COMADOS:

Git hub es sitio web que tiene por dentro u servidor de git donde cualquiera puede crear o clonar un repo y compartirlo

Es una interfaz visual del repo

Herramientas colaborativas mas importates

La red social de los programadores

Tus repos son el portafolio de proyectos como programados para que la gente vea lo que puedes hacer.

Nota: actualizar el archivo de txt

Cd proyecto1

Git Pull origin main sirve para traer

Git add espacio punto . sirve para agregar todos los archivos

Git status sirve para ver que se debe enviar o si no hay nada que enviar

Git commit espacio -m “entre comillas se le agrega titulo” sirve para enviar (sin commit se sale con esc shift zz)

Git push origin main “nombre de la rama a enviar a internet hub” sirve para enviar ramas a git hub

Git log sirve para ver el historial de la rama en la que me encuentro

Head sirve para apuntar al ultimo commit

Git checkout sirve para saltar de rama en rama“escribir el name de la rama a donde voy”

Git branch sirve para mostrarme todas las ramas que hay.

Git show sirve para mostrar lo ultimo en la rama que me encuentro

Cd : sirve para ir al home

Ls –al sirve para ver lo archivos que hay dentro de la carpeta

Contrl L sirve para limpiar pantalla

Pull sirve para traer

Git merge “nombre de la rama diferente a donde estoy” sirve para fucionar ramas

Debo estar en la rama donde voy hacer la fusión, y el git merge debo escribir de donde debo traer el código para fusionar

Git remote sirve para ver el origen remoto de nuestros archivos

Git remote –v sirve para mostrarnos los fetch (traer archivos) y los push (enviar archivos)

Git push orgin master : git envie al origen (https:…….(fetch) (push) la rama master

Git config –l sirve para ver la confirguacion de git

~ es un atajo para llegar a la carpeta home

Cd / me envía al home y termino con /

Pwd me muestra donde estoy parado /

Cd ~ y luego pwd me envía /c/User/Educacion-platzi

Git log sirve para ver todos los commit de proyecto

Git log --all sirve para ver todo los movimientos históricamente

Git log –all --graph sirve para ver rayitas de como funcionan las ramas

Git log –all –graph –decore --oneline sirve para ver toda la historia grafica

Git tag –a v0.1 –m “RESULTADO DE LA PRIMERAS…….” Sirve para crear versiones del código.

Git tag sirve para ver las versiones que se tienen en el código

Git show-ref –tags sirve para ver a que conocer el detalle del tag

Git tag –d “nombre del tag” ejp v0.1 sirve para borrar tag o versiones

Alians arbolito sirve para mostrarme la historia de las ramas

Git branch “nombre de la rama a creaer ejmp header” sirve para crear ramas

Git push origin “nombre de la rama a enviar a internet hub” sirve para enviar ramas a git hub

Git show-branch sirve para ver las ramas que existen y su historia

Git show—branch –all sirve para ver la historia con mas detalle

Gitk sirve para ver la historia de manera visual

Git clon mas el link de clon y pegar. Sirve para clonar un proyecto y trabajar en mi pc

. git ignore es una lista que los archivos que vamos a ignorar

Git clean –dry-run (ejecución en seco) es un simulador de lo que se va a borrar

Git clean –f sirve para darle la orden de borrar los archivos visto anteriormente

Git commit – amend sirve para pegar los cambios al commit del error.

Git grep –n “la palaba a buscar”

Git grep –c Sirve para contar el numero de veces que esa palabra se usa en el codigo

----------------------------------------------------------------------------------------------

Video 16 ramas o branches (fusiones = merge) minuto 6:06

Master es la rama principal

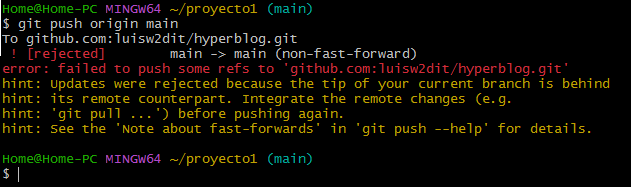
------------------------------------------------------------------------------------------------------

