Algunos conceptos git

¿Qué es git?

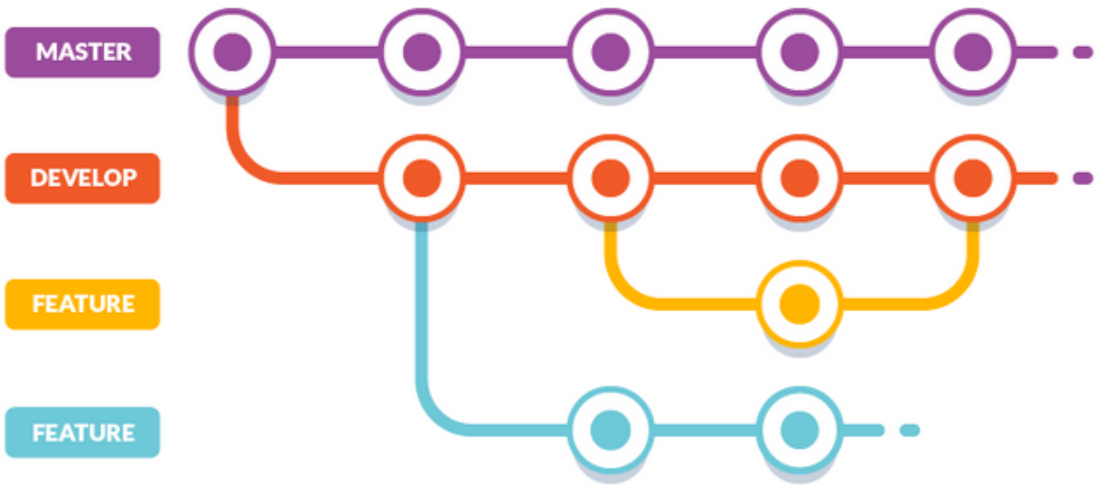
Git es un sistema de control de versiones distribuido, lo que significa que un clon local del proyecto es un repositorio de control de versiones completo.

¿Qué es GitHub?

Es una plataforma basada en la web donde los usuarios pueden alojar repositorios Git. Facilita compartir y colaborar fácilmente en proyectos con cualquier persona en cualquier momento.

¿Qué es Git Flow?

Gitflow es un modelo alternativo de creación de ramas en Git en el que se utilizan ramas de función y varias ramas principales.



Algunos conceptos git

Branches - Ramas

Una rama representa una línea independiente de desarrollo.

Las ramas sirven como una abstracción de los procesos de cambio, preparación y confirmación.

Se puededn ver como una forma de solicitar un nuevo directorio de trabajo

Comandos

```
git branch
```

Enumera todas las ramas de tu repositorio.

```
git branch <rama>
```

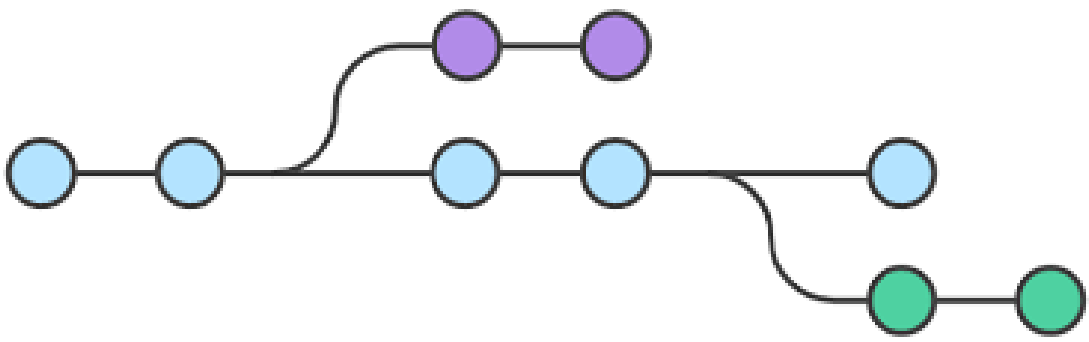
Crea una nueva rama llamada <rama>

```
git branch -d <rama>
```

Elimina la rama especificada.

```
git branch -m <rama>
```

Cambia el nombre de la rama actual a <rama>.

