Git & GitHub

Introducción a GIT

El Sistema de Control de Versiones (VCS) es un sistema que registra los cambios en un archivo o conjunto de archivos a lo largo del tiempo para que pueda recuperar versiones específicas más adelante. También permite revertir archivos a un estado anterior, revertir todo el proyecto a un estado anterior, comparar cambios a lo largo del tiempo, ver quién modificó por última vez algo que podría estar causando un problema, quién introdujo un problema y cuándo si pierdes archivos o se cometen errores en el proyecto, se podrán recuperar los archivos fácilmente.
Git es un sistema de control de versiones en el cual almacena cada versión de un archivo o proyecto como una 'instantánea' en lugar de una lista de cambios realizados en cada archivo permitiendo "impulsar" y "extraer" cambios hacia y desde instalaciones en otras computadoras. Esto lo convierte en lo que se conoce como un "Sistema de control de versiones distribuido" y permite que varios desarrolladores trabajen en el mismo proyecto.

Los tres estados

Git tiene tres estados principales en los que pueden residir sus archivos: modificado , preparado y confirmado :

- **Modificado** significa que ha cambiado el archivo pero aún no lo ha enviado a su base de datos.
- **Preparado** significa que ha marcado un archivo modificado en su versión actual para pasar a su próxima instantánea de confirmación.
- **Confirmado** significa que los datos se almacenan de forma segura en su base de datos local.

Secciones

Git cuenta con tres secciones principales de un proyecto Git:

- ➢ Directorio o Árbol de trabajo (proyecto).
- ➢ Área de preparación o índice.
- ➢ Repositorio o directorio Git.

El **directorio o árbol de trabajo** es una verificación única de una versión del proyecto. Estos archivos se extraen de la base de datos comprimida en el directorio Git y se colocan en el disco para que usted pueda usarlos o modificarlos.
El **área de preparación o índice** es un archivo, generalmente contenido en su directorio Git, que almacena información sobre lo que se incluirá en su próxima confirmación.
El **repositorio o directorio Git** es donde Git almacena los metadatos y la base de datos de objetos para su proyecto. Esta es la parte más importante de Git y es lo que se copia cuando se clona un repositorio desde otra computadora.

Flujo de trabajo

Entre el *Directorio o Árbol de trabajo* y el *Repositorio o directorio Git* existe un paso intermedio o área especial llamada *Área de preparación o índice (Figura 1)*, la cual es esencialmente los cambios que se proponen para la próxima confirmación. Una vez que se terminan de realizar los cambios, se almacenan en esta área (con el comando git add file1 file2). Se revisan que estos cambios estén bien, y si lo están, se procede al paso de confirmación (con el comando git commit -m "texto que describe la confirmación") donde se guardaran los cambios haciendo una instantánea. Estos son guardados permanentemente en el *repositorio o directorio Git*, la cual es una base de datos que guarda el historial de los cambios realizados a un proyecto mediante una "confirmación", la cual es un mensaje significativo para representar a esta instantánea. Estos mensajes sirven para describir e identificar cada estado del proyecto. Una vez que se haya realizado una confirmación, no vaciar el *Área de preparación o índice* quitando los últimos cambios. Esto sería un error. Lo que se tiene en esta área es la misma instantánea que se guardó en el *Repositorio o directorio Git*.

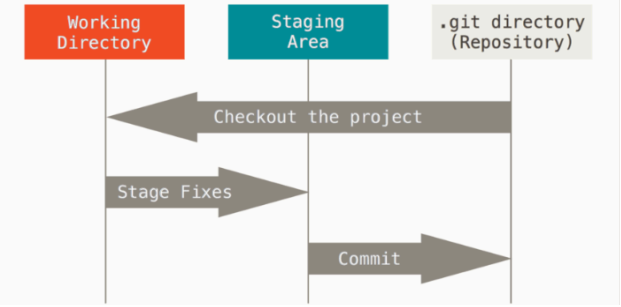


Figura 1. Árbol de trabajo, área de preparación y directorio Git

Comandos Básicos en GIT

En la terminal Git Bash, se iniciará de forma predeterminada en el directorio principal *(~)* o *C/Users/<User-of-windows-account>/ de su usuario.*

COMANDOS DE CONFIGURACIÓN (incorporación de credenciales)

Configurando el nombre de usuario en Git que se desea asociar a todos los commits en Git.
git config --global user.name "Tu-Nombre"

