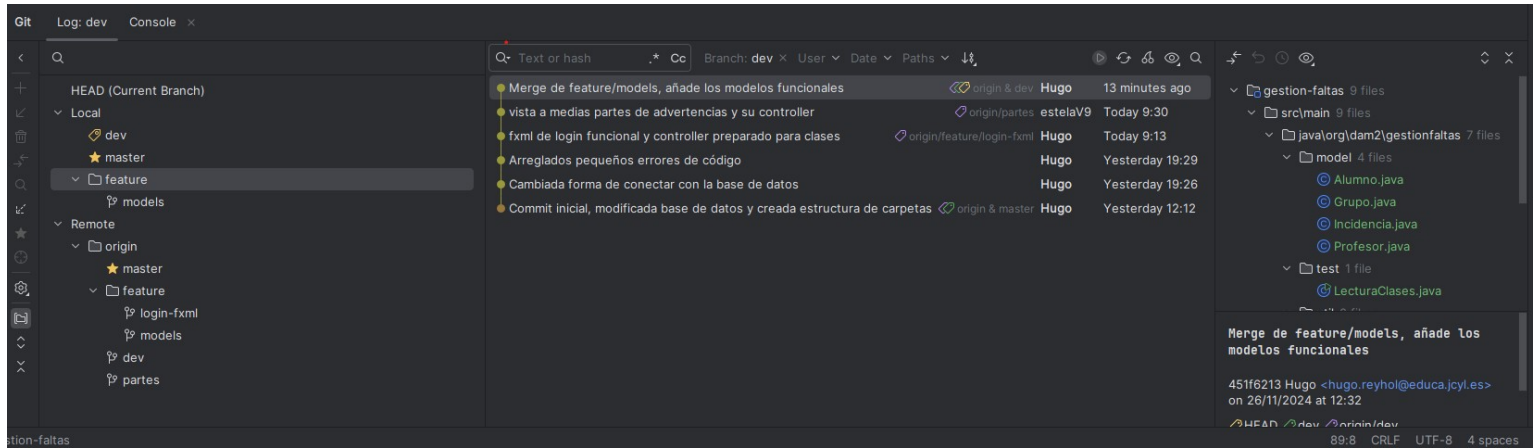


Trabajar en grupo con Git usando la interfaz gráfica de IntelliJ:

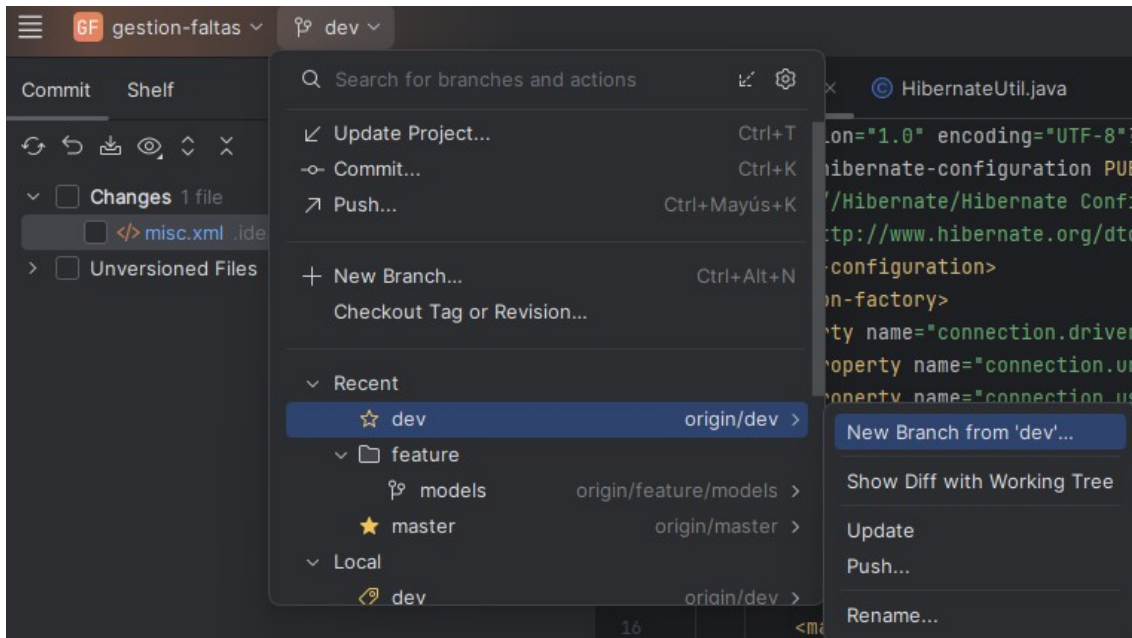
Antes de nada, en IntelliJ hay un apartado abajo a la izquierda que permite ver el estado de Git.



