Resolución de conflictos al hacer un merge 17/43

RECURSOS MARCADORES

Git nunca borra nada, a menos que nosotros se lo indiquemos. Cuando usamos los comandos `git merge` o `git checkout` estamos cambiando de rama o creando un nuevo *commit*, no borrando ramas ni *commits* (recuerda que puedes borrar commits con `git reset` y ramas con `git branch -d`).

Git es muy inteligente y puede resolver algunos conflictos automáticamente: cambios, nuevas líneas, entre otros. Pero algunas veces no sabe cómo resolver estas diferencias, por ejemplo, cuando dos ramas diferentes hacen cambios distintos a una misma línea.

Esto lo conocemos como **conflicto** y lo podemos resolver manualmente. Solo debemos hacer el *merge*, ir a nuestro editor de código y elegir si queremos quedarnos con alguna de estas dos versiones o algo diferente. Algunos editores de código como Visual Studio Code nos ayudan a resolver estos conflictos sin necesidad de borrar o escribir líneas de texto, basta con hacer clic en un botón y guardar el archivo.

Recuerda que siempre debemos crear un nuevo commit para aplicar los cambios del merge. Si Git puede resolver el conflicto, hará commit automáticamente. Pero, en caso de no pueda resolverlo, debemos solucionarlo y hacer el commit.

Los archivos con conflictos por el comando `git merge` entran en un nuevo estado que conocemos como ***Unmerged***. Funcionan muy parecido a los archivos en estado *Unstaged*, algo así como un estado intermedio entre Untracked y Unstaged. Solo debemos ejecutar `git add` para pasarlos al área de staging y `git commit` para aplicar los cambios en el repositorio.

Cómo revertir un merge

Si nos hemos equivocado y queremos cancelar el merge, debemos usar el siguiente comando:

```
git merge --abort
```

Conflictos en repositorios remotos

Al trabajar con otras personas, es necesario utilizar un repositorio remoto.

-Para copiar el repositorio remoto al directorio de trabajo local, se utiliza el comando `git clone <url>` , y para enviar cambios al repositorio remoto se utiliza `git push` .
-Para actualizar el repositorio local se hace uso del comando `git fetch` , luego se debe fusionar los datos traídos con los locales usando `git merge` .

- Para traer los datos y fusionarlos a la vez, en un solo comando, se usa `git pull` .
  - Para crear commits rápidamente, fusionando `git add` y `git commit -m ""` , usamos `git commit -am ""` .
  - Para generar nuevas ramas, hay que posicionarse sobre la rama que se desea copiar y utilizar el comando `git branch <nombre>` .
- Para saltar entre ramas, se usa el comando `git checkout <branch>`
  - Una vez realizado los cambios en la rama, estas deben fusionarse con `git merge` .
- El merge ocurre en la rama en la que se está posicionado. Por lo tanto, la rama a fusionar se transforma en la principal.
- Los merges también son commits.
- Los merges pueden generar conflictos, esto aborta la acción y pide que soluciones el problema manualmente, aceptando o rechazando los cambios que vienen.