Configurando el correo electrónico en Git que se desea asociar a todos los commits en Git (De preferencia usar el mismo correo electrónico que se usa para la cuenta de GitHub u otros servicios Git).
git config --global user.email "tu-email@ejemplo.com"

Verificar las configuraciones para el nombre de usuario (Devolverá el nombre de usuario configurado actualmente en Git).
git config --global user.name

Verificar las configuraciones para el correo electrónico (Devolverá el correo electrónico configurado actualmente en Git).
git config --global user.email

COMANDOS QUE FUNCIONAN EN GIT BASH

$ mkdir nombre_Carpeta	crear carpeta
$ cd Nombre_Carpeta	acceso a carpeta Nombre_Carpeta
$ pwd	determinar el directorio actual (verificar)
$ ls	lista archivos y carpetas
$ clear	para limpiar la consola
$ cd ..	regresa a carpeta anterior

COMANDOS PARA INICIALIZAR REPOSITORIO DE GIT

Inicializar el repositorio o comenzar a rastrear git en el repositorio indicado (ya teniendo un proyecto realizado o si se va a comenzar uno). Aparece mensaje de inicialización. (*versiones anteriores a2.28.0 de Git*).
$ git init

Inicializar el nuevo repositorio (*versión 2.28.0 o una posterior de Git*) use los comandos siguientes:
$ git init --initial-branch=main
$ git init -b main

Para garantizar la compatibilidad futura, se recomienda actualizar el nombre de la rama de *máster* a *main* con el siguiente comando:
$ git branch -M main

COMANDOS BÁSICOS DE GIT

Para conocer el estado en el que se encuentran los archivos del repositorio (puede ser Tracked files o Untracked files).
$ git status

Para comenzar a rastrear los archivos. Añade un archivo específico a la próxima confirmación.
$ git add nombre_del_archivo

Para comenzar a rastrear los archivos y pasarlos al área de preparación, adiciona todos los archivos de una sola vez a la próxima confirmación.
$ git add .

Para confirmar los cambios o archivos y guardarlos en el repositorio o directorio Git.
$ git commit -m "*Mensaje de confirmación*"

Ver historial de modificaciones hechas en nuestro proyecto. Aunque se hayan hecho en otro lugar (como GitHub) de todas formas ese historial aparecerá.
$ git log

Ver historial (resumido) de modificaciones hechas en nuestro proyecto. (en GitHub esto se ve dando click en el relojito que dice commits)
$ git log --oneline

Ver historial (extenso) de modificaciones hechas en nuestro proyecto.
$ git log -p

Mostrar la información de la persona autora del commit.
$ git log --author="user_name"

Mostrar la información del commit por fecha.
$ git log --since=1.month.ago --until=1.day.ago

Volver exactamente al punto inicial del proyecto.
$ git restore

Restaurar archivo a una versión anterior.
$ git restore --source [*numero_hash_de_la_versión*] [*nombre_archivo.extension*]

COMANDOS DE COLABORACIÓN

Para clonar un proyecto de github en nuestra computadora o en la carpeta local. El url sera el enlace a un repositorio (como GitHub); p.e: *https://github.com/LuisErnestoYnz/sistema-de-registro.git*
$ git clone url

Clonar el repositorio para una carpeta específica:
$ git clone <repositorio> <mi-proyecto-clone>

Configurar el git clone y clonar el repositorio desde una branch específica, diferente a la original, de esta manera:
$ git clone -branch new_feature <repositorio>

Bajar todos los archivos cuando ya hemos empezado un repositorio (e.j. en github hacemos cambios y se reflejan en el git de escritorio con este comando).
$ git pull

Envía nuestros archivos al repositorio remoto (p.e. GitHub).
$ git push

Muestra los cambios realizados en los archivos del proyecto comparando el árbol de trabajo con todos los cambios que aún no se han almacenado provisionalmente. Mostrará un signo "más" delante de las líneas que se han agregado, y un signo "menos" indica las líneas que se han eliminado. o haciendo click sobre los mensajes de confirmación en github.
$ git diff

Comparar el árbol de trabajo con la última confirmación:
$ git diff HEAD.

