Comandos git Jiménez Caiza Mateo

Configuración Básica

git config --global user.name "dasdo" //Configurar Nombre que salen en los commits

git config --global user.email dasdo1@gmail.com //Configurar Email

git config --global color.ui true //Marco de colores para los comando

Iniciando repositorio

git init //Iniciamos GIT en la carpeta donde esta el proyecto

git clone <url> //Clonamos el repositorio de github o bitbucket

git add //Añadimos todos los archivos para el commit

git commit -m "Texto que identifique por que se hizo el commit"//Hacemos el primer commit

git push origin master//subimos al repositorio

GIT ADD

git add //Añadimos todos los archivos para el commit

git add <archivo>//Añadimos el archivo para el commit

git add --all //Añadimos todos los archivos para el commit omitiendo los nuevos

git add \*.txt //Añadimos todos los archivos con la extensión especificada

git add docs/\*.txt //Añadimos todos los archivos dentro de un directorio y de una extensión especifica

git add docs/ //Añadimos todos los archivos dentro de un directorios

GIT COMMIT

git commit -m "Texto que identifique por que se hizo el commit" //Cargar en el HEAD los cambios realizados

git commit -a -m "Texto que identifique por que se hizo el commit" //Agregar y Cargar en el HEAD los cambios realizados

git commit -a //De haber conflictos los muestra

git commit --amend -m "Texto que identifique por que se hizo el commit"//Agregar al ultimo commit, este no se muestra como un nuevo commit en los logs. Se puede especificar un nuevo mensaje

GIT PUSH

git push <origien> <branch> //Subimos al repositorio

git push --tags //Subimos un tag

GIT LOG

git log //Muestra los logs de los commits

git log --oneline --stat //Muestras los cambios en los commits

git log --oneline --graph //Muestra graficos de los commits

GIT DIFF

git diff

git diff --staged //Muestra los cambios realizados a un archivo

GIT HEAD

git reset HEAD <archivo> //Saca un archivo del commit

git reset --soft HEAD^ //Devuelve el ultimo commit que se hizo y pone los cambios en staging

git reset --hard HEAD^ //Devuelve el ultimo commit y todos los cambios

git reset --hard HEAD^^ //Devuelve los 2 ultimo commit y todos los cambios

git log

git reset --hard <commit\_sha> //Rollback merge/commit

GIT REMOTE

git remote add origin <url> //Agregar repositorio remoto

git remote set-url origin <url> //Cambiar de remote

git remote rm <name/origin> //Remover repositorio

git remote -v //Muestra lista repositorios

git remote show origin //Muestra los branches remotos

git remote prune origin //Limpiar todos los branches eliminados

GIT BRANCH

git branch <nameBranch> //Crea un branch

git branch //Lista los branches

git branch -d <nameBranch> //Comando -d elimina el branch y lo une al master

git branch -D <nameBranch> //Elimina sin preguntar

GIT TAG

git tag //Muestra una lista de todos los tags

git tag -a <verison> - m "esta es la versión x" //Crea un nuevo tags

GIT REBASE

git rebase //Une el branch actual con el master, esto no se puede ver como un merge

git rebase --continue //cuando resolvemos los conflictos --continue continua la secuencia del rebase donde se pauso

git rebase --skip //Omite el conflicto y sigue su camino

git reabse --abort //Devuelve todo al principio del rebase

git rebase <nameBranch> //Para hacer un rebase a un branch en especifico

OTROS COMANDOS

git status //Lista un estado actual del repositorio con lista de archivos modificados o agregados

git checkout -- <file> //Quita del HEAD un archivo y le pone el estado de no trabajado

git checkout -b newlocalbranchname origin/branch-name //Crea un branch en base a uno online

git pull origin <nameBranch> //Busca los cambios nuevos y actualiza el repositorio

git checkout <nameBranch/tagname> //Cambiar de branch

git merge <nameBranch> //Une el branch actual con el especificado

git fetch //Verifica cambios en el repositorio online con el local

git rm <archivo> //Borrar un archivo del repositorio

FORK

git remote add upstream <url> //Descargar remote de un fork

git fetch upstream

git merge upstream/master //Merge con master de un fork