GIT HUB LLEVAR DEL REPO LOCAL LOS ARCHIVOS AL REPO EN LA ameINA DE GIT HUB (CLONAR):

Clone https copiar url

Git remote add origin + la dirección url OJO en consola de git

Video 19 minuto 5:24



rechazado

avance no rápido

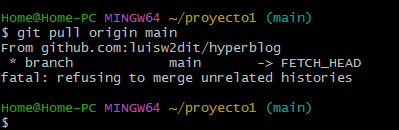
no empujó a algunos árbitros a

sugerencia

las actualizaciones fueron rechazadas porque la punta de su rama actual está detrás de su contraparte remota.

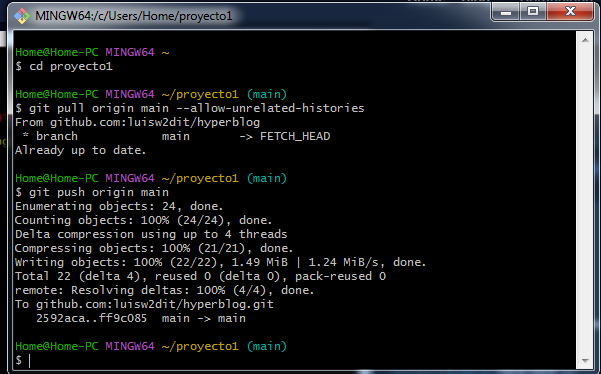
integra el cambio remoto (git pull) antes de empujar de nuevo.

vea la nota sobre fast forwards en la ayuda de git push para más detalles.

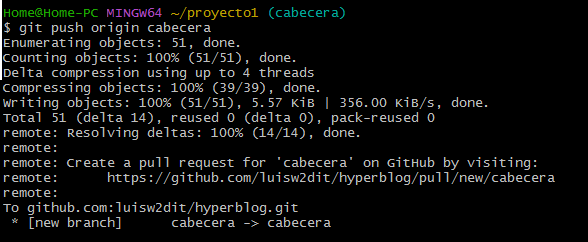


negarse a fusionar histoires no relacionadas

solución



CORREGIDO



CREAR LLAVE SSH: video 21 minuto 11:00

Ssh-keygen –t rsa –b 4096 –C “luis.munoz@w2dit.com”

PARA COMPROBAR QUE EL SERVIDOR DE LLAVES ESTE ENCENDIDO:

eval $(ssh-agent –s)

rta Agent pip 4724

AGREGAR LA LLAVE AL SERVIDOR ssh : video 21

SSH sirve para que NO nos pida claves

Ssh-add ~/.ssh/id\_rsa



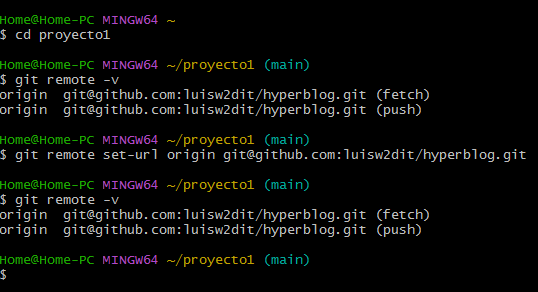
CAMBIAR EL NOMBRE DEL REPOSITORIO LLAMADO ORIGIN: oringin master es la rama remota

Sirve para que NO me pida clave

git remote –v sirve para mostrarme el nombre del repositorio.

Git remote set–url origin [git@github.com:luisw2dit/hyperblog.git](mailto:git@github.com:luisw2dit/hyperblog.git)

Esta corresponde a la url copia de la sección ssh



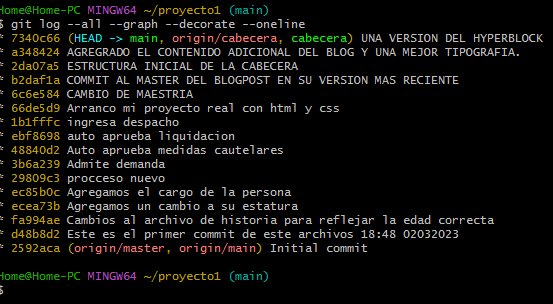
TRAER ULTIMA VERSION DEL SERVIDOR: video 22 minuto 5:09 error ojo

