cx git add .

git status

git log: historia de commits

tag

git show

head: versión mas reciente

escape shift z z para salir de poner mensaje al commit: editor de texto: vim

lo que tienen en numeral es para observaciones que no queremos que quedar

Escape + i: mensaje

Git diff numerocommit1 y numerocommit2: el orden de los commits afecta

10. COMO VOLVER EN EL TIEMPO

Directorio de trabajo es la carpeta

Git init genera un repositorio /.git/

Git add lo manda a staging: memoria ram: temporal

Git commit lo manda al repositorio

Repositorio es MASTER

Untracked y tracked

Commit es una versión de los archivos

Checkout hacia tu carpeta

Crear una rama que después quieres unir a otro proyecto

11. QUE ES UNA RAMA Y COMO FUNCIONA

Master: donde se tiene las versiones de los archivos.

Copias una versión y experimentas

Bugfix o hotfix: unes una rama con otra: merge

Versión head: es la versio final

12. USO DE RESET Y CHECKOUT

Ver que ha pasado

Git reset sof

Git reset hard ( se borra todo lo que esta en staging y al volver no se puede hacer commit)

Git log –stat

$ git checkout 6f36f0e636c3672af0c6e216454f0cddac6a46da Historia.txt

13.

git reset –-soft: volvemos al pasado sin poder volver al futuro, sin eliminar lo que esta en staging en cambio en hard si elimina lo que este en el staging: actualizar después el archivo, se mete por error como algo que rastrea git

git reset HEAD; mover cambios de staging a unstaged, no borra los archivos

# 14. flujo de trabajo básico en un repositorio

Directorio local: donde trabajo

Staging: cambios

Repositorio de trabajo; donde se almacenan los cambios

Repositorio remoto

Git clone url de repositorio de remoto

Git add

Git commit

Git push

Git fetch: traer actualización al repositorio local pero no el archivo

Git merge: traer el documento

Git pull: mezcla fetch y merge

# 15. RAMAS

Formas de hacer cambios sin afectar la rama principal

Commit mas reciente: HEAD

Master: rama principal

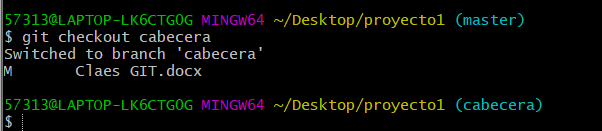
Rama es la copia del ultimo commit

Git commit –am: implica add y commit al miso tiempo: solo funciona para los que ya tenían add

Antes de hacer lo de ramas mirar que estemos en el master



Entonces creo la rama a partir del ultimo commit y ahora el HEAD le apunta tanto al master como a la cabecera que es una rama que va a trabajar otra persona



Ahora si estamos en el branch cabecera

Cuando creas una nueva rama te toca poner de nuevo add

Diferencia entre git shw y git log

Git log muestra el historial de los commits

Y git show

COMO QUEDA EL ARCHIVO HTML luego de volver a master: igual

Los archivos cambian automáticamente dependiendo de la rama

HEAD ES LA VERSION DE LOS ARCHIVOS QUE ESTOY VIENDO

# 16. MERGE

Vamos a hacer cambios en la cabecera

Y en master hacemos cambios en contenido

Git branch para ver que ramas hay

Escape shft zz

Hacerse en la rama master y hacer merge con cabecera

Git log –graph: para ver como se fusionan las ramas

:q para salir de vim

~$aes GIT.docx