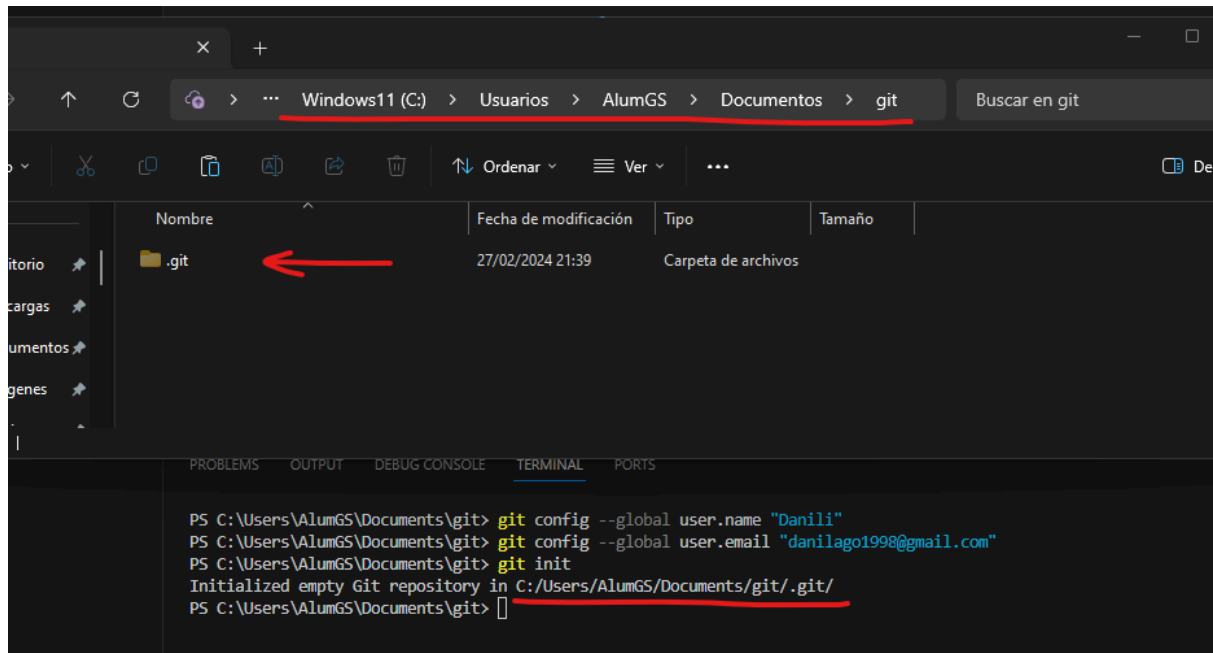
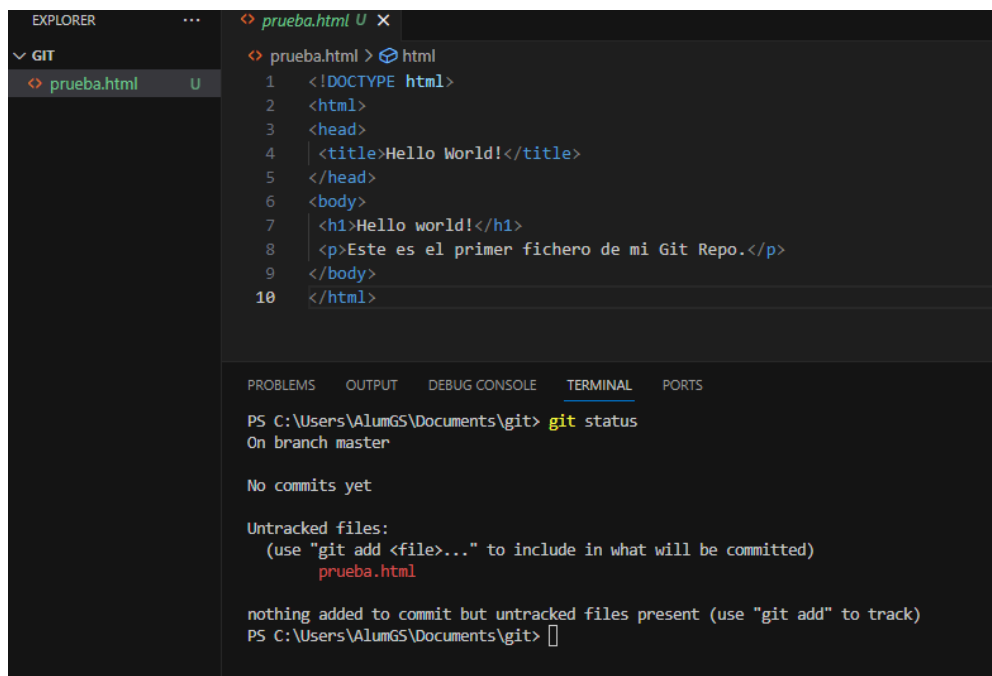


