educación IT
CODE YOUR FUTURE

Git:
Desarrollo colaborativo

Módulo 2

educación IT
CODE YOUR FUTURE

Combinación de ramas

Integrar cambios

Al trabajar con GIT y sus *branches*, se observa fácilmente que **el *branch* de una nueva funcionalidad siempre va a quedar por "encima" del *branch* master** que contiene el código estable (aquel que ya pasó por las pruebas necesarias para confirmar que es válido y puede estar en producción). Esto ocurre en el servidor, por ejemplo, durante el desarrollo de una aplicación web.

Cada vez que trabajemos sobre alguno de estos branches de funcionalidades, luego de testearlos y confirmar su aporte al proyecto, desearemos integrarlos a nuestro código oficial.

Para esto **GIT ofrece varias herramientas de integración, unificación, fusión o reubicación de trabajo**.

git merge

Si solamente se desean **incorporar los cambios nuevos** que hayamos ya pasado a través de pruebas oportunas, podemos utilizar el comando **`git merge`**:

```
> git merge <branch>
```

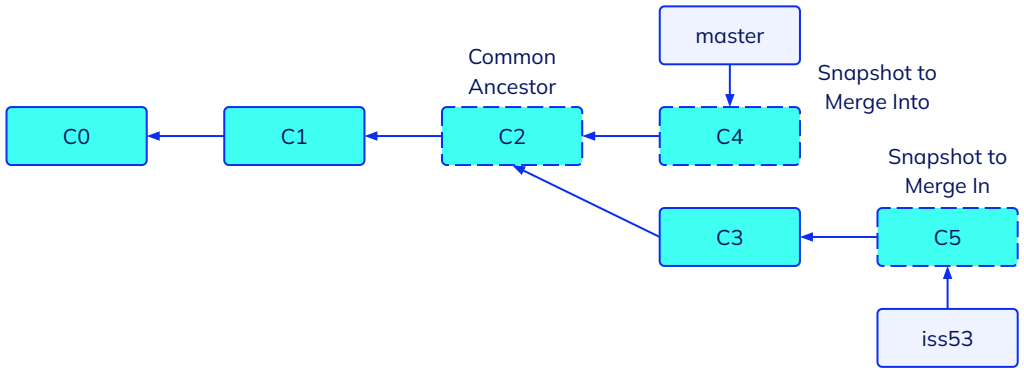
Según cómo esté distribuido el repositorio local, en muchos casos, aparecerá la frase ***"Fast forward"*** en la salida del comando.

GIT ha movido el apuntador hacia adelante, ya que la confirmación apuntada en la rama donde has fusionado estaba directamente arriba respecto a la confirmación actual.

Dicho de otro modo: cuando intentas fusionar una confirmación con otra confirmación accesible siguiendo directamente el historial de la primera; **GIT simplifica las cosas avanzando el puntero, ya que no hay ningún otro trabajo divergente a fusionar. Esto es lo que se denomina "avance rápido" ("fast forward").**

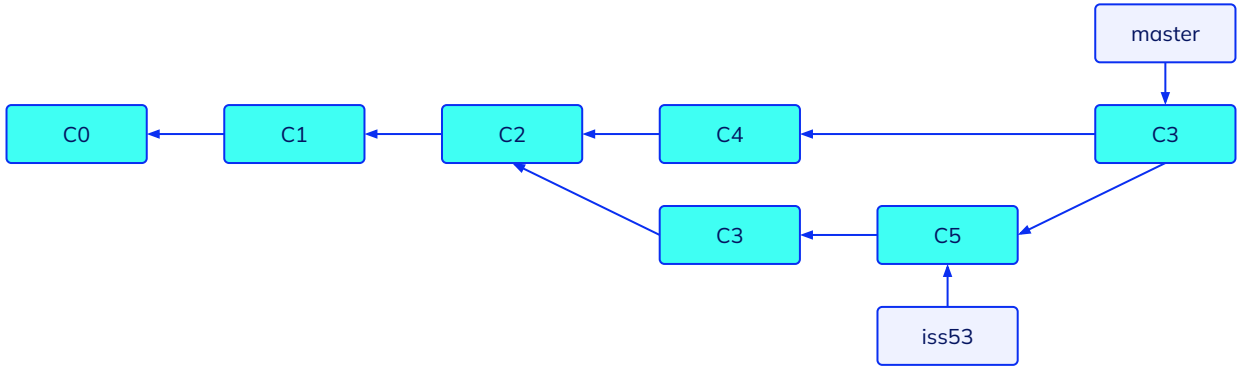
Los cambios realizados ya están en el *snapshot* del `commit` apuntado por el *branch master.*

