

Práctica 1: Repositorio GITHUB Desktop

Primero entraremos a la carpeta Practica1 y inicializaremos un repositorio, con el comando git Branch -M renombramos la rama master a main, luego creamos un nuevo archivo usando el comando cat.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop
$ cd Practica1/

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1
$ git init
Initialized empty Git repository in C:/Users/user/Desktop/Practica1/.git/

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (master)
$ git branch -M main

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (main)
$ ls
bin/  src/

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (main)
$ cat > archivo1.txt
hola
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (main)
$ ls
archivo1.txt  bin/  src/
```

Con el comando git add . añadimos todos los archivos a la etapa stage.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (main)
$ git add .

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practica1 (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .classpath
    new file:   .project
    new file:   .settings/org.eclipse.core.resources.prefs
    new file:   .settings/org.eclipse.jdt.core.prefs
    new file:   archivo1.txt
    new file:   bin/Main.class
    new file:   src/Main.java
```

Usando el comando `git commit -m "commit"` hacemos commit de los archivos que esten en la etapa stage.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git commit -m "Commit inicial"
[main (root-commit) be63021] Commit inicial
7 files changed, 53 insertions(+)
create mode 100644 .classpath
create mode 100644 .project
create mode 100644 .settings/org.eclipse.core.resources.prefs
create mode 100644 .settings/org.eclipse.jdt.core.prefs
create mode 100644 archivo1.txt
create mode 100644 bin/Main.class
create mode 100644 src/Main.java

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status
On branch main
nothing to commit, working tree clean
```

Creamos otro archivo con `cat` y lo añadimos a la etapa stage.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ cat > archivo2.txt
123
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ ls
archivo1.txt  archivo2.txt  bin/  src/

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    archivo2.txt

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

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git add archivo2.txt
```

Con el comando `git status` comprobamos el estado actual del repositorio.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   archivo2.txt
```

Con el comando `git restore --staged archivo2.txt` quitamos el archivo de la etapa de stage.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git restore --staged archivo2.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        archivo2.txt

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

Volvemos a añadir el archivo a la etapa stage y hacemos commit.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git add archivo2.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status -s
A  archivo2.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git commit -m "hacemos commit del archivo2"
[main b071696] hacemos commit del archivo2
1 file changed, 1 insertion(+)
create mode 100644 archivo2.txt
```

Con el comando `git checkout -b ramab` creamos otra rama.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git checkout -b ramab
Switched to a new branch 'ramab'
```

Usando el editor de texto nano editaremos el archivo1 para crear un conflicto entre ramas.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ nano archivo1.txt
```

```
GNU nano 7.1          archivo1.txt          Modifie
hola
que
tal
```

Añadimos el archivo a la etapa stage y hacemos commit.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git status
On branch ramab
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:   archivo1.txt

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

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git add archivo1.txt
warning: in the working copy of 'archivo1.txt', LF will be replaced by CRLF the
next time Git touches it

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git status -s
M  archivo1.txt
```

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git commit -m "se edito el archivo1"
[ramab 4ed5a8a] se edito el archivo1
1 file changed, 3 insertions(+), 1 deletion(-)
```

Nos cambiamos a la rama main.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git checkout main
Switched to branch 'main'

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ nano archivo1.txt
```

Con el editor nano editamos el archivo1, lo añadimos a la etapa stage y hacemos commit.

GNU nano 7.1	archivo1.txt	Modified
hola adios		

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status -s
M archivo1.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git add archivo1.txt
warning: in the working copy of 'archivo1.txt', LF will be replaced by CRLF the
next time Git touches it

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git status -s
M archivo1.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git commit -m "se agrego adios al archivo1"
[main cd931d6] se agrego adios al archivo1
1 file changed, 2 insertions(+), 1 deletion(-)
```

Nos cambiamos de nuevo a la ramab y probamos de hacer merge, y como podemos ver se ha creado un conflicto en el archivo1.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (main)
$ git checkout ramab
Switched to branch 'ramab'

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git merge main
Auto-merging archivo1.txt
CONFLICT (content): Merge conflict in archivo1.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Si usamos un editor de texto veremos que nos muestra debajo de HEAD el archivo tal cual como lo hemos editado en la ramab y debajo de ===== los cambios desde la rama main.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab|MERGING)
$ nano archivo1.txt
```

```
GNU nano 7.1 archivo1.txt
hola
<<<<<<< HEAD
que
tal
=====
adios
>>>>>>> main
```

Borramos las líneas que no queramos dejando el archivo como queramos y guardamos.

