

# GIT



## Contornos de Desarrollo

Nicolas Primoy Pucheta

## Ejercicio 2.1 Instalación y primeros pasos en Git

### Primeros pasos

Una vez instalado Git, realizamos la configuración utilizando los siguientes comandos:

```
git config --global user.name "Tu Nombre"  
git config --global user.email "tu.email@example.com"  
git config --global core.editor "visual code studio"
```

Verificamos que la configuración sea haya aplicado correctamente con el comando :

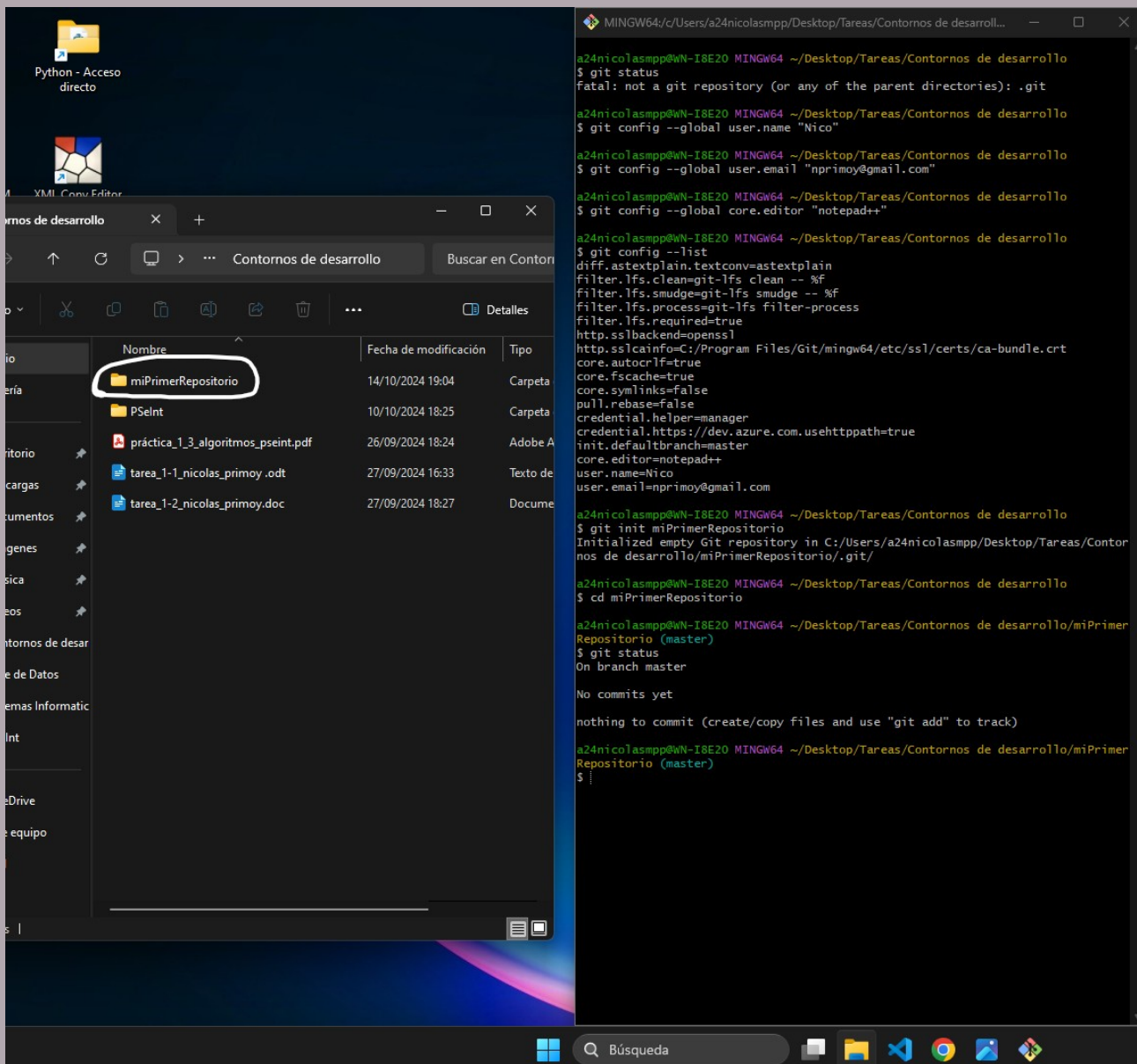
```
git config --list
```

Creamos nuestro primer repositorio con el comando:

```
git init miPrimerRepositorio
```

