CURSO GITHUB

El git lo usamos en local (no maneja el acceso a repositorio)

El acceso a repositorio lo lleva el GITHUB

cuando queramos hacer una copia en remoto (en otro servidor) usaremos el push y subimos nuestro repositorio a otro repositorio

Para hacer lo contrario (traerme el repositorio desde el servidor) usaremos el pull.

para el control acceso a recursos (permisos a archivos, etc)

Para ese control podemos usar servicios como BITBUCKET, GITHUB (HOSTEADOS -nos lo ofrecen otras personas) o GITOSIS (MANEJADO POR NOSOTROS MISMOS)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* GITHUB ES UNA PLATAFORMA DE DESARROLLO COLABORATIVO DE SOFTWARE PARA ALOJAR PROYECTOS \*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**GITHUB es muy popular porque la versión gratuíta permite hacer muchísimas cosas, siempre que el repositorio sea público**.

Si lo quieres privado A PAGAR 7$/mes

PARA ASOCIAR UN NUEVO REPOSITORIO REMOTO A LA RUTA DONDE TENGAMOS NUESTRO PROYECTO

git remote add origin https://github.com/Superamoto/udemy-heroes.git origin es el nombre ESTANDAR del repositorio remoto (así lo llamamos)

PARA REVISAR LOR REMOTOS QUE TENEMOS

git remote (nos da una lista de repositorios que tenemos en remoto)

git remote -v (nos da una lista de repositorios que tenemos en remoto con el fetch y el push)

PARA SUBIR ALGO AL REPOSITORIO REMOTO

git push -u origin master

**u para poner la rama master por defecto** (después no hay que nombrar origin en los push y pulls…)

origin es nuestro repositorio remoto. master es la rama que enviamos el repositorio local)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Mi primer repositorio en github

Usuario: carlosaguirreatos

Clave: T\*\*\*\*\*\*\*\*\*2\*

Email: [caguirreatos@outlook.es](mailto:caguirreatos@outlook.es)

git remote add origin https://github.com/carlosaguirreatos/udemy-heroes.git

git push -u origin master

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Cuando hacemos un git push **NO SE SUBEN LOS TAGS AUTOMÁTICAMENTE**. TENEMOS QUE IR SUBIÉNDOLOS A MANO

PARA SUBIR TAGS AL REPOSITORIO REMOTO (desde dentro de nuestro proyecto)

git push – tags

PARA TRAERNOS LOS CAMBIOS DESDE EL REPOSITORIO

git pull (a secas. No ponemos origin ya que antes en el primer push habíamos puesto con el parámetro “u” que origin fuera por defecto)

HABRIA QUE HACER UN GIT PULL CADA VEZ QUE VOLVAMOS A TRABAJAR O CUANDO SEPAMOS QUE HA HABIDO CAMBIOS

PARA CLONAR UN REPOSITORIO EN LA RUTA ACTUAL DE CARPETAS

En el repositorio pulsamos el botón clone or download, de ahí copiamos la dirección <https://github.com/carlosaguirreatos/udemy-heroes.git>

Ahora en la consola ejecutamos:

git clone <https://github.com/carlosaguirreatos/udemy-heroes.git> (se crea un clon con el mismo nombre que tiene en el repo remoto)

git clone <https://github.com/carlosaguirreatos/udemy-heroes.git> nombrenuevo (con otro nombre)

PARA DESCARGAR DESDE EL REPOSITORIO SIN QUE HAGA MERGE (COMO HACE EL PULL)

git fetch (lo hacemos cuando no sabemos exactamente qué ha cambiado en el repo y si puede entrar en conflicto con nuestro repo local)

**BUENAS PRÁCTICAS**

Cuando pueda haber cambios en el repo remoto que no conoczcamos y nosotros hagamos cambios en el repo local… ejecutar POR ÉSTE ORDEN:

Hacemos cambio en local

**git s**

**git commit -am “cambios hechos en el repo remoto”**

Ahora para combinar nuestro repo con el remoto…

**git fetch** (Nos traemos los cambios hechos en remoto pero sin hacer merge)

**git pull** (se abre un editor que nos dice que hizo un merge. Si hay conflicto arreglarlo)

**git push** (Finalmente subimos al repo remoto los cambios hechos en local)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**Markdown en Github**

|  |  |
| --- | --- |
| MD | HTML |
| # | H1 |
| ## | H2 |
| ### | H3 |
| #### | H4 |
| ##### | H5 |

Un gran poder requiere una gran \*\*responsabilidad\*\*

Ponme en \_cursiva\_ y también en \*cursiva\* o en \_\*\*cursiva y negrita\*\*\_

> chupi

1. item

2. item

3. item

\* subitem

\* subitem

\* subitem

![Batman](https://icon2.kisspng.com/20171220/ckt/batman-png-5a3a0566558728.11977439151375191035033852.jpg)

[Batman](https://icon2.kisspng.com/20171220/ckt/batman-png-5a3a0566558728.11977439151375191035033852.jpg)

[Visit GitHub!](www.github.com)

![Benjamin Bannekat](https://octodex.github.com/images/bannekat.png).

Para COLABORAR en repositorios remotos donde no estemos como colaboradores

FORK (clonar el repo y clonarlo a un repo nuestro) donde podemos hacer push, commits, clone, etc.

Una vez tenemos nuestras modificaciones y queremos que nos las admitan en el repo original hacemos un pull request

Ya que el fork es una especie de “rama” del original

Al hacer un fork en un repo público se nos hace como una copia en nuestro github. Para bajarlo a local

git clone [https://github.com/carlosaguirreatos/legion-del-mal.git demo-11](https://github.com/carlosaguirreatos/legion-del-mal.git%20demo-11)

Cómo podemos actualizar nuestro fork?

Para saber que en el repo original hubo cambios… podemos clonar otra vez y hacer:

git remote add upstream <url del repo original> (del original al local)

git remote -v saldría algo nuevo

git pull upstream master

nos aparece un editor para el merge lo dejamos así , salimos , :wq y se hace el merge

ahora para actualizar el repo nuestro git push

git fetch upstream

INTRODUCCIÓN A LOS FLUJOS DE TRABAJO

Para subir al upstream mi rama

git push origin rama\_silver

git push origin :rama-misiones para borrar la rama misiones del remoto

git remote prune origin

CUÁNDO HACER UN TAG? CUANDO HAGAMOS UN PUSH A PRODUCCIÓN ES LO RECOMENDABLE

git push –tags PARA SUBIRLOS A GITHUB

**NOTACIÓN DE LOS TAGS**

**v.x.y.z**

x Cambios a versiones mayores

y Cambios menores

z Parches o fix que hagamos

[semver.org](https://semver.org/lang/es/) para ver cómo se versiona

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

RAMA DE PRODUCCIÓN

No tiene por qué ser la rama master. Igual en un determinado punto interesa que otra rama sea la de producción

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

RELEASES Y TAGS (DIFERENCIA)

En Github parecen lo mismo pero no lo son.

Un reléase nos permite compartir archivos binarios y más información (ver vídeo 14 del github avanzado)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

PARA ADMINISTRADORES DE PROYECTOS…

ISSUES

MILESTONES son grupo sde issues o características o un determinado momento del tiempo.

http://guides.github.com/features/issues