Git & GitHub

Bryan Ivan Montiel Ortega

Octubre 2023

Índice

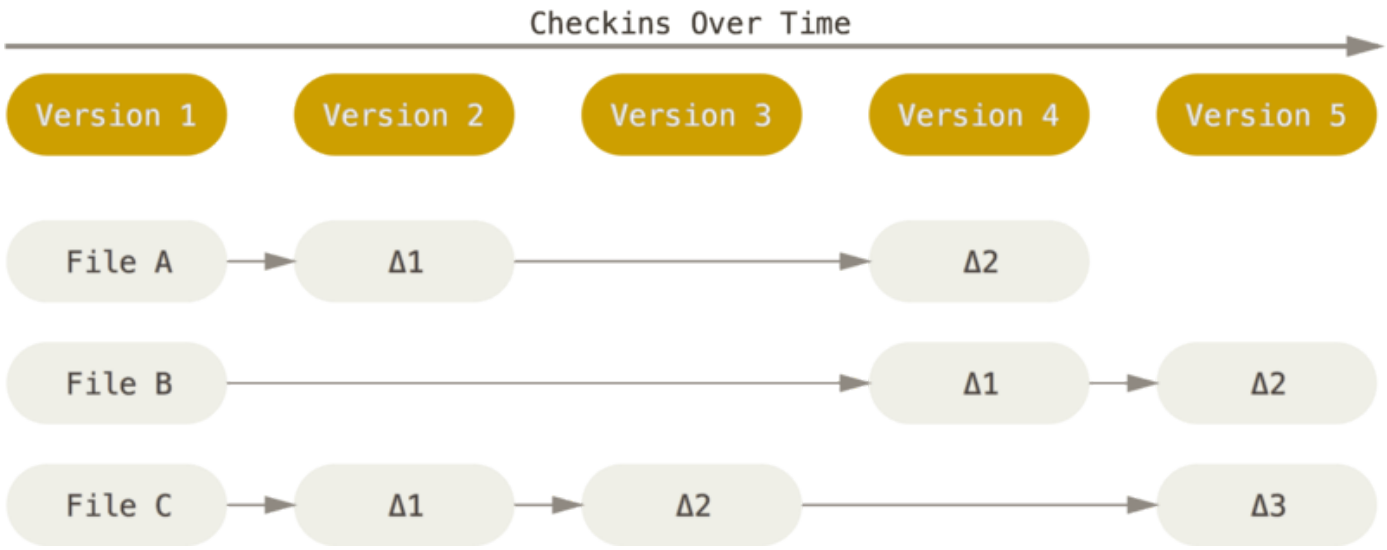
Introduction to Git

Git es un control de versiones distribuido (DVCS). La característica principal que distingue a Git como un sistema de control de versiones distribuido es su enfoque en la distribución y descentralización de los repositorios de código.

