

Práctica 04

Contenido

1	Creando alias para nuestros comandos.....	2
2	Cambios en los archivos.....	2
2.1	Comparar cambios en archivos con visual studio code	4
3	Actualizar mensajes de commits y deshacer commits	4
3.1	Actualizar el mensaje de un commit.....	4
3.2	Borrar un commit	4
4	Preparando un repositorio para viajes en el tiempo	5

1 Creando alias para nuestros comandos

Para mostrar el estado de los archivos del repositorio con una descripción corta podemos usar el siguiente comando:

(Pega aquí una captura de la ejecución del comando)

Podemos crear un alias para este comando de forma que podamos ejecutarlo escribiendo menos texto:

(Pega aquí una captura de la ejecución del comando)

Como vemos para crear un alias usamos la configuración global y ponemos alias. y después del punto ponemos el texto que servirá de alias. A continuación escribimos entre comillas el comando que queremos abreviar sin poner git delante.

2 Cambios en los archivos

Creamos un nuevo repositorio que se llame 03-instalaciones y lo abrimos en visual studio.

Hacemos el **git init** para inicializar el repositorio y creamos un archivo que se llame instalaciones.md con el siguiente contenido:

Lo agregamos al stage:

Y hacemos commit:

Deberíamos ver algo así en nuestro visual studio code:

Modificamos el archivo instalaciones y lo dejamos así:

(Hemos modificado una línea y añadido 3 líneas vacías debajo de la segunda línea con ...)

Guardamos los cambios, pero no hacemos add ni commit.

Vamos a comparar las modificaciones realizadas en este archivo, para ellos usamos el siguiente comando:

(Pega aquí una captura de la ejecución del comando)

La versión a del archivo instalaciones nos dice que tiene menos cosas que la versión b del mismo archivo

```
PS D:\Git\03-instalaciones> git diff
diff --git a/instalaciones.md b/instalaciones.md
index 4920948..d7266ac 100644
--- a/instalaciones.md
+++ b/instalaciones.md
@@ -3,5 +3,7 @@ Seguir estos pasos:

...

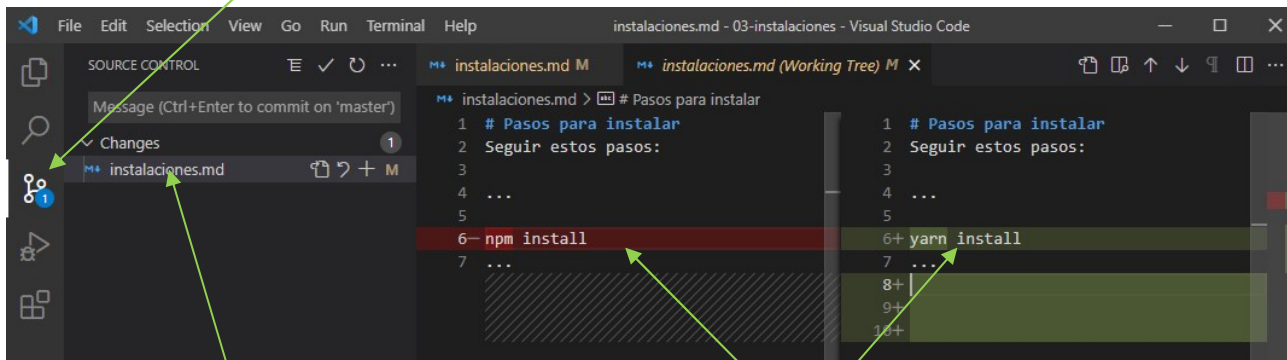
-npm install
+...
\ No newline at end of file
+yarn install
+.
```

Las líneas en color rojo y precedidas por un - nos indican cosas que la primera versión no tiene respecto a la segunda y las cosas en verde precedidas por un + nos indican cosas que la segunda versión tiene y la primera no.

El comando `git diff` nos compara las modificaciones de los archivos que no están en el escenario. Si queremos usar el comando con un archivo que hemos añadido al escenario usaremos lo siguiente:

2.1 Comparar cambios en archivos con visual studio code

Desde la opción de source control de visual studio podemos comparar las modificaciones realizadas en un archivo de formas más visual:



Hacemos click en el archivo del cual queremos ver las diferencias

3 Actualizar mensajes de commits y deshacer commits

3.1 Actualizar el mensaje de un commit

Para cambiar el mensaje del último commit realizado usaremos el siguiente comando:

(Pega aquí una captura de la ejecución del comando)

3.2 Borrar un commit

Modificamos el archivo instalaciones.md (escribimos cualquier cosa) y hacemos commit:

Para borrar el último commit realizado usaremos el siguiente comando:

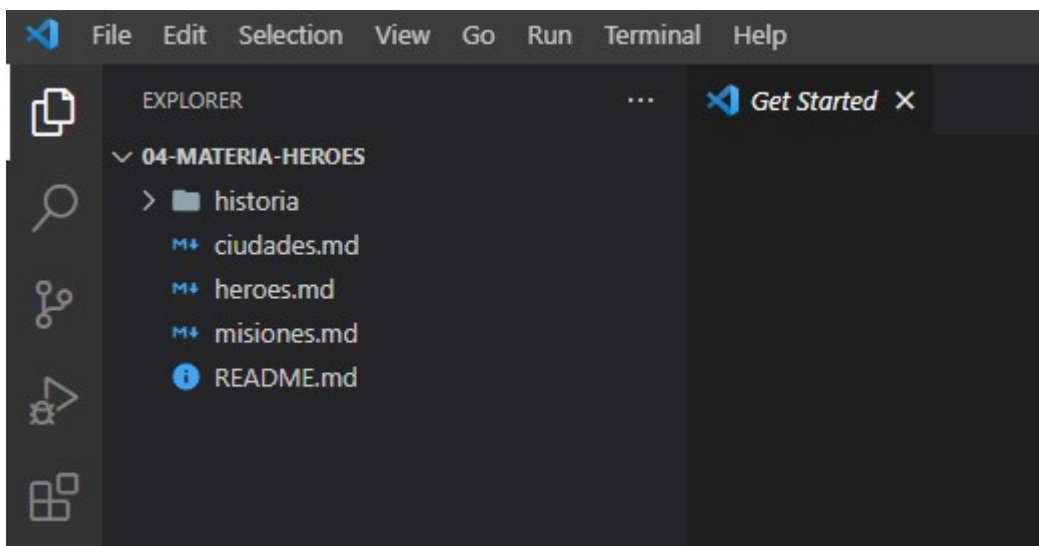
(Pega aquí una captura de la ejecución del comando)

Esto eliminará el commit y nos dejará el repositorio como estaba antes de hacer dicho commit. Es decir si teníamos archivos modificados volverán a aparecer como modificados. El mantener los cambios se debe al parámetro --soft.

4 Preparando un repositorio para viajes en el tiempo

Descomprimos el archivo Materia-Heroes y colocamos la carpeta Materia-Heroes en la carpeta donde estemos creando nuestros repositorios.

Renombramos la carpeta como: 04-Materia-Heroes y arrastramos la carpeta a visual estudio code. Deberíamos tener algo así:



Vamos a ir ejecutando los siguientes comandos:

Inicializamos el repositorio:

Añadimos el archivo README.md

Hacemos el commit de este archivo:

Añadimos el archivo misiones.md

Hacemos el commit de este archivo:

Añadimos el archivo heroes.md

Hacemos el commit de este archivo:

Añadimos el archivo ciudades.md

Hacemos el commit de este archivo:

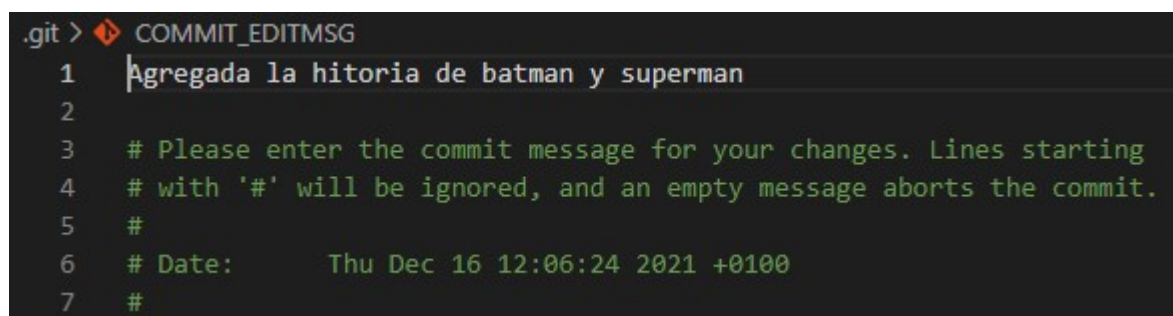
Agregamos la carpeta historia

Hacemos el commit de los archivos agregados:

Cambiamos el texto del último commit:

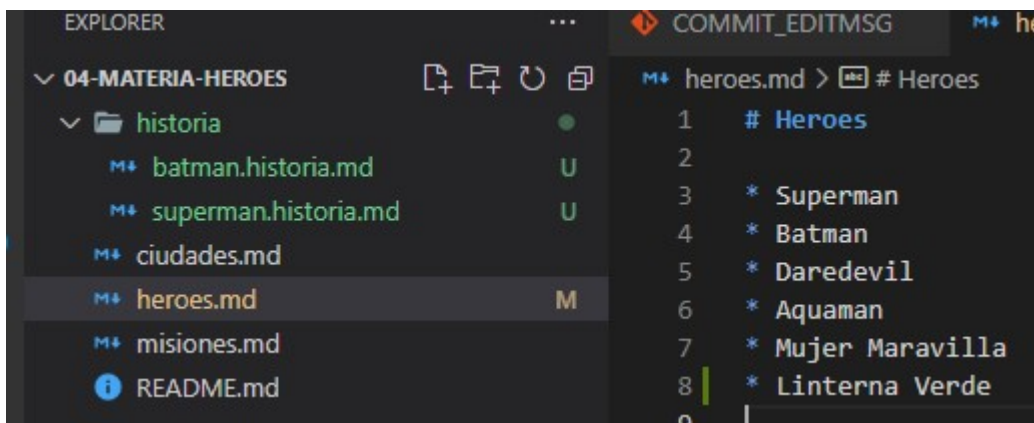
(Pega aquí una captura de la ejecución del comando)

Al ejecutar el comando sin poner al final `-m` “nuevo texto para el commit” se abrirá el editor y podremos cambiar en él el texto del commit:



```
.git > COMMIT_EDITMSG
1  Agregada la hitoria de batman y superman
2
3  # Please enter the commit message for your changes. Lines starting
4  # with '#' will be ignored, and an empty message aborts the commit.
5  #
6  # Date:      Thu Dec 16 12:06:24 2021 +0100
7  #
```

Editamos el archivo héroes.md y agregamos al héroeLinterna Verde:



Hacemos un nuevo commit:

Ejecutamos el comando `git lg` para ver un listado de los commits que hemos realizado:

```
PS D:\Git\04-Materia-Heroes> git lg
* ac3cf1d - (2 minutes ago) Agregamos el héroe Linterna Verde - Alvaro (HEAD -> master)
* 80de664 - (23 minutes ago) Agregada la hitoria de batman y superman - Alvaro
* 9dff538 - (24 minutes ago) heroes.md agregado - Alvaro
* 8bc807e - (25 minutes ago) Misiones agregado - Alvaro
* 421469d - (27 minutes ago) README agregado - Alvaro
```

5 Viajes en el tiempo, resets y reflogs

Vamos a añadir el héroe Robin en el archivo heroes.md:

En vez de hacer un nuevo commit para añadir este cambio queremos que haya un solo commit con la agragación de Robin y Linterna Verde.

Hacemos un `git lg`:

```
PS D:\Git\04-Materia-Heroes> git lg
* ac3cf1d - (2 minutes ago) Agregamos el héroe Linterna Verde - Alvaro (HEAD -> master)
* 80de664 - (23 minutes ago) Agregada la hitoria de batman y superman - Alvaro
* 9dff538 - (24 minutes ago) heroes.md agregado - Alvaro
* 8bc807e - (25 minutes ago) Misiones agregado - Alvaro
* 421469d - (27 minutes ago) README agregado - Alvaro
```

Vamos a regresar a un commit anterior usando como referencia el código hash del commit al que queremos volver:

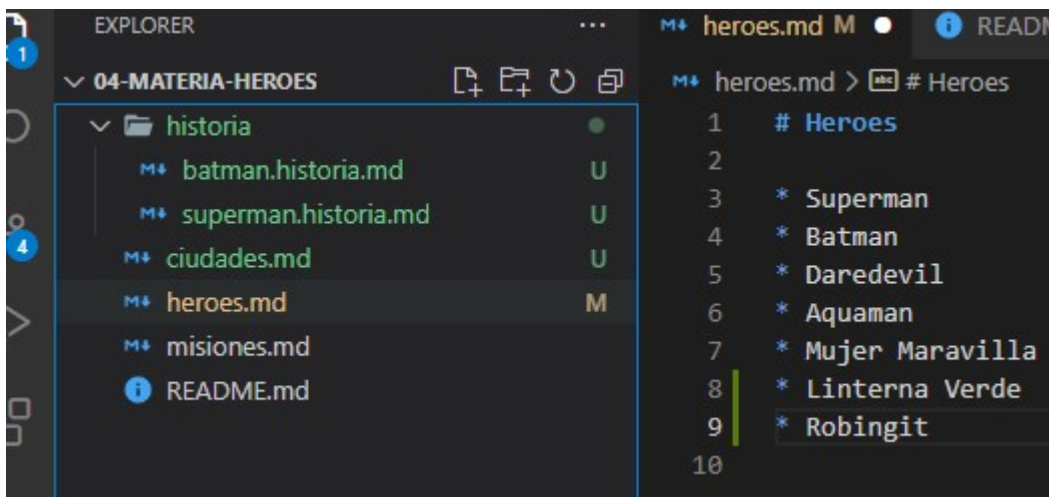
(Pega aquí una captura de la ejecución del comando)

