PRÁCTICA 1 PAT

Demostración descarga de programas:

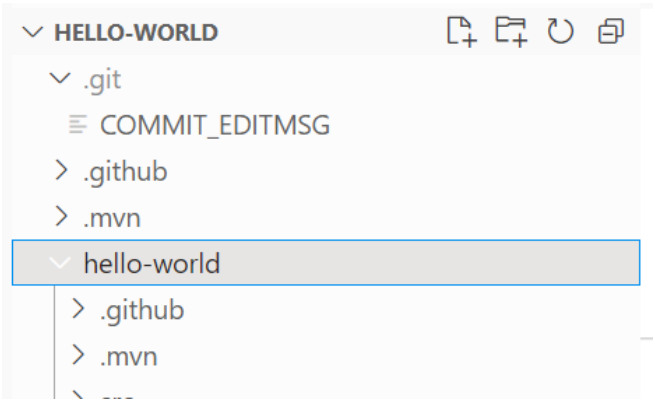
```
C:\Users\beatr>java -version
java version "17.0.2" 2022-01-18 LTS
Java(TM) SE Runtime Environment (build 17.0.2+8-LTS-86)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.2+8-LTS-86, mixed mode, sharing)

C:\Users\beatr>mvn -v
Apache Maven 3.8.4 (9b656c72d54e5bacbed989b64718c159fe39b537)
Maven home: C:\Program Files\Maven\apache-maven-3.8.4
Java version: 17.0.2, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17.0.2
Default locale: es_ES, platform encoding: Cp1252
OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"
```

```
PS C:\Users\beatr> choco -v
0.12.0
PS C:\Users\beatr>
```

Comandos git-hub:

1. **CLONE:**

Git clone sirve para hacer una copia o un clon de un repositorio que ya existe en un directorio nuevo.




```
gitpod /workspace/hello-world $ git clone https://github.com/gitt-3-pat/hello-world
Cloning into 'hello-world'...
remote: Enumerating objects: 38, done.
remote: Counting objects: 100% (38/38), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 38 (delta 1), reused 31 (delta 0), pack-reused 0
Receiving objects: 100% (38/38), 58.97 KiB | 6.55 MiB/s, done.
Resolving deltas: 100% (1/1), done.
gitpod /workspace/hello-world $
```

2. **STATUS:**

Git status comprueba las diferencias entre nuestros archivos de github. Comrpueba si están actualizados con las últimas modificaciones.

```
gitpod /workspace/hello-world $ 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)
        hello-world/

nothing added to commit but untracked files present (use "git add" to track)
gitpod /workspace/hello-world $
```

Si por ejemplo hiciéramos algunas modificaciones en el código y pusiéramos este comando, nos avisaría de que no está “up to date”

```
gitpod /workspace/hello-world $ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

3. **ADD:**

Git add lo utilizaremos cuando queramos actualizar en el entorno de ensayo algún cambio hecho en nuestro directorio de trabajo. Indicamos que queremos incluir actualizaciones en un archivo en concreto. Si escribimos git add . será que queremos actualizar todos los archivos.

```
gitpod /workspace/hello-world $ git add .
warning: adding embedded git repository: hello-world
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:
hint:   git submodule add <url> hello-world
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint:   git rm --cached hello-world
hint:
hint: See "git help submodule" for more information.
```

4. **COMMIT:**

Commit identifica los cambios hechos en el ambiente de trabajo. Le ponemos un comentario indicando qué es lo que hemos modificado

```
gitpod /workspace/hello-world $ git commit -m "hola que tal"
[main b56e450] hola que tal
 1 file changed, 1 insertion(+)
 create mode 160000 hello-world
gitpod /workspace/hello-world $
```

EXTRA: si queremos cambiar el mensaje que habíamos enviado en el anterior git commit, solo tenemos que escribir

```
git commit --amend -m "mensaje nuevo”
```

5. **PUSH:**

El comando git push sirve para cargar contenido en un repositorio remoto.

```
gitpod /workspace/hello-world $ git push
Everything up-to-date
gitpod /workspace/hello-world $
```

6. **CHECKOUT**

Sirve para cambiar entre ramas del repositorio, también vale para restaurar los archivos del campo de trabajo. En concreto, para eliminar los cambios locales que no han sido añadidos al área de preparación

```
gitpod /workspace/hello-world $ git checkout
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
```

7. GIT REVERT

Sirve para deshacer algún cambio que hayamos hecho. Para saber qué commit queremos deshacer podemos utilizar este comando:

```
gitpod /workspace/hello-world $ git log --oneline
48fe276 (HEAD -> main, upstream/main, origin/main, origin/HEAD) Merge pull request #1 from gitt-3-pat/
feature/1
5b68377 Primera iteracion
5038239 Initial commit
```

Luego, utilizando git revert + código de commit, podremos borrar el que queramos.