
python™


git


GitHub

Objetivos de la clase:

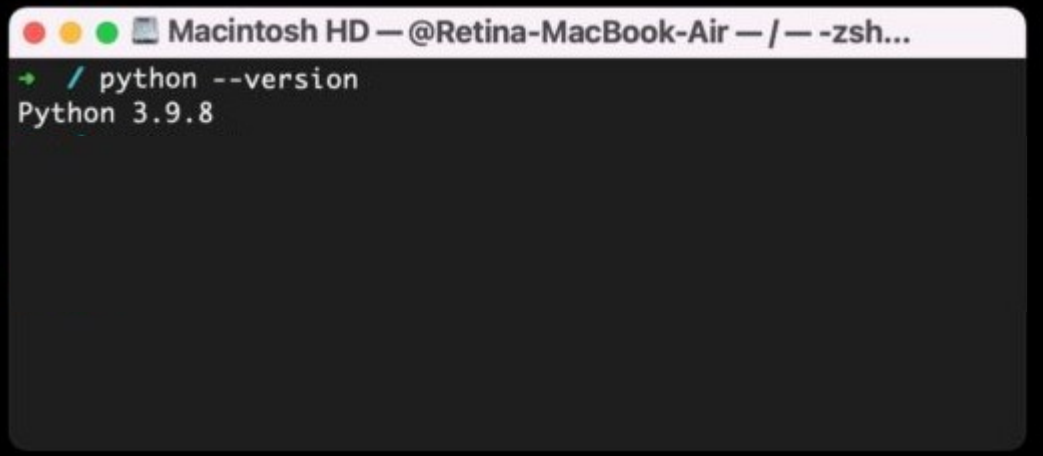
- Entender qué problema resuelve Git.
- Crear un repositorio local, versionar cambios y ver historial.
- Conectar con GitHub usando SSH y subir el trabajo.
- Explicar el flujo de tarea semanal (fork → cambios → PR).

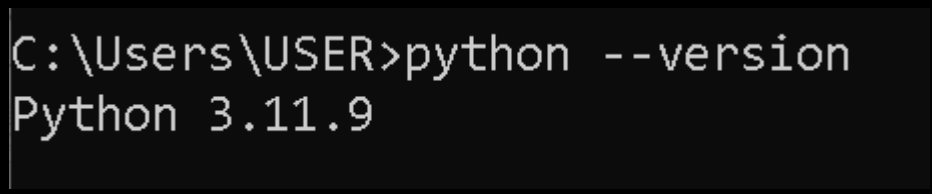
Primer paso

- Para poder aprovechar todas las herramientas que podemos usar con el conocimiento de Python que tenemos hasta ahora necesitaremos empezar a usar mas nuestra computadora e instalar mas programas, es muy importante entender este proceso y mantener buenas practicas para no tener problemas en el futuro
  - Instalar Python
  - Instalar Git
  - Crear cuenta en GitHub (mencionaremos alternativas mas adelante)

Al instalar Python

- Podemos hacerlo desde la tienda de Microsoft o directamente con la pagina de Python para cualquier sistema operativo https://www.python.org/downloads
- Asegurarse de marcar que sea incluido en el PATH
- Verificar con la consola (imagen de sistema, comand prompt o bash)
- Usar Google de manera responsable