Realizar conexión de github con la carpeta local (de la pc)
$ git remote add origin [*url de carpeta en github*]

Muestra conexiones remotas.
$ git remote -v

enviar el proyecto actual de la computadora local a un repositorio remoto (p.e. GitHub).
$ git branch -M main
$ git push u -origin main

COMANDOS PARA RAMAS (BRANCHES)

Muestra las ramas que se tienen.
$ git branch

Crea una nueva rama con nombre *nombre_rama*

$ git checkout -b nombre_rama

Cambiar de la rama actual a otra rama (en *nombre_rama* puede ir el nombre de la rama principal o el nombre de otra rama que queremos trabajar).
$ git switch nombre_rama

Enviar nuestros archivos al repositorio remoto. En *nombre_rama* es el nombre de una rama distinta de main
$ git push origin nombre_rama

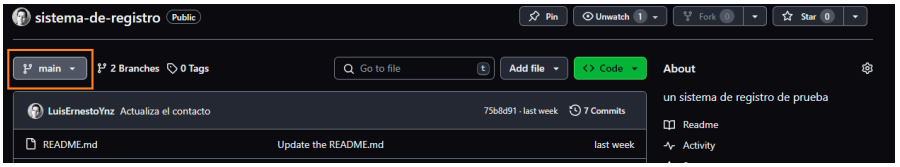
Poner contenido de la rama actual en *nombre_rama* (puede ir el nombre de la rama principal o el nombre de otra rama que queremos trabajar).
$ git merge *nombre_rama*

Trabajando con repositorios en GitHub (Branches, Merge, Conflicts)

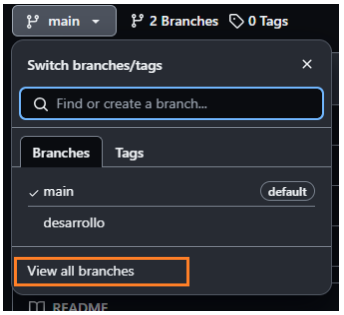
BRANCHES

CREACIÓN DE RAMAS EN REPOSITORIOS EN GITHUB

En GitHub.com, situarnos en la página principal del repositorio.
En la vista del árbol de archivos de la izquierda, seleccionar la opción de main.

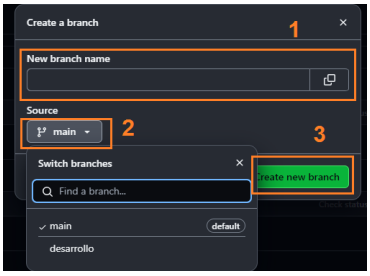


Hacer clic en Ver todas las ramas.



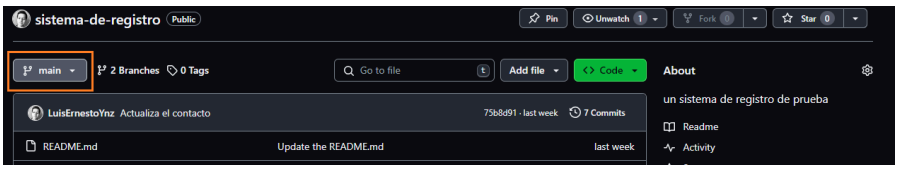
Haz clic en Nueva rama.

1. En "Nombre de rama", escriba un nombre para la rama.
2. En "source", elegir un origen para la rama. Si el repositorio es una bifurcación, Seleccione del menú desplegable la rama y haga clic en una rama.
3. clic en Crear rama.

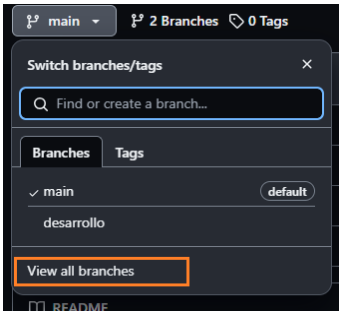


ELIMINACIÓN DE RAMAS EN REPOSITORIOS EN GITHUB

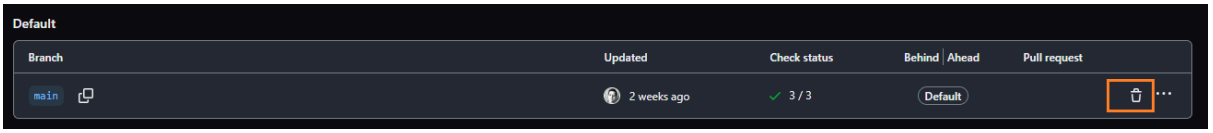
En GitHub.com, navega a la página principal del repositorio.



Hacer clic en Ver todas las ramas.