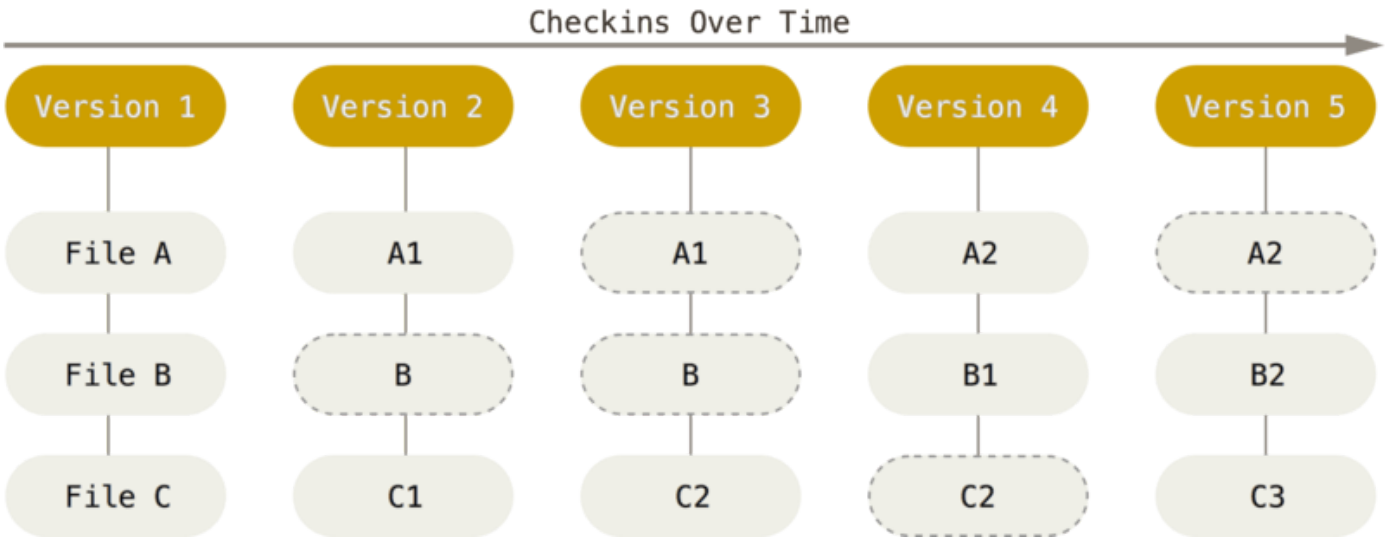
Git almacena y maneja la información de forma muy diferente a esos otros sistemas, a pesar de que su interfaz de usuario es bastante similar.

Copias instantáneas, no diferencias

La principal diferencia entre Git y cualquier otro VCS (incluyendo Subversion y sus amigos) es la forma en la que manejan sus datos. Conceptualmente, la mayoría de los otros sistemas almacenan la información como una lista de cambios en los archivos. Estos sistemas (CVS, Subversion, Perforce, Bazaar, etc.) manejan la información que almacenan como un conjunto de archivos y las modificaciones hechas a cada uno de ellos a través del tiempo.



Git no maneja ni almacena sus datos de esta forma. Git maneja sus datos como un conjunto de copias instantáneas de un sistema de archivos miniatura. Cada vez que confirmas un cambio, o guardas el estado de tu proyecto en Git, él básicamente toma una foto del aspecto de todos tus archivos en ese momento y guarda una referencia a esa copia instantánea. Para ser eficiente, si los archivos no se han modificado Git no almacena el archivo de nuevo, sino un enlace al archivo anterior idéntico que ya tiene almacenado. Git maneja sus datos como una secuencia de copias instantáneas.



Esta es una diferencia importante entre Git y prácticamente todos los demás VCS. Hace que Git reconsidere casi todos los aspectos del control de versiones que muchos de los demás sistemas copiaron de la generación anterior. Esto hace que Git se parezca más a un sistema de archivos miniatura con algunas herramientas poderosas desarrolladas sobre él, que a un VCS.

Git tiene integridad

Todo en Git es verificado mediante una suma de comprobación (checksum) antes de ser almacenado, y es identificado a partir de ese momento mediante dicha suma. Esto significa que es imposible cambiar los contenidos de cualquier archivo o directorio sin que Git lo sepa. Esta funcionalidad está integrada en Git al más bajo nivel y es parte integral de su filosofía. No puedes perder información durante su transmisión o sufrir corrupción de archivos sin que Git sea capaz de detectarlo.

El mecanismo que usa Git para generar esta suma de comprobación se conoce como hash SHA-1. Se trata de una cadena de 40 caracteres hexadecimales (0-9 y a-f), y se calcula con base en los contenidos del archivo o estructura del directorio en Git. Un hash SHA-1 se ve de la siguiente forma:

```
24b9da6552252987aa493b52f8696cd6d3b00373
```

Git guarda todo no por nombre de archivo, sino por el valor hash de sus contenidos.

Git generalmente solo añade información

Cuando realizas acciones en Git, casi todas ellas sólo añaden información a la base de datos de Git. Es muy difícil conseguir que el sistema haga algo que no se pueda enmendar, o que de algún modo borre información. Como en cualquier VCS, puedes perder o estropear cambios que no has confirmado

todavía. Pero después de confirmar una copia instantánea en Git es muy difícil perderla, especialmente si envías tu base de datos a otro repositorio con regularidad.

