


- Adjunta una captura de pantalla del repositorio que has creado (es decir que se vea el fichero ".git")

 Símbolo del sistema

```
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\pablo>git --version
git version 2.44.0.windows.1

C:\Users\pablo>git config --global user.name "Pablo"

C:\Users\pablo>git config --global user.email "pablokh06@gmail.com"

C:\Users\pablo>cd C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git init
Initialized empty Git repository in C:/Users/pablo/Desktop/UT4_PracticaGuiada_5_GIT/.git/

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>_
```

- Ejecuta ahora un git status para que veas el cambio de estado. Adjunta captura de pantalla y comenta las diferencias con el anterior git status que ejecutaste.

La mayor diferencia que vemos es que en el primer status es que el archivo esta sin seguimiento y no está en la zona de preparación a diferencia de la segunda, donde esta ya con su seguimiento

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        index.html

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git add index.html

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>_
```

- Vuelve a ejecutar un git status y observa como git ha detectado que el fichero se ha modificado (adjunta captura de pantalla de lo que devuelve el comando).

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md
        bluestyle.css
```

- Ejecuta un git status y observa lo que devuelve. Adjunta captura y comenta lo que te devuelve el comando.

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git add .

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md
        new file:   bluestyle.css
        new file:   index.html

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>
```

- Adjunta captura de lo que devuelve el comando de commit, a continuación ejecuta un git status y comenta lo que devuelve también.

ha hecho un save y ahora está todo limpio

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -m "Primer commit!"
[master (root-commit) df146de] Primer commit!
3 files changed, 29 insertions(+)
create mode 100644 README.md
create mode 100644 bluestyle.css
create mode 100644 index.html
```

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master
nothing to commit, working tree clean
```

- Ahora modifica algo a uno de los 3 ficheros (lo que quieras), guarda el cambio y ejecuta un git status. Adjunta captura y comenta lo que te devuelve.

me devuelve que el archivo index ha sido modificado y no está en seguimiento

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>S
```

- Adjunta captura de lo que devuelve el comando de commit -a, a continuación ejecuta un git status y comenta lo que devuelve también.

Se informa de texto modificado

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -a -m "Modificado y comiteado sin pasar por staging"
[master 9fe9a9d] Modificado y comiteado sin pasar por staging
1 file changed, 1 insertion(+), 1 deletion(-)
```

- Crea un fichero llamado "config.txt" en el directorio y haz un git status. Observa como el archivo se reconoce por git en el area de Archivos sin seguimiento.

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -a -m "Modificado y comiteado sin pasar por staging"
[master 9fe9a9d] Modificado y comiteado sin pasar por staging
1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        config.txt

nothing added to commit but untracked files present (use "git add" to track)

```

- Crea un fichero que se llame ".gitignore" y ponle el siguiente contenido : config.txt. Haz un git status y observa como el archivo config.txt ya no aparece.

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore

nothing added to commit but untracked files present (use "git add" to track)

```

- Ejecuta un "git add ." y un commit.

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git add .

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -m "Segundo commit!"
[master a748ea8] Segundo commit!
1 file changed, 1 insertion(+)
 create mode 100644 .gitignore

```

- Adjunta captura de lo que devuelve el comando y comenta lo que ves.

veo no solo los commit creado sino por quien ha sido creado en este caso por mi con la respectiva fecha, además te dice cuando ha pasado por staging y cuando no

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git log
commit a748ea8d0bf757943b1d6666175adcc86cf72d1b (HEAD -> master)
Author: Pablo <pablokh06@gmail.com>
Date: Mon Mar 4 20:27:02 2024 +0000
```

Segundo commit!

```
commit 9fe9a9d3915d90282bb960aefd0063b5ca2069f8
Author: Pablo <pablokh06@gmail.com>
Date: Mon Mar 4 20:12:49 2024 +0000
```

Modificado y comiteado sin pasar por staging

```
commit df146de8cc53b3b22fa6f9d99c966cea9e4b4970
Author: Pablo <pablokh06@gmail.com>
Date: Mon Mar 4 20:01:17 2024 +0000
```

Primer commit!

- Ejecuta el comando `git checkout` para cambiarte de rama y ejecuta el comando `git branch` de nuevo. Adjunta captura y comenta lo que ves.

Podemos observar como se ha cambiado la rama con éxito y el asterisco que nos dice que rama usamos que prueba que efectivamente el cambio se ha ejecutado correctamente

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch primerarama

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch
* master
  primerarama

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git checkout primerarama
Switched to branch 'primerarama'

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch
  master
* primerarama

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>
```

- Modifica el fichero `index.html` y añádele algo. Por ejemplo una nueva línea tipo:

Nueva línea para trabajar con branches!

Guarda el fichero, ejecuta un `git status` y comenta lo que devuelve.

nos devuelve esto mostrando que en nuestra rama "primerarama" se ha modificado el index

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch primerarama
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

- Añade los cambios a la zona de staging (como hemos hecho antes en la práctica) y vuelve a ejecutar un git status. Adjunta captura y comenta lo que ves.

se ha cambiado correctamente en la primerarama

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git add .