Git pull

**VERSIONES:** Video 23



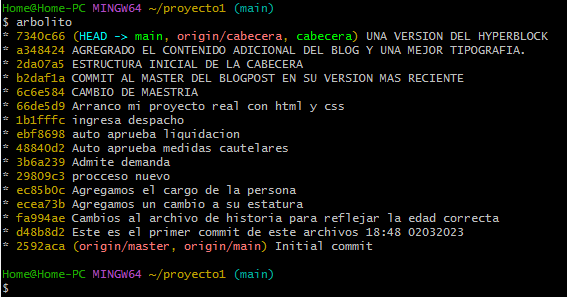
Sirve para ver la historia comprimida



**CREAR ALIANS:**

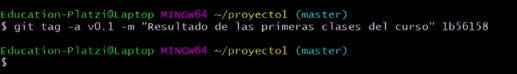
Cuando tengo un commando tan largo lo renombro para llamarlo luego con el alians



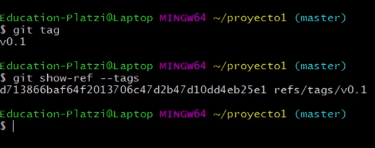


CREAR VERSIONES (TAG):

Copiar has del comit (la referencia del comit)



Git tag sirve para ver las versiones que se tienen en el código



PARA ENVIAR UN TAG

NOTA: ojo siempre se debe realizar un pull y luego un push.

Git pull origin main

Git push origin --tags

BORRAR TAG (VERSIONES):

Git tag –d “nombre del tag” ejp v0.1 paso seguido se debe borrar la referencia

Git push origin :refs/tags/nombre del tag

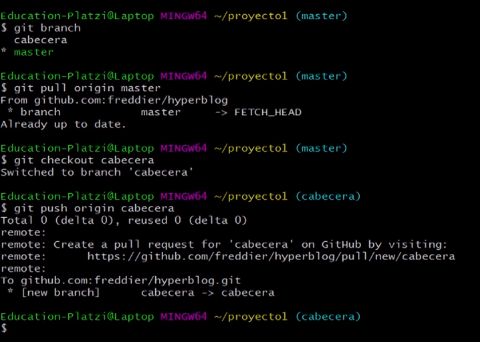
Video 24 ENVIAR UNA RAMA A GIT HUB

Git branch sirve para identificar las ramas

Git pull origin main sirve para traer la rama origin del hub (internet)

Git checkout cabecera sirve para pararme en la rama que necesito enviar ejp cabecera

Git push origin cabecera sirve para enviar a origin la rama en la que estoy parado ejemp cabecera

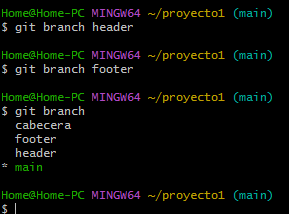


Video 24 CREAR RAMAS (header y footer)

En main

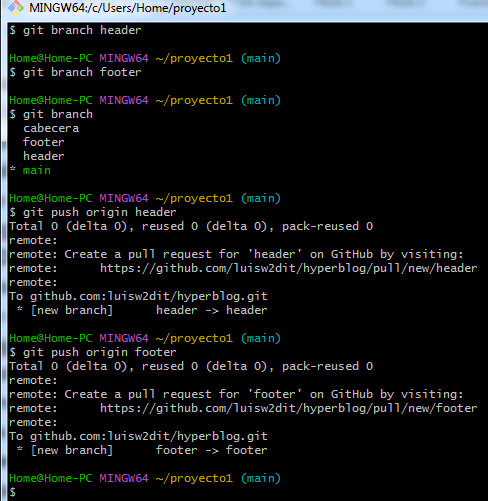
Git branch header

Git branch footer



ENVIAR RAMAS A INTERNET GIT HUB

Git push origin “nombre de la rama a enviar a internet hub” sirve para enviar ramas a git hub



VIDEO 25: TRAER UN PROYECTO PARA UNIRME COMO NUEVO MIEMBRO:

Git clon y pego la url

AGREGAR A PARTICIPANTE DEL PROYECTO

En git hub setting del repositorio

Collaborates agregar a otros colaboradores con el correo.

