remueve archivos del árbol de trabajo y de la zona de preparación.

Examinar estados y el historial

- bisect: Use la búsqueda binaria para encontrar el commit que introdujo un error.
- diff: Muestra los cambios entre commits existentes y los archivos en el directorio de trabajo.
- grep: Muestra los commits que coinciden con un patrón.
- log: Muestra el historial de cambios realizados en el repositorio.
- show: Muestra varios tipos de objetos.
- status: Muestra el estado actual de los archivos en el repositorio.

Crecer, marcar y ajustar tu historial común

- branch: Crea, lista o elimina ramas.
- commit: Confirma los cambios realizados en los archivos del repositorio.
- merge: Combina los cambios de dos o más ramas en la rama actual.
- rebase: Reaplica los commits de otra rama en la parte superior de la rama base.
- reset: Restablece la rama actual al commit especificado.
- switch: Intercambio entre ramas.
- tag: Crea, lista, elimina o verifica una etiqueta firmada con GPG.

Colaboración

- fetch: Descarga objetos y referencias del repositorio remoto u otro repositorio.
- pull: Actualiza el repositorio local obteniendo e integrando los cambios existentes en el repositorio remoto.
- push: Envía los cambios confirmados al repositorio remoto.

Para mostrar el listado de comandos se utiliza `git help`.



```
[erickneur@Ericks-MacBook-Pro Repository % git help
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  restore    Restore working tree files
  rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect    Use binary search to find the commit that introduced a bug
  diff      Show changes between commits, commit and working tree, etc
  grep      Print lines matching a pattern
  log       Show commit logs
  show      Show various types of objects
  status    Show the working tree status

grow, mark and tweak your common history
  branch   List, create, or delete branches
  commit   Record changes to the repository
  merge    Join two or more development histories together
  rebase   Reapply commits on top of another base tip
  reset   Reset current HEAD to the specified state
  switch  Switch branches
  tag     Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
  fetch   Download objects and refs from another repository
  pull    Fetch from and integrate with another repository or a local branch
  push    Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
```