Ramas (Branch):

- master: Código funcional, versión estable que se entregará
- dev: Rama de desarrollo, a partir de ella se crean las sub-ramas para añadir mejoras, arreglar bugs, etc.... y se vuelve a unificar cuando el código de cada sub-rama está listo
- feature/<nombre>: Sub-ramas que añaden funcionalidad
- fix/<nombre>: Sub-ramas que arreglan bugs

Para crear una rama elegís la rama a partir de la cual vais a crearla y pulsáis “Nueva rama de ‘<rama padre>’...”.



Una vez hecho esto la rama debería cambiar automáticamente a la nueva.

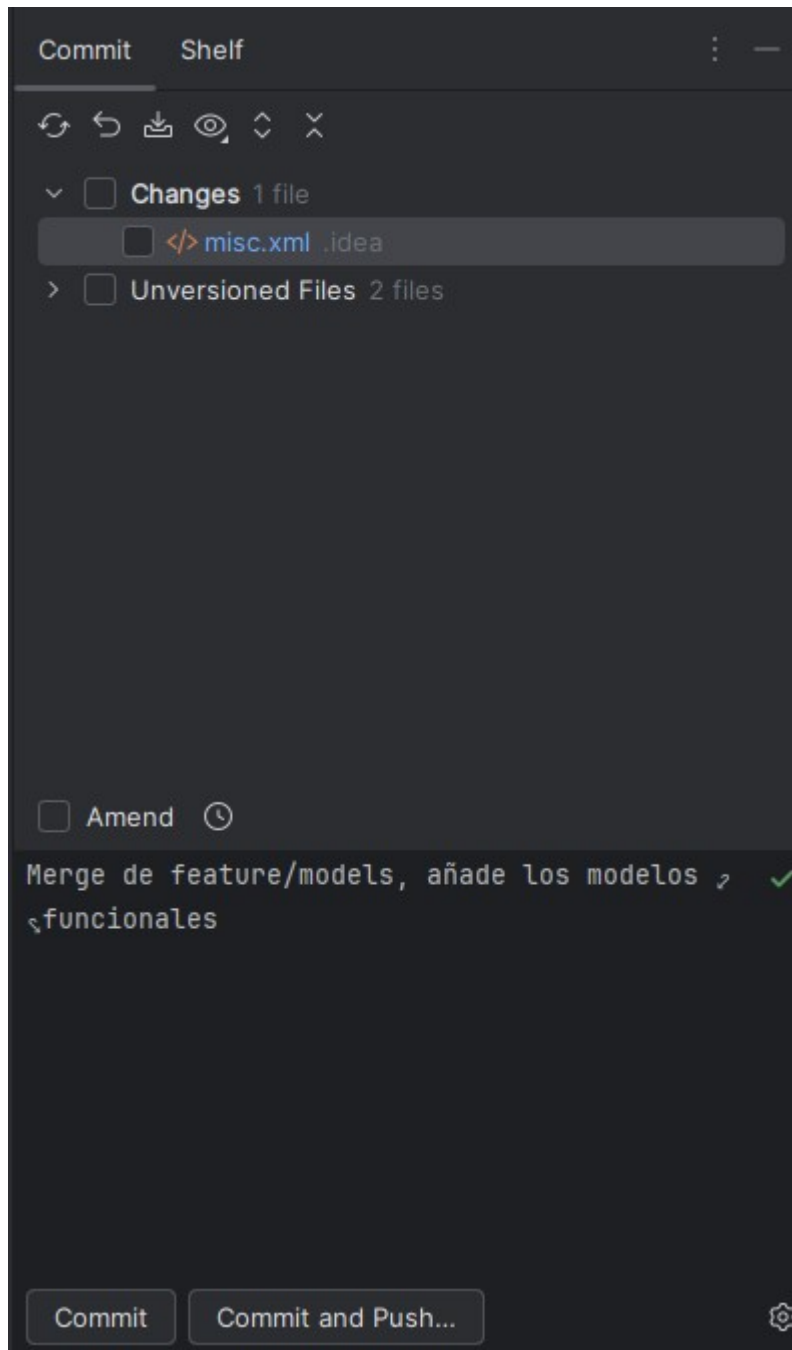
Si no lo ha hecho, para movernos entre ramas podéis usar la opción “Checkout” que aparece al pulsar clic derecho en una rama en cualquiera de los 2 menús mostrados anteriormente. Tened en cuenta que si habéis hecho cambios que no habéis guardado(commit) en archivos de una rama y os vais a cambiar a otra rama que contiene los mismos archivos tendréis problemas para hacerlo ya


que detectará que esos cambios se van a perder. (Se puede hacer un commit o si no estás seguro de querer mantener ese código meterlo en el stash/shelf)

Guardar cambios (Commit):

Para guardar los cambios hechos en el código se usan los commits, para hacer un commit id al

apartado de arriba a la izquierda.

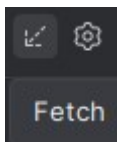


- Changes: todos los archivos que habéis añadido a Git y tienen cambios pendientes
- Unversioned Files: los archivos que nunca habéis añadido a Git
- Espacio oscuro: mensaje del commit, sirve para identificar que se ha hecho fácilmente
-  Rollback: restaura los archivos a como estaban en el anterior commit
- Commit: guarda el commit

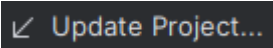
Los archivos de la carpeta .idea como el que se ve en el ejemplo nunca hay que guardarlos en un commit ya que son archivos de configuración de IntelliJ.

Actualizar un proyecto ya existente (Pull y Fetch):

En el mismo menú que se ve en el apartado de las ramas pulsar la flecha con rayas discontinuas.

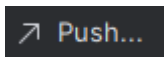


Esta opción también aparece en *Menu de hamburguesa > Git > Fetch*

Su funcionalidad es actualizar la información de nuestro repositorio local con el repositorio remoto, pero no modifica nuestro trabajo local, para ello usaremos .

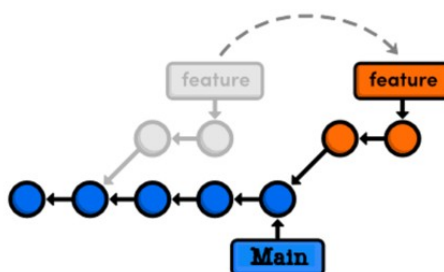
Fetch es útil a la hora de subir nuestros cambios a GitHub ya que podemos comprobar si alguien ha añadido algún commit a la rama que vamos a actualizar, si es así habría que comprobar si nuestro código rompe el ya existente y solucionarlo en caso de hacerlo.

Subir los commits a GitHub u otro servidor (Push):

En el mismo menú que se ve en el apartado de las ramas pulsar "Push..." .

Juntar una sub-rama con su rama padre (Merge):

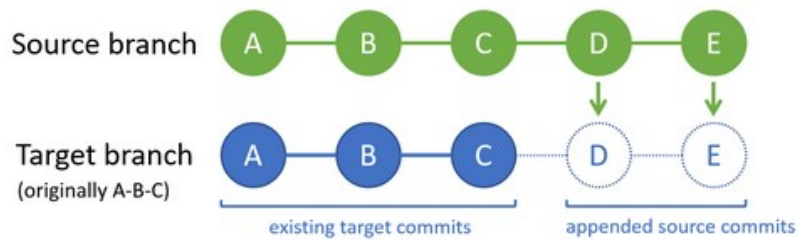
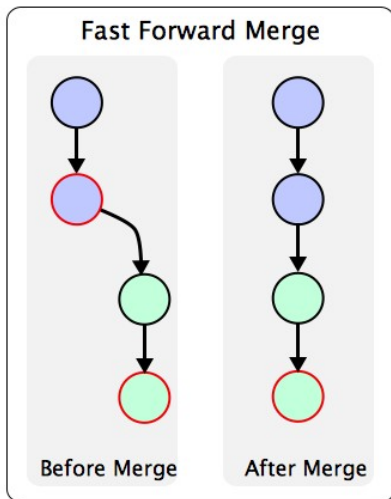
Primero habría que hacer un rebase de nuestra rama sobre la rama que vamos a modificar, esto desplaza los commits de nuestra rama actual encima de los de la rama a juntar, esto se puede hacer en cualquiera de los 2 menús que aparecen al principio del Word.



