PRÁCTICA 1 PAT

Entorno de Desarrollo

Guadalupe Martínez Blanco
3º GITT-B

Link repositorio Github: https://github.com/gmblanco/ci-cd

1.- Explicación de comandos:

- **Clone:**

  El comando $git\ clone$ se ha utilizado para realizar una copia local de un repositorio Git existente (remoto). Al clonar un repositorio, se crea una copia completa del historial de versiones, ramas y archivos en el ordenador local que ejecuta el comando.

  El comando $git\ clone\ https://github.com/gitt-3-pat/p1$ se ha empleado para realizar una copia completa del repositorio existente $gitt-3-pat/p1$ en el ordenador local.

```
@gmblanco →/workspaces/ci-cd-p1 (main) $ cd /workspaces
-pat/p1
ls p1
cat p1/README.md
@gmblanco →/workspaces $ git clone https://github.com/gitt-3-pat/p1
fatal: destination path 'p1' already exists and is not an empty directory.
@gmblanco →/workspaces $ ls p1
README.md
```

```
@gmblanco →/workspaces $ cat p1/README.md
# Práctica 1

Un repositorio para empezar a usar [git](https://git-scm.com/) y Github

## ¿Como probar en la nube?

[Github-Codespaces](https://github.com/features/codespaces)

## Comandos git básicos

```
git clone https://github.com/gitt-3-pat/p1
git status
git add .
git commit -m "TU MENSAJE"
git push

git checkout -b feature/1
git checkout main
```

¿Cómo escribir un README.md con formato?
```

Ilustración 1.- Uso comando git clone

- **Fork**:
  El $fork$ sirve para bifurcar en un nuevo repositorio que comparte la configuración y el código contenido del repositorio original. Es útil para proponer cambios al repositorio original en proyectos con varios colaboradores.
  La rama tiene el mismo contenido que el main pero permite que la información no se machaque de forma que cada persona trabaja en su rama.

```
@gmblanco →/workspaces $ cd /workspaces
co/p1-fork
ls p1-fork
cat p1-fork/README.md
@gmblanco →/workspaces $ git clone https://github.com/gmblanco/p1-for
k
Cloning into 'p1-fork'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (6/6), done.
@gmblanco →/workspaces $ ls p1-fork
README.md
@gmblanco →/workspaces $ cat p1-fork/README.md
# Práctica 1

Un repositorio para empezar a usar [git](https://git-scm.com/) y Githu
b

## ¿Como probar en la nube?

[Github-Codespaces](https://github.com/features/codespaces)

## Comandos git básicos

```
git clone https://github.com/gitt-3-pat/p1
git status
git add .
git commit -m "TU MENSAJE"
git push

git checkout -b feature/1
git checkout main
```

¿Cómo escribir un README.md con formato?
```

Ilustración 2.- Uso ramificaciones y fork

- **Status**:
  El comando $git\ status$ se utiliza para copiar un repositorio existente junto con todo su historial de versiones desde un repositorio remoto al ordenador local donde se ejecuta el comando.

  Al ejecutar este comando, indica que nos encontramos en la rama principal $main$ y que la rama creada anteriormente está actualizada respecto a la rama principal en el repositorio remoto.

```
@gmblanco →/workspaces/ci-cd-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)
        src/

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

Ilustración 3.- Uso comando git status

- **Add:**
  El comando $git\ add$ se utiliza para agregar los cambios realizados a un área temporal llamada $staging\ area$. Prepara los cambios para ser incluidos en la próxima confirmación.
  En este caso se añaden al directorio actual.

```
@gmblanco →/workspaces/ci-cd-p1 (main) $ git add .
```

- **Commit:**
  El comando $git\ commit$ confirma los cambios del staging área y registra un nuevo commit de los cambios. Se utiliza junto con un mensaje que describe los cambios realizados.

```
@gmblanco →/workspaces/ci-cd-p1 (main) $ git commit -m "feat: homepage"
[main e575202] feat: homepage
 1 file changed, 5 insertions(+)
 create mode 100644 src/index.html
```

Ilustración 4.- Uso comando git commit

Confirma los cambios en el repositorio local y registra un nuevo commit con el mensaje descriptivo "feat: homepage".

- **Push:**
  El comando $git\ push$ se utiliza para enviar los commits locales a un repositorio remoto. Además, actualiza el historial en el servidor con los cambios realizados en local.