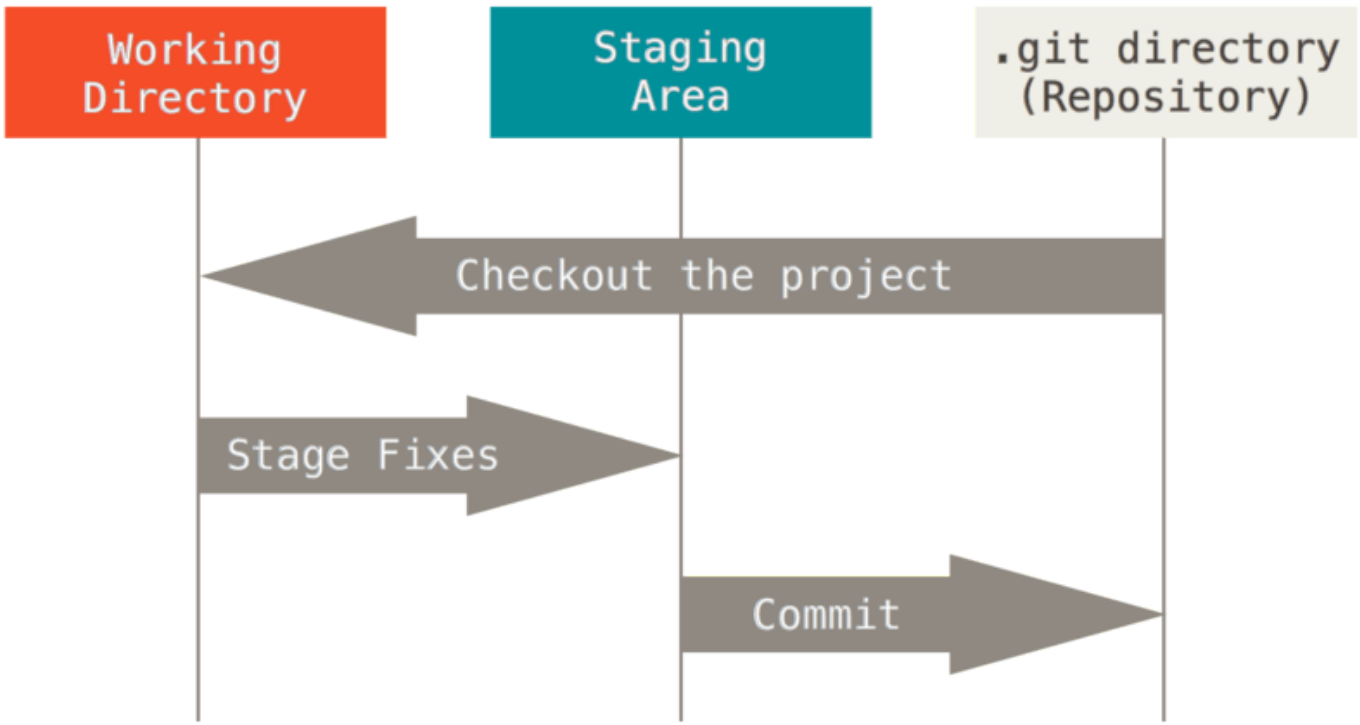
Esto hace que usar Git sea un placer, porque sabemos que podemos experimentar sin peligro de estropear gravemente las cosas. Para un análisis más exhaustivo de cómo almacena Git su información y cómo puedes recuperar datos aparentemente perdidos, ver Deshacer Cosas.

Los Tres Estados

Git tiene tres estados principales en los que se pueden encontrar tus archivos: confirmado (committed), modificado (modified), y preparado (staged).

- Confirmado: significa que los datos están almacenados de manera segura en tu base de datos local.
- Modificado: significa que has modificado el archivo pero todavía no lo has confirmado a tu base de datos.
- Preparado: significa que has marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.