```
GNU nano 7.1 archivo1.txt Modified
hola
que
tal
adios
```

Añadimos el archivo a stage, hacemos commit y ahora si probamos a hacer merge veremos que los conflictos ya se han resuelto.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab|MERGING)
$ git add archivo1.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab|MERGING)
$ git commit -m "Conflictos resueltos"
[ramab c32f82b] Conflictos resueltos

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git merge main
Already up to date.
```

Con el comando git log --oneline podemos ver el historial de commits en una línea.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git log --oneline
c32f82b (HEAD -> ramab) Conflictos resueltos
cd931d6 (main) se agrego adios al archivo1
4ed5a8a se edito el archivo1
ddfcdbf hacemos commit del archivo2
739a00b Commit inicial
```

Creamos otra rama llamada ramac.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git checkout -b ramac
Switched to a new branch 'ramac'
```

Nos cambiamos a la ramab, creamos otro archivo, lo añadimos a stage y hacemos commit.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramac)
$ git checkout ramab
Switched to branch 'ramab'

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ cat > archivo3.txt
prueba
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git add archivo3.txt

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramab)
$ git commit -m "agregamos el archivo3"
[ramab f57e2d7] agregamos el archivo3
1 file changed, 1 insertion(+)
create mode 100644 archivo3.txt
```

Nos cambiamos a la ramac y hacemos merge de la ramab.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical1 (ramab)
$ git checkout ramac
Switched to branch 'ramac'

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical1 (ramac)
$ git merge ramab
Updating c32f82b..f57e2d7
Fast-forward
 archivo3.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 archivo3.txt
```

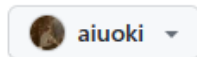

Vamos a github y creamos un nuevo repositorio publico.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner *





Repository name *

/ Practica1

✔ Practica1 is available.

Great repository names are short and memorable. Need inspiration? How about [effective-pancake](#) ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **None** ▼

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None** ▼

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

Create repository

Indicamos el servidor donde vamos a subir los cambios. Y con el comando git push -u origin main subimos los cambios al servidor.

```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramac)
$ git remote add origin https://github.com/aiuoki/Practical.git

user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramac)
$ git push -u origin main
Enumerating objects: 18, done.
Counting objects: 100% (18/18), done.
Delta compression using up to 6 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (18/18), 2.14 KiB | 729.00 KiB/s, done.
Total 18 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/aiuoki/Practical.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```


Subimos los cambios de la ramab.



```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramac)
$ git push -u origin ramab
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 6 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 827 bytes | 827.00 KiB/s, done.
Total 9 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 1 local object.
remote:
remote: Create a pull request for 'ramab' on GitHub by visiting:
remote:   https://github.com/aiuoki/Practical/pull/new/ramab
remote:
To https://github.com/aiuoki/Practical.git
 * [new branch]      ramab -> ramab
branch 'ramab' set up to track 'origin/ramab'.
```


Y también los cambios de la ramac.


```
user@DESKTOP-3CCBA44 MINGW64 ~/Desktop/Practical (ramac)
$ git push -u origin ramac
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'ramac' on GitHub by visiting:
remote:   https://github.com/aiuoki/Practical/pull/new/ramac
remote:
To https://github.com/aiuoki/Practical.git
 * [new branch]      ramac -> ramac
branch 'ramac' set up to track 'origin/ramac'.
```




Finalmente si vamos a nuestro repositorio de github veremos que los cambios se han subido correctamente con las 3 ramas creadas.



 **Practica1** Public








 Pin  Unwatch 1

 **ramab** had recent pushes 1 minute ago [Compare & pull request](#)

 **ramac** had recent pushes 1 minute ago [Compare & pull request](#)

 **main**  **3** branches  **0** tags [Go to file](#) [Add file](#) [Code](#)

 **aiuoki** se agrego adios al archivo1 cd931d6 17 minutes ago  **3** commits

 .settings	Commit inicial	39 minutes ago
 bin	Commit inicial	39 minutes ago
 src	Commit inicial	39 minutes ago
 .classpath	Commit inicial	39 minutes ago
 .project	Commit inicial	39 minutes ago
 archivo1.txt	se agrego adios al archivo1	17 minutes ago
 archivo2.txt	hacemos commit del archivo2	38 minutes ago

Help people interested in this repository understand your project by adding a README. [Add a README](#)

En la misma carpeta Practica1 adjunto una guía creada por mi con los comandos esenciales de GIT.