# Práctica Guiada: Primeros pasos con Git

Creamos el repositorio local:



Creamos un archivo prueba.html y comprobamos el estado del repositorio:



Añadimos nuestro archivo a la zona de staging utilizando el comando:

- **git add "nombre\_fichero"**

En mi caso: **git add prueba.html**

```
PS C:\Users\AlumGS\Documents\git> git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\AlumGS\Documents\git> git add prueba.html
PS C:\Users\AlumGS\Documents\git> git status
On branch master

No commits yet

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

PS C:\Users\AlumGS\Documents\git> 
```

Podemos observar que en el primer git status nuestro archivo no está siendo rastreado por Git, en el segundo git status nuestro archivo se encuentra en la zona de staging, listos para ser commiteados.

## 6. Añadiendo más de un archivo al repositorio.

Hemos creado dos archivos nuevos y modificado prueba.html. Ahora usamos: `<git add .>`

```
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:   prueba.html

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

PS C:\Users\AlumGS\Documents\git> git add .
PS C:\Users\AlumGS\Documents\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:   prueba.html

PS C:\Users\AlumGS\Documents\git> 
```

## 7. Git Commit

Podemos observar que se ha hecho commit de nuestros 3 archivos y si usamos `git status` veremos que no hay ningún proyecto en la zona de staging por lo que no hay nada que comitear (working tree clean).

```
PS C:\Users\AlumGS\Documents\git> git commit -m "Primer commit!"
[master (root-commit) 8a4c07d] Primer commit!
3 files changed, 22 insertions(+)
create mode 100644 README.MD
create mode 100644 bluestyle.css
create mode 100644 prueba.html
PS C:\Users\AlumGS\Documents\git> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\AlumGS\Documents\git> █
```

Ahora si realizamos algún cambio y queremos hacer un commit sin pasar por el entorno staging, ejecutamos el siguiente comando:

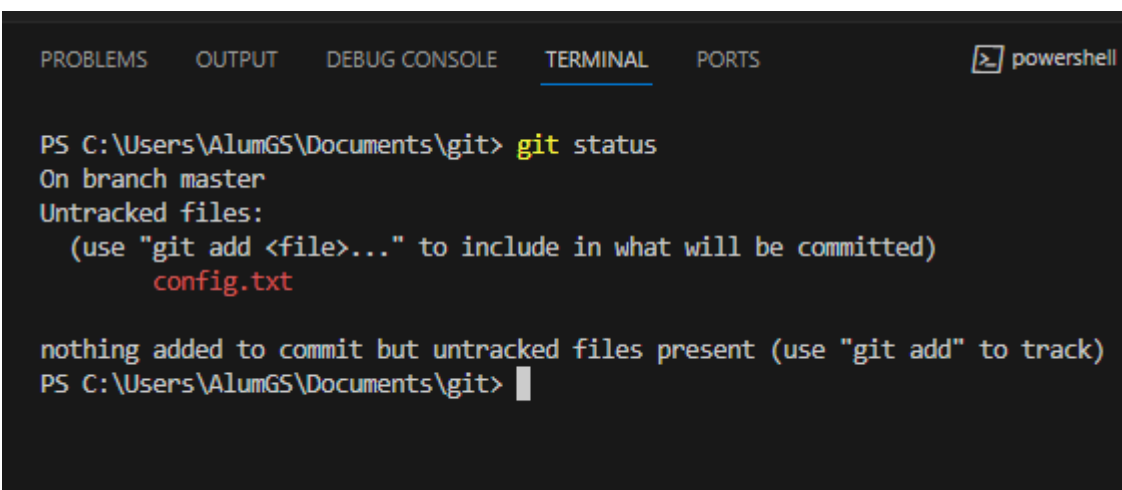
**`git commit -a -m "Modificado y comiteado sin pasar por staging"`**