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch primerarama
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   index.html
```

- Confirma los cambios y actualiza el repositorio, es decir ejecuta un commit como hemos visto. Vuelve a ejecutar un git status. Adjunta captura y comenta lo que ves.

se muestra como se ha guardado en el commit y ha quedado limpio

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -m "Tercer commit!"
[primerarama 50580ab] Tercer commit!
1 file changed, 1 insertion(+)

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch primerarama
nothing to commit, working tree clean
```

- Descargar la imagen adjunta a esta práctica (git-icon.jpeg) y pégala en el directorio de trabajo donde tenemos el repositorio. Añade en el fichero index.html la siguiente línea donde creas conveniente:

```

1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |
6  | <title>Hello Mundo!</title>
7  <link rel="stylesheet" href="bluestyle.css">
8
9  </head>
10
11 <body>
12 |
13 | <h1>Hello world!</h1>
14
15 | <p>Este es el primer fichero de mi Git Repo.</p>
16 | <p>Nueva línea para trabajar con branches!</p>
17 | <div></div>
18 |
19 |
20 </body>
21
22 </html>

```

- Ejecuta un git status y comenta lo que ves. ¿En qué zona está el archivo modificado y el nuevo que hemos agregado?

se ha modificado el archivo index que no está en commit y el archivo imagen que no está en seguimiento

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git status
On branch primerarama
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        git-icon.jpeg

no changes added to commit (use "git add" and/or "git commit -a")

```

- Añade el fichero a la zona de staging (es decir git add . ), confirma los cambios y actualiza el repositorio, es decir ejecuta un commit como hemos visto. Vuelve a ejecutar un git status. Adjunta captura y comenta lo que ves.

vemos que se han guardado lo nuevo insertado y lo cambiado

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git add .

```

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git commit -m "Cuarto commit!"
[primerarama 7a3eb6c] Cuarto commit!
2 files changed, 2 insertions(+)
create mode 100644 git-icon.jpeg
```

- Cambia a la rama “master” con un: git checkout master

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git checkout master
Switched to branch 'master'
```

- Vuelve ejecutar el comando “ls” para listar los ficheros del directorio actual y comenta lo que ves respecto al anterior ls que has hecho. ¿A que se deben las diferencias? ¿Que le ha pasado al contenido del fichero index.html?

realmente no pude ejecutar esto antes así que no se cuáles son las diferencias (estaba usando ls y en windows se usa dir me entere tarde)

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: B0C6-1729

Directorio de C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT

04/03/2024  20:53    <DIR>          .
04/03/2024  20:53    <DIR>          ..
04/03/2024  20:24                10 .gitignore
04/03/2024  19:56               112 bluestyle.css
04/03/2024  20:19                 0 config.txt
04/03/2024  20:53               233 index.html
04/03/2024  19:55                17 README.md
                5 archivos                372 bytes
                2 dirs  349.733.888.000 bytes libres
```

- Adjunta captura de lo que devuelve el comando de merge, a continuación vuelve a ejecutar un “ls” y vuelve a comprobar el contenido del fichero index.html. ¿Que ha ocurrido?

ha cambiado el peso además que ha añadido los cambios que he hecho en la primera rama

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git merge primerarama
Updating a748ea8..7a3eb6c
Fast-forward
 git-icon.jpeg | Bin 0 -> 3423 bytes
 index.html    |   3 +++
2 files changed, 3 insertions(+)
create mode 100644 git-icon.jpeg
```



```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: B0C6-1729

Directorio de C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT

04/03/2024  21:01    <DIR>          .
04/03/2024  21:01    <DIR>          ..
04/03/2024  20:24             10 .gitignore
04/03/2024  19:56            112 bluestyle.css
04/03/2024  20:19              0 config.txt
04/03/2024  21:01           3.423 git-icon.jpeg
04/03/2024  21:01           359 index.html
04/03/2024  19:55           17 README.md
               6 archivos              3.921 bytes
               2 dirs 349.707.091.968 bytes libres

```

- Crea una nueva rama como hemos visto. Comprueba que se haya creado y bórrala. Borra también la rama que creamos al principio de la práctica (es decir deja solo la rama master). Escribe los comandos que has usado y adjunta capturas.

git branch segundarama, git branch, git branch -d primerarama, git branch -d segundarama, git branch

creamos la segunda rama, vemos que se ha creado, borramos la primera, borramos la segunda y comprobamos cual queda

```

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch segundarama

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch
* master
  primerarama
  segundarama

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch -d primerarama
Deleted branch primerarama (was 7a3eb6c).

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch -d segundarama
Deleted branch segundarama (was 7a3eb6c).

C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git branch
* master

```

- Sacar una lista de commits que has hecho en tu rama master y vete hasta el primero que hiciste (con un checkout + id). Observa el contenido del fichero index.html. Adjunta captura y comenta lo que ves.

se ve el guardado del primer commit sin modificaciones imagen etc

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |
6  | <title>Hello World!</title>
7  | <link rel="stylesheet" href="bluestyle.css">
8  |
9  </head>
10
11 <body>
12 |
13 | <h1>Hello world!</h1>
14 |
15 | <p>Este es el primer fichero de mi Git Repo.</p>
16 |
17 </body>
18
19 </html>
```

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git checkout df146de
Note: switching to 'df146de'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using -c with the switch command. Example:

```
git switch -c <new-branch-name>
```

Or undo this operation with:

```
git switch -
```

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at df146de Primer commit!

- Vuelve hacer otro checkout pero al último commit de nuevo. Adjunta captura y comenta lo que ves.

ahora esta como en el último commit, con todos los cambios hechos y las modificaciones y añadidos puestos

```
C:\Users\pablo\Desktop\UT4_PracticaGuiada_5_GIT>git checkout 7a3eb6c
Previous HEAD position was df146de Primer commit!
HEAD is now at 7a3eb6c Cuarto commit!
```

```
<> index.html > html > body
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |
6  | <title>Hello Mundo!</title>
7  | <link rel="stylesheet" href="bluestyle.css">
8  |
9  | </head>
10
11 <body>
12 |
13 | <h1>Hello world!</h1>
14 |
15 | <p>Este es el primer fichero de mi Git Repo.</p>
16 | <p>Nueva línea para trabajar con branches!</p>
17 | <div></div>
18 |
19 |
20 | </body>
21
22 </html>
```