Comprobamos el estado del repositorio utilizando el comando:

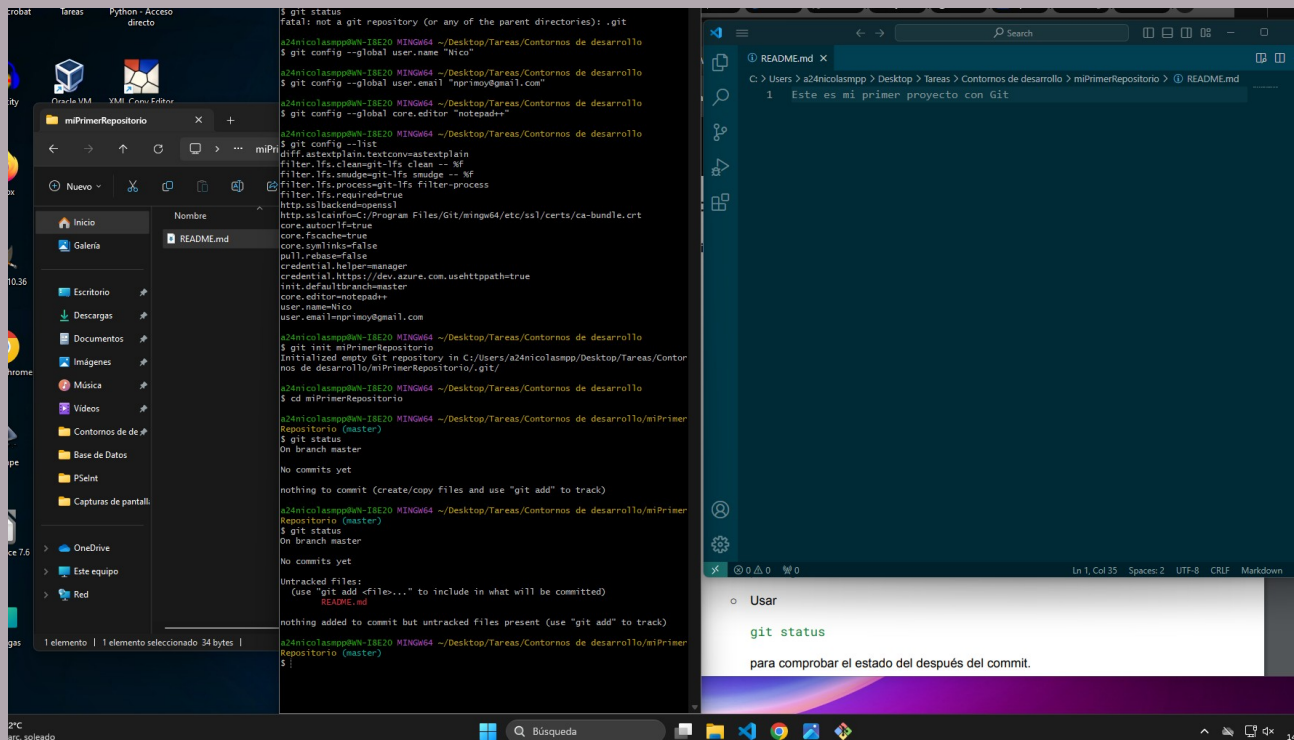
```
git status
```



## Primer commit

Para realizar nuestro primer commit creamos un archivo “README.md” en nuestra carpeta de git, con una descripción.