La opción `-a` automáticamente preparará cada archivo cambiado que ya esté siendo rastreado.

```
PS C:\Users\AlumGS\Documents\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:   prueba.html

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\AlumGS\Documents\git> git commit -a -m "Modificado y comiteado sin pasar por staging"
[master 43e80c5] Modificado y comiteado sin pasar por staging
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\AlumGS\Documents\git> █
```

## 8. Ignorando ficheros (.gitignore)



The screenshot shows a PowerShell terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active. The command `git status` has been executed, showing that the file `config.txt` is untracked. The output indicates that nothing has been added to the commit because there are untracked files present.

```
PS C:\Users\AlumGS\Documents\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)
PS C:\Users\AlumGS\Documents\git> █
```

Ahora creamos un archivo **.gitignore** y escribimos dentro de él nuestro archivo **config.txt**

```
PS C:\Users\AlumGS\Documents\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)
PS C:\Users\AlumGS\Documents\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)
PS C:\Users\AlumGS\Documents\git> █
```

Ahora hacemos un “**git add .**” seguido de un “**commit**”

```
PS C:\Users\AlumGS\Documents\git> git add .
PS C:\Users\AlumGS\Documents\git> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   .gitignore

PS C:\Users\AlumGS\Documents\git> git commit -a -m
error: switch `m' requires a value
PS C:\Users\AlumGS\Documents\git> git commit -m "ignorando ficheros"
[master cbb1ad5] ignorando ficheros
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
PS C:\Users\AlumGS\Documents\git> git status
On branch master
nothing to commit, working tree clean
PS C:\Users\AlumGS\Documents\git> █
```

## 9. Historial de commits (Log)

Para ver el historial de los commits que se han hecho en un repositorio se usa el comando “log”. Ejecutando el siguiente comando: **git log**

De esta forma podrás ver los commits que hemos hecho.

Nota: también puedes ejecutar el comando **git log --oneline**

```
PS C:\Users\AlumGS\Documents\git> git log
commit cbb1ad5b4ec43588b83053fab89966cb0ab738b4 (HEAD -> master)
Author: Danili <danilago1998@gmail.com>
Date: Thu Feb 29 21:40:06 2024 +0000
```

ignorando ficheros

```
commit 43e80c54e587be2b7dc85438bf6f631299beb614
Author: Danili <danilago1998@gmail.com>
Date: Tue Feb 27 22:07:12 2024 +0000
```

Modificado y comiteado sin pasar por staging

```
commit 8a4c07d2e745c93d4f369f0c22e909cad4666fad
Author: Danili <danilago1998@gmail.com>
Date: Tue Feb 27 21:59:26 2024 +0000
```

Primer commit!

```
PS C:\Users\AlumGS\Documents\git> git log --oneline
>>
cbb1ad5 (HEAD -> master) ignorando ficheros
43e80c5 Modificado y comiteado sin pasar por staging
8a4c07d Primer commit!
PS C:\Users\AlumGS\Documents\git> █
```

Efectivamente, podemos observar que con **git log** se muestra nuestro historial de commits, de más recientes a más antiguos junto con su fecha, comentario, id del commit y el usuario que lo realizó.

