# **GIT**

Repositorio Plantilla: <https://github.com/Garcia091/2021-Plantilla-Heroe.git>

Despliegue: <https://garcia091.github.io/2021-Plantilla-Heroe/>

Interfaz de usuario gráfica

Descripción generada automáticamente

# **Ejercicio de clase:**

1. Clonar el repositorio:

$ git clone <https://github.com/Garcia091/2021-Plantilla-Heroe.git>

1. Listar todas las carpeta para ver la carpeta .git

$ ls -al

Imagen que contiene Texto

Descripción generada automáticamente

1. Cambiar el título de la aplicación por Ejercicio de clase git

Texto

Descripción generada automáticamente

1. Realizar un git status para agregar los cambios al staging

$ git status

Texto

Descripción generada automáticamente

1. Agregar la información

Agregar archivo con la misma extención

git add \*.html

Texto

Descripción generada automáticamente

1. Hacer commit con los cambios actuales

git commit -m "Agregando nuevo título al html"

1. Aplicar los otros tipos de adición en git
2. Agregar un nuevo archivo excluir.js y añadirlos al ara de trabajo de Git y luego excluirlos

git add -A

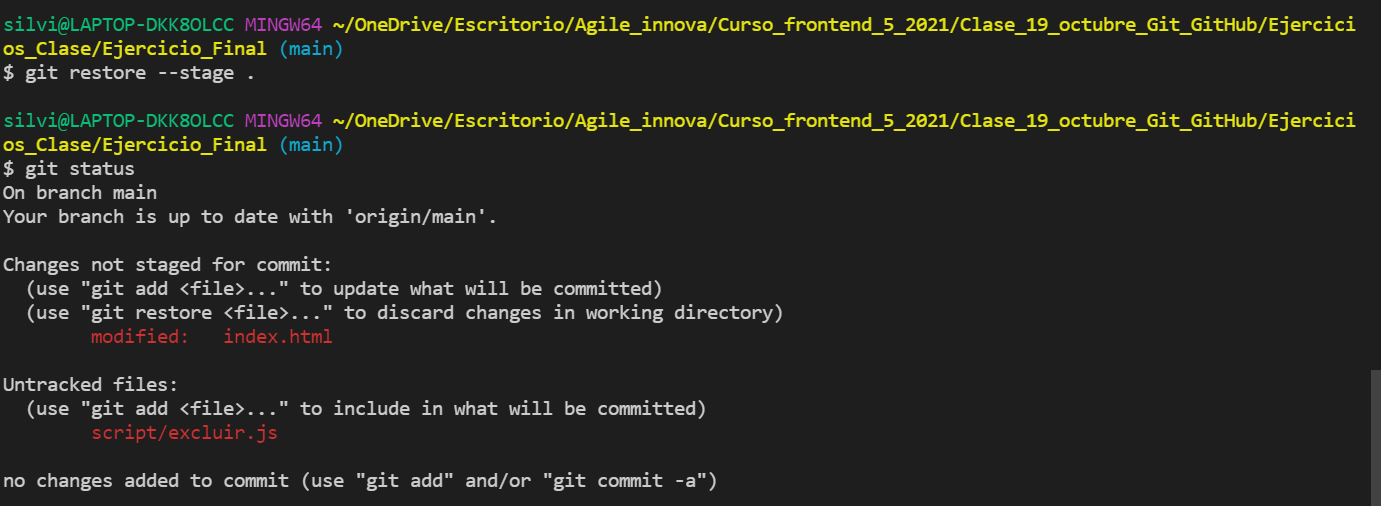
git reset script/main.js

Texto

Descripción generada automáticamente

1. Descartar cambios en el directorio de trabajo

$ git restore --stage .



1. Verificar el historial de commit

$ git log

Texto

Descripción generada automáticamente

1. Renombrar la rama de master a main, si aún no la has cambiado.

git branch -M main

1. Subir la información al repositorio

Eliminar el archivo de prueba excluir.js

git add .

git commit -m “Agregando nuevo título al html”

git push origin main

Texto

Descripción generada automáticamente

1. Git log para ver el historial de nuevo

Texto

Descripción generada automáticamente

# **Hacer cambios en el repositorio para ver cómo podemos volver a una versión anterior.**

1. Hacer cambios al interior de los archivos

Texto

Descripción generada automáticamente

Agregar estilos

Texto

Descripción generada automáticamente

Verificar los cambios con git status

Texto

Descripción generada automáticamente

Revisar los cambios realizados al interior de los archivos

$ git diff

q para salir del editor de texto de git

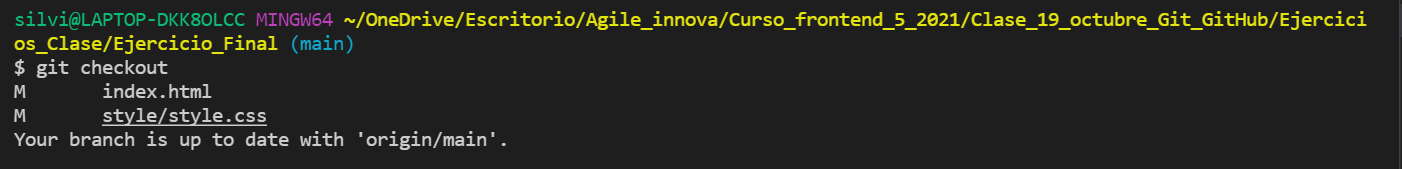
Texto

Descripción generada automáticamente

Los cambios que realizamos no nos gusta, para regresar al estado del último commit realizado, ejecutamos el siguiente comando