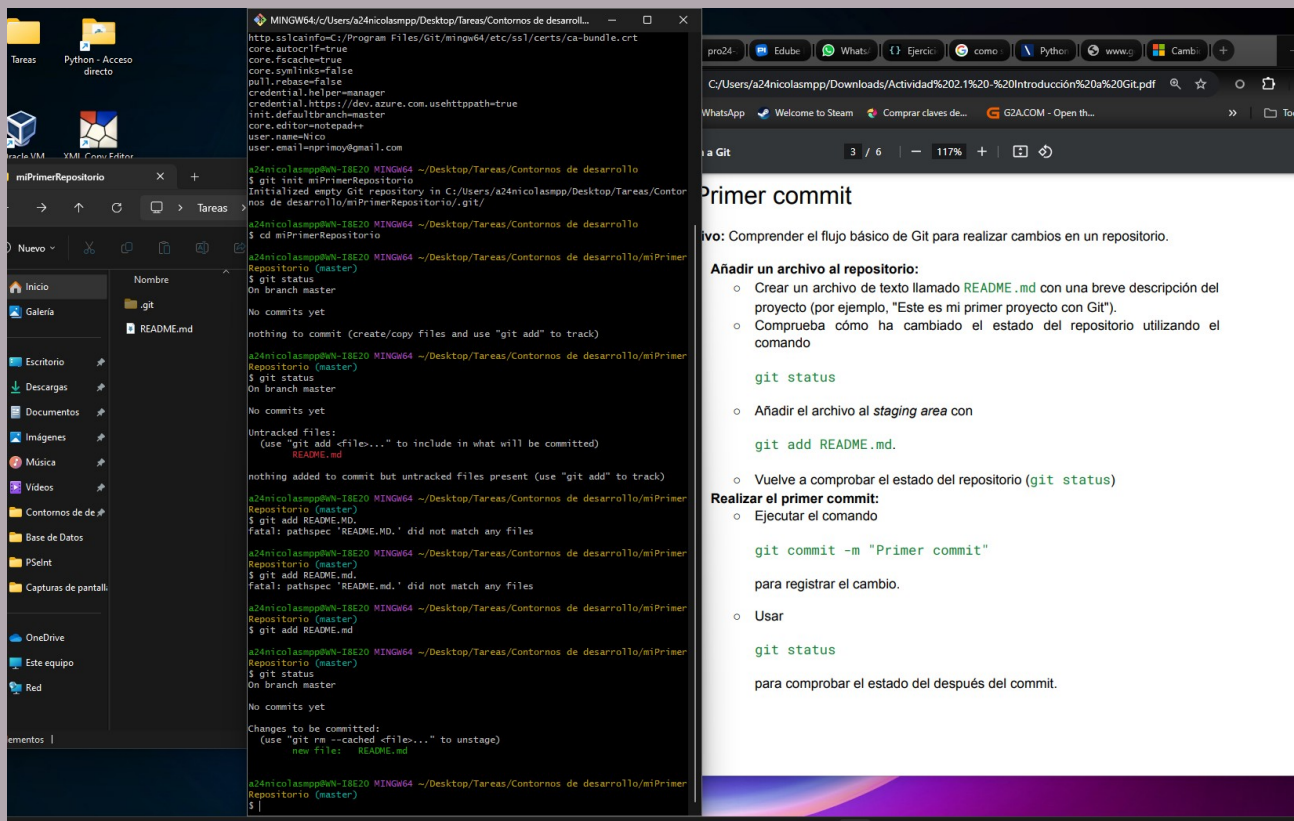
Verificamos el estado del repositorio utilizando: *git status*



Para añadir el archivo al staging area utilizamos el siguiente comando:

*git add README.md*

Y volvemos a verificar el estado con: *git status*

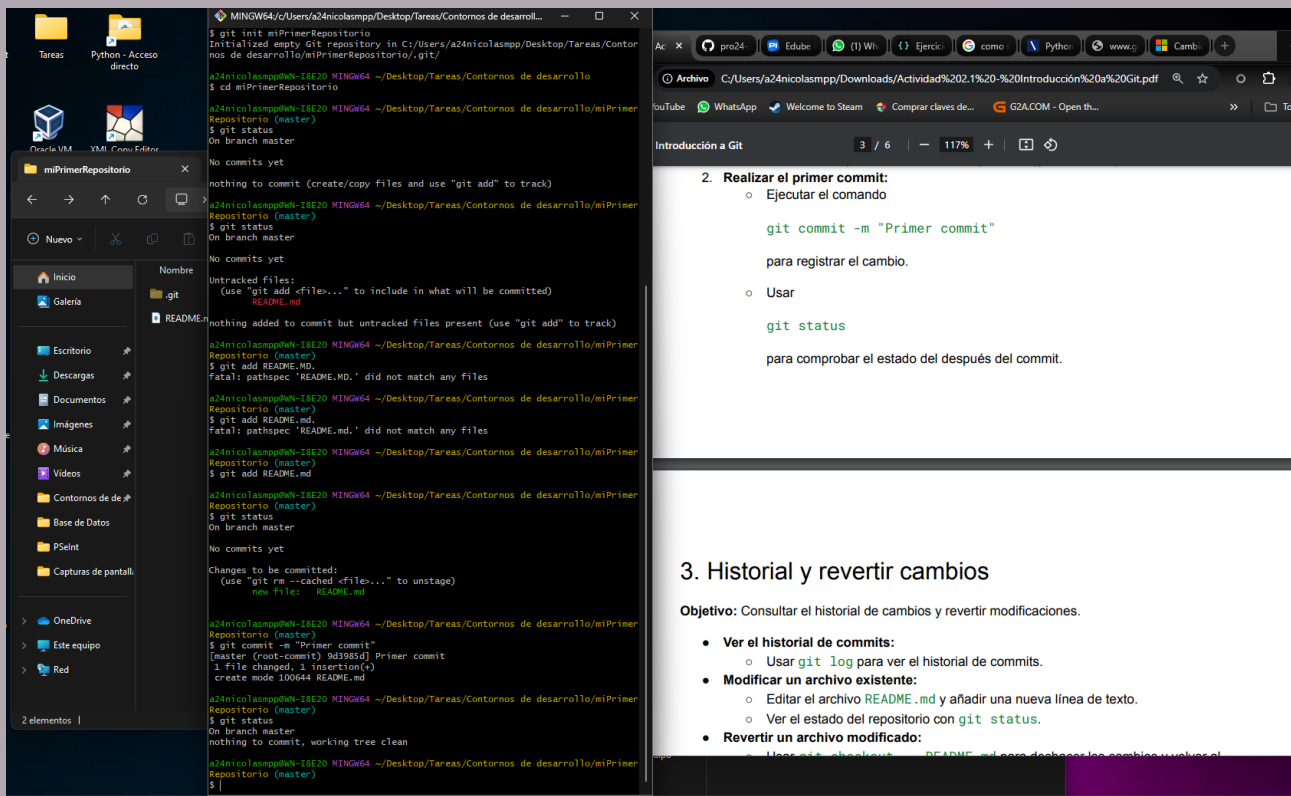
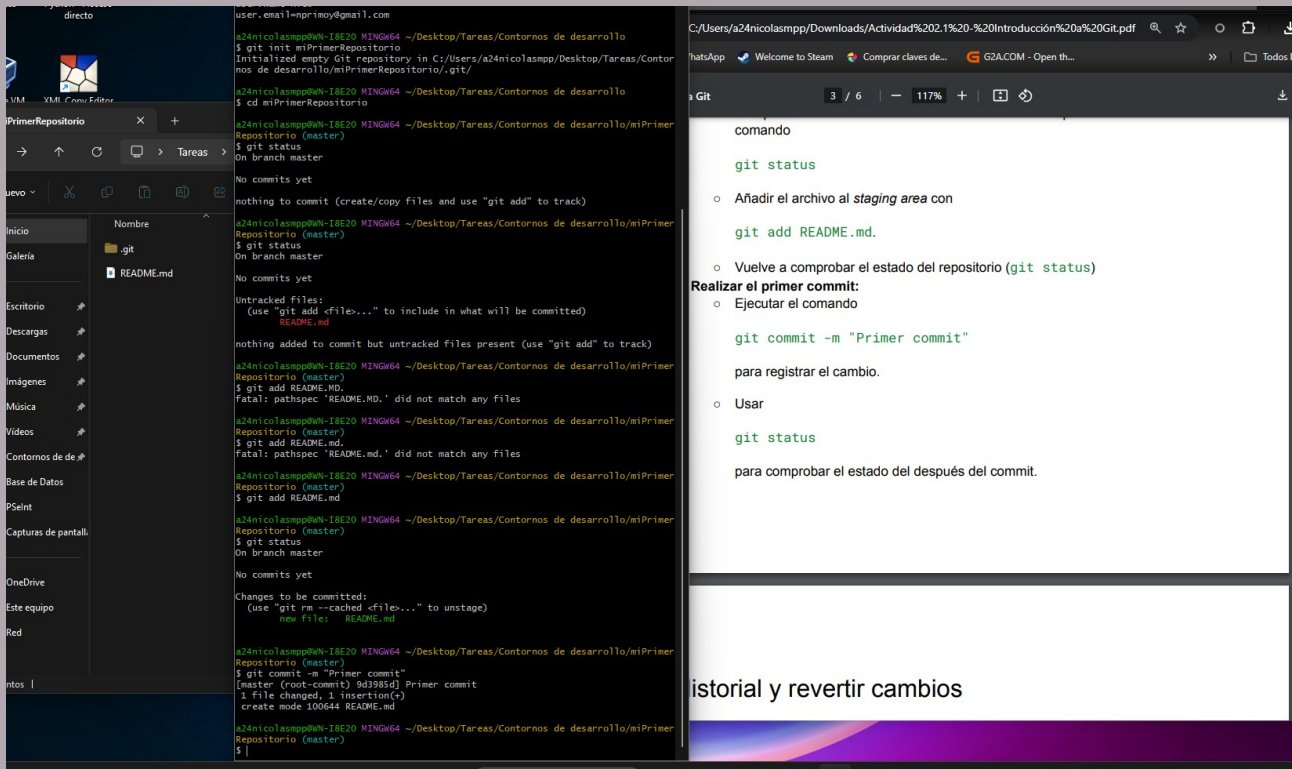


