Contenido

[Iniciar un proyecto 2](#_Toc5558620)

[Eliminar un archivo del repositorio 3](#_Toc5558621)

[Crear snapshot 4](#_Toc5558622)

[Eliminar un commit o un snapshot 5](#_Toc5558623)

[Eliminar un cambio 5](#_Toc5558624)

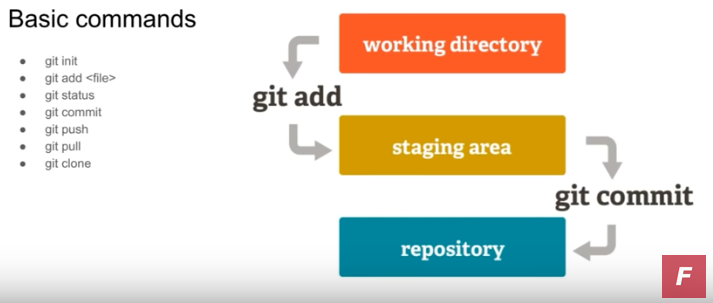
[Ignorar archivos en el proyecto 7](#_Toc5558625)

[Crear versiones alternativas de un proyecto 7](#_Toc5558626)

[Subiendo un repositorio en github 9](#_Toc5558627)

[Clonar un repositorio 10](#_Toc5558628)

[Sincronizar un Proyecto 10](#_Toc5558629)

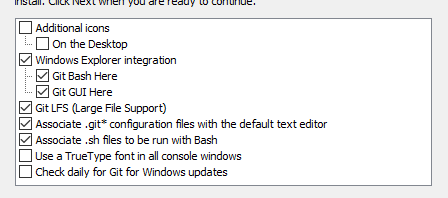


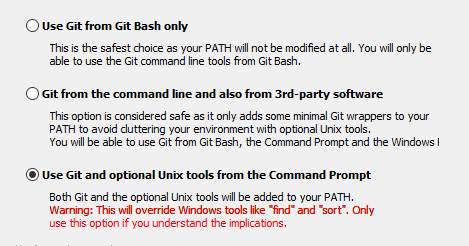
En git existen tres áreas

Working directory: donde se trabaja con todos los archivos.

Standing área: son los archivos que se preparan para el guardado. Cuando se tienen la versión completa de un software se pasan los archivos al standing área,

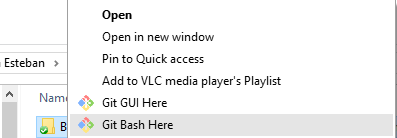
Instalación:





## Iniciar un proyecto

En la carpeta donde esta el proyecto damos clic derecho y damos la opción de git bash



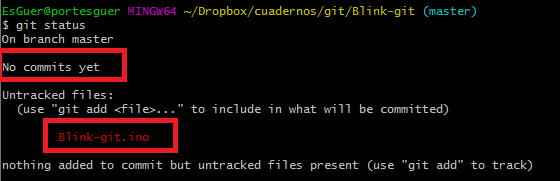
Luego que se habrá la ventana de terminal, damos el comando git init



Luego de esto git crea una carpeta oculta. git



Ahora damos el comando git status



Como indica la salida no hay commits, y el archivo Blink-git.ino se encuentra en el working área,

Git add

Para agregar un archivo al staging área damos comando: git add <nombre del archivo>

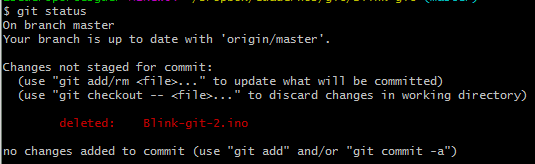
Configurar usuarios y email:

Git config –global user.mail “[correo@tal.com](mailto:correo@tal.com)”

Git config –global user.name “Mario-bros”

## Eliminar un archivo del repositorio

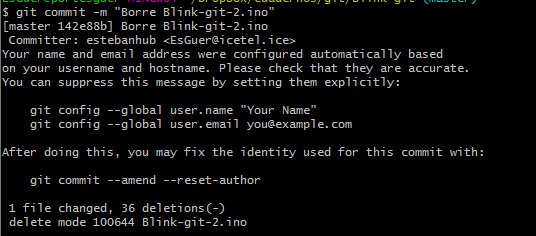
Aunque se elimine de la carpeta del proyecto queda en el repositorio para quietarlo de repositorio es:



Git rm <nombre del archivo>



Esto se va a reflejar en el próximo commit:

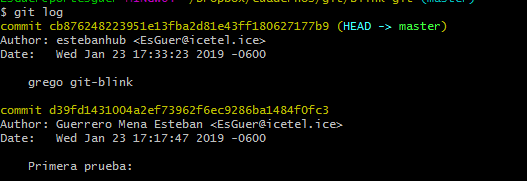


## Crear snapshot

Damos el commando git commint o git commint<nombre del archivo>

Habre un vim y en la primera línea le ponemos un nombre que distingue a ese snapshot