Esto nos lleva a las tres secciones principales de un proyecto de Git: El directorio de Git (Git directory), el directorio de trabajo (working directory), y el área de preparación (staging area).



El directorio de Git es donde se almacenan los metadatos y la base de datos de objetos para tu proyecto. Es la parte más importante de Git, y es lo que se copia cuando clonas un repositorio desde otra computadora.

El directorio de trabajo es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que los puedas usar o modificar.

El área de preparación es un archivo, generalmente contenido en tu directorio de Git, que almacena información acerca de lo que va a ir en tu próxima confirmación. A veces se le denomina índice (“index”), pero se está convirtiendo en estándar el referirse a ella como el área de preparación.

- El flujo de trabajo básico en Git es algo así:
- Modificas una serie de archivos en tu directorio de trabajo.
- Preparas los archivos, añadiéndolos a tu área de preparación.

Confirmas los cambios, lo que toma los archivos ten el área de preparación y almacena esa copia instantánea de manera permanente en tu directorio de Git.

Si una versión concreta de un archivo está en el directorio de Git, se considera confirmada (committed). Si ha sufrido cambios desde que se obtuvo del repositorio, pero ha sido añadida al área de preparación, está preparada (staged). Y si ha sufrido cambios desde que se obtuvo del repositorio, pero no se ha preparado, está modificada (modified).

Basic Commands in Git

La siguiente lista se basa en la lista compartida por GitHub, puede revisarse la dirección en la sección de bibliografía.

Configuración de Herramientas

$ git config --global

Para la primera configuración de Git en tu PC. Va a afectar a todo lo que se haga desde git en esa máquina (PC).

$ git config --global user.name "[name]"

Establece el nombre que desea esté anexado a sus transacciones de commit. Normalmente la crea en Users.

$ git config --global user.email "[email address]"

Establece el e-mail que desea esté anexado a sus transacciones de commit.

Crear Repositorios

$ git init [project-name]

Para indicar que queremos trabajar en un directorio con Git. Es cuando queremos inicializar el contexto de un control de versiones ‘aqui’ refiriéndose a la raíz donde se realiza. Crea un nuevo repositorio local con el nombre especificado.

$ git clone [url]

Descarga un proyecto y toda su historia de versión.

Efectuar Cambios

$ git status

Enumera todos los archivos nuevos o modificados que se deben confirmar. Puede verse los cambios en los archivos y el estado en que se encuentran, en ‘stage’. Si no existe algo, este estado te mostrará que no hay nada para hacerle ‘commit’.

$ git add [file]

Toma una instantánea del archivo para preparar la versión. Si es la primera vez, creará la primer 'fotografía' de tu proyecto.

$ git reset [file]

Mueve el archivo del área de espera, pero preserva su contenido. Sirve para revisar cual archivo ha tenido algún cambio, posiblemente inesperado, y puedes darle un 'checkout' para recuperar un estado antes de pasar por 'commit'.

$ git diff

Muestra las diferencias de archivos que no se han enviado aún al área de espera. Puedes revisar ambos archivos y decidir por cual cambio optar, Git muestra las diferencias entre archivos con ayuda de comentarios

$ git diff --staged

Muestra las diferencias del archivo entre el área de espera y la última versión del archivo.

$ git commit -m "[descriptive message]"

Registra las instantáneas del archivo permanentemente en el historial de versión. Si no ingresas el mensaje no dejará hacer el 'commit'.

Cambios Grupales

$ git branch

Enumera todas las ramas en el repositorio actual.

$ git branch [branch-name]

Crea una nueva rama

$ git checkout [branch-name]

Cambia a la rama especificada y actualiza el directorio activo.

$ git merge [branch]

Combina el historial de la rama especificada con la rama actual

$ git branch -d [branch-name]

Borra la rama especificada

$ git tag [tag-name]

Puede hacer un tag, utilizado para algún punto importante o de referencia, si se coloca sin nombre este mostrará el listado de 'tags'

Nombres de archivo de refactorización

$ git rm [file]

Borra el archivo del directorio activo y pone en el área de espera el archivo borrado.