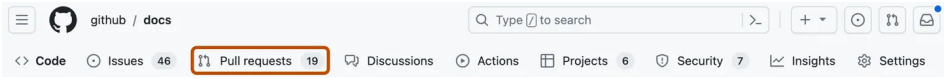
Junto a la rama que quieres borrar, haz clic en el icono de bote de basura. Esto eliminará la rama.



MERGE

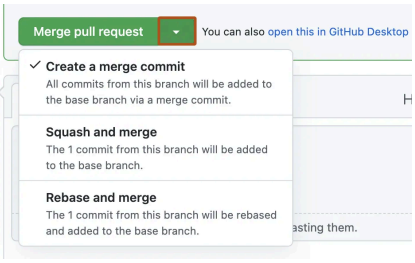
Combinación de una solicitud de incorporación de cambios
Cualquier persona con acceso de escritura al repositorio puede completar la fusión.
En una solicitud de extracción, propones que los cambios que hayas hecho en una rama de encabezado se fusionen en una rama base. Por defecto, cualquier solicitud de extracción se puede fusionar en cualquier momento, a menos que la rama de encabezado esté en conflicto con la rama base. Sin embargo, puede que existan restricciones sobre cuándo puedes fusionar una solicitud de cambios en una rama específica. Por ejemplo, puede que solo puedas fusionar una solicitud de extracción en la rama predeterminada si están pasando las verificaciones de estado requeridas. Los administradores del repositorio pueden agregar restricciones como esta a las ramas mediante reglas de protección de rama.
Se puede configurar una solicitud de cambios para que se fusione automáticamente cuando se cumplan todos los requisitos de fusión.
El repositorio podría configurarse para que la rama de encabezado para una solicitud de cambios se borre automáticamente cuando fusiones una solicitud de cambios.

Combinación de una solicitud de incorporación de cambios
En GitHub.com nos situamos en la página del repositorio. Hacer clic en Solicitudes de incorporación de cambios (pull request).



En la lista "Pull Requests" (Solicitudes de extracción), hacer clic en la **solicitud de extracción** que deseas fusionar.

Desplázate hasta la parte inferior de la solicitud de incorporación de cambios.
Para combinar todas las confirmaciones en la rama base, haga clic en **Combinar solicitud de incorporación de cambios**. Si no se muestra la opción Combinar solicitud de incorporación de cambios, haz clic en el menú desplegable de combinación y selecciona **Crear una confirmación de combinación**.



Para combinar con "squash" las confirmaciones en una sola, hacer clic en el menú desplegable de combinación, selecciona **Combinar y fusionar** y después, haz clic en el botón **Squash y combinar (squash and merge)**.

Fusiona mediante cambio de base las confirmaciones de forma individual en la rama base; para ello, haz clic en el menú desplegable de combinación, selecciona **Fusionar mediante cambio de base** y, después, haz clic en el botón **Fusionar mediante cambio de base y combinar**.

Nota: Fusionar mediante cambio de base y combinar siempre actualizará la información de la persona que confirma el cambio y creará SHA de confirmación. Si se te solicita, escribe un mensaje de confirmación o acepta el mensaje predeterminado.

Haga clic en **Confirm merge**, **Confirm squash and merge** o **Confirm rebase and merge**.
Opcionalmente, eliminar la rama. Esto mantiene ordenado el listado de ramas en tu repositorio.

CONFLICTS

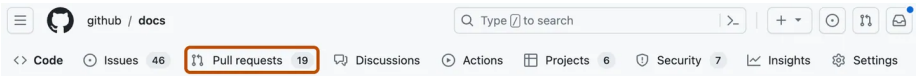
Los conflictos de fusión suceden cuando fusionas ramas que tienen confirmaciones de cambios contrapuestas y Git necesita de la ayuda del usuario para decidir qué cambios incorporar en la fusión final.
Por lo general, Git puede resolver las diferencias entre las ramas y fusionarlas automáticamente. Generalmente, los cambios están en diferentes líneas o incluso en diferentes archivos, lo que hace que sea simple para los equipos comprender la fusión. Sin embargo, a veces hay cambios contrapuestos que Git no puede resolver sin tu ayuda. Por lo general, los conflictos de fusión suceden cuando las personas realizan diferentes cambios en la misma línea en el mismo archivo o cuando una persona edita un archivo y otra persona elimina el mismo archivo.

Se deben resolver todos los conflictos de fusión antes de poder fusionar una solicitud de extracción en GitHub. Si tiene un conflicto de combinación entre la rama de comparación y la rama base de la solicitud de incorporación de cambios, puede ver una lista de los archivos con cambios en conflicto encima del botón Combinar solicitud de incorporación de cambios. El botón Combinar solicitud de incorporación de cambios se desactiva hasta que haya resuelto todos los conflictos entre la rama de comparación y la rama base.

