



Centro de Astropartículas y  
Física de Altas Energías  
Universidad Zaragoza



Departamento de  
Física Teórica  
Universidad Zaragoza

# Tutorial: python y flavio

Jorge Alda,  
Universidad de Zaragoza/CAPA

[jalda@unizar.es](mailto:jalda@unizar.es)

Slides Example  
Mayo 2022

Un ambiente virtual (venv) permite “encapsular” los paquetes de python que necesitas de forma aislada del resto del sistema operativo.

Para crear un venv en el directorio actual:

#### Linux

```
python3 -m venv envname
```

#### Windows

```
C:\Path\to\python -m venv envname
```

Esto crea un subdirectorio envname con los archivos necesarios para usar el venv.

Ahora hay que activar el `textttvenv`

### Linux

```
source envname/bin/activate
```

### Windows

```
envname\Scripts\Activate.ps1
```

Comprueba que estás dentro del `venv` usando el comando `python -m pip list`.

Para salir del `venv`, usa el comando

### Windows & Linux

```
deactivate
```

Activa el `venv`, e instala los siguientes paquetes usando `pip`:

- `numpy`
- `scipy`
- `matplotlib`
- `jupyter`
- `flavio`

- git es un sistema de control de versiones. Permite mantener una historia de todos los cambios de un proyecto de código y trabajar en paralelo con varias versiones (ramas).
  - Cliente de línea de comandos en <https://git-scm.com/downloads>
  - También hay clientes gráficos o integrados en IDE (VSCode).
- GitHub es un servidor de git. Permite mantener una copia “en la nube” del proyecto y su historia, editar de forma colaborativa, y compartir el código (open software).
  - Crear una cuenta en <https://github.com/>
  - Cliente de línea de comandos en <https://github.com/cli/cli>
- Conecta git con la cuenta de GitHub con el comando `gh auth login` y sigue las instrucciones (cuenta de GitHub básica, protocolo HTTPS, autenticación usando browser).

- 1 Entra en [github.com](https://github.com) con tu cuenta, pulsa el botón + en la esquina superior derecha y crea un nuevo repositorio. El nombre del repositorio es test-github, y añade un README.
- 2 En la página del repositorio, pulsa el botón verde “Code” y copia la dirección. En la consola de comandos, navega al directorio donde quieras crear tu proyecto y ejecuta `git clone https://github.com/your-name/test-git.git`. El archivo README.md se habrá copiado a tu ordenador.

- 3 Crea un archivo `test.txt` en el directorio y guárdalo.
- 4 Ejecuta `git status`. Verás que el archivo `test.txt` ha sido modificado desde la última entrada en la historia del repositorio.
- 5 Ejecuta `git add test.txt`. Esto añade el archivo a la lista de preparación (stage), pero aún no a la historia.
- 6 Vuelve a ejecutar `git status`, el archivo está en el stage.
- 7 Una vez que hayas preparado todos los archivos modificados, puedes añadir una nueva entrada (commit) a la historia local con el comando `git commit -m "Descripción del commit"` (intenta ser descriptivo).
- 8 Puedes ver todos los commits locales con `git log`.

- 9 Para subir los nuevos commits a GitHub, usa `git push`.
- 10 En GitHub puedes ver la historia del repositorio.
- 11 Si hay cambios en GitHub que no están en el ordenador, puedes incorporarlos con `git pull`. **Importante!:** Haz pull antes de `commit+push`.

GitHub tiene otras funcionalidades avanzadas (issues, pull requests,...) que no utilizaremos de momento.



Hasta ahora, solo hemos usado historias lineales: cada commit solo tiene un predecesor o un sucesor. Una historia con varias ramas sirve, por ejemplo, para probar nuevos conceptos mientras se mantiene (y desarrolla) la versión original.

- 1 Crea una rama con el comando `git branch rama1`. La rama `rama1` tiene la misma historia hasta este punto que la rama `main`.
- 2 Puedes ver todas las ramas del repositorio (locales+remotas) con `git branch -a`
- 3 Para cambiar la rama activa, usa `git checkout rama1`.