**git log --oneline** es lo mismo pero solo nos muestra el id del commit y su comentario.

## 10. Trabajando con Branches (Ramas)

### 10. Trabajando con Branches (Ramas)

En Git, una **rama** es una nueva versión **separada** del repositorio principal (o rama master). Por defecto, estaremos trabajando en la rama **master**, pero especialmente cuando trabajemos de forma colaborativa con otros compañeros puede ser útil no trabajar en la rama master, si no crear un **rama aparte** para nosotros sobre la que poder realizar nuestros cambios **sin interferir** en los cambios que se puedan estar haciendo en la **master**.

Una vez terminados los cambios en nuestra rama, podremos unir nuestra rama nuevamente a la Master para que se reflejen tus cambios con un "merge" (ya veremos lo que es esto).



En definitiva, las ramas te permiten **trabajar** en diferentes partes de un proyecto sin afectar la rama master (o rama principal). Cuando el trabajo está completo, una rama puede fusionarse con el proyecto principal.

Incluso puedes cambiar/movertte entre ramas y trabajar en diferentes proyectos sin que interfieran entre sí.

## 11. Creando una nueva rama en Git

Podemos observar que nos indica haber cambiado de rama con el asterisco:

```
PS C:\Users\AlumGS\Documents\git> git branch rama-test
PS C:\Users\AlumGS\Documents\git> git branch
* master
  rama-test
PS C:\Users\AlumGS\Documents\git> git checkout rama-test
Switched to branch 'rama-test'
PS C:\Users\AlumGS\Documents\git> git branch
  master
* rama-test
PS C:\Users\AlumGS\Documents\git> █
```

Si ahora modificamos nuestro archivo html y hacemos un git status, nos mostrará que dentro de la rama "rama-test", prueba.html ha sido modificado

```
PS C:\Users\AlumGS\Documents\git> git status
On branch rama-test
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:   prueba.html

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

Ahora podemos añadirlo a la zona de staging, listo para ser comiteado:

```
PS C:\Users\AlumGS\Documents\git> git add prueba.html
PS C:\Users\AlumGS\Documents\git> git status
On branch rama-test
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   prueba.html

PS C:\Users\AlumGS\Documents\git> █
```

```
PS C:\Users\AlumGS\Documents\git> git commit -m "prueba2"
[rama-test 95bf910] prueba2
 1 file changed, 1 insertion(+)
PS C:\Users\AlumGS\Documents\git> git status
On branch rama-test
nothing to commit, working tree clean
PS C:\Users\AlumGS\Documents\git> █
```

Ahora si añadimos una imagen a nuestro archivo html:

```
PS C:\Users\AlumGS\Documents\git> git status
On branch rama-test
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:   prueba.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")
PS C:\Users\AlumGS\Documents\git>
```

El archivo html fue modificado pero nuestra imagen aparece como untracked, por lo que habrá que añadirla.

```
PS C:\Users\AlumGS\Documents\git> git add .
PS C:\Users\AlumGS\Documents\git> git commit -m "prueba2ConImagen"
[rama-test ccc39a6] prueba2ConImagen
 2 files changed, 1 insertion(+)
 create mode 100644 git-icon.jpeg
PS C:\Users\AlumGS\Documents\git> git status
On branch rama-test
nothing to commit, working tree clean
PS C:\Users\AlumGS\Documents\git>
```

Nuestra imagen fue agregada a la zona de staging del archivo html y posteriormente fue comiteada.

## 12. Moverse entre Ramas (Branches)

```
PS C:\Users\AlumGS\Documents\git> ls

Directorio: C:\Users\AlumGS\Documents\git

Mode                LastWriteTime         Length Name
----                -
-a----            29/02/2024    22:00           10 .gitignore
-a----            27/02/2024    22:10          112 bluestyle.css
-a----            29/02/2024    21:34           0 config.txt
-a----            29/02/2024    22:14         3423 git-icon.jpeg
-a----            29/02/2024    22:15          317 prueba.html
-a----            27/02/2024    22:10           99 README.MD