Realizamos nuestro primer commit y verificamos los cambios:

*git commit -m "Primer commit"*

*git status*





# Historial y revertir cambios

Comprobamos el historial de commits con el siguiente comando:

*git log*

```
No commits yet

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

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

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md
fatal: pathspec 'README.md.' did not match any files

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md
fatal: pathspec 'README.md.' did not match any files

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
On branch master

No commits yet

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

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git commit -m "Primer commit"
[master (root-commit) 9d3985d] Primer commit
 1 file changed, 1 insertion(+),
 create mode 100644 README.md

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git log
commit 9d3985da51165f8d0b50edb88864b2b58e98867c (HEAD -> master)
Author: Nico <nprimoy@gmail.com>
Date:   Mon Oct 14 19:18:08 2024 +0200

    Primer commit

a24nicolasmp@N-I8E20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$
```

## 3. Historial y revertir cambios

**Objetivo:** Consultar el historial de cambios y revertir modificaciones.

- **Ver el historial de commits:**
  - Usar `git log` para ver el historial de commits.
- **Modificar un archivo existente:**
  - Editar el archivo `README.md` y añadir una nueva línea de texto.
  - Ver el estado del repositorio con `git status`.
- **Revertir un archivo modificado:**
  - Usar `git checkout -- README.md` para deshacer los cambios y volver al estado anterior del archivo.
- **Hacer un nuevo commit:** Modificar el archivo de nuevo, añadirlo al staging con `git add` y realizar un nuevo commit.

## 4. Trabajo con ramas

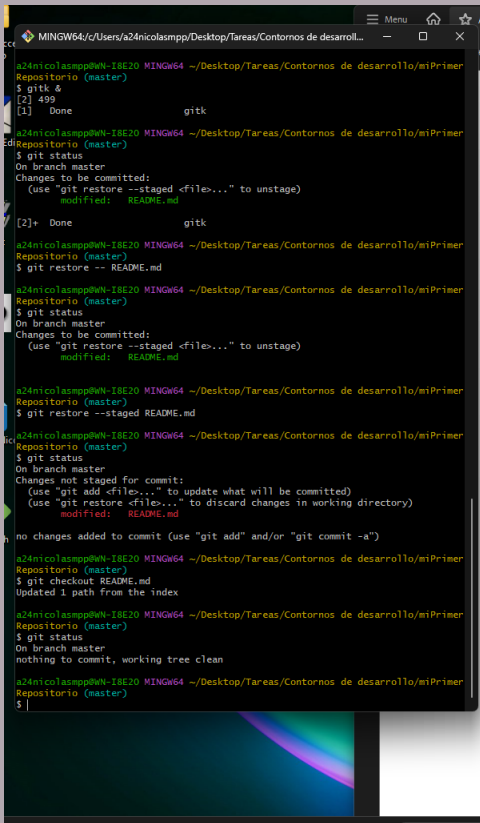
**Objetivo:** Crear y trabajar con ramas para desarrollar características de manera aislada.

**Ejercicios:**

- **Crear una rama nueva:**
  - Crear una rama llamada `nueva-funcionalidad` usando `git branch nueva-funcionalidad`.
  - Cambiar a la nueva rama con `git checkout nueva-funcionalidad`.