```
@gmblanco →/workspaces/ci-cd-p1 (main) $ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 384 bytes | 384.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/gmblanco/ci-cd-p1
   e3620c5..e575202  main -> main
```

Showing **1 changed file** with **1 addition** and **1 deletion**.

2 src/index.html

```
...  ...   @@ -1,5 +1,5 @@
1    1     <!DOCTYPE html>
2    2     <html>
3    3     <head><title></title></head>
4    -     <body></body>
     4   + <body><body><p>Hola gmblanco </p></body></body>
5    5     </html>
```

Ilustración 5.- Uso git push e historial de cambios

Este comando se emplea para enviar los commits locales de la rama principal $main$ al repositorio remoto $origin$. De esta forma, se actualiza el historial remoto con los cambios realizados.

- **Checkout:**

El comando $git\ checkout$ se utiliza para cambiar entre ramas o deshacer cambios en el directorio donde se está trabajando. Además, puede utilizarse para crear nuevas ramas, cambiar a una rama existente o deshacer cambios en archivos específicos.

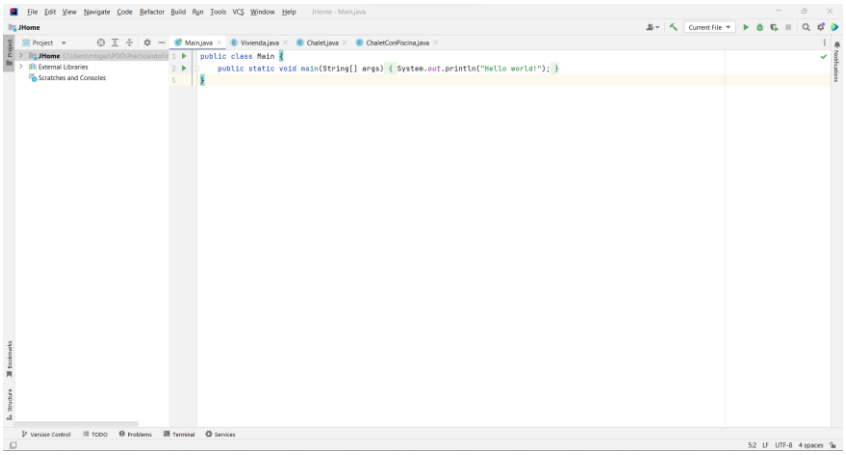
Con el comando $git\ checkout - b$ se crea una nueva rama $feat/add - body$ y se cambia a ella directamente. Como esta rama no existía previamente, este comando permite crearla si no existe.

```
@gmblanco ➜/workspaces/ci-cd-p1 (main) $ git checkout -b feat/add-body
Switched to a new branch 'feat/add-body'
```

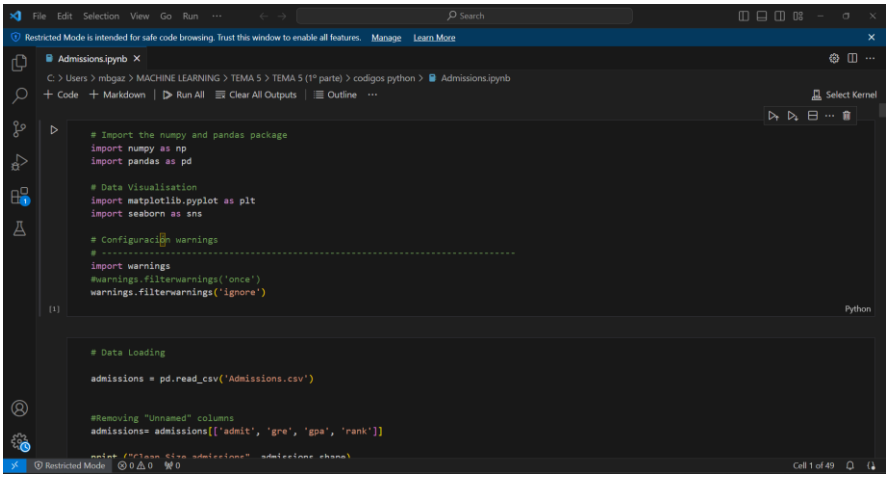
2.- Entorno de desarrollo Java:

- EDITOR DE CÓDIGO FUENTE:

  IntelliJ: es un entorno de desarrollo integrado para el desarrollo de programas informáticos. Se empleará principalmente para el leguaje de programación Java durante la asignatura.



Visual Studio Code: Visual Studio Code es un programa diseñado para editar el código fuente en entornos como Windows, Linux, macOS y la web. Ofrece funciones como depuración, gestión integrada de Git, destacado de la sintaxis y la capacidad de reorganizar y mejorar la estructura del código, entre otras funcionalidades.



Java 17 JDK: Se ha descargado e instalado correctamente la versión JDK 17 de java en el ordenador

```
C:\Users\mbgaz>java --version
java 17.0.10 2024-01-16 LTS
Java(TM) SE Runtime Environment (build 17.0.10+11-LTS-240)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.10+11-LTS-240, mixed mode, sharing)
```

Maven: Se ha descargado e instalado Maven

```
C:\Users\mbgaz>mvn -version
Apache Maven 3.9.6 (bc0240f3c744dd6b6ec2920b3cd08dcc295161ae)
Maven home: C:\Users\mbgaz\Downloads\apache-maven-3.9.6-bin\apache-maven-3.9.6
Java version: 17.0.10, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17
Default locale: es_ES, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
```