Git.

Git es un control de versiones, a diferencia de una copiar exacta, es que con el control de versiones, puedes guardar los cambios en tus ficheros, pero, guardando un histórico de las modificaciones, pudiendo recuperar los cambios anteriores al actual (o el fichero original).

Github es un hibrido entre un control de versiones y una red social.

El Git tiene 3 estados para una carpeta (que es el area del trabajo).

Comandos git

Git.add 🡪lo sube al almacén.

Git.init🡪 inicia el control de versiones

Git.commit🡪comitea el archivo

Git.remote🡪prepara los cambios para subirlos a la nube

Git.push🡪 sube el repositorio

Man git🡪 abre el manual

Git status 🡪 miramos el estado de los archivos

Git remote add origin “URL del repositorio”🡪 creamos la conexión entre el repositorio local y el GitHub

Git push origin master🡪 Sube los archivos al repositorio remoto

Git pull origin 🡪 Baja los repositorios del repositorio remoto

Git diff 🡪 muestra las diferencias entre dos archivos.

Git fetch Origin 🡪 Creamos una rama para bajarnos el conflicto (Hacemos un merge de la rama)

git tag -a v1.0 -m "Version 1 Alpha" 🡪 Añadimos el tag “v1.0”

git push origin v1.0 🡪 Este comando sube el tag v1.0

git push origin tags 🡪 subimos todos los tags a la vez

Comandos Linux

Echo “Texto” >> nombre archivo (Para añadirlo al final, con “>” lo sobreescribimos.

Al instalar el git, nos instalara una consola (donde pondremos los comandos anteriores de Git), entonces nos situamos en la carpeta donde queremos crear el repositorio y usamos el comando “Git.init” (nos creará un “.git” y ya sabremos que es un repositorio local).

//BUSCAR QUE SIGNIFICAN LAS TRES LICENCIAS DIFERENTES

Para fusionar una rama con otra usamos el comando “git checkout”

//Estando en otra rama

Git **checkout** master //Nos cambiamos a la rama master.

Git merge desarrollo //Juntamos la rama desarrollo con la rama master

Git branch –d “nombre de la rama” //Con esto eliminamos la rama.

Git chekout –b “nombre de la rama” //Creamos la rama y nos movemos a ella

*Es el equivalente a:*

*Git branch desarrollo*

*Git chekcout desarrollo*

Git checkout “Número de hash” //Con esto volvemos al estado anterior de nuestro programa, dependiendo del hash que le pongamos

El último commit siempre es el “HEAD” y la rama final es la master

Git reset –numero hash.

Dentro del reset tenemos 3 reset diferentes

Reset hard // Borramos el commit y el código

Reset soft //Con este reset, pasamos el commit al area del trabajo y lo podemos recuperar con el git add (nos quedamos con el código)

¿? //Se queda en el área de preparación

Cd 🡪 nos movemos a la carpeta donde crearemos el “entorno de trabajo”

(Creamos la carpeta con MKDIR)

Git init 🡪 Creamos el repositorio

Git add 🡪 Añadimos el archivo al repositorio (git add “nombre del archivo”).

Git status🡪 vemos el estado de los trabajos

Git commit -m “Comentario”🡪 Lo subimos al repositorio LOCAL con un comentario

Creamos un repositorio en el github

Primero nos creamos el repositorio en github

Hacemos un clon en local

Trabajamos en nuestra area de trabajo

Subimos los cambios.