Nombre Completo: María Tatiana Sanagua \_

Taller de Git – GitHub

1. ¿Cómo se crea un repositorio en Git?

Para crear un repositorio remoto en GitHub se debe ir New repository, elegir el nombre para el repositorio. Opcionalmente escribir alguna descripción del mismo. Elegir la visibilidad que tendrá el repositorio. Se puede incluir un archivo ReadMe y luego presionar Crear. Una vez creado ya tendremos el link a ese repositorio.

Para crear un repositorio local hay 2 opciones:

1. Crear un repositorio local en la PC y vincularlo a un repositorio remoto ya creado en GitHub:

Crear una carpeta vacía para inicializar nuestro repositorio, usar el comando git init, se creará un nuevo subdirectorio .git en el directorio de trabajo actual:

$**git init**

Luego debemos vincular nuestro repositorio con el repositorio (ya creado) en GitHub con el comando:

$**git remote add** **origin** <url\_repo\_github>

1. Clonar un repositorio
2. ¿Cómo se clona un repositorio en Git?

$**git clone** "url del repositorio"

1. ¿Cómo se crea una rama en Git?

$**git branch** **nombre\_rama**

$**git checkout** **nombre\_rama**

**O**

$**git checkout** **–b** **nombre\_rama**

1. ¿Cómo se fusionan dos ramas en Git?

$**git merge** **<nombre\_de\_tu\_rama>** **-m** “mensaje del merge”

1. ¿Cómo se resuelve un conflicto de fusión en Git?

Los conflictos suceden cuando dos ramas distintas hacen cambios en las mismas líneas de código o en los mismos archivos y Git no puede fusionar automáticamente esos cambios. Se debe identificar que archivos tienen conflictos y se los puede solucionar en el mismo código.

Una vez que se decide que cambios se van a mantener se debe hacer $git add nuevamente, luego continuar con la fusión con $git merge –continue, $git commit –m “Mensaje de fusión” y volver a subir los cambios con git push

1. ¿Cómo se hace un push en Git?

$**git push origin** <nombre\_de\_tu\_rama>

1. ¿Cómo se hace un pull en Git?

$**git pull** **origin** "nombre del remoto" 🡪 para traer la última versión

$**git pull** **--rebase** **origin** "nombre del remoto" 🡪 para comparar lo que yo tengo con lo que está en el repositorio.

1. ¿Cómo se crea una etiqueta en Git?

Una etiqueta se utiliza para marcar puntos específicos en la historia del repositorio.

$git tag nombre\_etiqueta 🡪 etiqueta ligera

$git tag –a nombre\_etiqueta –m “Mensaje de la etiqueta” 🡪 etiqueta anotada

$git push –tags 🡪 para subir todas las etiquetas al repositorio remoto

$git show nombre\_etiqueta 🡪para ver la información de la etiqueta junto a la del commit

1. ¿Cómo se revierte un commit en Git?¨

Si los cambios no se subieron al repositorio remoto se puede deshacer un commit y volver al estado anterior de la rama:

$ git reset --hard HEAD^

Si los cambios ya se subieron al remoto se debe hacer:

$ git revert -n HEAD~n..HEAD 🡪 para revertir “n” commits

$ git commit -m "Reverti ultimo commit"

$ git push --force origin nombre\_rama

1. ¿Cómo se hace un rebase en Git?

Rebase se utiliza para cambiar la base de una rama. En lugar de fusionar los cambios de una rama con otra, como en un merge, el rebase reaplica los commits de una rama en la parte superior de otra rama.

Primero tenemos que cambiarnos a la rama destino donde se desean aplicar los cambios:

$git checkout main

$git pull origin main 🡪actualizar la rama

$git rebase nombre\_rama

1. ¿Cómo se resuelve un conflicto de rebase en Git?

Si hay conflictos se debe resolver modificando los archivos con conflictos y decidir que va de manera permanente y hacer:

$git add nombre\_archivo

$git rebase --continue

$git rebase --skip

$git push origin nombre\_rama

1. ¿Cómo se usa el comando cherry-pick en Git?

El comando $git cherry-pick se utiliza en Git para aplicar un commit específico de una rama a otra. Básicamente, toma un commit existente y lo aplica en la rama actual, creando un nuevo commit con los mismos cambios.

$git checkout tu\_rama\_destino 🡪 Cambiarse a la rama destino

$git log 🡪 Obtener el identificador del commit

$git cherry-pick hash\_del\_commit

1. ¿Cómo se usa el comando stash en Git?