Realizamos una modificación en el archivo y verificamos el estado con *git status*

En la imagen se puede comprobar que el estado del README.md se ve en rojo ya que esta modificado en local pero no actualizado.



```
a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ gitk &
[1] Done gitk

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git restore --staged README.md

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git restore --staged README.md

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ 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:   README.md

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

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git checkout README.md
Updated 1 path from the index

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolaspp@WN-IE20 MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimerRepositorio (master)
$
```

### 3. Historial y revertir cambios

Objetivo: Consultar el historial de cambios y revertir modificaciones.

- Ver el historial de commits:
  - Usar `git log` para ver el historial de commits.
- Modificar un archivo existente:
  - Editar el archivo `README.md` y añadir una nueva línea de texto. **estoy x aca**
  - Ver el estado del repositorio con `git status`.
- Revertir un archivo modificado:
  - Usar `git checkout -- README.md` para deshacer los cambios y volver al estado anterior del archivo.
- Hacer un nuevo commit: Modificar el archivo de nuevo, añadirlo al staging con `git add` y realizar un nuevo commit.

### 4. Trabajo con ramas

Objetivo: Crear y trabajar con ramas para desarrollar características de manera aislada.

Ejercicios:

- Crear una rama nueva:
  - Crear una rama llamada `nueva-funcionalidad` usando `git branch nueva-funcionalidad`.
  - Cambiar a la nueva rama con `git checkout nueva-funcionalidad`.

Nota: se puede crear una rama y cambiarte a ella con un solo comando:

Agregamos la modificación con el comando *git add README.md*

Al estar actualizado vuelve a estar en verde



```
MINGW64/C:/Users/a24nicolas/Desktop/Tareas/Contornos de desarrollo...
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:   README.md

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

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git checkout README.md
Updated 1 path from the index

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md
fatal: pathspec 'README.md.' did not match any files

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git add README.md

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$ git status
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

a24nicolas@MINGW64 ~/Desktop/Tareas/Contornos de desarrollo/miPrimer
Repositorio (master)
$
```

Objetivo: Consultar el historial de cambios y revertir modificaciones.

- Ver el historial de commits:
  - Usar `git log` para ver el historial de commits.
- Modificar un archivo existente:
  - Editar el archivo `README.md` y añadir una nueva línea de texto. **estoy x aca**
  - Ver el estado del repositorio con `git status`.
- Revertir un archivo modificado:
  - Usar `git checkout -- README.md` para deshacer los cambios y volver al estado anterior del archivo.
- Hacer un nuevo commit: Modificar el archivo de nuevo, añadirlo al staging con `git add` y realizar un nuevo commit.

## 4. Trabajo con ramas

Objetivo: Crear y trabajar con ramas para desarrollar características de manera aislada.

Ejercicios:

- Crear una rama nueva:
  - Crear una rama llamada `nueva-funcionalidad` usando  
`git branch nueva-funcionalidad`.
  - Cambiar a la nueva rama con `git checkout nueva-funcionalidad`.  
  
Nota: se puede crear una rama y cambiarte a ella con un solo comando:  
`git checkout -b nueva-funcionalidad`.
- Hacer cambios en la nueva rama:

## Trabajo con ramas

Creamos una rama llamada “nueva-funcionalidad”, cambiamos a la rama y verificamos el estado usando los siguientes comandos:

*git branch nueva-funcionalidad*  
*git checkout nueva-funcionalidad*  
*git status*

Realizamos cambios en el archivo README.md, lo agregamos, subimos con un commit y verificamos estado:

```
git add README.md
git commit -m "commit rama"
git status
```

#### 4. Trabajo con ramas

**Objetivo:** Crear y trabajar con ramas para desarrollar características de manera aislada.

**Ejercicios:**

- **Crear una rama nueva:**
  - Crear una rama llamada nueva-funcionalidad usando  
`git branch nueva-funcionalidad.`
  - Cambiar a la nueva rama con `git checkout nueva-funcionalidad.`  
Nota: se puede crear una rama y cambiarte a ella con un solo comando:  
`git checkout -b nueva-funcionalidad.`
- **Hacer cambios en la nueva rama:**
  - Editar el archivo README.md para añadir una nueva sección.
  - Realizar un commit con los cambios.
- **Cambiar de rama:** Volver a la rama principal (main) con `git checkout main` y comprobar que los cambios no se ven en esta rama.
- **Unir ramas (merge):**
  - Unir la rama nueva-funcionalidad a main usando `git merge nueva-funcionalidad.`

