

### CONFIGURAR HERRAMIENTAS:

Inicia un nuevo repositorio u obtiene uno de una URL existente

```
$ git config --global user.name "[name]"
```

Establece el nombre que desea esté anexado a sus transacciones de commit

```
$ git config --global user.email "[email address]"
```

Establece el e-mail que desea esté anexado a sus transacciones de commit

```
$ git config --global color.ui auto
```

Habilita la útil colorización del producto de la línea de comando

```
$ git config --list
```

Va mostrarnos una lista de todas las variables que esten config y que git pueda encontrar hasta el punto



### CREAR REPOSITORIOS:

Inicia un nuevo repositorio u obtiene uno de una URL existente

```
$ git init [project-name]
```

Crea un nuevo repositorio local con el nombre especificado

```
$ git clone [url]
```

Descarga un proyecto y toda su historia de versión



### EFECTUAR CAMBIOS:

Revisa las ediciones y elabora una transacción de commit

```
$ git status
```

Enumera todos los archivos nuevos o modificados que se deben confirmar

```
$ git diff
```

Muestra las diferencias de archivos que no se han enviado aún al área de espera

```
$ git add [file]
```

Toma una instantánea del archivo para preparar la versión

```
$ git diff --staged
```

Muestra las diferencias del archivo entre el área de espera y la última versión del archivo

```
$ git reset [file]
```

Mueve el archivo del área de espera, pero preserva su contenido

```
$ git commit -m "[descriptive message]"
```



### CAMBIOS GRUPALES:

Nombra una serie de commits y combina esfuerzos ya culminados

```
$ git branch
```

Enumera todas las ramas en el repositorio actual

```
$ git branch [branch-name]
```

Crea una nueva rama

```
$ git checkout [branch-name]
```

Cambia a la rama especificada y actualiza el directorio activo

```
$ git merge [branch]
```

Combina el historial de la rama especificada con la rama actual

```
$ git branch -d [branch-name]
```

Borra la rama especificada



### NOMBRES DEL ARCHIVO DE REFACTORIZACIÓN:

Reubica y retira los archivos con versión

```
$ git rm [file]
```

Borra el archivo del directorio activo y pone en el área de espera el archivo borrado

```
$ git rm --cached [file]
```

Retira el archivo del control de versiones, pero preserva el archivo a nivel local

```
$ git mv [file-original] [file-renamed]
```

Cambia el nombre del archivo y lo prepara para commit



### SUPRIMIR TRACKING

Excluye los archivos temporales y las rutas

*.log	*.log	*.log	*.log
build/	build/	build/	build/
temp-*	temp-*	temp-*	temp-*

Un archivo de texto llamado .gitignore suprime la creación accidental de versiones de archivos y rutas que concuerdan con los patrones especificados

```
$ git ls-files --other --ignored --exclude-standard
```

Enumera todos los archivos ignorados en este proyecto



## GUARDAR FRAGMENTOS

Almacena y restaura cambios incompletos

```
$ git stash
```

Almacena temporalmente todos los archivos  
tracked modificados

```
$ git stash pop
```

Restaura los archivos guardados más recientemente

```
$ git stash list
```

Enumera todos los sets de cambios en guardado  
rápido

```
$ git stash drop
```

Elimina el set de cambios en guardado rápido más  
reciente



### REPASAR HISTORIAL

Navega e inspecciona la evolución de los archivos de proyecto

```
$ git log
```

Enumera el historial de la versión para la rama actual

```
$ git log --follow [file]
```

Enumera el historial de versión para el archivo, incluidos los cambios de nombre

```
$ git diff [first-branch]...[second-branch]
```

Muestra las diferencias de contenido entre dos ramas

```
$ git show [commit]
```

Produce metadatos y cambios de contenido del commit especificado





## REHACER COMMITS

Borra errores y elabora historial de reemplazo

```
$ git reset [commit]
```

Deshace todos los commits después de [commit],  
preservando los cambios localmente

```
$ git reset --hard [commit]
```

Desecha todo el historial y regresa al commit  
especificado



## SINCRONIZAR CAMBIOS

Registrar un marcador de repositorio e intercambiar historial de versión

```
$ git fetch [bookmark]
```

Descarga todo el historial del marcador del repositorio

```
$ git merge [bookmark]/[branch]
```

Combina la rama del marcador con la rama local actual

```
$ git merge --abort
```

Para revertir un merge

```
$ git push [alias] [branch]
```

Carga todos los commits de la rama local al GitHub

```
$ git pull
```

Descarga el historial del marcador e incorpora cambios



Mostrar	
\$ ls	Lista archivos en el directorio o carpeta
\$ ls -a	Lista todos los archivos, incluyendo los archivos ocultos
\$ ls -l	Muestra toda la información de una carpeta: usuario, grupo, permisos, tamaño, fecha y hora de creación.
\$ ls -R	Muestra las carpetas y los archivos contenidos en ellos de manera recursiva
\$ pwd	Muestra la carpeta en la que se está trabajando actualmente
more [Nombre del archivo]	Muestra el contenido de un archivo



## Actividad mochila

## Comandos de GitHub Sthephania Martheyn

Mostrar	
\$ history	Muestra historia de los comandos que hemos usado
\$ cat	Muestra el contenido de un archivo y lo muestra en la terminal
\$ code	Se abre la visual de code



## Actividad mochila

## Comandos de GitHub Sthephania Martheyn

### Crear

\$ mkdir  
[Carpeta]

Crea una nueva  
directorio o carpeta

\$ touch [Nombre  
del archivo]

Crea un nuevo  
archivo



### Eliminar

\$ rm [Nombre  
del archivo]

Elimina un archivo

\$ rmdir [Nombre  
de la carpeta]

Elimina una carpeta  
vacía

\$ rm -r [Nombre  
de la carpeta]

Elimina una carpeta  
y su contenido



### Copiar/Mover/Renombrar

```
$ mv  
[ruta/archivo1]  
[ruta/archivo2]
```

Renombra archivos  
(archivo2 no debe existir  
o será sobrescrito)

```
$ mv  
[ruta/carpeta1]  
[ruta/carpeta2]
```

Renombra la carpeta1  
como carpeta2 (carpeta  
2 no debe existir)

```
$ mv  
[ruta/carpeta1]  
[ruta/carpeta2]
```

Mueve contenido de  
carpeta1 a carpeta2 (carpeta  
2 debe existir)

```
$ cp  
[ruta/archivo1]  
[ruta/archivo2]  
$ cp  
[ruta/archivo1]  
[ruta/archivo2]
```

Copia un archivo o  
carpeta

opción: -r

Indica que copie  
recursivamente el contenido  
de las subcarpetas



### Navegación entre carpetas

\$ cd

Nos permite navegar entre carpetas.

\$ cd/

ir a la ruta principal

\$ cd ~

ir a la ruta de tu usuario

\$ cd ..

me permite regresar hacia atras

\$ cd carpeta/subcarpeta

Navegar a una ruta dentro de la carpeta que nos encontramos.





### Navegación entre carpetas

**\$ clear**

Limpia la pantalla de la terminal

**\$ comando  
--help**

Muestra ayuda del comando



### Atajos de teclado

\$ ctrl + c

Finaliza un proceso  
vigente que está  
corriendo en la terminal

\$ ctrl + l

Limpia la pantalla de  
la termina



### Caracteres especiales

**“”** (comillas)

Nos permiten utilizar términos que consistan en más de una palabra

**.** (el punto)

Permite hacer referencia al directorio donde estamos ubicados actualmente