Para ver los cambios realizados damos git log



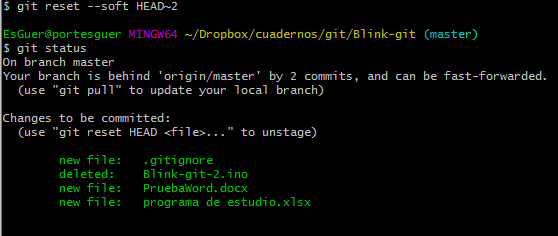
Vemos ya tenemos dos hash y muestra quien es el autor de cada snapshot y que archivos agrego



Ahora vemos que con git status me indica que no hay archivos para utilizar commint

## Eliminar un commit o un snapshot

Git reset - - soft HEAD~<NUMERO DE COMMITS A ELIMINAR>

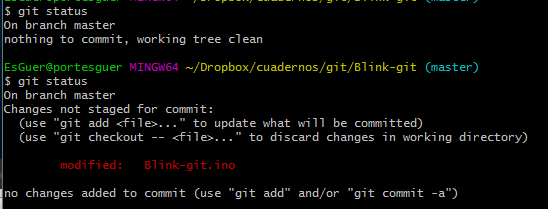


## Eliminar un cambio

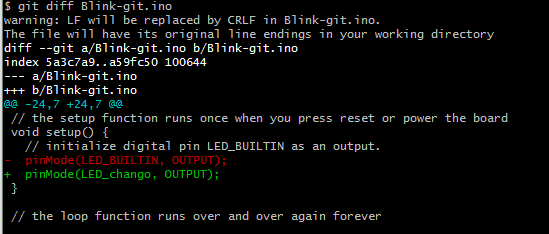
Cambiamos el contenido del archivo Blink-git-ino:

|  |  |
| --- | --- |
|  |  |

Si damos git status vemos que no indica que Blink-git.ino ha cambiado

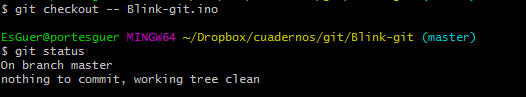


Para ver las diferencias damos el comando git diff <nombre del archivo>



Como vemos nos muestra las líneas que cambiaron

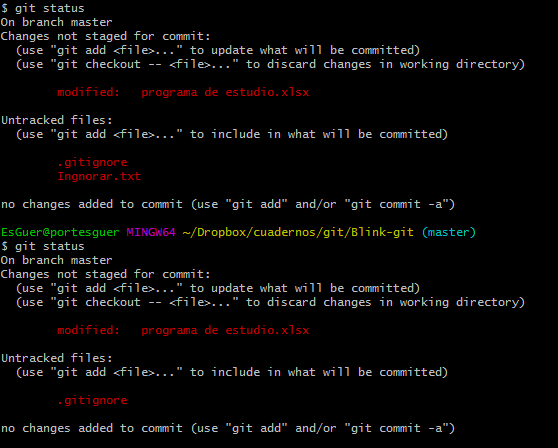
Para hacer que git lo devuelva a su estado anterior damos el comando: git checkout – Blink-git.ino



## Ignorar archivos en el proyecto

Para ignorar archivos durante el proyecto lo que debemos hacer es incluir un archivo que se llama .gitingnore y dentro se incluyen los archivos a ignorar.

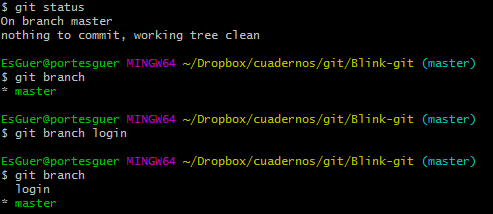
Dentro de este archivo vamo a ingnorar ingnorar.txt



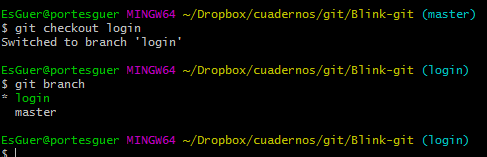
Recordar agregar este archivo y agregarlo al repositorio.

## Crear versiones alternativas de un proyecto (RAMAS)

Damos el comando git branch, para este ejemplo vamos a crear una versión llamada login:

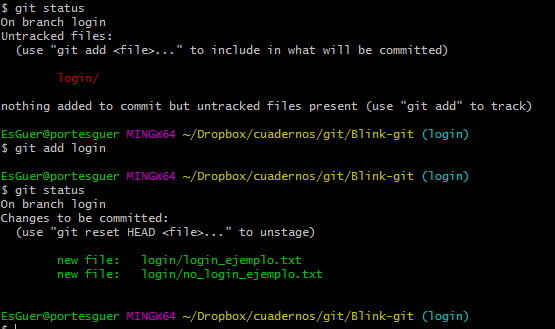


Para navegar entre versiones utilizamos nuevamente el comando git checkout



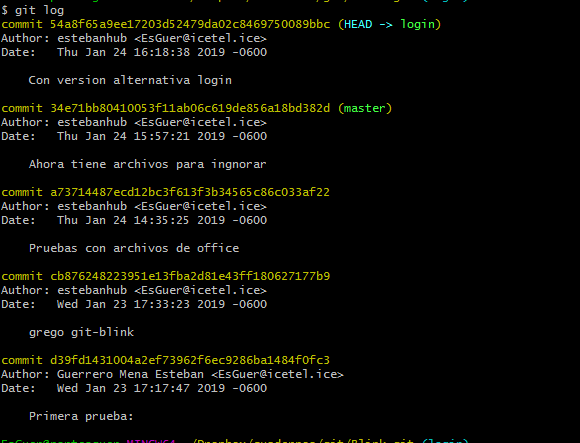
Para este ejemplo se crea una carpeta login y dentro login.txt