$ git rm --cached [file]

Retira el archivo del control de versiones, pero preserva el archivo a nivel local

$ git mv [file-original] [file-renamed]

Cambia el nombre del archivo y lo prepara para commit

Guardar Fragmentos

$ git stash

Almacena temporalmente todos los archivos tracked modificados

$ git stash list

Enumera todos los sets de cambios en guardado rápido

$ git stash pop

Restaura los archivos guardados más recientemente

$ git stash drop

Elimina el set de cambios en guardado rápido más reciente

Repasar Historial

$ git log

Enumera el historial de la versión para la rama actual

$ git log --follow [file]

Enumera el historial de versión para el archivo, incluidos los cambios de nombre

$ git diff [first-branch]...[second-branch]

Muestra las diferencias de contenido entre dos ramas

$ git show [commit]

Produce metadatos y cambios de contenido del commit especificado

Rehacer commits

$ git reset [commit]

Deshace todos los commits después de [commit], preservando los cambios localmente

$ git reset --hard [commit]

Desecha todo el historial y regresa al commit especificado

Sincronizar Cambios

$ git fetch [bookmark]

Descarga todo el historial del marcador del repositorio

$ git merge [bookmark]/[branch]

Combina la rama del marcador con la rama local actual

$ git push [alias] [branch]

Carga todos los commits de la rama local al GitHub

$ git pull

Descarga el historial del marcador e incorpora cambios

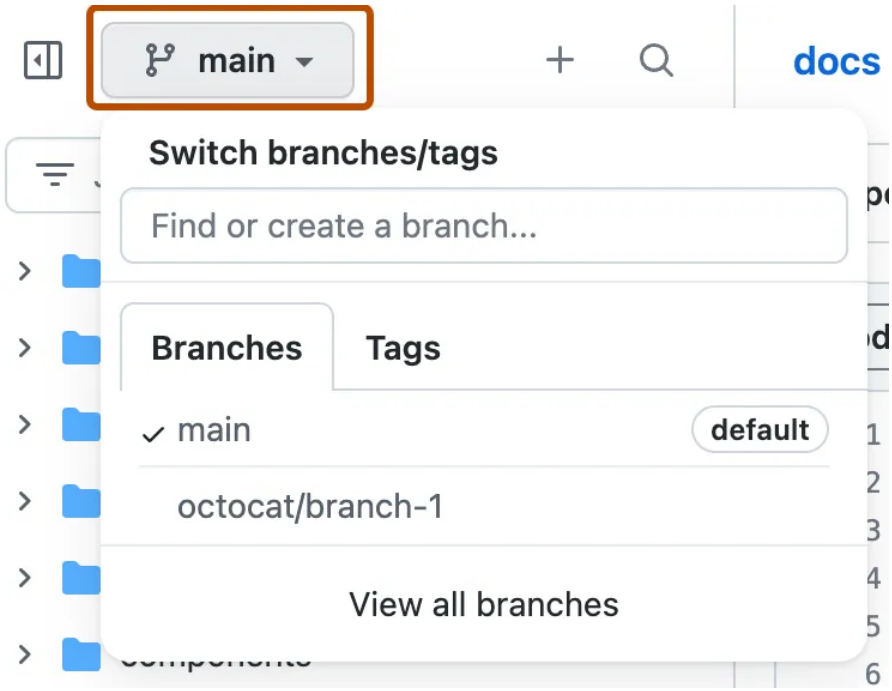
Working with repositories on GitHub (Branches, Merge, Conflicts)

Crear y eliminar ramas en tu repositorio

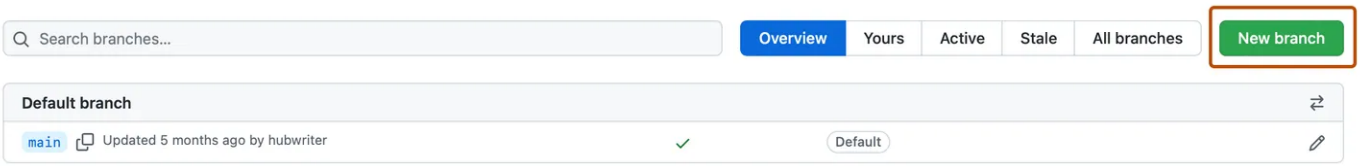
Puedes crear una rama de diferentes maneras enGitHub.

