Introducción

A continuación, se explican una serie de comandos utilizados en la primera práctica.

¿Para qué se usa el *git fork*?

Mediante el *fork* se ha creado un nuevo repositorio igual que el original (*main*, en este caso). Es decir, hemos creado una bifurcación del código donde podremos editar el código y realizar distintos cambios para, más tarde, proponerlos al repositorio remoto (mediante un *pull request*). El repositorio bifurcado es un nuevo repositorio que será completamente independiente del repositorio original del que se hizo el *fork*.

¿Para qué se usa el *git clone*?

Este comando ha servido para clonar el repositorio remoto de la *url* dada. De esta manera, se ha creado una copia local en *Codespace* para poder acceder y editar los archivos que contiene. Estos cambios se podrán sincronizar con el repositorio remoto más adelante (mediante *commits*).

```
@MariaBegara ➜/workspaces $ ls
ci-cd
@MariaBegara ➜/workspaces $ git clone https://github.com/MariaBegara/p1
Cloning into 'p1'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 5 (from 1)
Receiving objects: 100% (6/6), done.
```

¿Para qué se usa el *git status*?

Nos permite visualizar las discrepancias que existen entre el directorio de trabajo y del área del entorno de ensayo. Se ha empleado para verificar si la rama estaba o no actualizada con el repositorio; se ha comprobado si era necesario hacer un *add* de los cambios para más tarde hacer un *commit*.

```
@MariaBegara ➜/workspaces/p1 (main) $ git status
On branch main
Your branch is up to date with 'origin/main'.

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

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

¿Para qué se utiliza *git add*?

Permite añadir los cambios que se quieren actualizar al repositorio desde el área del entorno de ensayo. Se puede seleccionar un único cambio o todos (*git add* .).

```
@MariaBegara ➜/workspaces/p1 (main) $ git add .
```

¿Para qué se utiliza git commit?

Se emplea para actualizar los cambios. Para añadir una etiqueta que defina dicho cambio, se debe añadir "-m mensaje".

```
@MariaBegara ➜/workspaces/p1 (main) $ git commit -m "añadir git.txt"
[main a86f493] añadir git.txt
 1 file changed, 42 insertions(+)
 create mode 100644 git.txt
```

¿Para qué se utiliza git push?

Permite subir los cambios actualizados con un *commit* al repositorio remoto.

```
@MariaBegara ➜/workspaces/p1 (main) $ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.16 KiB | 1.16 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MariaBegara/p1
   07720b5..a86f493  main -> main
```

¿Para qué se utiliza git checkout?

Se utiliza para poder ir de una rama a otra del repositorio. En este caso se ha vuelto a la rama *main* para, posteriormente, hacer un *pull request*.

```
@MariaBegara ➜/workspaces/ci-cd/src (feat/add-body) $ git chec
kout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Estos pasos también pueden realizarse para hacer un único cambio en el repositorio: “Añadir el archivo git.pdf”.

```
@MariaBegara ➜/workspaces/p1 (main) $ git add git.pdf
@MariaBegara ➜/workspaces/p1 (main) $ git commit -m "añadir git.pdf"
[main f738f96] añadir git.pdf
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 git.pdf
@MariaBegara ➜/workspaces/p1 (main) $ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 108.73 KiB | 21.75 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MariaBegara/p1
   492dffb..f738f96  main -> main
```