El comando git stash se utiliza para guardar temporalmente cambios no comprometidos (en el directorio de trabajo) para poder cambiar de rama o realizar otras operaciones sin perder los cambios actuales.

$git stash

$git stash save "Mensaje descriptivo"

$git stash list 🡪 Lista todos los stash

$git stash apply 🡪 Aplica el último stash

$git stash apply stash@{n} 🡪 Para aplicar el n stash de la lista

1. ¿Cómo se crea un pull request en GitHub?

Pull request sirve principalmente para revisar y comparar código entre ramas antes de ser fusionadas. Cuando se pushea un cambio de una rama GitHub da la posibilidad de realizar un Compare & Pull request entre la rama que subió el cambio y otra rama existente. Se muestra cuantos commits de diferencia hay entre una rama y la otra y si las ramas son capaces de fusionarse automáticamente o hay que resolver conflictos. Es posible agregar una descripción y un titulo al pull request. Una vez que se crea el PR se puede realizar el merge entre las ramas y eliminar la rama fusionada.

1. ¿Cómo se revisan los cambios en un pull request en GitHub?

Une vez que se crea el PR, en la pestaña Files es posible ver los archivos y compararlos. En caso de haber conflictos se podrá decidir que cambios quedan o cuales no.

1. ¿Cómo se acepta un pull request en GitHub?

Luego de haber revisado y solucionado los conflictos, se puede hacer un merge del pull request, se puede añadir comentarios y eliminar la rama fusionada.

1. ¿Cómo se trabaja con submódulos en Git?

Un submódulo en Git es un repositorio Git dentro de otro repositorio Git. El repositorio principal (también llamado repositorio padre) tiene un enlace al repositorio secundario (repositorio hijo) en una ubicación específica, pero no almacena directamente su contenido en el directorio de trabajo.

$git **submodule** add **Url\_repositorio** **ruta\_local**

Para realizar cambios en el submódulo:

**$cd** nombre\_del\_submodulo

# Realizar cambios, hacer commits, etc.

$git add .

$git commit -m "Mensaje del commit en el submódulo"

$git push origin main # o la rama correspondiente

$cd ..

$git add nombre\_del\_submodulo

$git commit -m "Actualizar referencia del submódulo"

Para actualizar el submódulo:

$cd nombre\_del\_submodulo

$git checkout main # o la rama que desees

$git pull origin main # o la rama correspondiente

$cd ..

Par desvincular el submódulo

$git submodule deinit -f ruta\_local

Eliminar las entradas del submódulo del índice

$git rm --cached ruta\_local

Eliminar los archivos del submódulo del sistema de archivos

$rm -rf ruta\_local

1. ¿Cómo se usa Git bisect para encontrar un commit problemático?

Ayuda a encontrar el commit específico que introdujo un problema (bug) en tu código.

$git bisect start 🡪 Iniciar el proceso

$git bisect good hash\_commit

$git bisect bad hash\_commit

$git bisect reset 🡪 Finalizar el proceso

1. ¿Cómo se configura un flujo de trabajo de Git flow?

El flujo de trabajo de Git flow es una convención popular para organizar las ramas de un repositorio Git en un entorno de desarrollo colaborativo. Está diseñado para facilitar la colaboración entre equipos y gestionar versiones de software.

1. Instalar git flow
2. Inicializar git flow en un repositorio existente: $git flow init
3. Configurar ramas principal de producción y de desarrollo
4. Poner nombre a las ramas de funciones (feature), lanzamientos (release), correciones (hotfix), corrección de errores (bugfix) y soporte (support).
5. Crear una nueva rama de función

$git flow feature start nombre\_de\_la\_funcionalidad

Trabajar en la nueva función y realizar commits

...

Finalizar la rama de función

$git flow feature finish nombre\_de\_la\_funcionalidad

1. Crear una nueva rama de lanzamiento

$git flow release start version

Realizar tareas de preparación para el lanzamiento (actualizar la versión, etc.)

...

Finalizar el lanzamiento

$git flow release finish versión

1. Crear una nueva rama de hotfix

$git flow hotfix start nombre\_del\_hotfix

Realizar correcciones y commits

...

Finalizar el hotfix

$git flow hotfix finish nombre\_del\_hotfix

1. Crear una nueva rama de soporte

$git flow support start version

Realizar correcciones en la rama de soporte y commits

...

Finalizar la rama de soporte

$git flow support finish versión

1. ¿Cómo se configura un flujo de trabajo de GitHub flow?