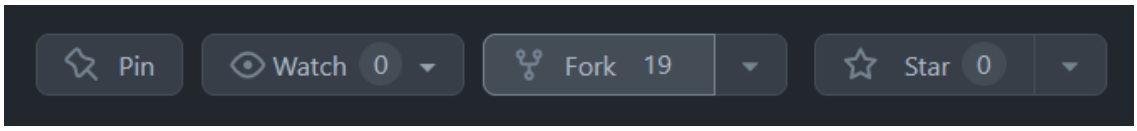


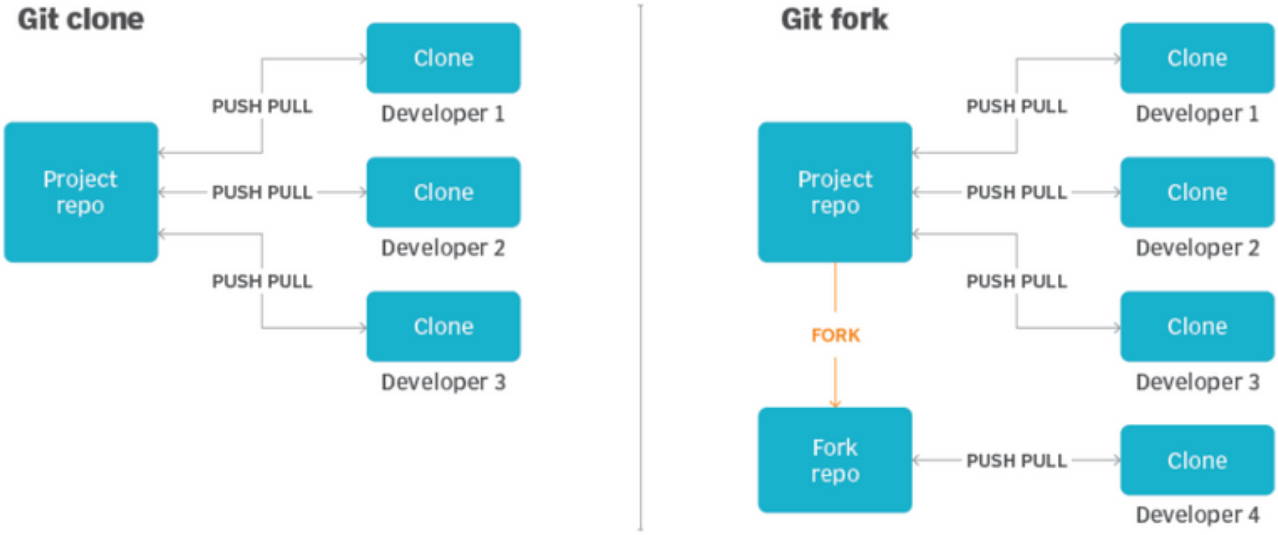
Algunos conceptos git

Fork

Una bifurcación (*Fork*) es un nuevo repositorio que comparte la configuración de visibilidad y código con el repositorio “ascendente” original.

Las bifurcaciones se suelen usar para iterar ideas o cambios antes de que se vuelvan a proponer al repositorio ascendente, como en proyectos de código abierto o cuando un usuario no tiene acceso de escritura al repositorio ascendente.

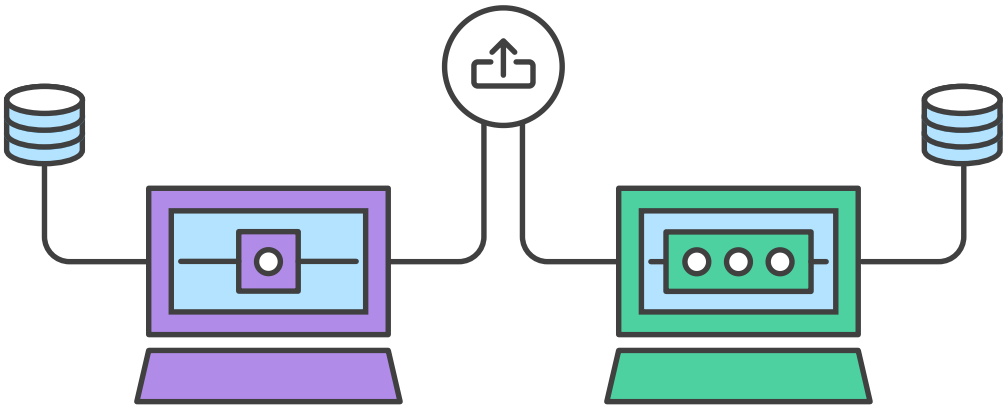




Algunos conceptos git

Pull Request

Un pull request es una petición que el propietario de un fork de un repositorio hace al propietario del repositorio original para que este último incorpore los commits que están en el fork



