{desafío}
latam_

Terminal, Git, GitHub y GitHub Pages _

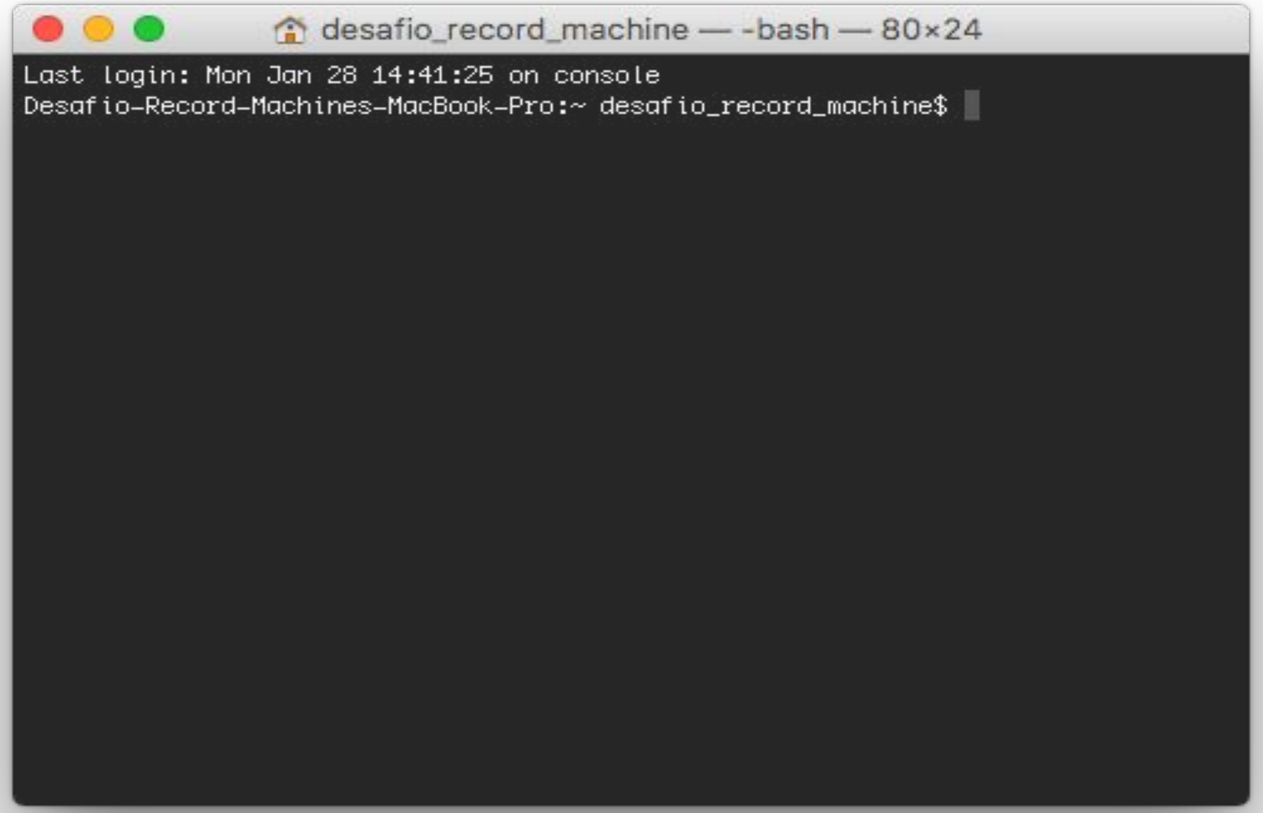
Parte I

Terminal

¿Que haremos en esta unidad?

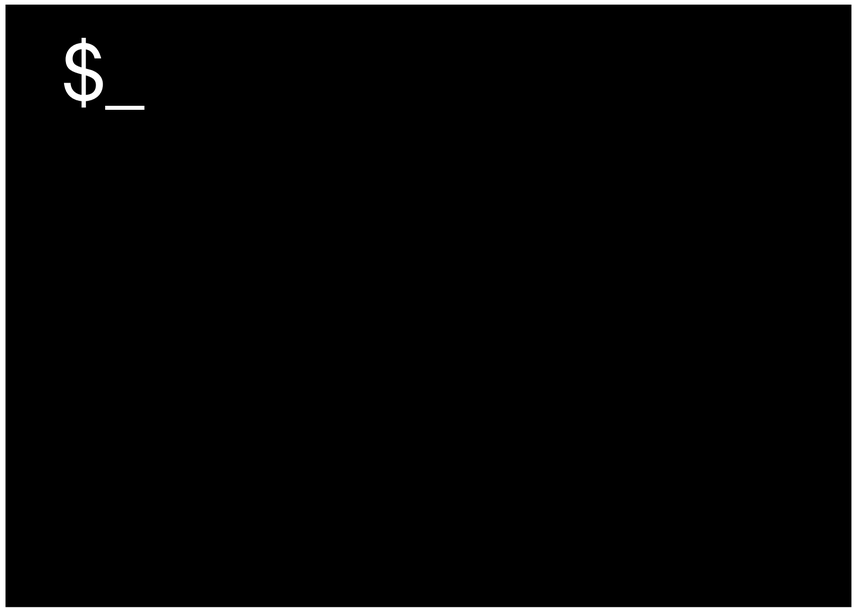
- Conoceremos el terminal
- Aprenderemos a controlar las versiones de nuestro código
- Respaldaremos el código de forma online
- Subiremos nuestro sitio a un dominio

¿Que es Terminal?