Creación de una rama en la página de información general de ramas

1. En GitHub.com, navega a la página principal del repositorio.
2. En la vista de árbol de archivos de la izquierda, selecciona el menú desplegable de ramas de y, a continuación, haz clic en Ver todas las ramas. También puedes encontrar el menú desplegable de ramas en la parte superior del editor de archivos integrado.



3. Haz clic en Nueva rama.



4. En "Nombre de rama", escriba un nombre para la rama.
5. En "Origen de rama", elija un origen para la rama.
    - Si el repositorio es una bifurcación, seleccione el menú desplegable del repositorio y haga clic en la bifurcación o en el repositorio ascendente.

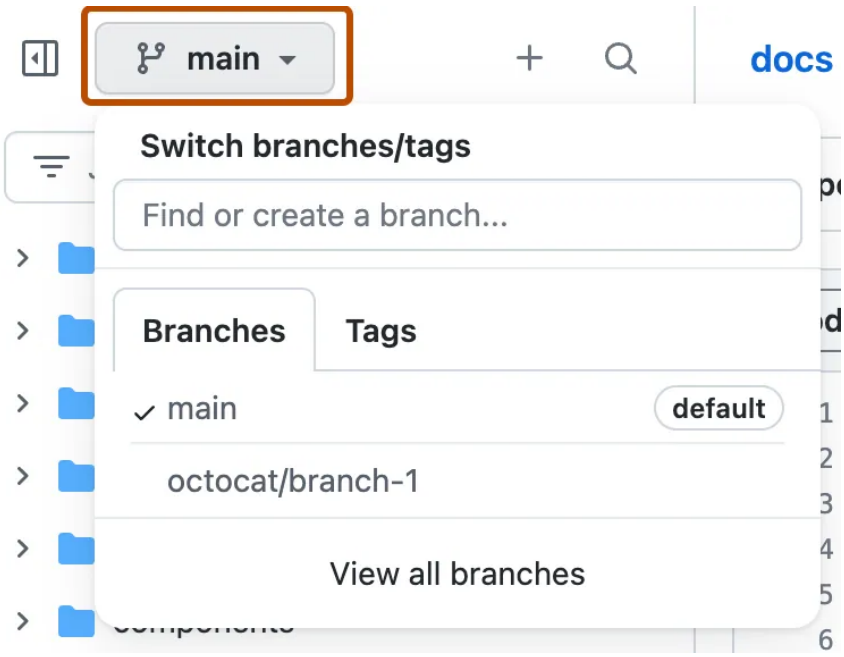
- Seleccione el menú desplegable rama y haga clic en una rama.

6. Haz clic en Crear rama.

Cómo eliminar una rama

Si la rama que quieres borrar está asociada con una solicitud de cambios abierta, debes fusionar o cerrar dicha solicitud antes de borrar la rama.

1. En GitHub.com, navega a la página principal del repositorio.
2. En la vista de árbol de archivos de la izquierda, selecciona el menú desplegable de ramas de y, a continuación, haz clic en Ver todas las ramas. También puedes encontrar el menú desplegable de ramas en la parte superior del editor de archivos integrado.



3. Junto a la rama que quieres borrar, haz clic en 🗑️.

4. Si la rama está asociada con al menos una solicitud de incorporación de cambios abierta, la eliminación de la rama cerrará las solicitudes de incorporación de cambios. Lea la advertencia y haga clic en Eliminar.