Resolución de conflictos de combinación

Para resolver un conflicto de fusión, debes editar de forma manual el archivo conflictivo para seleccionar los cambios que quieres mantener en la fusión final.
Si tu conflicto de fusión es ocasionado por cambios de líneas contrapuestos, como cuando las personas realizan diferentes cambios en la misma línea del mismo archivo en diferentes ramas en tu repositorio de Git, lo puedes resolver en GitHub usando el editor de conflictos.

En el nombre del repositorio, haga clic en Solicitudes de incorporación de cambios (Pull request).



En la lista de "Pull Requests" (Solicitudes de extracción), haz clic en la solicitud de extracción con un conflicto de fusión que quieres resolver.

Junto a la parte inferior de la solicitud de incorporación de cambios, haga clic en Resolver conflictos.

Nota: Si el botón Resolver conflictos está desactivado, significa que el conflicto de combinación de la solicitud de incorporación de cambios es demasiado complejo para resolverlo en GitHub. Debes resolver el conflicto de fusión utilizando un cliente de Git alterno, o utilizando Git en la línea de comandos.

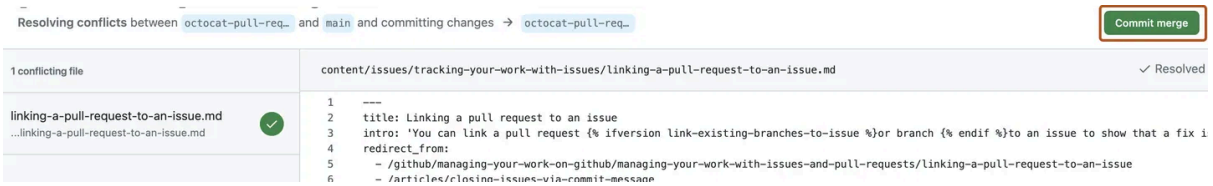
Decide si quieres mantener únicamente los cambios de tu rama, mantener únicamente los cambios de las demás ramas, o hacer un cambio nuevo, el cual puede incorporar cambios de ambas ramas. Elimine los marcadores `<<<<<<<`, `=======` y `>>>>>>>` en conflicto, y realice los cambios deseados en la combinación final.

Si tienes más de un conflicto de fusión en tu archivo, desplázate hacia abajo hasta el siguiente conjunto de marcadores de conflicto y repite los pasos cuatro y cinco para resolver el conflicto de fusión.

Una vez que haya resuelto todos los conflictos en el archivo, haga clic en **Marcar como resueltos**.

Si tienes más de un archivo con conflictos, selecciona el siguiente archivo que quieres editar del lado izquierdo de la página en "conflicting files" (archivos conflictivos) y repite los pasos cuatro a siete hasta que hayas resuelto todos los conflictos de fusión de tu solicitud de extracción.

Una vez que haya resuelto todos los conflictos de fusión mediante combinación, haga clic en **Confirmar combinación**. Esto fusiona toda la rama de base con tu rama de encabezado.



Si se te solicita, revisa la rama para la que vas a confirmar.

Si la rama principal es la rama predeterminada del repositorio, puedes escoger ya sea actualizar esta rama con los cambios que hiciste para resolver el conflicto, o crear una rama nueva y utilizarla como la rama principal de la solicitud de extracción.

Si eliges crear una rama nueva, ingresa un nombre para ésta.

Si la rama principal de tu solicitud de extracción está protegida, debes crear una rama nueva. No tendrás la opción para actualizar la rama protegida.

Haz clic en Crear rama y actualizar mi solicitud de incorporación de cambios o Entiendo, continuar la actualización de RAMA. El texto del botón corresponde a la acción que estás realizando.

Para fusionar mediante combinación la solicitud de incorporación de cambios, haga clic en **Combinar solicitud de incorporación de cambios**.

REFERENCIAS:

GIT
https://git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F

COMANDOS
https://learn.microsoft.com/es-es/training/modules/intro-to-git/
https://www.aluracursos.com/blog/iniciando-repositorio-con-git

CREACION Y ELIMINACION DE BRANCHES
https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-and-deleting-branches-within-your-repository

MERGE
https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request?tool=webui

CONFLICTS
https://docs.github.com/es/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-on-github