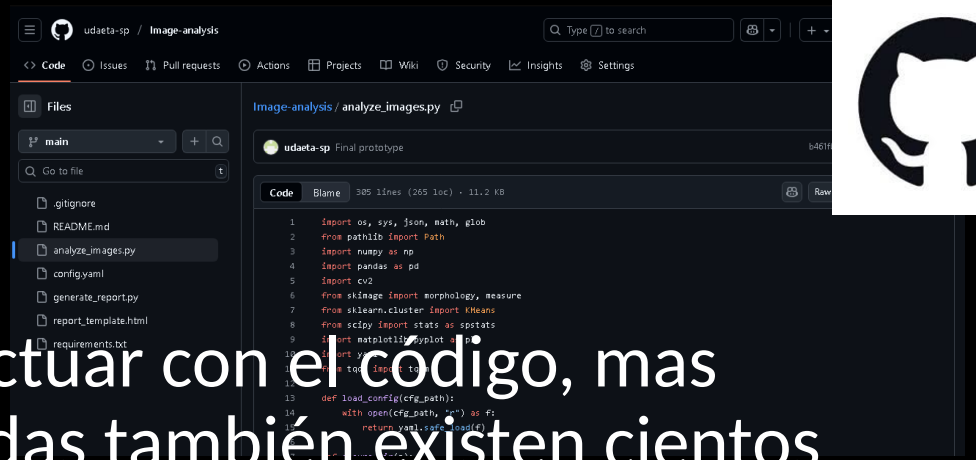


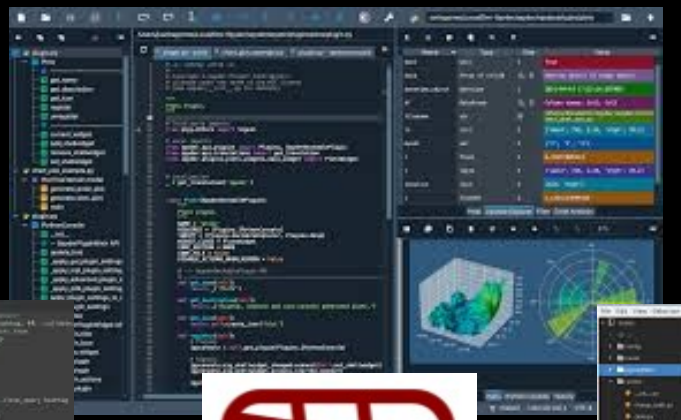
¿Que podemos hacer ahora?

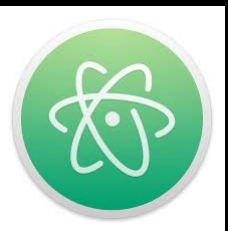
- Nuestra computadora aprendió un nuevo idioma para comunicarse con nosotros
- Al abrir archivos .py estos se ejecutarán como código de Python
- Podemos instalar editores de texto como jupyter, visual studio code, atom, Emacs o usar el Notepad para editar y crear programas en Python y ejecutarlos

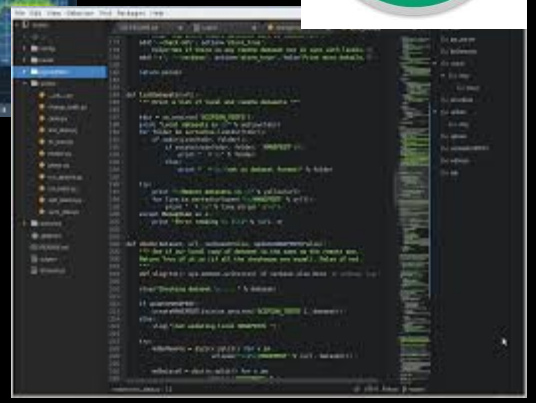
Pausa publicitaria

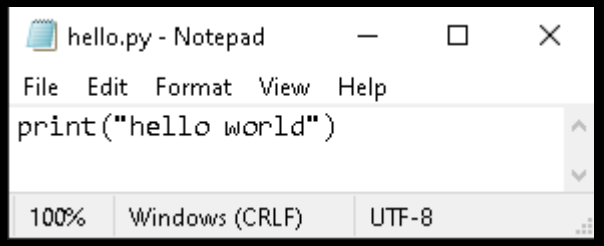
- Existen cientos de interfaces para interactuar con el codigo, mas adelante veremos algunas, pero para todas también existen cientos de videos explicando sus funciones herramientas y configuración, la experiencia de entender e instalar distintos editores es muy valiosa en la formación de un programador