#### 5. Resolución de conflictos

**Objetivo:** Gestionar conflictos al hacer merges entre ramas.

```
PrimerRepositorio (master)
$ git add
Nothing specified, nothing added.
hint: Maybe you wanted to say 'git add .' ?
hint: Disable this message with 'git config advice.addEmptyPathsSpec false'

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git add README.md
fatal: pathspec 'README.md' did not match any files

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git add README.md

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git commit^C

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git commit -m "segundo commit"
[master c1c470] segundo commit
1 file changed, 4 insertions(+)

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git branch nueva-funcionalidad

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (master)
$ git checkout nueva-funcionalidad
Switched to branch 'nueva-funcionalidad'

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (nueva-funcionalidad)
$ git add README.md

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (nueva-funcionalidad)
$ git commit -m "Commit rama"
[nueva-funcionalidad 95fec8] commit rama
1 file changed, 6 insertions(+), 1 deletion(-)

a24nicolas@ppw-18E20-MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/mi
PrimerRepositorio (nueva-funcionalidad)
$ |
```

Volvemos al master, verificamos el estado y realizamos un merge de la rama “nueva-funcionalidad”:

```
git checkout master
git status
git merge nueva-funcionalidad
```

## 4. Trabajo con ramas

**Objetivo:** Crear y trabajar con ramas para desarrollar características de manera aislada.

**Ejercicios:**

- **Crear una rama nueva:**
  - Crear una rama llamada `nueva-funcionalidad` usando  
`git branch nueva-funcionalidad.`
  - Cambiar a la nueva rama con `git checkout nueva-funcionalidad`.  
  
Nota: se puede crear una rama y cambiarte a ella con un solo comando:  
`git checkout -b nueva-funcionalidad.`
- **Hacer cambios en la nueva rama:**
  - Editar el archivo `README.md` para añadir una nueva sección.
  - Realizar un commit con los cambios.
- **Cambiar de rama:** Volver a la rama principal (`main`) con `git checkout main` y comprobar que los cambios no se ven en esta rama.
- **Unir ramas (merge):**
  - Unir la rama `nueva-funcionalidad` a `main` usando `git merge nueva-funcionalidad`.

## 5. Resolución de conflictos

**Objetivo:** Gestionar conflictos al hacer merges entre ramas.

```
a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git commit -m "segundo commit"
(master c1ce470) segundo commit
1 file changed, 4 insertions(+)

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git branch nueva-funcionalidad

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git checkout nueva-funcionalidad
Switched to branch 'nueva-funcionalidad'

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (nueva-funcionalidad)
$ git add README.md

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (nueva-funcionalidad)
$ git commit -m "Commit rama"
[nueva-funcionalidad 95ee5c8] commit rama
1 file changed, 6 insertions(+), 1 deletion(-)

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (nueva-funcionalidad)
$ git checkout master
Switched to branch 'master'

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git status
On branch master
nothing to commit, working tree clean

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git merge nueva-funcionalidad
Updating c1ce470..95ee5c8
Fast-forward
 README.md | 7 ++++++
 1 file changed, 6 insertions(+), 1 deletion(-)

a24nicolaspp@WN-IBE20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$
```

Buscando el error realizamos un commit en el master volvemos a cambiar a la rama y realizamos otro commit en la misma linea que el commit en el master:

```
git add README.md
git commit -m "tercer commit master"
git checkout nueva-funcionalidad
git add README.md
git commit -m "commit rama2"
```

11

Ejercicios:

- **Generar un conflicto:**
  - En la rama `main`, modificar una línea en el archivo `README.md` y hacer un commit.
  - Cambiar a la rama `nueva-funcionalidad`, modificar la misma línea de texto y hacer un commit.
- **Intentar fusionar las ramas:**
  - Al intentar fusionar `nueva-funcionalidad` con `main`, surgirá un conflicto.
- **Resolver el conflicto:**
  - Usar el editor de texto para resolver el conflicto manualmente y después hacer el commit de la resolución. En el commit resultante, deben mantener las dos modificaciones del archivo `README.md`, primero la realizada en la rama `main` y luego la realizada en la rama `nueva-funcionalidad`.