Hacemos un nuevo commit:

Ahora decidimos que los últimos cambios realizados no están bien y que queremos volver al punto en el que agregamos el archivo de heroes. Y además queremos que los archivos de los que se hubiera hecho commit se queden fuera del stage si es que los habíamos subido. Para ello hacemos un git reset --mixed:

(Pega aquí una captura de la ejecución del comando)

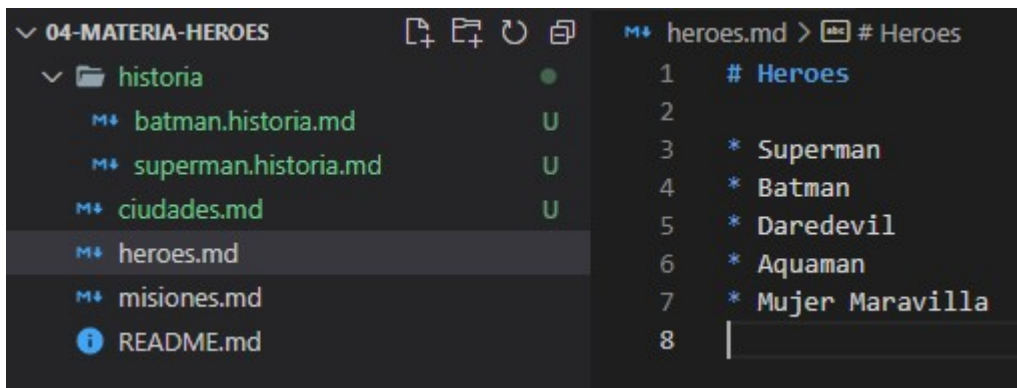
Si no ponemos una opción del modo de reset (--soft --mixed etc) el tipo de reset que se hace es el --mixed ya que es el modo por defecto.



Vemos que los archivos han pasado a estar con la U de untracked es decir sin seguimiento pero que los últimos cambios realizados en heroes.md se han mantenido.

Si queremos hacer un reset que borre los cambios usaremos la opción --hard:

(Pega aquí una captura de la ejecución del comando)



The image shows a file explorer on the left with the following structure:

- 04-MATERIA-HEROES
 - historia
 - batman.historia.md (U)
 - superman.historia.md (U)
 - ciudades.md (U)
 - heroes.md (selected)
 - misiones.md
 - README.md (i)

On the right, the content of `heroes.md` is displayed:

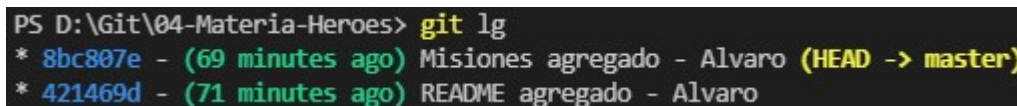
```
# Heroes
* Superman
* Batman
* Daredevil
* Aquaman
* Mujer Maravilla
```

Vemos que el archivo `heroes.md` ha vuelto al estado del commit al que hemos regresado.

Hacemos un reset hard al commit donde agregamos las misiones:

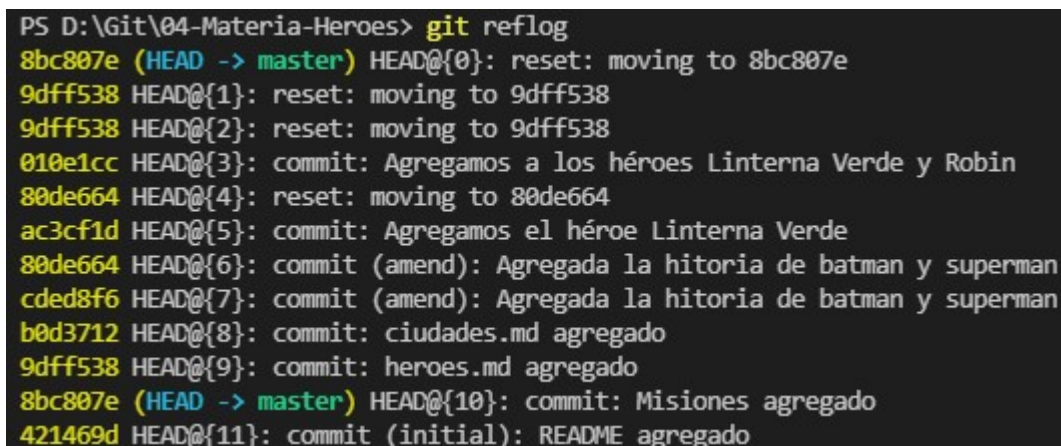
(Pega aquí una captura de la ejecución del comando)