Debemos de agregar los archivos y carpetas:



Git ignora las capetas vacías

Con git log vemos los cambios:



Cuando cambiamos de versión se suman, restan o modifican archivos en la carpeta

## Subiendo un repositorio en github

Entramos a git hub y creamos un nuevo repositorio en este caso es <https://github.com/estebanHub/probandoGit.git>

git remote add origin <https://github.com/estebanHub/probandoGit.git>

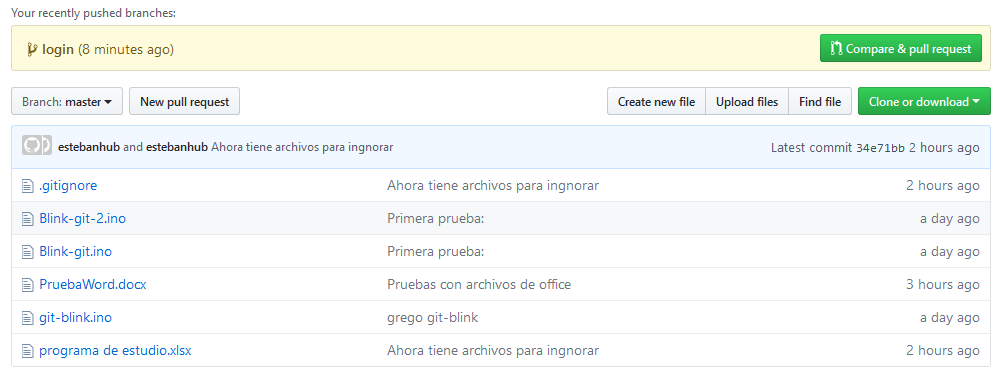
git push -u origin master

recordar que debemos de colocar el usuario y la contraseña de github

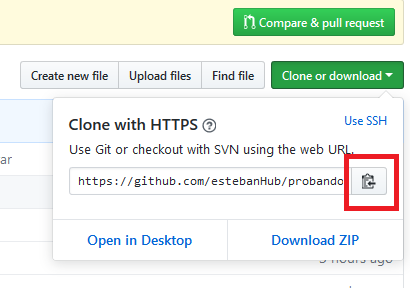
si queremos ver cual origen esta configurardo damos el comando

git remote –v





## Clonar un repositorio

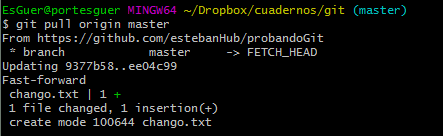


Es este icono copiamos la dirección

Git clone <https://github.com/estebanHub/probandoGit.git>

## Sincronizar un Proyecto

Cuando damos un syn en la interface gráfica el por debajo hace un pull



# Ramas

Resumen de comandos:

|  |  |
| --- | --- |
| Comando | Comentario |
| Git status | Nos muestra en qué estado están los archivos del proyecto |
| Git init | Inicializa un proyecto en git |
| Git add <nombre del archivo> | Agrega el archivo al staging área |
| Git add . | Agrega todos los archivos |
| Git rm <nombre del archivo> | Quita el archivo del staging área, hasta que se de un commit se borra por completo. |
| Git commit | Agrega el archivo al repositorio |
| Git commit –m “<mensaje…>” | Agrego los archivos al repositorio pero sin abrir el vim |
| git reset --soft HEAD~1 | Elimina el ultimo commit (en el ejemplo se va un 1 commit para atras) |
| Git config –global user.mail “<correo electronico>” | Configurar el correo |
| Git config –global user.name“<nombre>” | Configure el usuario |
| Git diff <nombre del archivo> | Muestra las líneas que cambiaron en un archivo |
| git checkout -- <nombre del archivo> | Devuelve a la versión guardada en el repositorio |
| git checkout -- <nombre de la versión> | Se pasa dentro de las versiones del proyecto |
| git remote add origin <direction de github> | Especifica en que repositorio debe de almacenarse los archivos |
| Git remote -v | Muestra cual el el origin del proyecto |
| git push -u origin <nombre de la rama> | Sube los archivo de una rama en especifico |
| git pull origin <nombre de la rama> | Ejemplo:  git pull origin master  sincroniza el codigo del repositorio en github y el código local. |
| Git log | Ve los commits de la rama, donde indica HEAD es el repositorio donde estamos trabajando. |
| Git log – decorate | Ve los commits de todas las ramas |
| Git branch | Ver las ramas del proyecto |
|  |  |