## 6. Visualización de las ramas

**Objetivo:** utilizar la utilidad `gitk` para visualizar las ramas.

**Ejercicios:**

- **Utilización de `gitk`**
  - Ejecuta el comando `gitk &` en la terminal de `git`. Se abrirá una herramienta visual con el siguiente aspecto

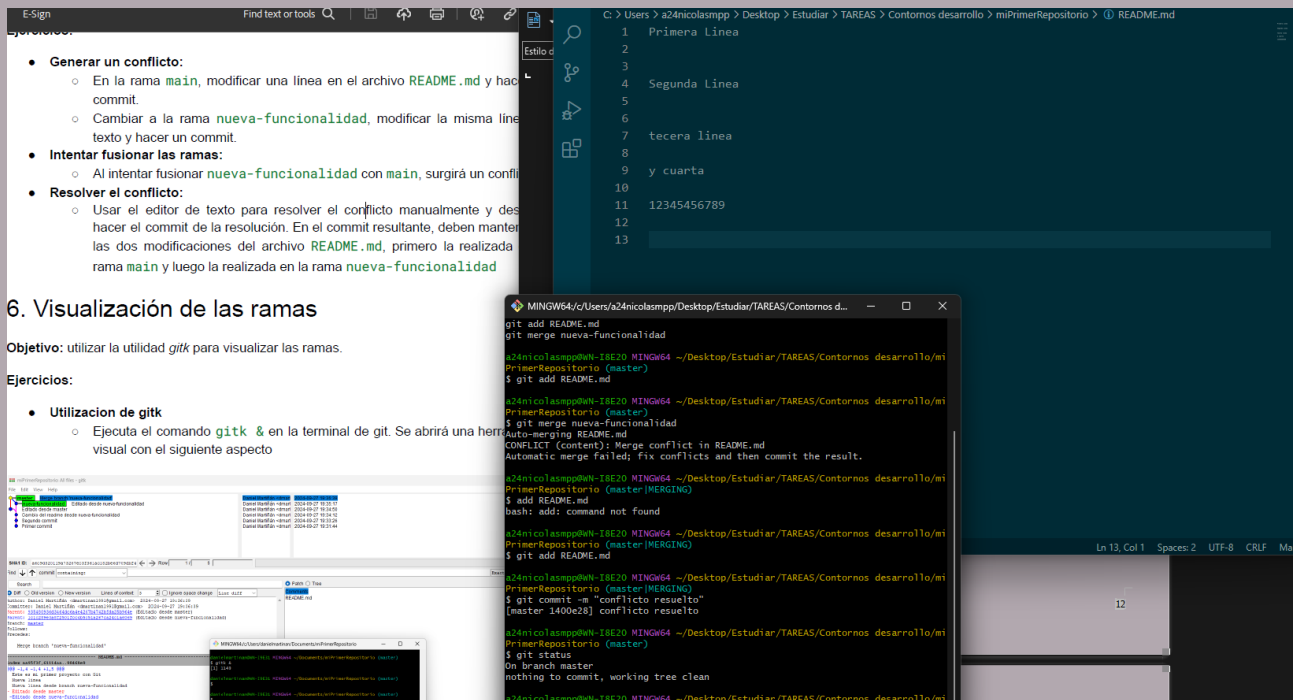
```

a24nicolasmp@WN-18E20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (nueva-funcionalidad)
$ gitk
a24nicolasmp@WN-18E20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (nueva-funcionalidad)
$ git checkout master
Switched to branch 'master'
a24nicolasmp@WN-18E20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ gitk
git add README.md
git merge nueva-funcionalidad
a24nicolasmp@WN-18E20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master)
$ git merge nueva-funcionalidad
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
a24nicolasmp@WN-18E20 MINGW64 ~/Desktop/Estudiar/TAREAS/Contornos desarrollo/miPrimerRepositorio (master) (MERGING)
$

```

Resolvemos el conflicto desde el editor de texto y volvemos a subir el archivo:

*git add README.md*  
*git commit -m “conflicto resuelto”*  
*git status*

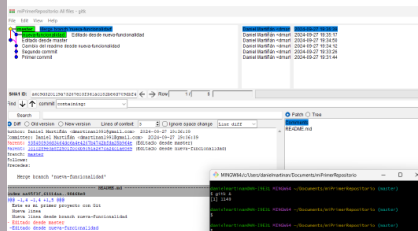


## 6. Visualización de las ramas

Objetivo: utilizar la utilidad `gitk` para visualizar las ramas.

Ejercicios:

- Utilización de gitk**
  - Ejecuta el comando `gitk &` en la terminal de git. Se abrirá una herramienta visual con el siguiente aspecto



Utilizamos el comando *gitk* para visualizar las modificaciones que se realizaron en la reposición.