VIDEO 26: FUCIONAR LA RAMA

Git checkout main sirve para ir a la rama maestra

Git merge header

VIDEO 27: PULL REQUEST: Sirve para que otros miembros del equipo puedan ver los cambios que yo hice y aprobarlos y al aprobrarlos se auto ejecuta el merge, y luego se deben fucionar los cabios con la rama maestra. Para eso se realiza un pull request, es característica de github.

Dev ops es un admon del entorno de desarrollo.

VIDEO 28:

VIDEO 29\_ FORK (difurcaciones): es como un tenedor, un camino que se abre en dos, y quiere decir que una persona que no hace parte del proyecto, lo puede clonar para colaborar sin hacer un push.

Es tomar una copia del estado actual del proyecto y clonarlo, luego lo traigo a mi computador para trabajarlo y lo traigo.

Me ubico en la carpeta que quiero que descargar el proyecto y copio la url de clon,

Git clon mas el link pegar

VIDEO 30: archivos para ignorar

. git ignore es una lista que los archivos que vamos a ignorar

No guardar archivos binarios (fotos) en un repositorio

VIDEO 31: Referencia archivos binarios desde un externo

VIDEO 32: editar readme

VIDEO 34 : pagina con git hub

VIDEO 35: rebase es agarrar una rama entera y pegarla completa a la rama maestar, solo usar pera repositorios locales, rebase reescribe la historia del repositorio, es un parche.

VIDEO 36: Guardar modifiaciones en un lugar antes de realizar un comit : git stash y con git stash list. Puedo ver los guardados temporal git stash pop sirve para regresarlo.

VIDEO 37: borrar los archivos que no necesito

Git clean –dry-run (ejecución en seco) es un simulador de lo que se va a borrar

Git clean –f sirve para darle la orden de borrar los archivos visto anteriormente

VIDEO 38: traer un commit viejo a la rama master

Git cherry-pick “agregar el numero del commit” desde master : sirve para traer un commit viejo de una rama a mi rama master

Luego se debe hacer un merge

Git merge “nombre de la rama diferente a donde estoy” sirve para fucionar ramas

Debo estar en la rama donde voy hacer la fusión, y el git merge debo escribir de donde debo traer el código para fusionar. OJO ES MALA PRACTICA.. USAR CON RESPONSABILIDAD.

git cherry-pick puede ser útil para deshacer cambios. Por ejemplo, supongamos que una confirmación se aplica accidentalmente en la rama equivocada. Puedes cambiar a la rama correcta y ejecutar cherry-pick en la confirmación para aplicarla a donde debería estar.

VIDEO 39: todo se rompe y reseteo todo antes del accidente

Git reset –HARD “pegar el comint antes del accidente”

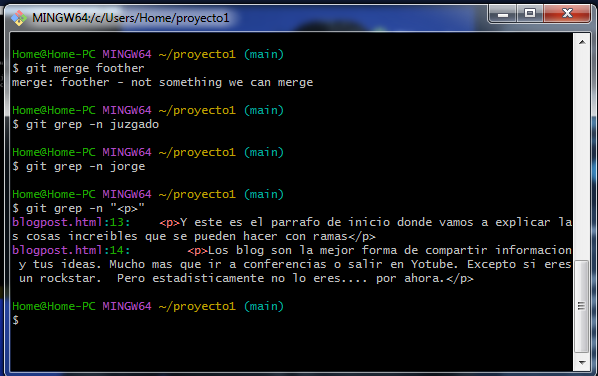
VIDEO 40: remendar un comit para remendar un error.

Git commit – amend sirve para pegar los cambios al commit del error.

VIDEO 41: buscar palabras en los archivos en el codigo

Git grep –n “la palaba a buscar”

Git grep –c Sirve para contar el numero de veces que esa palabra se usa en el codigo



Git log –S “nombre palabra buscar” Buscar en la historia de los commit una palabra especifica

VIDEO 42: