EDUCA2 – *¡Domina las Matemáticas y Programación!*

Curso de Git

Preparador: Alexanyer Naranjo

Guía Teórica N°2

Lista de Comandos Básicos

A continuación, se indicará un listado de comandos básicos y más utilizados de Git, su funcionalidad y un respectivo breve ejemplo para observar cómo ejecutar la instrucción indicada:

- **git config:** Utilizado para establecer una configuración específica de usuario, como sería el caso de nombre de usuario, email, algoritmo de preferencia, entre otros.

```
git config --global user.name “educa2”
git config --global user.email “educa2ucv@gmail.com”
```

- **git init:** Permite la creación de un nuevo repositorio Git.

```
git init
```

- **git add:** Nos permite agregar archivos al Área de Ensayo (*Staging Area*).

```
git add <archivo>
git add .
```

- **git rm:** Remover archivo del Área de Ensayo (Staging Area).

```
git rm --cached <archivo>
```

- **git clone:** Se utiliza con el propósito de realizar una clonación (copia) de un respectivo repositorio.

```
git clone <url>
```

- **git commit:** Hace *commit* a los archivos que indiquemos, de esta manera quedan guardados nuestras modificaciones.

```
git commit -m “Texto que identifique el motivo del commit”
```

- **git status:** Muestra una lista de los archivos que se han cambiado junto con los archivos que están por ser añadidos o comprometidos.

```
git status
```

- **git push:** Envía los cambios que se han realizado en la rama principal de los repositorios remotos que están asociados con el directorio que está trabajando.

```
git push <origin> <branch>
```

- **git checkout:** Utilizado para la creación de ramas o cambias entre las ramas ya creadas. Por ejemplo, el siguiente comando crea una nueva rama y se cambia a ella.

```
git checkout <branch>
```

- **git remote:** Esta instrucción se realiza para conectar a un repositorio remoto.

```
git remote add origin <url>
```

- **git branch:** Su uso principal se encuentra en la de listar, crear y borrar ramas.

```
git branch
git branch <nuevo>
git branch -d <nombreRama>
```

- **git pull:** Busca los cambios realizados y actualiza el repositorio.

```
git pull origin <branch>
```

- **git merge:** Permite fusionar una rama con otra rama activa.

```
git merge <branch>
```

- **git diff:** Nos permite ver las diferencias entre el último commit realizado y la situación actual del proyecto.

```
git diff
```

- **git tag:** Git tiene la posibilidad de etiquetar puntos específicos del historial como importantes. Esta funcionalidad se usa típicamente para marcar versiones de lanzamiento (v1.0, por ejemplo).

```
git tag -a v1.4 -m “Mi versión 1.4”
```

- **git log:** Muestra una lista de *commits* en una determinada rama junto con todos los detalles de las respectivas confirmaciones realizadas.

```
git log
git log --oneline --decorate --all --graph
```

- **git reset:** Se utiliza para mover el proyecto a un *commit* anterior eliminando todos los posteriores del historial de *commits*.

```
git reset --soft <idCommit>
git reset --hard <idCommit>
```