PS C:\Users\AlumGS\Documents\git>
```



Ahora hacemos un **git checkout nombre\_de\_la\_rama**

```
PS C:\Users\AlumGS\Documents\git> git checkout master
Switched to branch 'master'
PS C:\Users\AlumGS\Documents\git> ls

Directorio: C:\Users\AlumGS\Documents\git

Mode                LastWriteTime         Length Name
----                -
-a----             29/02/2024     22:00           10 .gitignore
-a----             27/02/2024     22:10          112 bluestyle.css
-a----             29/02/2024     21:34           0 config.txt
-a----             29/02/2024     22:23          230 prueba.html
-a----             27/02/2024     22:10           99 README.MD

PS C:\Users\AlumGS\Documents\git> 
```

Podemos observar que en la rama **master** tenemos nuestros archivos originales y en la rama **rama-test** aparecen nuestros archivos modificados junto a la imagen que agregamos. Nuestro archivo html ya no tiene los cambios que se hicieron en la rama **rama-test** y en el **ls** que hemos hecho, tiene más caracteres (length) y una fecha más reciente.

### 13. Fusionar ramas (Merge)

Podemos ver que nos ha fusionado la rama **rama-test** con la rama **master**, en un solo archivo

```
PS C:\Users\AlumGS\Documents\git> git merge rama-test
Updating cbb1ad5..ccc39a6
Fast-forward
 git-icon.jpeg | Bin 0 -> 3423 bytes
 prueba.html   | 2 ++
2 files changed, 2 insertions(+)
create mode 100644 git-icon.jpeg
PS C:\Users\AlumGS\Documents\git> ls

Directorio: C:\Users\AlumGS\Documents\git

Mode                LastWriteTime         Length Name
----                -
-a----             29/02/2024     22:00           10 .gitignore
-a----             27/02/2024     22:10          112 bluestyle.css
-a----             29/02/2024     21:34           0 config.txt
-a----             29/02/2024     22:29          3423 git-icon.jpeg
-a----             29/02/2024     22:29          317 prueba.html
-a----             27/02/2024     22:10           99 README.MD

PS C:\Users\AlumGS\Documents\git> 
```



```

PS C:\Users\AlumGS\Documents\git> git branch rama-test2
PS C:\Users\AlumGS\Documents\git> git branch
* master
  rama-test
  rama-test2
PS C:\Users\AlumGS\Documents\git> git branch -d rama-test2
Deleted branch rama-test2 (was ccc39a6).
PS C:\Users\AlumGS\Documents\git> git branch -d rama-test
Deleted branch rama-test (was ccc39a6).
PS C:\Users\AlumGS\Documents\git> git branch
* master
PS C:\Users\AlumGS\Documents\git> 

```

Podemos ver que usando **git branch rama-test2** nos ha creado una nueva rama, con **git branch** comprobamos las ramas creadas y con **git branch -d <nombre-rama>** podemos borrar las ramas que queramos.

## 14. Volver a un commit específico

```

PS C:\Users\AlumGS\Documents\git> git log --oneline
ccc39a6 (HEAD -> master) prueba2ConImagen
95bf910 prueba2
cbb1ad5 ignorando ficheros
43e80c5 Modificado y comiteado sin pasar por staging
8a4c07d Primer commit!
PS C:\Users\AlumGS\Documents\git> git checkout 8a4c07d
Note: switching to '8a4c07d'.

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 8a4c07d Primer commit!
PS C:\Users\AlumGS\Documents\git> 

```

Usando **git checkout + ID** podemos volver al primer commit que guardamos, mostrándonos cómo eran nuestros archivos en ese commit.

```

PS C:\Users\AlumGS\Documents\git> git checkout ccc39a6
Previous HEAD position was 8a4c07d Primer commit!
HEAD is now at ccc39a6 prueba2ConImagen

```