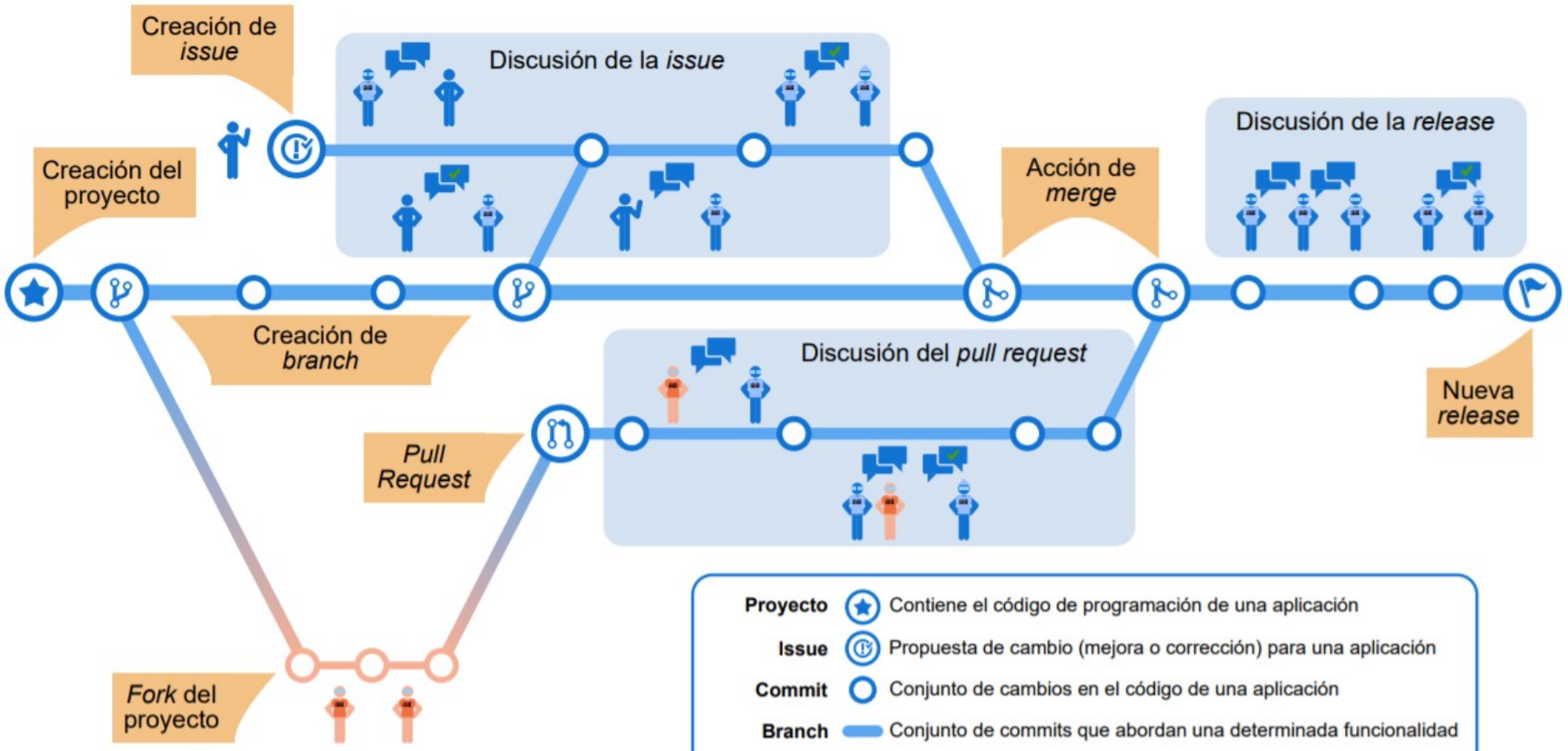
¿Y como buscar y compartir programas?

- Git es un sistema de control de versiones
- Por qué usarlo
  - Seguridad: recuperar versiones anteriores.
  - Colaboración: varias personas trabajando sin pisarse (branches + merge).
  - Trazabilidad: quién cambió qué y por qué (commits con mensaje).
  - Experimentación: probar ideas en una rama sin romper main.
- Si manejamos bases de datos, programas de procesamiento, interfaces o reportes de resultados nos permite organizar mejor y escalar nuestros proyectos

¿Y donde pasa esto?

- **GitHub es una plataforma en la nube**
  - Servicio que **aloja repositorios**
  - Facilita **compartir, colaborar y mostrar** trabajo (issues, PR, revisión).
- **Git = local** **GitHub = remoto** (host/colaboración).