{desafío}
latam_

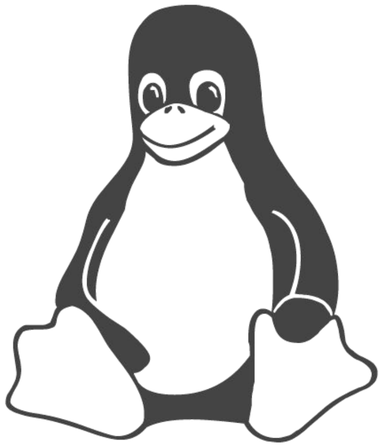
¿Qué pasa con Windows?



1. Pausa el video.
2. Descarga la terminal desde el siguiente link → gitforwindows
3. Luego, instala la terminal en tu computador.
4. Y, ¡listo!

Iniciar Terminal

LINUX

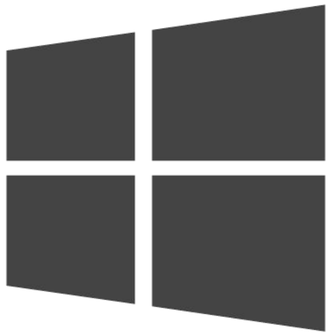


Presiona ctrl + alt + t

MAC

⌘ + espacio, busca por spotlight la "terminal"

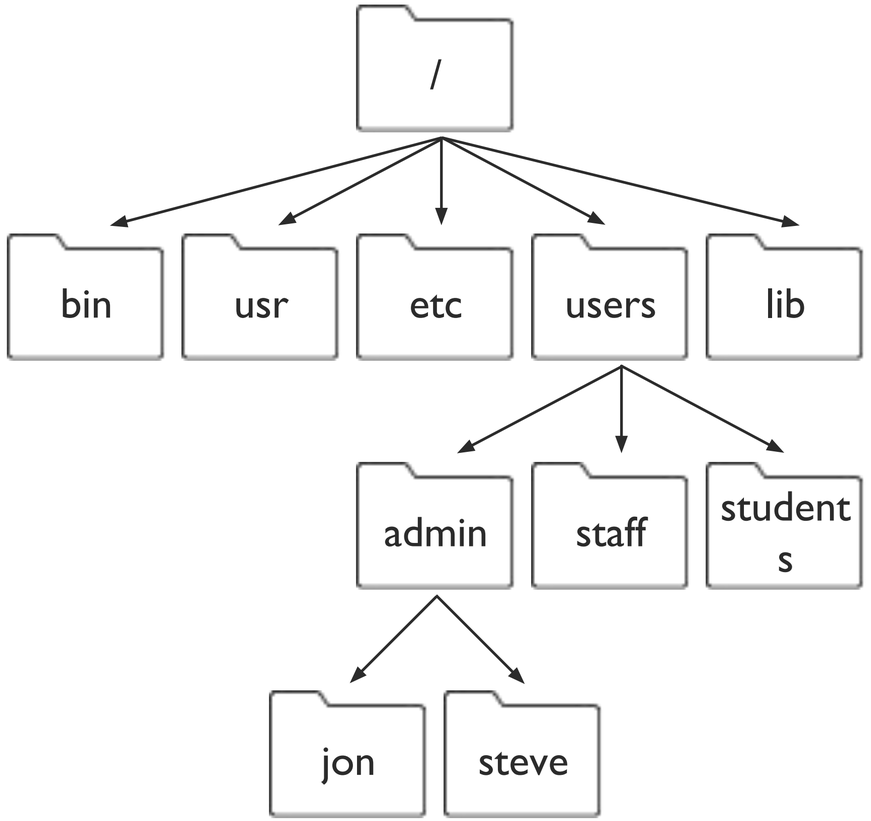
WINDOWS



Busca el programa "git bash" y ábrelo

Comandos

Estructura de Directorios



pwd

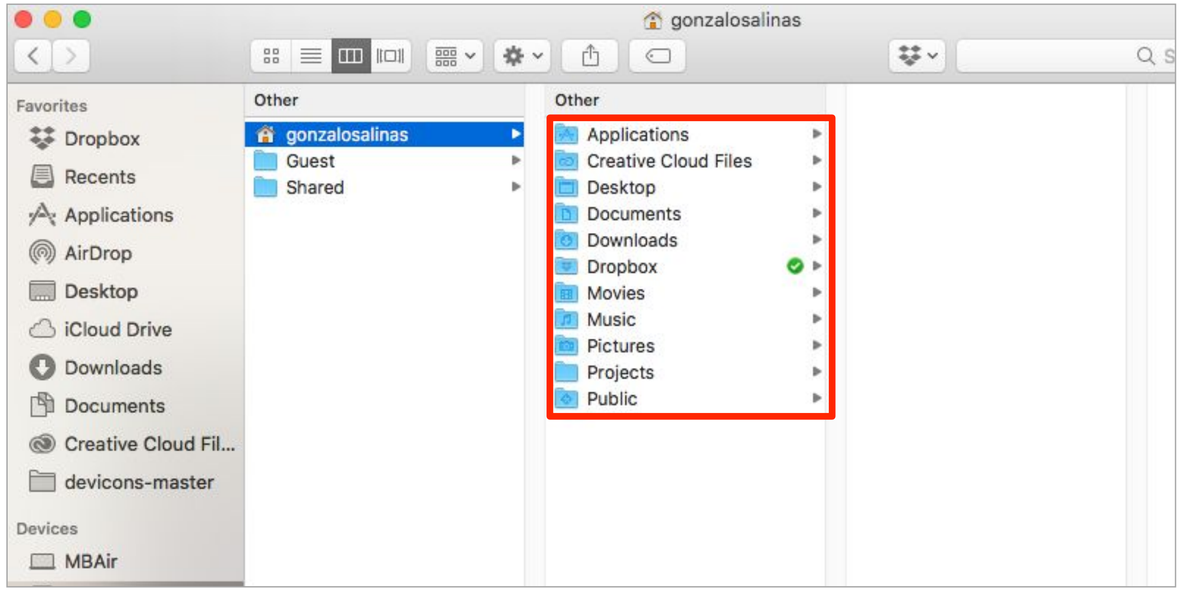
Comando que sirve para saber en qué directorio nos encontramos

```
$ pwd
users/mi_usuario
```

