Git & GitHub

Pull request

Los pull requests son la forma de contribuir a un proyecto grupal o de código abierto.

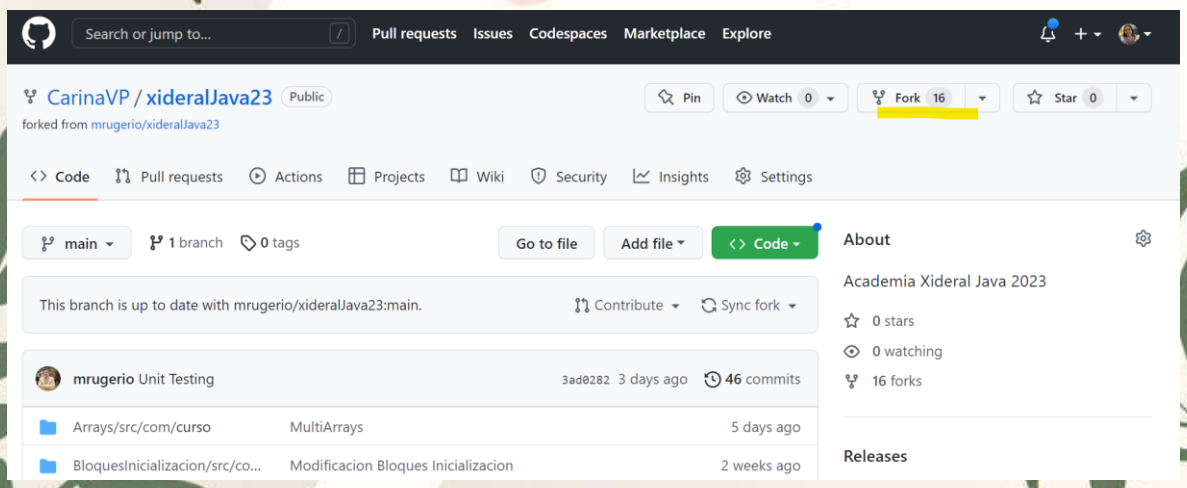
Por ejemplo, un usuario llamado Carina realiza un fork de un repositorio de Miguel y le efectúa algunos cambios. Ahora Carina puede hacer un pull request a Miguel, pero dependerá de Miguel aceptar o declinarlo. Es como decir: "Miguel, ¿podrías por favor extraer (pull) mis cambios?"

¿Cómo hacer un pull request?

1. Realizar un fork desde el repositorio principal.
2. Clonar el repositorio: clonarlo a la computadora para trabajarlo localmente.
3. Crear una rama nueva.
4. Realizar cambios y confirmarlos: Se hacen cambios esenciales al proyecto y se guardan.
5. Guardarlos a la rama creada.
6. Enviar los cambios a GitHub.
7. Crear un pull request: si el pull request es aceptado se confirma por correo.

fork

Es una de las operativas comunes con el trabajo en Git y GitHub, sirve para **crear una copia de un repositorio en tu cuenta de usuario**. Ese repositorio copiado será básicamente un clon del repositorio desde el que se hace el fork, pero a partir de entonces **el fork vivirá en un espacio diferente y podrá evolucionar de manera distinta**, a tu propio cargo, es decir, es una copia de un repositorio, pero **creado en tu propia cuenta de GitHub, donde sí que tienes permisos de escritura**.



Git rebase

Es **un comando que reproduce *commits* o confirmaciones de cambio** una por una, en la parte superior de una determinada rama. Otra de las características de este comando es que **no guarda la historia de la rama secundaria,** sino que se encarga de reescribir la historia de la rama principal, integrando las confirmaciones de las dos ramas y cambiando el HEAD al último *commit* que ubica.

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.

git rebase --interactive OTHER-BRANCH-NAME

Para cambiar de base todas las confirmaciones entre otra rama y el estado de rama actual.

git rebase --interactive HEAD~7

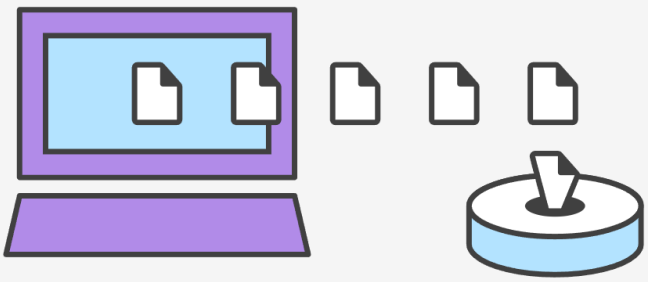
Para cambiar de base las últimas confirmaciones en tu rama actual.

Hay seis comandos disponibles mientras se cambia la base:

- ***pick***: indica que el *commit* o confirmación de cambio se ha incluido.
- ***reword***: después de su uso, la reorganización se detiene y permite modificar el mensaje de *commit*.
- ***edit***: permite modificar el *commit*, es decir, agregar o eliminar información de este.
- ***squash***: permite combinar uno o más *commits* en uno solo.
- ***fixup***: similar al comando *squash*, a diferencia de que el *commit* que se va a fusionar tiene descartado el mensaje.
- ***exec:*** ejecuta los comandos de *shell* contra un *commit*.