¿Cómo se desarrolla en GitHub?
¿Cuáles son los principales elementos de discusión?



Proyecto	Contiene el código de programación de una aplicación
Issue	Propuesta de cambio (mejora o corrección) para una aplicación
Commit	Conjunto de cambios en el código de una aplicación
Branch	Conjunto de commits que abordan una determinada funcionalidad
Fork	Creación de una nueva branch
Pull Request	Propuesta de incorporación de una branch en otra
Merge	Unión de dos branches
Release	Lanzamiento de una versión de una aplicación

bcnparticipa.cat/decidimjam

som-research.uoc.edu

Como crear mi primer repositorio

- Antes de empezar tenemos que aprender a movernos entre nuestros archivos con la computadora

```
cd                  :: muestra la ruta actual
cd Carpeta          :: entrar a Carpeta
cd ..               :: subir un nivel
cd \                :: ir a la raíz de la unidad actual
cd /d D:\Proy     :: cambiar de unidad y carpeta (útil al pasar de C:\ a D:\)
```

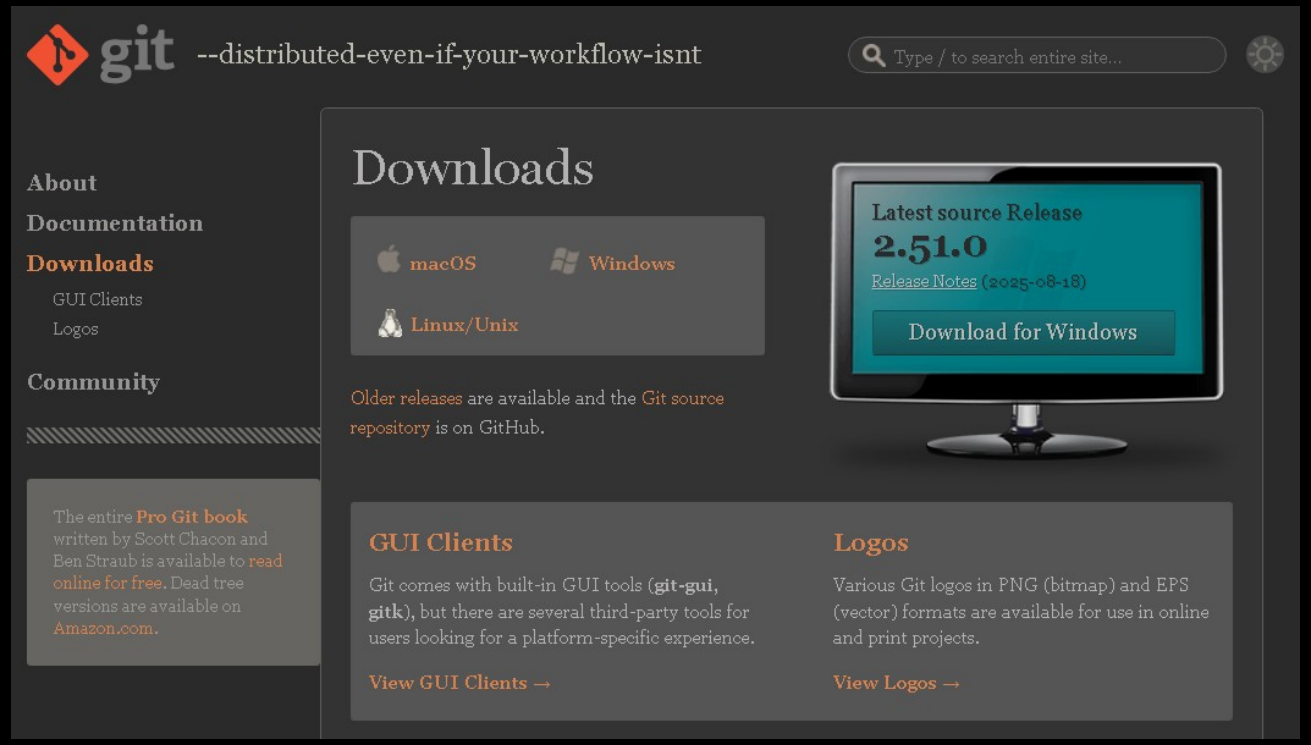
```
dir                 :: lista archivos y carpetas
dir /b              :: listado simple (solo nombres)
dir /ad             :: solo carpetas
dir /ah             :: archivos ocultos
dir /s              :: incluye subcarpetas
```