Supongamos ahora que el *branch* que queremos fusionar con nuestro trabajo actual no es lineal, es decir, no tiene como ancestro a nuestro *branch master.* En este caso, **los branches se verán bifurcados respecto a algún ancestro en común** probablemente:



Debido a que la confirmación en la rama actual no es ancestro directo de la rama que pretendes fusionar, Git tiene cierto trabajo extra que hacer. **Realizará una fusión a tres bandas, utilizando las dos instantáneas apuntadas por el extremo de cada una de las ramas y por el ancestro común a ambas.**

En lugar de simplemente avanzar el apuntador de la rama, Git crea una **nueva instantánea *(snapshot)* resultante de la fusión** a tres bandas; y crea automáticamente una nueva **confirmación de cambios (`commit`) que apunta a ella.** Nos referimos a este proceso como **"fusión confirmada"** y su particularidad es que **tiene más de un padre.**



Resolver conflictos

Resolución de conflictos

En algunas ocasiones, **los procesos de fusión no suelen ser fluidos.**

Si hay modificaciones dispares en una misma porción de un mismo archivo en las dos ramas distintas que pretendes fusionar, Git no será capaz de fusionarlas directamente.

Por ejemplo, si en tu trabajo del *problema #53* has modificado una misma porción que también ha sido modificada en el *problema hotfix*, verás un conflicto como el que se muestra en el bloque debajo.

```
> git merge iss53
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

Git no crea automáticamente una nueva fusión confirmada (**`merge commit`**), sino que **hace una pausa en el proceso, esperando a que tú resuelvas el conflicto.**

Para ver qué archivos permanecen sin fusionar en un determinado momento conflictivo de una fusión, puedes usar el comando **`git status`**:

```
> git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

Unmerged paths:
  (use "git add <file>..." to mark resolution)

    both modified:      index.html

no changes added to commit (use "git add" and/or "git commit -a")
```

Todo aquello que sea conflictivo y no se haya podido resolver, se marca como **"sin fusionar"** (**unmerged**).

Git añade a los archivos conflictivos unos marcadores especiales de resolución de conflictos que te guiarán cuando abras manualmente los archivos implicados y los edites para corregirlos.

El archivo conflictivo contendrá algo como:

```
<<<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=======
<div id="footer">
 please contact us at support@github.com
</div>
>>>>>>> iss53:index.html
```

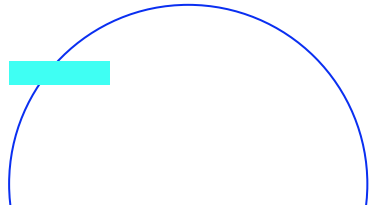
Donde dice que la versión en **HEAD** (la rama ***master*** que habías activado antes de lanzar el comando de fusión) contiene lo indicado en la parte superior del bloque (todo lo que está encima de **=======**) y que la versión en **iss53** contiene el resto, lo indicado en la parte inferior del bloque.

Para resolver el conflicto, debes elegir manualmente el contenido de uno o de otro lado. Por ejemplo, puedes optar por cambiar el bloque, dejándolo así:

```
<div id="footer">
please contact us at email.support@github.com
</div>
```

Esta corrección contiene un poco de ambas partes y se han eliminado completamente las líneas **<<<<<<< , ======= y >>>>>>>**.

Tras resolver todos los bloques conflictivos, debes lanzar comandos **`git add` para marcar cada archivo modificado. Marcar archivos como preparados (staged) indica a Git que sus conflictos han sido resueltos.**



¡Sigamos trabajando!