Git stash

El comando git stash coge los cambios sin confirmar (tanto los que están preparados como los que no), los guarda aparte para usarlos más adelante y, acto seguido, los deshace en el código en el que estás trabajando.



git stash

Mis cambios serán almacenados en una pila

git stash list

Para ver lo que hay en el stash solo ~~tengo que~~se escribir git stash list y nos mostrará la lista de las cosas que guarde en el stash

git stash show -p stash@{0}

Para ver qué cambios hay en un stash se puede colocar git stash show -p y le pone el nombre, en este caso stash@{0}, {i}, i=n índice que quiere ver.

git stash save "Agregado texto al index.html"

Se puede guardar cambios en el stash con un mensaje como si fuera un commit, de esta forma lo hará más descriptivo y le puede ayudar a tener un mejor control del stash.

git stash pop

Para recuperar sus cambios del stash, el stash es una pila, lo que significa que lo último que entra siempre será lo primero en salir. Así que cuando haga pop siempre va a sacar el último cambio a menos que le diga lo contrario.

git stash pop stash@{1}

git stash pop 1

Para aplicar los cambios de un stash especifico (con su índice).

git stash drop NOMBRE-DEL-STASH

git stash drop 1

Para remover los cambios guardados en stash sin aplicarlos.

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto

git stash clear

Comando para limpiar todo del stash.

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto

Con formato: Fuente: Arial Rounded MT Bold

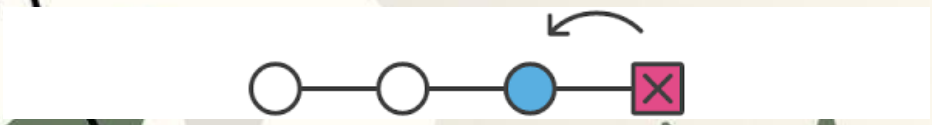
Git clearstash

Limpia el árbol de trabajo mediante la eliminación recursiva de archivos que no están bajo cont de versiones, a partir del directorio actual.

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto

Normalmente, solo se eliminan los archivos desconocidos para Git, pero si -x se especifica la op también se eliminan los archivos ignorados. Esto puede, por ejemplo, ser útil para eliminar todos los productos de compilación.

Con formato: Justificado



git clean --dry-run

Revisar que archivos no tienen seguimiento

Con formato: Fuente: Arial Rounded MT Bold

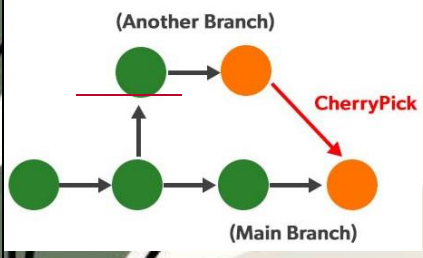
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto

Con formato: Justificado

git clean -d	Cuando no se especifica <pathspec>, git clean no recurrirá a directorios sin seguimiento para evitar eliminar demasiado. Especifique -d para que también se repita en dichos directorios.
git clean -F	Si la variable de configuración de Git clean.requireForce no se establece en false, git clean se negará a eliminar archivos o directorios a menos que se indique -f o -i.
git clean -i	Muestra lo que se haría y limpie los archivos de forma interactiva. Consulte "Modo interactivo" para obtener más detalles.
git clean -n	En realidad, no elimine nada, solo muestre lo que se haría.
git clean -q	Solo informa los errores, pero no los archivos que se eliminan con éxito.
git clean -X	Elimina solo los archivos ignorados por Git. Esto puede ser útil pa reconstruir todo desde cero, pero conserva los archivos creados manualmente.

Git cherry

El Cherry pick es una característica de git nos va a permitir traernos commits de un branch a otro sin tener que hacer un merge o un rebase.



CASOS:

1. Por alguna razón tengo otra rama donde hice un p commits que nunca subí al repositorio remoto y r uno de esos commits.
2. Quiero probar un conjunto de cambios particular traerme un commit desde la rama de un compañe
3. Estaba trabajando y luego de un par de commits l un punto donde quedé bloqueado, guardé cambio moví a trabajar en otra funcionalidad. (Creo que el punto 1 ahora tiene más sentido)
4. Un bug encontrado en develop necesita moverse a una rama hotfix que eventualmente se hará merge a master, y así podamos reparar algo en producción.

Con formato: Justificado
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático
Con formato: Justificado
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Negrita, Color de fuente: Fondo 1
Con formato ...
Con formato ...
Con formato ...
Con formato: Fuente: 14 pto
Con formato ...

git cherry-pick --abort

Cuando nos encontramos con problemas durante el proceso siempre podemos utilizar el siguiente comando para abortar la operación.

git cherry-pick <commit-SHA>

Donde <commit_SHA> es el identificar corto del commit q queremos seleccionar y que está conformado por 7 dígi (mejor conocido como short SHA-1 o short SHA.

git log --oneline

Obtener el short SHA-1, en tu terminal podrá ver un listado commits con su respectivo short SHA y su commit messag

```
HP@DESKTOP-3UQQHPL MINGW64 ~/Documents (master)
$ git log --oneline
4bce868 (HEAD -> master, origin/master) Evaluación semana 1
```

¿Conflictos?

git add .

git cherry-pick --continue

Los solucionamos de la misma forma en que lo hace durante un merge regular y luego ejecutamos.

git cherry-pick --abort

Si por alguna razón cambiamos de parecer y queremos abortar la misión.

El uso del cherry-pick puede generar **duplicidad** de commits en cuanto a contenido, debido a que este proceso siempre creará un short SHA diferente, recuerda que dos short SHA diferentes harán que Git piense que son historias totalmente diferentes.

Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto, Color de fuente: Automático
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Justificado
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: Arial Rounded MT Bold, 16 pto, Color de fuente: Énfasis 6
Con formato: Fuente: (Predeterminada) Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Justificado
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: Arial Rounded MT Bold
Con formato: Fuente: (Predeterminada) Times New Roman, 12 pto
Con formato: Fuente: (Predeterminada) Times New Roman, 14 pto, Color de fuente: Automático
Con formato: Justificado
Con formato: Fuente: (Predeterminada) Times New Roman, 14 pto