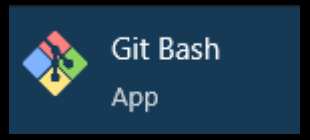
```
type nul > nuevo.txt          :: crea archivo vacío
echo hola > nota.txt          :: crea (o sobrescribe) con texto
echo otra línea >> nota.txt   :: agrega al final
mkdir MiCarpeta               :: crea carpeta
```

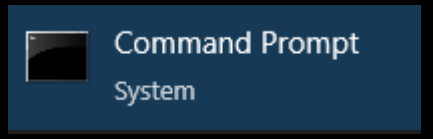
```
del archivo.txt               :: borra archivo
del *.log                     :: borra por patrón
del /s /q *.tmp               :: borra .tmp en subcarpetas (silencioso)

rmdir MiCarpeta               :: borra carpeta vacía
rmdir /s /q MiCarpeta         :: borra carpeta y todo su contenido
```

Instalamos git

- Lo hacemos desde la pagina y usamos Google con prudencia
- Veremos una nueva app llamada git bash o git cmd, no confundir con la consola/terminal/imagen de sistema/command prompt







Ahora si el primer repositorio

- Confirmo primero que haya instalado correctamente
- Voy a la carpeta que se convertirá en mi repositorio (cd)

```
>git init
```

- Genero los archivos necesarios para que la carpeta se convierta en un repositorio (la podemos ver en archivos como oculta)

```
C:\Users\USER>git --version
git version 2.50.1.windows.1
```

¡Y listo!

¿Que significa tener un repositorio?

- Podemos clasificar el avance de nuestro proyecto para entender como git almacena nuestra información

```
git add archivo.txt          # preparar archivo para commit
git add .                    # preparar todo lo cambiado
```

Lo que hago en la carpeta usando el explorador de archivos o la consola para crear o editar archivos

Lo que decido documentar con git para poder hacer un seguimiento de mi progreso

```
git commit -m "Mensaje"      # crear snapshot (commit)
```

Un checkpoint del proyecto donde esta, acompañado de un mensaje describiendo el avance logrado

¿Y como lo asocio a GitHub?

- Para subir mi repositorio

```
# 1) Iniciar y primer snapshot
git init
git add .
git commit -m "Primer commit"
git branch -M main

# 2) Conectar con GitHub (remoto 'origin')
git remote add origin git@github.com:USUARIO/REPO.git
#   (o) git remote add origin https://github.com/USUARIO/REPO.git

# 3) Subir por primera vez y dejar upstream configurado
git push -u origin main
```

- Para copiar un repositorio

```
# 1) Traer el repo a la compu
git clone git@github.com:OWNER/REPO.git
cd REPO

# 2) Crear rama de trabajo (recomendado)
git checkout -b feature/mi-cambio

# 3) Editar, preparar y confirmar cambios
git add .
git commit -m "Describir el cambio"

# 4) Subir tu rama al remoto
git push -u origin feature/mi-cambio
```

Algunas recomendaciones

```
:: 1) Clonar y entrar
git clone <URL_DEL_REPO>
cd <CARPETA_DEL_REPO>

:: 2) Crear entorno virtual
python -m venv .venv

:: 3) Activar entorno (CMD)
.\.venv\Scripts\activate

:: 4) Actualizar pip e instalar dependencias
python -m pip install --upgrade pip
python -m pip install -r requirements.txt

:: 5) Ejecutar el proyecto (según el repo)
python main.py
```

- Muchos repositorios tienen sus dependencias en archivos .txt, si usamos un entorno virtual podemos asegurarnos que las compatibilidades sean apropiadas para ejecutar nuestro código
- Es importante escribir un archivo ReadMe.md que nos pueda guiar para entender como funciona el código y principalmente como ejecutarlo