git checkout

Permite conocer los archivo que tienen modificaciones o el estado que tenga el archivo



git checkout -- .

Texto

Descripción generada automáticamente

# Historial de commit

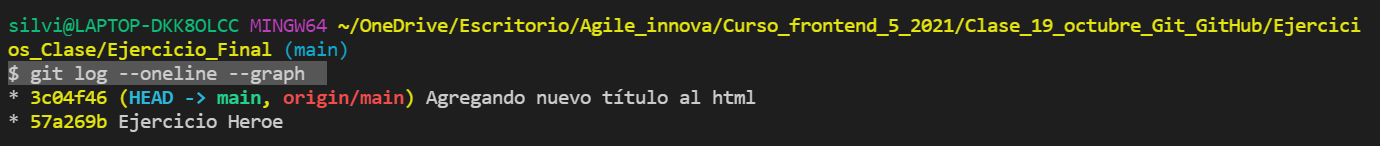
git log

Texto

Descripción generada automáticamente

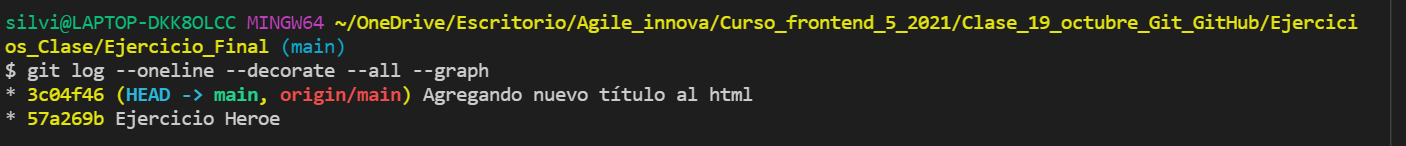
Ver el resumen del historial

$ git log --oneline –graph



Otra forma de mostrar el historial de las versiones

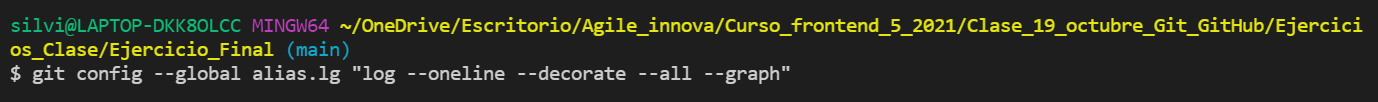
$ git log --oneline --decorate --all –graph



# Alias:

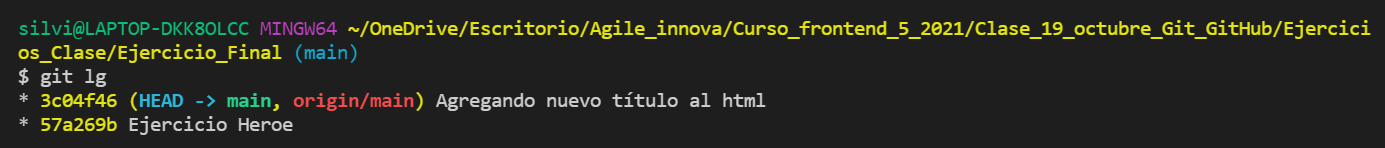
Crear un alias

git config --global alias.lg "log --oneline --decorate --all --graph"