Si borras una rama de encabezado después de haber fusionado su solicitud de extracción, GitHub verificará cualquier solicitud de extracción abierta en el mismo repositorio que especifique la rama borrada como su rama base. GitHub actualiza automáticamente cualquier solicitud de extracción, cambiando su rama base a la rama base de la solicitud de extracción que se ha fusionado.

Acerca de las fusiones de las solicitudes de extracción

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 o conjuntos de reglas de repositorio.

Puedes configurar una solicitud de cambios para que se fusione automáticamente cuando se cumplan todos los requisitos de fusión.

Si la solicitud de incorporación de cambios tiene conflictos de combinación, o bien si quiere probar los cambios antes de la combinación, puede extraer del repositorio local la solicitud de incorporación de cambios y combinarla desde la línea de comandos.

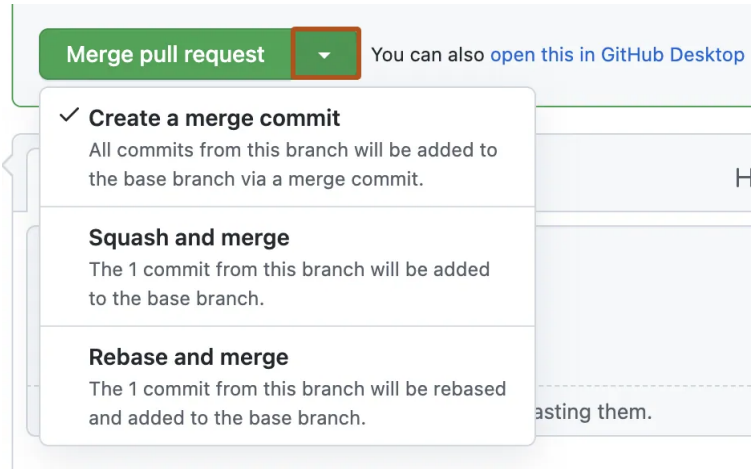
El repositorio podría configurarse para que la rama de encabezado para una solicitud de cambios se borre automáticamente cuando fusiones una solicitud de cambios.

Puedes vincular una solicitud de incorporación de cambios a una incidencia para mostrar que una corrección está en curso y para cerrar automáticamente la incidencia cuando un usuario fusione mediante combinación dicha solicitud.

Si decide que no quiere que los cambios en una rama de tema se combinen con la rama ascendente, puede cerrar la solicitud de incorporación de cambios sin combinarla.

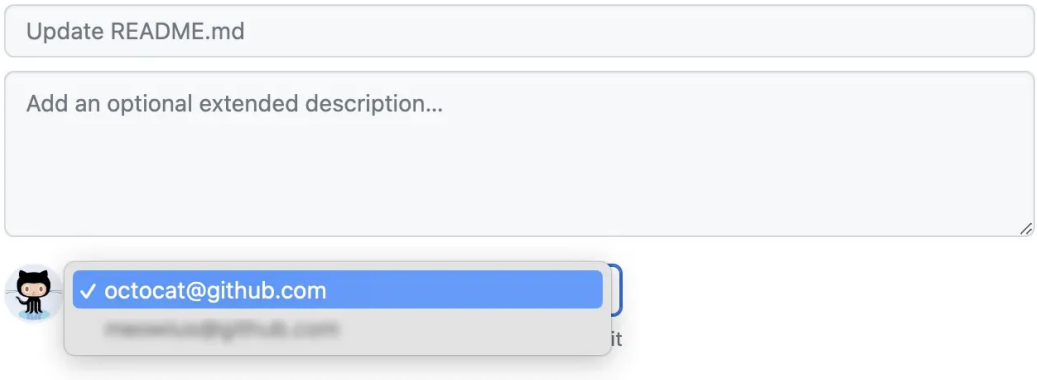
Combinación de una solicitud de incorporación de cambios

1. En el nombre del repositorio, haga clic en Solicitudes de incorporación de cambios.
2. En la lista "Pull Requests" (Solicitudes de extracción), haz clic en la solicitud de extracción que deseas fusionar.
3. Desplázate hasta la parte inferior de la solicitud de incorporación de cambios. Según las opciones de fusión habilitadas para tu repositorio, puedes:
    - 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, haz clic en el menú desplegable de combinación, selecciona Combinar y fusionar y, después, haz clic en el botón Squash y combinar.
- 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.
-

