Bloque 3 Rubrica 2: Tutorial online de Git

Configuración inicial

Git config user.name “nombre_de_usuario”: este comando configura el nombre del usuario en el proyecto en el que se está trabajando.

Git config user.email “<correo@ejemplo.com>”: este comando configura el email del usuario en el proyecto en el que se está trabajando.

Git config -- global user.name “nombre_de_usuario”: este comando configura el nombre del usuario de forma global.

Git config -- global user.email “<correo@ejemplo.com>”: este comando configura el email del usuario de forma global.

Crear un repositorio nuevo

Git init: este comando crea una carpeta .git que contendrá los datos del repositorio sobre el que trabajaremos.

Git clone: este comando crea una copia de un repositorio ya existente que este en internet o en nuestro propio ordenador.

El staging index

Git add nombre_del_repositorio: Este comando añade los repositorios o archivos al staging index para poder trabajar sobre él.

Git add . : Este comando añade todos los archivos que hayamos modificado al staging index.

Git status: Este comando nos permite examinar el estado de nuestra carpeta de trabajo, los archivos que se añadirán en el siguiente commit, o archivos que aun no se han añadido a la carpeta de trabajo.

Enviando cambios

Git commit: Este comando confirma los cambios realizados en los archivos y envia estos cambios al directorio de trabajo, al hacer un commit es obligatorio poner un mensaje en el que se especifique con la mayor brevedad posible que cambios tiene ese commit.

Eliminando archivos

Git rm nombre_del_archivo: Este comando permite eliminar archivos del staging index y posteriormente, tras hacer un commit, en el directorio de trabajo.

Deshaciendo cambios

Git reset HEAD nombre_del_archivo: Este comando permite eliminar del staging index un archivo concreto con lo que pasara a ya no estar vigilado por el commit.

Git checkout -- nombre_del_archivo: Este comando permite revertir un archivo a su situación anterior para poder trabajar con el a partir de ese punto.

Examinando el registro

Git log: Este comando nos permite visualizar los distintos commits que se han hecho a lo largo de la historia del proyecto y los mensajes que contenian cada uno de los commits.

Git log --oneline: Este comando nos permite visualizar los distintos commits que se han hecho a lo largo de la historia del proyecto y los mensajes que contenian cada uno de los commits con la diferencia con el anterior comando de que cada commit y su mensaje solo ocupa una linea.

Creando ramas

Git branch nombre_de_la_rama: Este comando permite crear una nueva rama a través de de un commit en especifico.

Git Branch: Este comando permite visualizar las ramas existentes en un repositorio.

Git checkout nombre_de_la_rama: Este comando permite cambiar de rama especificando el nombre de la rama que queremos como destino.

Git checkout -b nombre_de_la_rama: Este comando permite crear una nueva rama y al mismo tiempo desplazarte a la misma.

Git branch -d nombre_de_la_rama: Este comando permite borrar la rama especificada.

Fusionando ramas

Git log -- oneline --decorate: Este comando nos permite ver los distintos commits en una sola línea y además con colores que nos dan cierta información sobre las características de los mismos

Git merge nombre_de_la_rama: Este comando permite unir desde la rama de destino (generalmente master) otra rama que especifiquemos aplicando los cambios de la rama origen a la de destino.

Examinando commits previos

Git checkout código_del_commit: Este comando nos permite volver a commits pasados permitiéndonos así crear nuevas ramas o reacer el proyecto desde ese punto.

Creando tags

Git tag nombre_del_tag: Este comando crea una etiqueta o “seudonimo” al commit en el que estemos.

Git tag nombre_del_tag código_del_commit: Este comando crea una etiqueta o “seudonimo” al commit que especifiquemos.

Git tag: Este comando permite ver la lista de las distintas etiquetas que hemos creado en el proyecto.

Push a repositorios remotos

Git remote add nombre_del_remoto dirección_del_repositorio: Este comando permite añadir repositorios remotos.

Git push nombre_del_remoto rama_para_enviar: Este comando permite enviar al repositorio remoto que especifiquemos una rama concreta.

Actualizando repositorios con el comando pull

Git branch --all: Este comando permite ver todas las ramas ya sean locales o remotas del proyecto en el que estemos.

Git pull nombre_de_la_rama: Este comando permite pasarnos los datos que contiene un repositorio remoto a nuestro repositorio local.

Git pull --rebase: Este comando permite pasarnos los datos de un repositorio remoto sin perder los cambios que hemos hecho en nuestro repositorio local.