ls

```
$ ls
Applications
Desktop
Download
Movie
Music
Pictures
Public
```

- Muestra una lista de los archivos que hay dentro de un directorio específico.



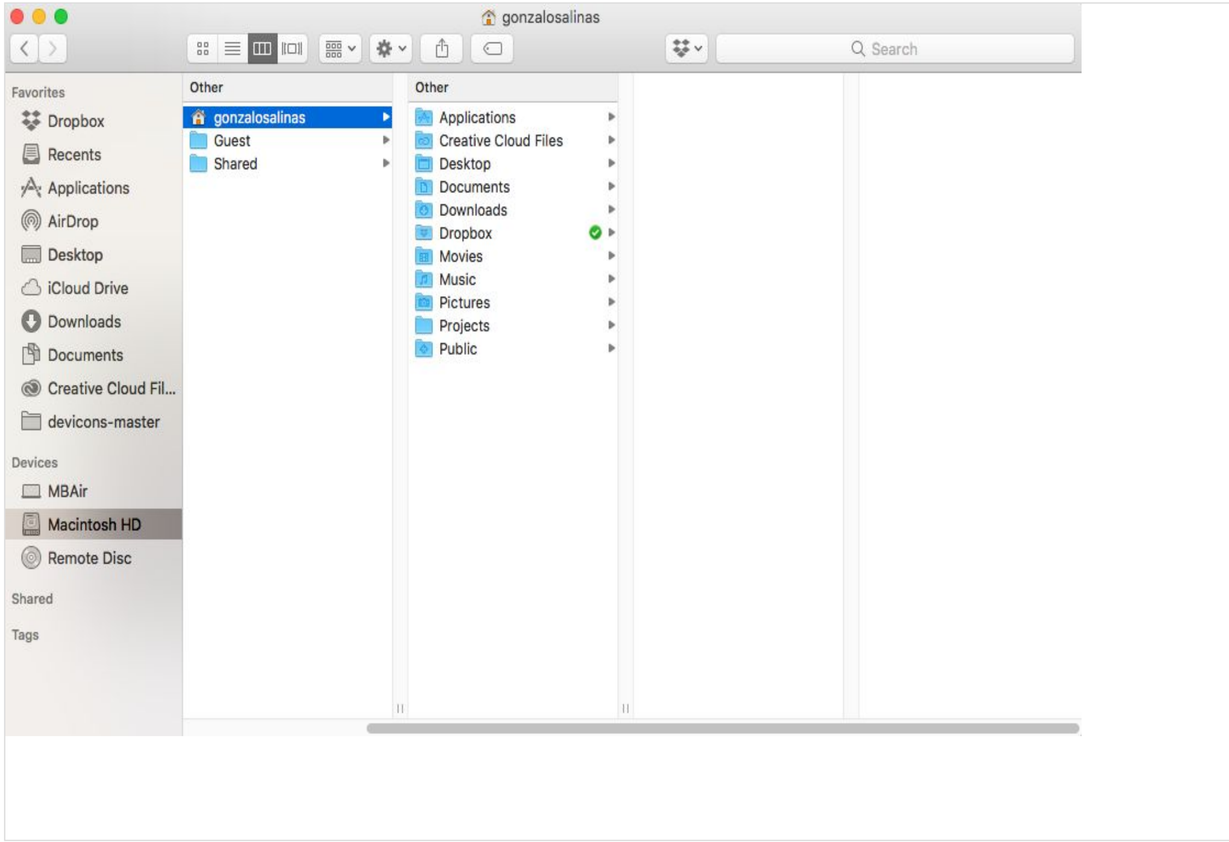
ls -a

- Este comando nos mostrará el listado de todos los archivos, incluyendo los archivos ocultos.
- Los archivos ocultos aparecen siempre con un “ . ” delante de ellos.

cd

```
$ cd
```

- Se utiliza para navegar entre directorios.

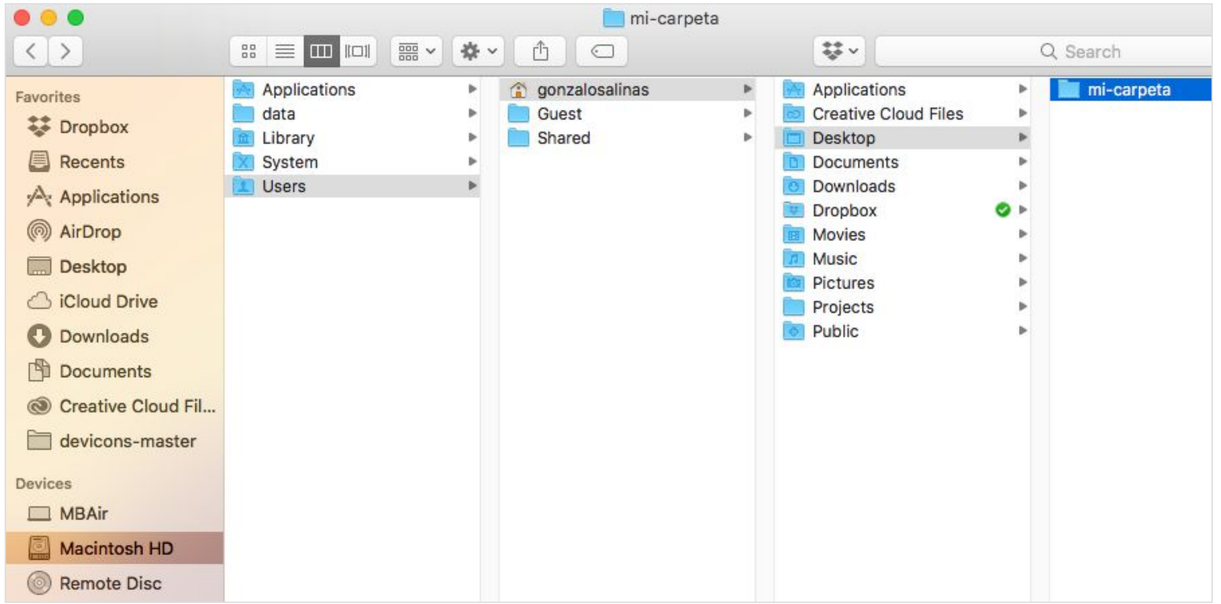


{desafío} latam_

cd [ruta a directorio]

```
$ cd Desktop/mi-carpeta
ruta: /Desktop/mi carpeta
```

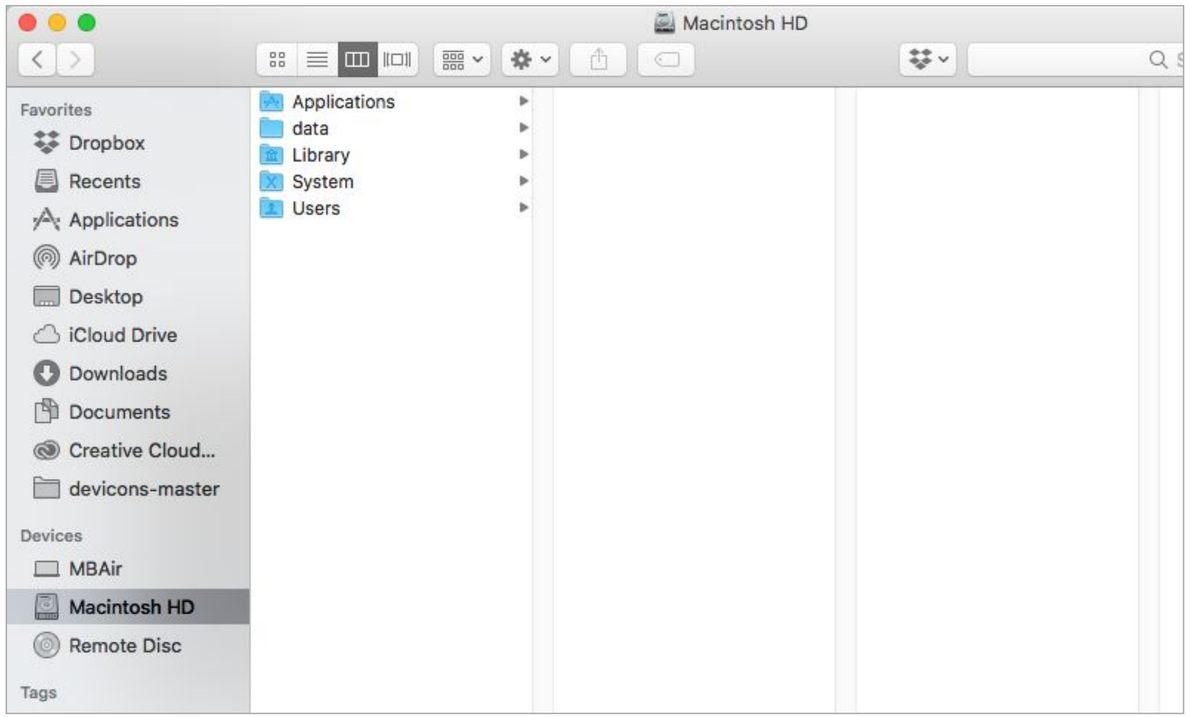
- Este comando nos llevará a una carpeta que especifiquemos.
- Este comando es igual a elegir una carpeta en el navegador de archivos.



cd /

```
$ cd /
ruta: /
```

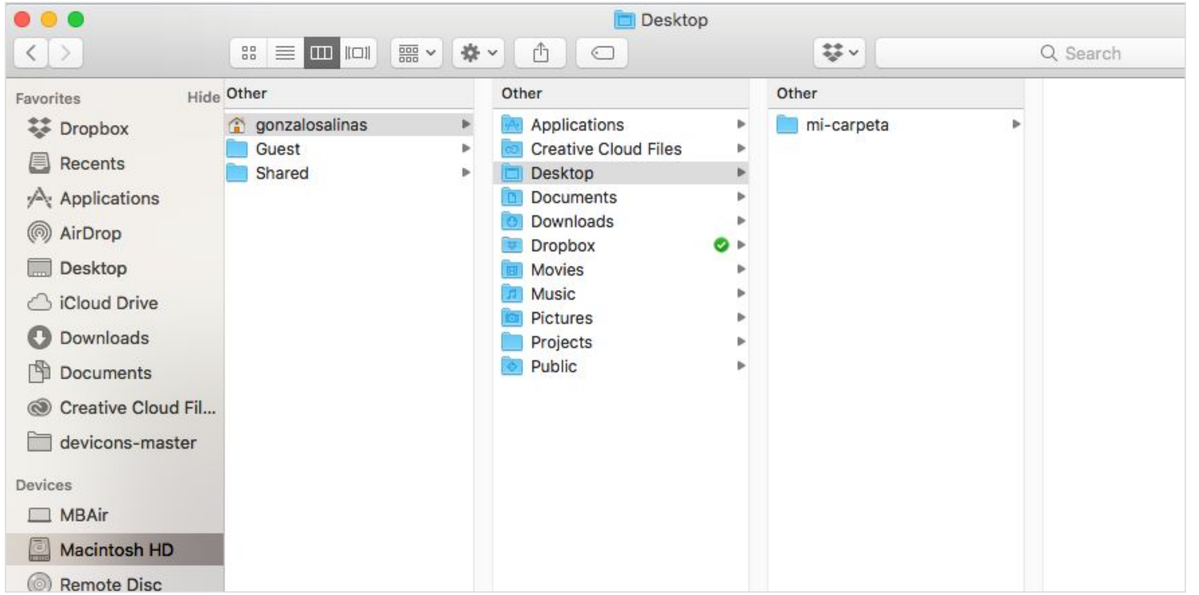
- Este comando nos llevará a la carpeta raíz.



cd ..

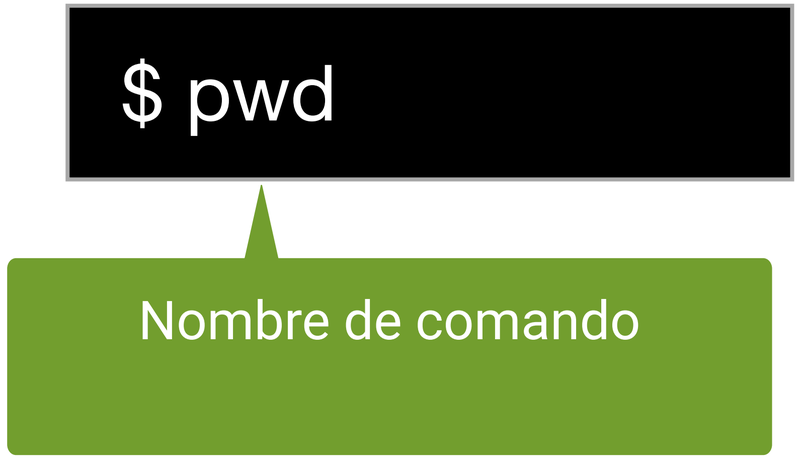
```
$ cd ..
```

- Sirve para ir hacia una carpeta atrás (carpeta contenedora).

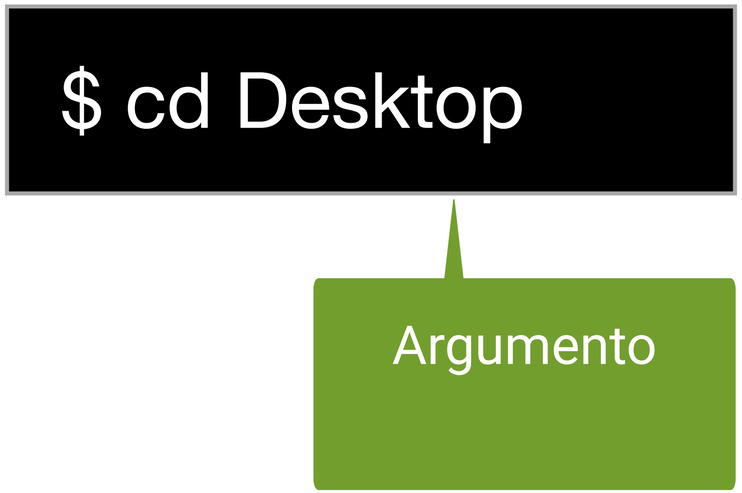


Anatomía de comandos

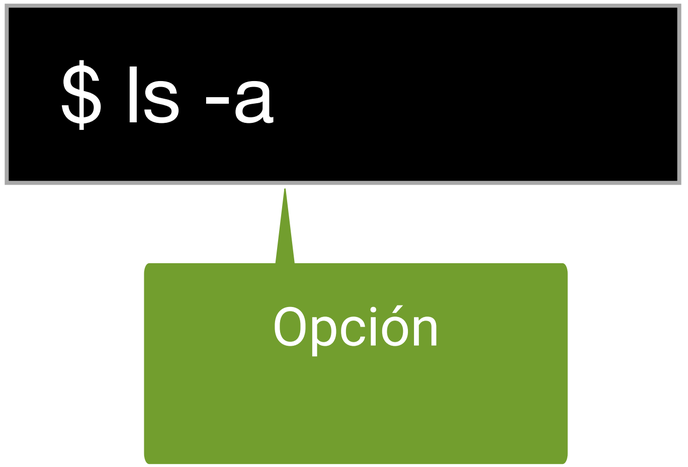
Anatomía de un comando



Anatomía de un comando



Anatomía de un comando



Nuestro sistema operativo puede ser sensible a las mayúsculas

cd es distinto a **CD**, **cD** o **Cd**

cd hola es distinto a **cd Hola**

Administración de archivos y directorios

Dentro del terminal existen varios comandos que sirven para manejar archivos y directorios

mkdir [directorio]

```
$ mkdir carpeta_1
(ruta: users/mi_usuario/Desktop)
```

- Comando que crea un directorio vacío.

touch [archivo]

```
$ touch index.html
(ruta: users/mi_usuario/Desktop)
```

- Comando que crea un archivo vacío.
- El archivo creado debe contener un nombre y la extensión a la cual pertenece.

cp

```
$cp index.html
(ruta: users/mi_usuario/Desktop)
```

- Comando que sirve para copiar archivos y directorios.

mv

Comando que sirve para cambiar el nombre de un archivo
y/o mover un archivo de un directorio a otro

mv [archivo] [nuevo nombre]

Con esta combinación podemos cambiar el nombre de un archivo

```
$ mv index.html  archivo.html
(ruta: users/mi_usuario/Desktop)
```

Para hacerlo debemos indicar el nombre actual del archivo,
con el nuevo nombre que quieras agregar

mv [ruta archivo] [nueva ruta archivo]

```
 $ mv archivo.html  ../Desktop/mi
-carpeta
 (ruta:
users/mi_usuario/Desktop/mi-carpeta)
```

- Con esta combinación podemos mover un archivo de un directorio a otro.
- Para cambiar el archivo debes escribir la ruta del archivo, espacio, y luego la nueva ruta del archivo.

rm [nombre archivo]

Comando que sirve para eliminar un archivo especificado por su nombre

```
$ rm index.html
(ruta: users/mi_usuario/Desktop)
```

rm -r [nombre directorio]

Comando que sirve para eliminar por completo un directorio

```
$ rm index.html
(ruta: users/mi_usuario/Desktop)
```

Recapitulación de comandos

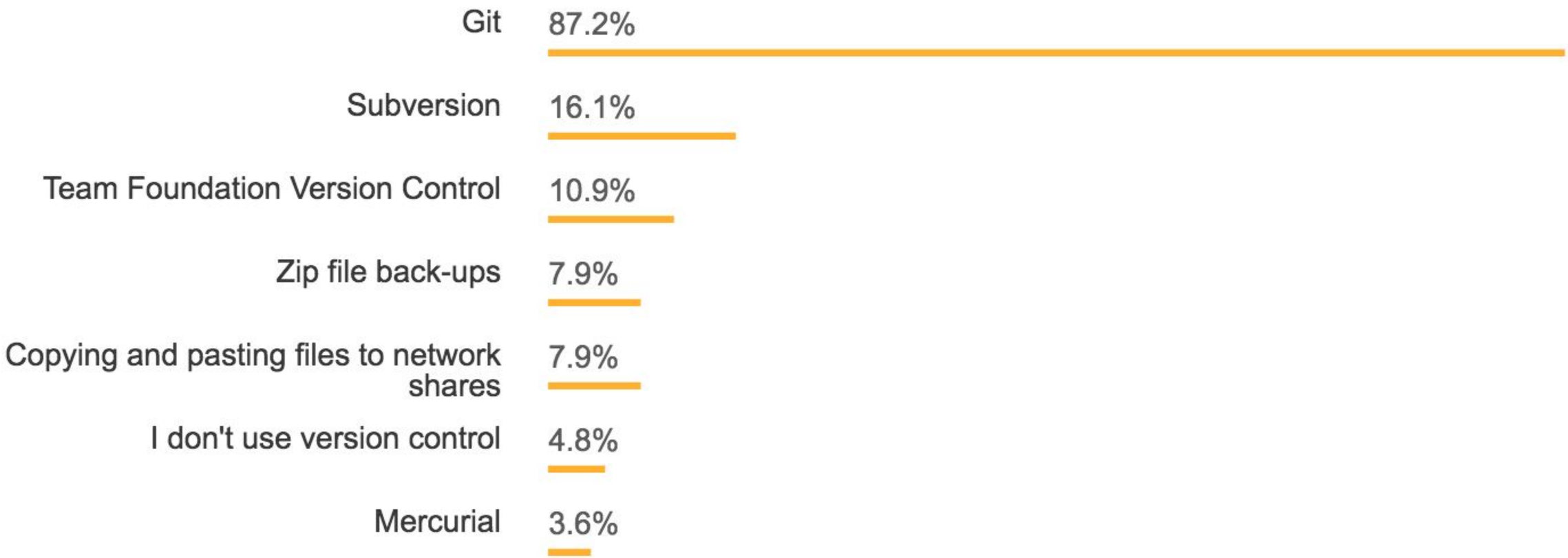
Comando	Explicación
pwd	Muestra en que directorio te encuentras
cd	Cambia de directorio a uno especificado
cd ..	Permite ir a un directorio anterior al actual
cd /	Lleva a la raíz
cd ~	Lleva al directorio $home
ls	Muestra todos los archivos y directorios de la carpeta actual
ls -a	Muestra los archivos, los archivos ocultos y los directorios de la carpeta actual
touch	Crea un archivo
mv	Renombra y mueve un archivo
rm	Borra un archivo
mkdir	Crea un directorio
rmdir	Elimina un directorio vacío
rm -r	Elimina todos los archivos de un directorio, incluído el directorio

Git

¿Qué es Git?

Git es un software de control de versiones open source, diseñado para mantener archivos de manera eficaz y confiable.

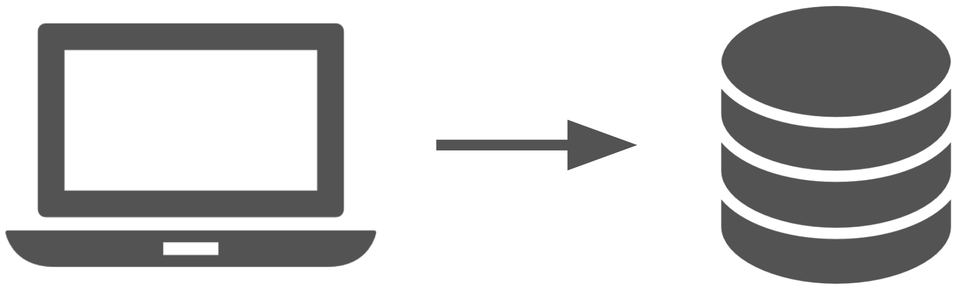

Google
facebook
Microsoft
twitter
LinkedIn
NETFLIX
PostgreSQL
ANDROID
RAILS
Qt
GNOME
eclipse


Git
87.2%
Subversion
16.1%
Team Foundation Version Control
10.9%
Zip file back-ups
7.9%
Copying and pasting files to network shares
7.9%
I don't use version control
4.8%
Mercurial
3.6%

Git nos permite:

- Recuperar versiones anteriores de nuestro código
- Recuperar archivos borrados
- Ayudar a gestionar cambios realizados por otras personas
- Administrar un proyecto donde trabajan múltiples desarrolladores

¿Qué es “control de versiones”?



Es un sistema que registra los cambios realizados en un archivo o conjunto de archivos a lo largo del tiempo, de modo que puedas recuperar versiones específicas más adelante.

¿Cuándo utilizar Git?

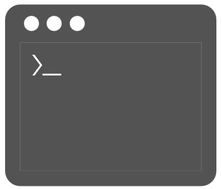
Se recomienda utilizar **siempre**.

Formas de utilizar Git

GIT

Editores de texto

Programas de versionamiento



Terminal

Instalar Git

Git en Mac

1. Ingresar al terminal y escribir **git --version** para verificar si está instalado.
2. Si no esta instalado, descarga la última versión desde el siguiente link.
3. Seguir las instrucciones de instalación.
4. Volver al terminal y escribir nuevamente **git - -version** para verificar que todo resulto bien.

Git en Linux

```
$ sudo apt-get update
$ sudo apt-get install git
```

```
$ git  --version
git version 2.14.3
```

1. Ingresar al terminal y escribe **sudo apt-get update**.
2. Luego escribe **sudo apt-get install git**.
3. Verificar que la instalación resultó bien escribiendo **git - -version**.

Git en Windows

Anteriormente descargamos *Git for Windows*
e instalamos el terminal en nuestro computador.

Configurar Git

```
$ git config --global user.name “Tu nombre”
```

- Escribir en el terminal el comando **git config --global user.name**, seguido de ***tu nombre*** dentro de comillas dobles.

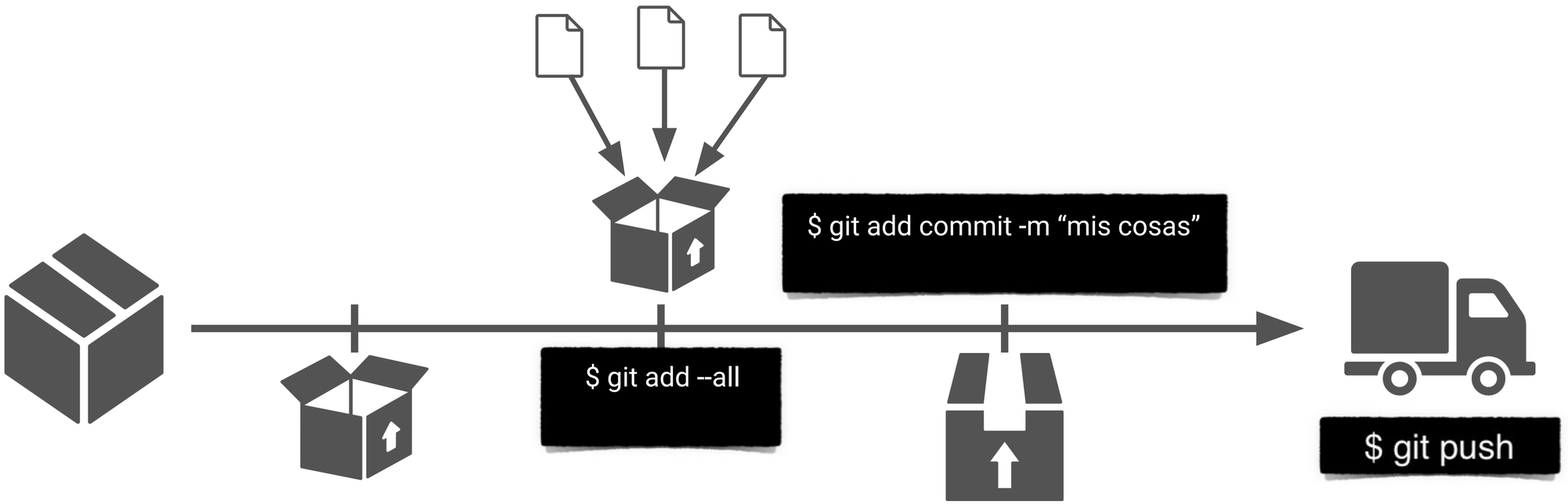
```
$ git config - -list
user.name= tu nombre
user.mail= tucorreo@mail.com
```

- Revisar si la configuración está correcta usando el comando **git config - -list**.