Ahora con el comando lg ejecutamos todo la línea anterior

git lg



para observar todos los alias que hemos creado

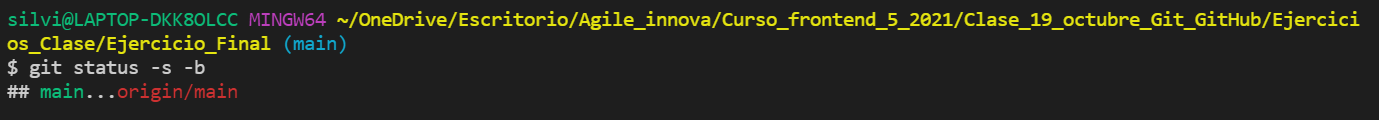
git config --global -l

Texto

Descripción generada automáticamente

# **Para saber la rama y los cambios realizados**

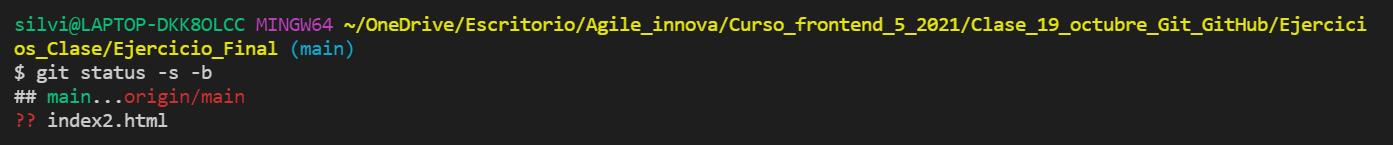
git status -s -b



Crear un archivo de prueba

Texto

Descripción generada automáticamente con confianza media

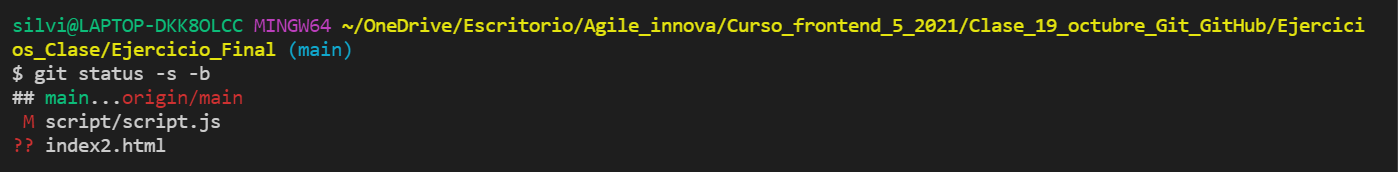


Realicemos un cambio a ver qué pasa con el comando anterior. Nos permite ver los cambios de forma resumida

Texto

Descripción generada automáticamente

Ejecutar el comando



# **Ya tenemos una versión estable de nuestra aplicación, vamos a crear ramas para desarrollar**

Las Ramas son conocidas como líneas de tiempo

1. Crear una rama developers

$ git checkout -b developers

Texto

Descripción generada automáticamente

1. Listar todas las ramas

$ git branch

Texto

Descripción generada automáticamente

1. Crear cambios en la rama developers

Crear un Pie de pagina donde se evidencia el nombre del programador y el año en curso. Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Dar estilos

Texto

Descripción generada automáticamente

Si me cambio nuevamente a la rama main que pasa con los cambios?

Los cambios no están asignados a ninguna rama lo que debemos hacer es cambiarnos nuevamente a la rema developers y hacer commit y empujar los cambios

1. Agregar los cambios al directorio de trabajo

Git add .

Git commit -m “Agregar footer”

Git push origin developers

Texto

Descripción generada automáticamente

Los cambios estan en la rama developers

Si cambiamos a la rama main los cambios de footer no deberian versen

Texto

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

# **Combinar las ramas**

Vamos a combinar la rama main con el contenido de la rama developers

Estar en la rama main

$ git checkout main

Texto

Descripción generada automáticamente

Combinar las ramas

$ git merge developers

Texto

Descripción generada automáticamente

**Controlar conflictos al combinar ramas**

**En la rama main y en developer agregar los siguientes cambios.**

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

**Hacer lo mismo en developer pero intercambiar la información**

**Developers**

Texto

Descripción generada automáticamente

Agregar datos y hacer commit en la rama developers

Texto

Descripción generada automáticamente

Combinar los cambios de las ramas:

Merge:

1. Fast-forward : se dispara cuando el git no reconoce cambios y se puede integrar a la rama principal sin ningún problema
2. Uniones automáticas: El sistema combinas los cambios de automáticamente
3. Uniones manuales: cuando salen conflictos, git pide actualización manual y hace un merge commit para guardar los cambios.



Texto

Descripción generada automáticamente

Tenemos el conflicto creado, en su editor de texto validar qué información desea conservar

Cuando hacemos un merge, el por defecto genera un commit, cuando no se puede completar el proceso la rama en la cual estabas trabajando quedará de esta manera. Soluciona el conflicto y guarda los cambios.

Texto

Descripción generada automáticamente

Guardar cambios.

Texto

Descripción generada automáticamente

Si necesitas borrar la rama lo puedes hacer con el siguiente comando

git branch -d developers

# **Regresar a una versión anterior**

Cambiar los colores de la página a petición del cliente, sus nuevos colores son rojos y verdes

Texto

Descripción generada automáticamente

Guardar los cambios en la rama principal

Texto

Descripción generada automáticamente

Luego de mostrar el resultado final, no era lo que el cliente deseaba por tal motivo debemos regresar una versión anterior

$ git reset --hard HEAD^

Imagen que contiene Texto

Descripción generada automáticamente

Guardar los cambios y enviarlos a la rama.

Si queremos regresar a una versión diferente hacemos lo siguiente

$ git log

Texto

Descripción generada automáticamente

Escoger el commit al cual deseas regresar