4. Si se te solicita, escribe un mensaje de confirmación o acepta el mensaje predeterminado.
5. Si tienes más de una dirección de correo electrónico asociada a la cuenta en GitHub.com, haz clic en el menú desplegable de la dirección de correo electrónico y selecciona la que se usará como dirección de correo electrónico del creador de Git. Únicamente las direcciones de correo electrónico verificadas aparecen en el menú desplegable. Si ha habilitado la privacidad de la dirección de correo electrónico, una dirección de correo electrónico no responder del creador de la confirmación será la predeterminada.



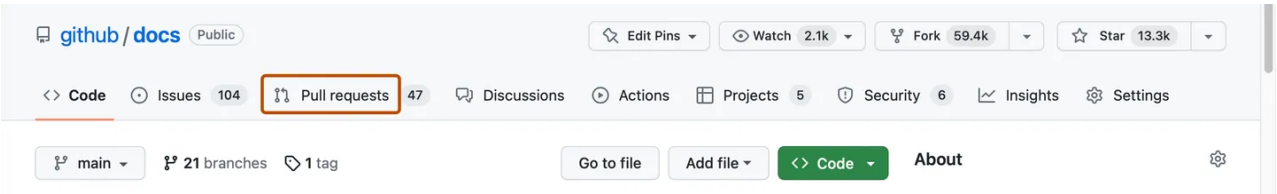
6. Haga clic en Confirm merge, Confirm squash and merge o Confirm rebase and merge.
7. Opcionalmente, elimine la rama. Esto mantiene ordenado el listado de ramas en tu repositorio.

Resolver un conflicto de fusión en GitHub

Puedes resolver conflictos de fusión simples que impliquen realizar cambios de líneas en GitHub, usando el editor de conflictos.

Solo puedes resolver los conflictos de fusión en GitHub que hayan sido provocados por realizar cambios de líneas, como cuando las personas hacen cambios diferentes en la misma línea del mismo archivo en ramas diferentes de tu repositorio de Git.

1. En el nombre del repositorio, haga clic en Solicitudes de incorporación de cambios.



2. En la lista de "Pull Requests" (Solicitudes de extracción), haz clic en la solicitud de extracción con un conflicto de fusión que quieres resolver.
3. Junto a la parte inferior de la solicitud de incorporación de cambios, haga clic en Resolver conflictos.

4. 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.
5. 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.
6. Una vez que haya resuelto todos los conflictos en el archivo, haga clic en Marcar como resueltos.

7. 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.
8. 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.

9. 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.
10. Para fusionar mediante combinación la solicitud de incorporación de cambios, haga clic en Combinar solicitud de incorporación de cambios.

Bibliografia

1. Git - fundamentos de git. Git-scm.com. Recuperado el 27 de octubre de 2023, de https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Fundamentos-de-Git
2. GItHub. “Combinación de Una Solicitud de Incorporación de Cambios - Documentación de GitHub.” GitHub Docs, docs.github.com/es/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/merging-a-pull-request. Accessed 27 Oct. 2023.
3. GitHub. “Crear Y Eliminar Ramas En Tu Repositorio - Documentación de GitHub.” GitHub Docs, 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. Accessed 27 Oct. 2023.
4. GitHub. “Resolver Un Conflicto de Fusión En GitHub - Documentación de GitHub.” GitHub Docs, docs.github.com/es/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-on-github. Accessed 27 Oct. 2023.
5. Las líneas de comando de Git más comúnmente usadas., HOJA DE REFERENCIA PARA GITHUB GIT. Github.com. Recuperado el 27 de octubre de 2023, de https://training.github.com/downloads/es_ES/github-git-cheat-sheet.pdf
6. MoureDev. (2023, febrero). Curso de GIT y GITHUB desde CERO para PRINCIPIANTES. YouTube. https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Fundamentos-de-Git