Para ello estando en nuestra rama haremos clic derecho en la rama padre, generalmente dev, y pulsaremos "Rebase '<nuestra rama>' onto '<rama padre>'".

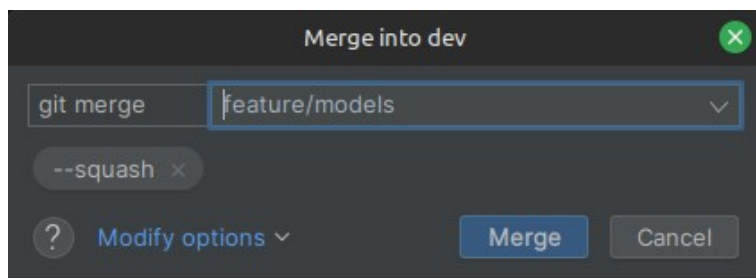
Después de hacer este proceso habría que comprobar que el código sigue funcionando correctamente, en caso de hacerlo haremos un merge.

Merge por defecto copia todos los commits de una rama a la otra, esto no nos interesa ya que en proyectos grandes nos ensucia dev, para poder hacer merges limpios tenemos 2 opciones:



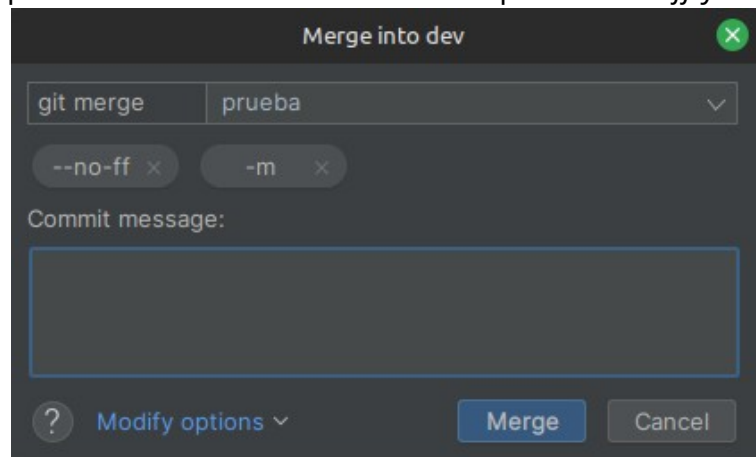
- Usamos un tipo especial llamado squash: Esto nos copiará todos los cambios a la rama que recibe el merge para que podamos hacer un único commit.

Para hacer este merge nos posicionamos a la rama que queremos copiar los cambios, en este caso dev, vamos a *Menú de hamburguesa > Git > Merge...*, se nos abrirá la siguiente ventana, en ella



seleccionaremos la rama que tiene los cambios que queremos implementar y en *Modify options* seleccionaremos `--squash`, después de esto solo nos queda hacer un commit de dev explicando que rama hemos juntado y que cambios implementa esa rama.

- Usamos un tipo llamado no-ff: Esta opción nos hace un merge desactivando el fast forward, la opción que pega todos los commits, para hacerlo abrimos la misma ventana que en el caso anterior pero esta vez seleccionaremos las opciones `--no-ff` y `-m`, introducimos el comentario del commit y nos



creará algo parecido al caso anterior solo que ahora podemos ver la rama en el editor.



Eliminar un commit:

Un commit generalmente se borra con clic derecho sobre un commit en el menú de Git y pulsando *Drop Commit*, pero en el caso de que hayamos usado el merge no-ff no podremos eliminarlo usando

Drop, para borrar estos hay que dar clic derecho al commit del que divergen las ramas y pulsar *Reset current branch to here...*, esto devolverá la rama al estado antes de hacer el merge pero tendremos que borrar manualmente los archivos que se añadieron o modificaron, esto se hace con el rollback.