Si hacemos un nuevo `git lg`:



```
PS D:\Git\04-Materia-Heroes> git lg
* 8bc807e - (69 minutes ago) Misiones agregado - Alvaro (HEAD -> master)
* 421469d - (71 minutes ago) README agregado - Alvaro
```

Vemos que sólo nos quedan dos commits. Si ahora nos damos cuenta de que estaba todo bien y que queremos volver a un commit de los que borramos, todavía es posible ya que git guarda un historial:

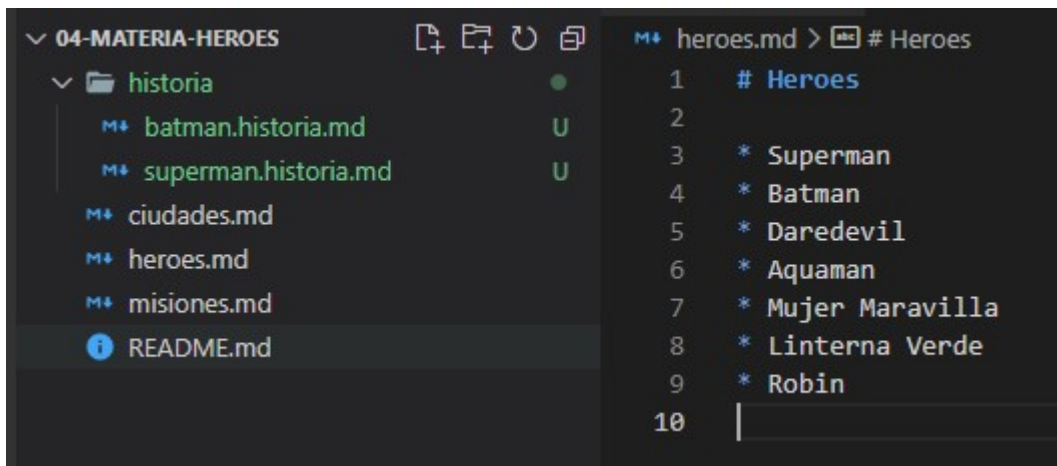


```
PS D:\Git\04-Materia-Heroes> git reflog
8bc807e (HEAD -> master) HEAD@{0}: reset: moving to 8bc807e
9dff538 HEAD@{1}: reset: moving to 9dff538
9dff538 HEAD@{2}: reset: moving to 9dff538
010e1cc HEAD@{3}: commit: Agregamos a los héroesLinterna Verde y Robin
80de664 HEAD@{4}: reset: moving to 80de664
ac3cf1d HEAD@{5}: commit: Agregamos el héroeLinterna Verde
80de664 HEAD@{6}: commit (amend): Agregada la hitoria de batman y superman
cded8f6 HEAD@{7}: commit (amend): Agregada la hitoria de batman y superman
b0d3712 HEAD@{8}: commit: ciudades.md agregado
9dff538 HEAD@{9}: commit: heroes.md agregado
8bc807e (HEAD -> master) HEAD@{10}: commit: Misiones agregado
421469d HEAD@{11}: commit (initial): README agregado
```

Como podemos ver en este historial se guardan tanto los commit como los reset que se fueron realizando. Esto nos permite ver los hash de los commit borrados y si por ejemplo queremos

volver al commit en el que agregamos a los héroes Linterna Verde y Robin no tenemos más que ejecutar el siguiente comando:

Vemos que todo vuelve a ese punto:



```
04-MATERIA-HEROES
├── historia
│   ├── batman.historia.md
│   └── superman.historia.md
├── ciudades.md
├── heroes.md
├── misiones.md
└── README.md
```

```
heroes.md > # Heroes
1  # Heroes
2
3  * Superman
4  * Batman
5  * Daredevil
6  * Aquaman
7  * Mujer Maravilla
8  * Linterna Verde
9  * Robin
10
```

