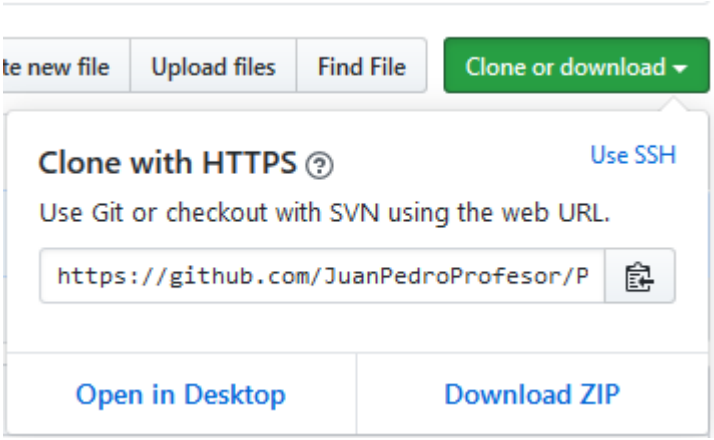
UD1
Mini Tutorial de Git - Procedimiento Básico

Vamos a partir de la base que tenemos ya creado un repositorio en github; podemos usar cualquier otro servidor Git siempre que tengamos una URL que nos permita tener un acceso al mismo. También debemos de tener instalado Git en nuestro ordenador. Puedes encontrar Git en el siguiente enlace: https://git-scm.com/downloads

- Ya tenemos el repositorio de Github creado. Necesitamos una URL para poder acceder a él mismo, en GitHub el enlace estará en:



Operaciones básicas

- Clonando el repositorio en nuestro equipo (local):

  Para obtener una copia de nuestro repositorio en nuestra máquina local tenemos que hacer un clonado. Para ello seguimos los siguientes pasos:

  git clone url_de_tu_repositorio

  Por defecto cuando clonamos nos situamos en la rama **master**.

  Para ver cuantas ramas tenemos creadas por debajo de nuestra rama “master” verlo con:

  git branch → *Te da las que ya tienes en local pero si quieres ver todas, usa el parámetro* ***--all***

  ((La rama marcada con “*” es la rama en la que estás situado))

Ojo al dato!!! Si en vez de clonar un repositorio ya previamente creado quisiéramos crear uno, en un directorio donde ya tenemos un proyecto en local deberíamos de usar los siguientes comandos dentro del directorio que queremos "guitear":

```
git init    → Inicializa el directorio actual como un directorio
git add *.c    →  Añade todos los archivos con extensión .c al repositorio
(es un ejemplo).
git commit -m 'versión inicial del proyecto'  → Hacemos comit de los cambios.
```

- Crear una nueva rama:

  Es "altamente aconsejable" trabajar siempre con una rama específica para cada tarea o modificación sustancial del proyecto. Para cambiar de rama, o si no existe, se crea esa rama usamos el comando "checkout" con el parámetro -b.

  `git checkout -b mi_nueva_rama` → El parámetro -b se utiliza para crear la rama si no existe. Si simplemente se quiere cambiar de rama, no usaremos el -b

  ¡¡¡ Ojo, esta rama se crea en local !!!

- Añadir archivo:

  Vale, parece que ya estamos en una nueva rama. Imaginemos que ya tenemos un archivo de código que queremos añadir a la rama. Git exige que para cada nuevo archivo debemos de añadirlo a Git de una forma explícita para que Git pueda realizar Para ello usamos el comando:

  git add mi_nuevo_archivo

  ¡¡¡ Si utilizamos el símbolo punto "." se añadirán todos los archivos/carpetas que no estén traqueados !!!

- Guardar cambios:

  Esta es la operación que más se veces se realiza. Se trata de ir guardando en local los cambios, también llamado en el argot "comitear". Hay que añadir un comentario y ojo, que sea corto pero preciso:

  git commit -am "descripción de mi primer commit"

- Subir los cambios a la rama remota:

  Vamos a publicar nuestros cambios en local al remoto

  git push origin mi_nueva_rama

¿Qué es “origin”? En git, “origin” es un sustituto de la url del repositorio remoto, en vez de poner toda la url, como tu directorio en local ya está asociado a tu repositorio en remoto, simplificamos y usamos “origin”.

- Fusionar tu rama con la master:

Esto es lo que se le llama realizar un “merge”. Siempre tener en cuenta que vamos a fusionar 2 ramas, pues bien, **es muy importante primero hacer un checkout de la que se va a quedar con toda la fusión**, es decir, la rama que contendrá el contenido de las 2 ramas a fusionar. Seguiremos los siguientes pasos:

git checkout master → (cambia de tu rama local a la master)

git pull origin master → (este comando actualiza tu rama “master” local con todos los cambios que puedan haber ocurrido en tu rama “master” en remoto (otros desarrolladores pueden haber modificado esta). Resumiendo, es como si sincronizáramos, no vaya a ser que algún otro desarrollador hubiera cambiado algo en la rama).

git merge nombre_rama_fusionar → (fusiona a la rama en la que estás actualmente, el contenido de la rama que le indiques “**nombre_rama_fusionar**”))

git commit -am “comentario_corto_preciso” (guardamos/comiteamos los cambios de la fusión. Este paso no es realmente necesario si el merge ha sido completado con éxito, sin conflictos)

git push origin master (y por fin subimos actualizamos los cambios de nuestra rama “master” en local a nuestra rama “master” en el servidor).

Buenas prácticas

→ Es importante que la rama “master” siempre esté bien mantenida. Todo lo que vuelques/fusiones en la master tiene que estar bien testeado, con revisión de código, sin conflictos.

- Para cambios importantes o nuevas “features” en tu programa: ¡¡¡¡ Crea una nueva rama !!!!

- No anides ramas dentro de otras; como mucho tres niveles: la master, un nivel de subrama, y dentro de esta si acaso otro nivel. Lo ideal es solo un nivel que hereden todas de la master.

Fin! 🎉  ⚡

Enlaces

Resumen abreviado de los comandos más usados en Git:

https://www.hostinger.es/tutoriales/comandos-de-git

Un ejemplo de uso:

http://tombatossals.github.io/git-puesto-en-practica/ejemplos/

Chuleta de comandos:

https://elbauldelprogramador.com/mini-tutorial-y-chuleta-de-comandos-git/