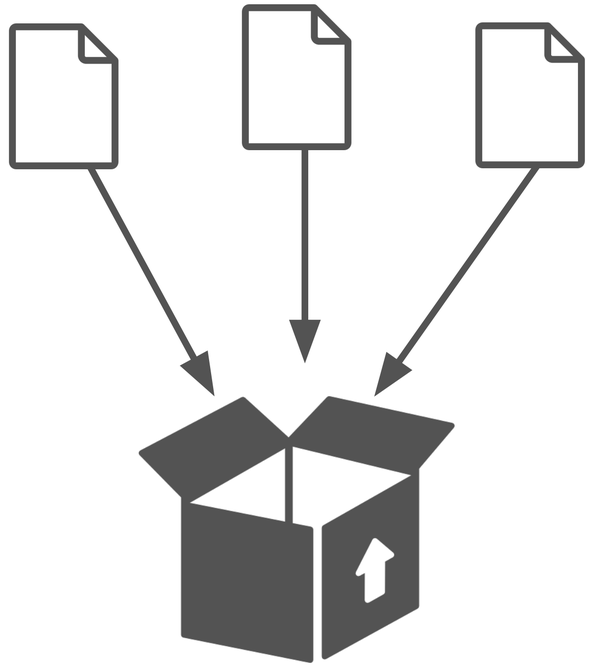
Utilizando Git

```
$ git init
```

Flujo de trabajo de Git



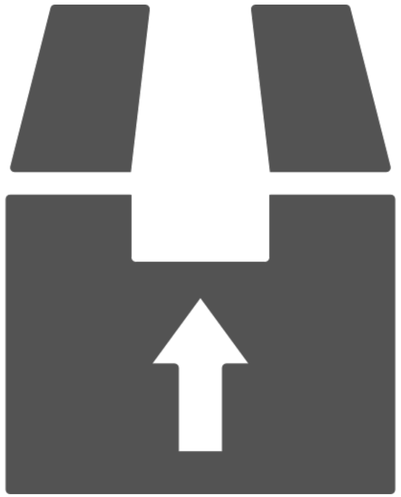
Agregar cosas a la caja



```
$ git add --all
```

- Ahora que está abierta, es momento de agregar todas nuestras cosas a la caja.
- Esta acción es igual a escribir **git add --all** o uno a uno con **git add [nombre_de_archivo]** .

Cerrar y etiquetar la caja



```
$ git commit -m “mis cosas”
```

- Luego de agregar todas nuestras cosas, habrá que etiquetar y enlistar las cosas de nuestra caja.
- Esta acción es igual a escribir **git commit -m “mis cosas”**.

Enviar la caja

```
$ git push origin master
```

- A continuación debemos enviar la caja a destino.
- Esta acción es igual a escribir **git push origin master**.

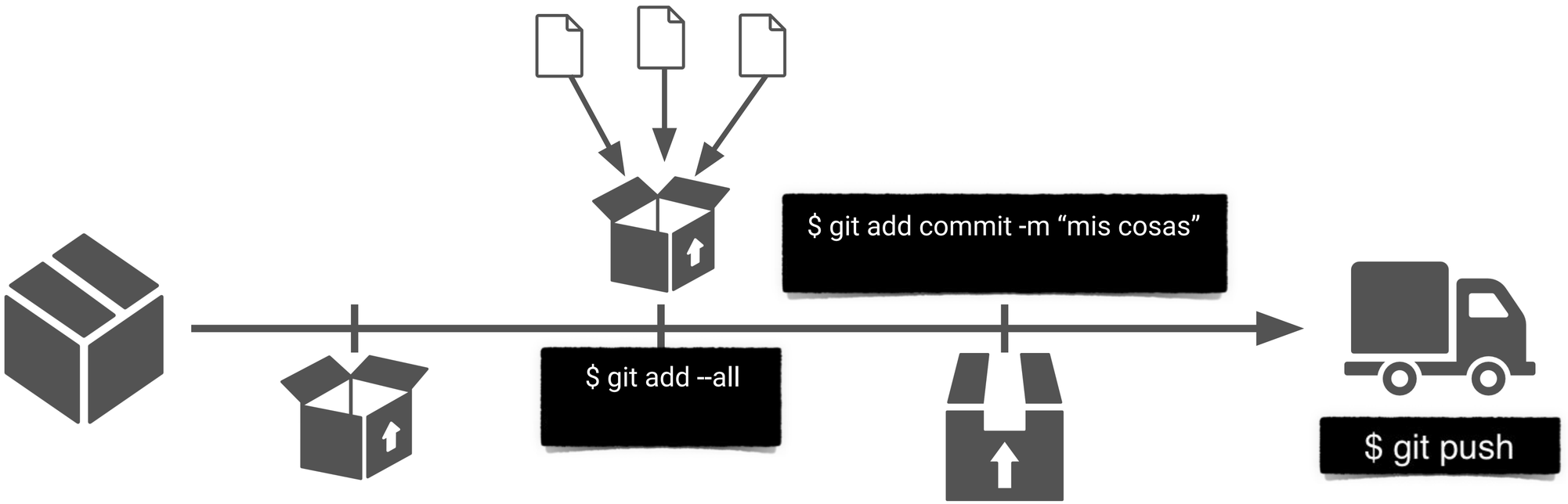
Revisar la caja

```
$ git status
On branch master
nothing to commit, working tree clean/
mi-usuario/Desktop/mi-carpeta)
```

- **git status** nos ayudará a saber en qué parte del flujo de trabajo nos encontramos.
- Es es similar a revisar el proceso dentro de una lista.

¿Local o remoto?

Flujo de trabajo de Git



{desafío} latam_

Academia de talentos digitales

www.desafiolatam.com