COMANDOS AVANZADOS DE GIT

Fork

Los forks o bifurcaciones son una característica única de GitHub en la que se crea una copia exacta del estado actual de un repositorio directamente en GitHub. Este repositorio podrá servir como otro origen y se podrá clonar (como cualquier otro repositorio). En pocas palabras, lo podremos utilizar como un nuevo repositorio git cualquiera

Un fork es como una bifurcación del repositorio completo. Comparte una historia en común con el original, pero de repente se bifurca y pueden aparecer varios cambios, ya que ambos proyectos podrán ser modificados en paralelo y para estar al día un colaborador tendrá que estar actualizando su fork con la información del original.

Al hacer un fork de un poryecto en GitHub, te conviertes en dueño del repositorio fork, puedes trabajar en este con todos los permisos, pero es un repositorio completamente diferente que el original, teniendo solamente alguna historia en común (como crédito al creado o creadora original).

Los forks son importantes porque es la manera en la que funciona el open source, ya que, una persona puede no ser colaborador de un proyecto, pero puede contribuír al mismo, haciendo mejor software que pueda ser utilizado por cualquiera.

Como se hace un fork

Al hacer un fork, GitHub sabe que se hizo el fork del proyecto, por lo que se le permite al colaborador hacer pull request desde su repositorio propio.

Cuando trabajas en un proyecto que existe en diferentes repositorios remotos (normalmente a causa de un fork), es muy probable que desees poder trabajar con ambos repositorios. Para esto, puedes generar un remoto adicional desde consola.

```
git remote add <nombre_del_remoto> <url_del_remoto>
git remote upstream https://github.com/freddier/hyperblog
```

Al crear un remoto adicional, podremos hacer pull desde el nuevo origen. En caso de tener permisos, podremos hacer fetch y push.

```
git pull <remoto> <rama>
git pull upstream master
```

Este pull nos traerá los cambios del remoto, por lo que se estará al día en el proyecto. El flujo de trabajo cambia, en adelante se estará trabajando haciendo pull desde el upstream y push al origin para pasar a hacer pull request.

```
git pull upstream master
 git push origin master
```

Pull Request

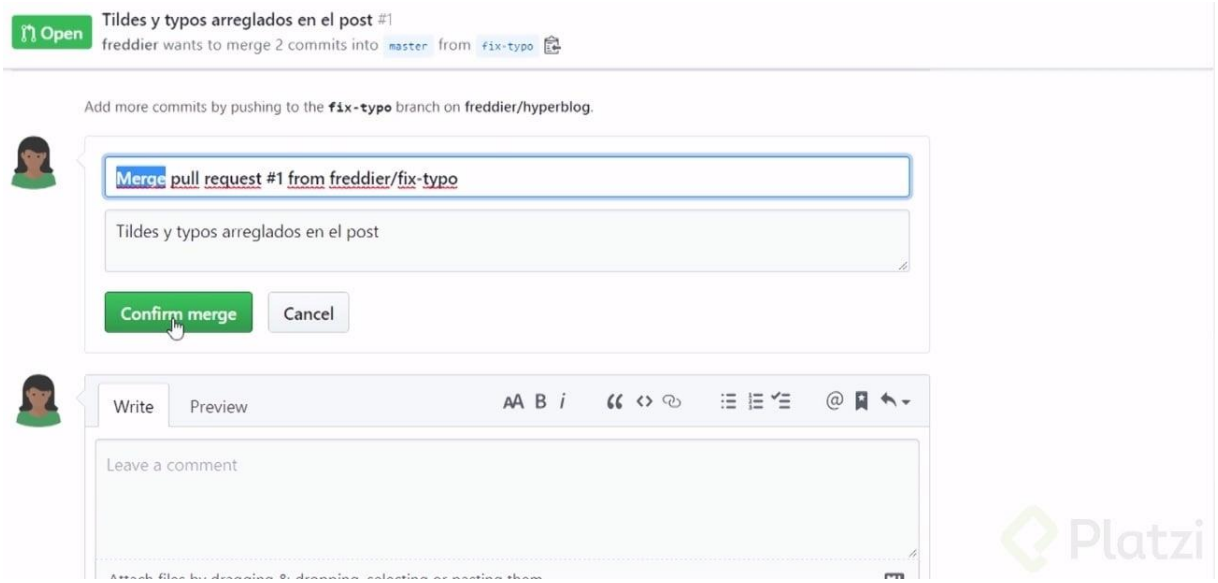
El sistema de control de versiones git, ofrece una solución muy conveniente a la hora de llevar control sobre el desarrollo de un proyecto con el cual podemos ver el progreso del trabajo solo con inspeccionar el cambio realizado en cada commit. Los pull request permiten no solo llevar de forma más ordenada las tareas en la etapa del desarrollo, sino también crear propuestas o cambios que puedan ser integrados posteriormente a dicho proyecto.

Básicamente un pull request es una petición para integrar nuestras propuestas o cambios de código a un proyecto.

Cuando estás trabajando como parte de un equipo, debes ser un poco más cuidadoso, es allí donde entra en juego el uso de los pull request, cada vez que creas un cambio, es recomendable subir al repositorio central la nueva rama de trabajo, digamos «front-page» y con un pull request estarías haciendo una petición de que esta rama sea incluida en master, entonces, de esta forma puedes dar una explicación mucho más detallada de lo que hace tu código (más allá del mensaje del commit) y la persona encargada de la integración puede descargar tu rama, probar los cambios y aprobar o rechazar la petición, según sea el caso, inclusive dejar un comentario para que hagas algún arreglo si es requerido.

