

# **GIT – Control de Versiones**

## **Instalación**



# COMANDOS ESENCIALES

---

`git init` - Sirve para iniciar un nuevo repositorio en local

`git status` - Sirve para controlar el estado de nuestro repositorio

`git add` - Sirve para agregar cambios a nuestro staging

`git checkout <branch>` - Sirve para movernos de una rama a otra

# COMANDOS ESENCIALES

---

`git commit -m` – Sirve para escribir un mensaje en el cambio que queremos versionar

`git push` – Sirve para enviar los cambios desde el repo local al repo remoto

`git remote update origin --prune` – Sirve para traer todas las ramas remotas hacia el repositorio local

# COMANDOS ESENCIALES

---

**git branch – Sirve para ver las ramas disponibles en el repositorio**

**git remote -v – Sirve para ver la dirección del repositorio remoto vinculado**

**git remote add origin <[url]> – Sirve para vincular el repo local a un repo remoto**

# COMANDOS ESENCIALES

---

`git clone <[url]>` - Sirve para clonar un repositorio remoto a nuestro equipo



# COMANDOS ESENCIALES

---

**Es necesario configurar las credenciales de forma inicial para que Git pueda hacer login**

```
git config --global user.name "[your_name]"  
git config --global user.email "[your_email]"
```

Para ver las variables configuradas

```
git config --get [key_name]  
e.g.:git config --get user.name
```

# SECUENCIAS SUGERIDAS

---

`git init` - Creo el repositorio

Voy a GitHub / GitLab y me creo un repositorio remoto

`git remote add origin [url]` - Agregó la URL del repositorio remoto

`git remote -v` - Veo que se haya creado la referencia remota para mi repo local

# SECUENCIAS SUGERIDAS

---

`git status` - Veo el estado del repositorio y la rama en la que estoy

`git add .` - Agrego todos los cambios

`git commit -m` - Creo un commit con un mensaje

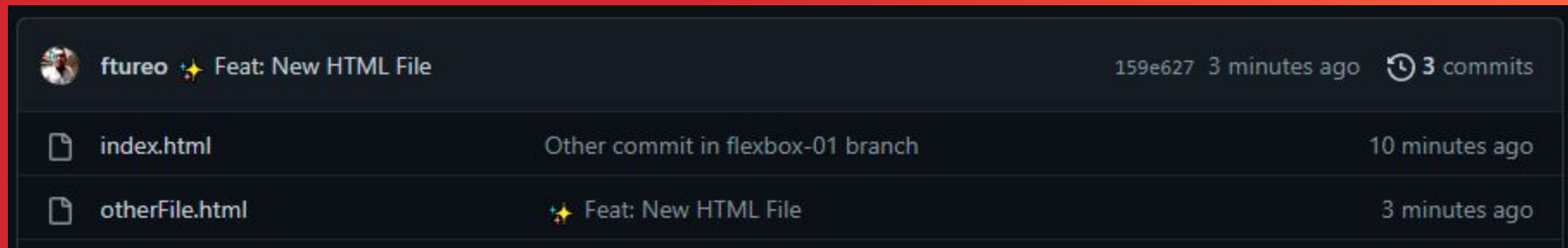
`git push origin [branch]` - Publico la nueva versión de mi repositorio local en el remoto






# BUENAS PRÁCTICAS PARA COMMITS

Si querés agregar iconos en los mensajes de tus commits, esto puede servirte:

<https://gist.github.com/parmentf/035de27d6ed1dce0b36a>



The screenshot shows a GitHub commit history interface. At the top, a commit by user 'ftureo' is shown with a star icon and the message 'Feat: New HTML File'. Below this, a table lists files changed in the commit: 'index.html' and 'otherFile.html'. The 'index.html' entry is linked to 'Other commit in flexbox-01 branch' and is dated '10 minutes ago'. The 'otherFile.html' entry is linked to 'Feat: New HTML File' (with a star icon) and is dated '3 minutes ago'.

 ftureo ✨	Feat: New HTML File	159e627 3 minutes ago	🕒 3 commits
 index.html	Other commit in flexbox-01 branch	10 minutes ago	
 otherFile.html	✨ Feat: New HTML File	3 minutes ago	

# BUENAS PRÁCTICAS PARA COMMITS

<https://midu.dev/buenas-practicas-escribir-commits-git/>

<https://codigofacilito.com/articulos/buenas-practicas-en-commits-de-git>

<https://medium.com/@jmz12/buenas-pr%C3%A1cticas-para-commits-5eb4c86b9a47>

# **GIT / GITHUB**

## **LET'S GO TO THE CODE**