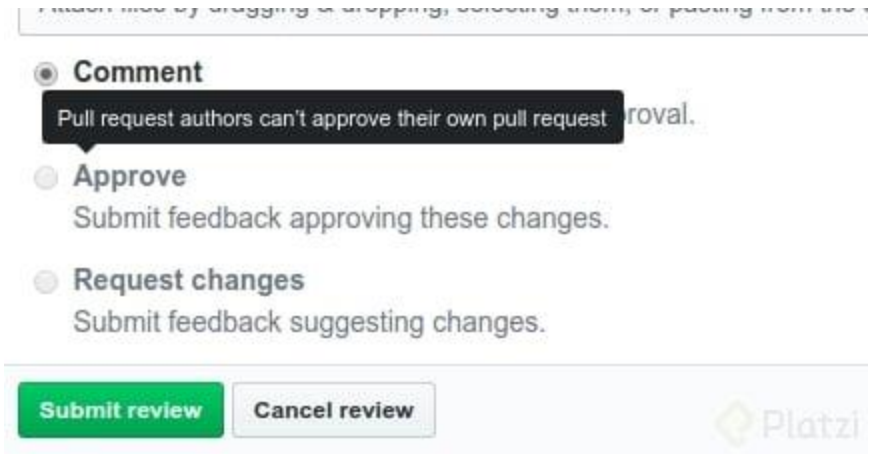
Crear un pull request

1. Crea una rama paralela: Antes de hacer cambios en el código, utiliza el comando git checkout -b <rama> para crear una nueva rama. Así, podrás hacer tus modificaciones sin afectar la rama principal (por ejemplo, master).
2. Realiza commits: Después de hacer cambios en los archivos, usa git commit -am '<Comentario>' para hacer un commit con un mensaje descriptivo.
3. Sube los cambios: Usa git push origin <rama> para subir tus cambios de la rama local al repositorio remoto. Reemplaza <rama> con el nombre de tu rama.
4. Crea un pull request: En el repositorio remoto (como GitHub), crea un nuevo pull request. Selecciona la rama principal como destino y tu rama con los cambios como comparación.
5. Feedback: Los revisores examinarán los cambios. Usa la sección de comentarios del pull request para discutir los cambios y proporcionar feedback adicional.
6. Realiza los cambios solicitados: Si se solicitan cambios, regresa a tu rama local y haz las modificaciones necesarias. Luego, sube los cambios al repositorio remoto usando git push origin <rama>.



Aceptar un pull request

1. Acepta los cambios en GitHub: Si estás satisfecho con los cambios propuestos en el pull request y consideras que están listos para ser fusionados con la rama principal, acepta el pull request en GitHub. De esta forma, los cambios se fusionarán en la rama principal del repositorio.
2. Realiza el merge en la rama principal: Después de aceptar el pull request, selecciona la opción para realizar el merge en GitHub. Esto combinará los cambios de la rama con los cambios existentes en la rama principal (master).



Rebase

El comando `git rebase` te permite cambiar fácilmente una serie de confirmaciones, modificando el historial de tu repositorio. Puedes reordenar, editar o combinar confirmaciones.

Normalmente, se usaría git rebase para lo siguiente:

- Editar mensajes de confirmación previos.
- Combinar varias confirmaciones en una.
- Eliminar o revertir confirmaciones que ya no son necesarias.

Cambiar de base las confirmaciones de una rama
Para cambiar de base todas las confirmaciones entre otra rama y el estado de rama actual, puedes ingresar el siguiente comando en tu shell (ya sea el símbolo del sistema para Windows o la terminal para Mac y Linux):

```
git rebase --interactive OTHER-BRANCH-NAME
```

Cambiar de base las confirmaciones en un momento especifico

Para cambiar de base las últimas confirmaciones en tu rama actual, puedes ingresar el siguiente comando en tu shell:

```
git rebase --interactive HEAD~7
```

Comandos disponibles mientras se cambia de base

Hay seis comandos disponibles mientras se cambia la base:

pick

pick simplemente significa que se incluye la confirmación. Reordenar los comandos pick cambia el orden de las confirmaciones cuando la fusión mediante cambio de base está en curso. Si eliges no incluir una confirmación, debes eliminar la línea completa.

reword

El comando reword es similar a pick, pero después de usarlo, la fusión mediante cambio de base se pausará y le dará la oportunidad de modificar el mensaje de confirmación. Cualquier cambio hecho por la confirmación no se ve afectado.

edit

Si elige realizar edit en una confirmación, se le dará la oportunidad de modificar la confirmación, lo que significa que puede agregar o cambiar la confirmación por completo. También puedes realizar más confirmaciones antes de continuar con el cambio de base. Esto te permite dividir una confirmación grande en otras más pequeñas o eliminar cambios erróneos hechos en una confirmación.

squash

Este comando te permite combinar dos o más confirmaciones en una única confirmación. Una confirmación se combina en la confirmación de arriba. Git te da la oportunidad de escribir un mensaje de confirmación nuevo describiendo ambos cambios.

fixup

Esto es similar a squash, pero la confirmación que se va a combinar tiene su mensaje descartado. La confirmación simplemente se fusiona en la confirmación de arriba y el mensaje de la confirmación anterior se usa para describir ambos cambios.

exec

Esto te permite ejecutar comandos shell de forma arbitraria con una confirmación.

Fuentes bibliográficas:

- https://platzi.com/clases/1557-git-github/19978-creando-un-fork-contribuyendo-a-un-repositorio/#:~:text=Un%20fork%20es%20como%20una%20bifurcaci%C3%B3n%20del%20repositorio%20completo.
- https://platzi.com/clases/1557-git-github/19957-utilizando-pull-requests-en-github/
- https://styde.net/pull-request-en-github/
-