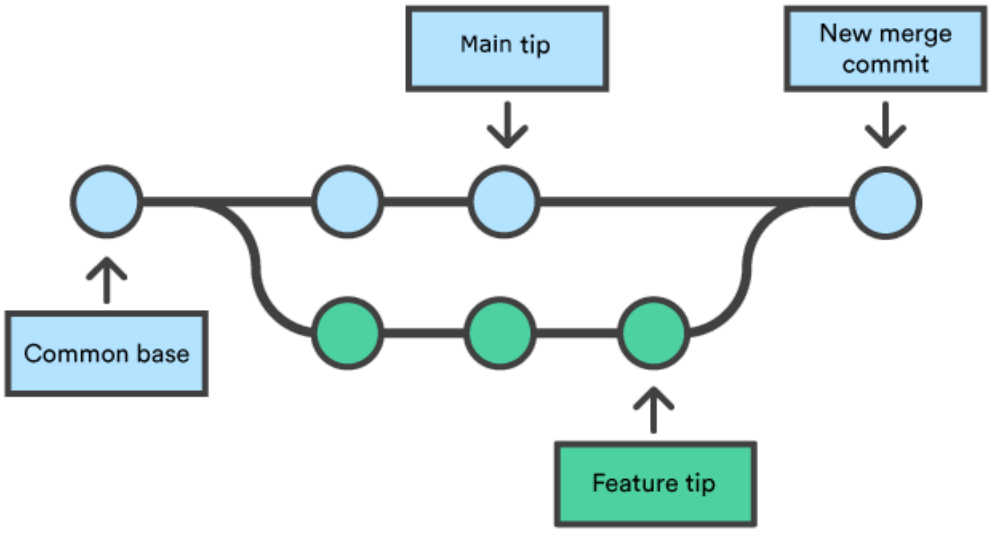
En su forma más sencilla, las solicitudes de incorporación de cambios son un mecanismo para que los desarrolladores notifiquen a los miembros de su equipo que han terminado una función.

Una vez la rama de función está lista, el desarrollador realiza la solicitud de incorporación de cambios. Así, todas las personas involucradas saben que deben revisar el código y fusionarlo con la rama main.

Algunos conceptos git

Merge

La fusión ('*Merge*') es la forma que tiene Git de volver a unir un historial bifurcado. El comando *git merge* permite tomar las líneas independientes de desarrollo creadas por *git branch* e integrarlas en una sola rama.



Funcionamiento:

-*git merge* combinará varias secuencias de confirmaciones en un historial unificado. En los casos de uso más frecuentes, git merge se utiliza para combinar dos ramas.

Ejemplo:

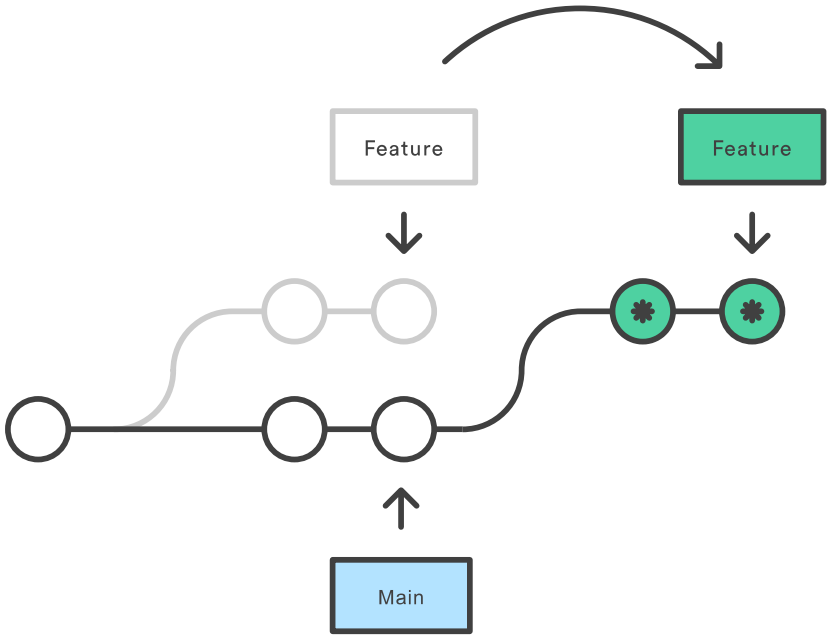
```
git merge <rama1>
```

Esto hace un merge de la rama <rama1> dentro de la rama actual

Algunos conceptos git

Rebase

Este comando integra las modificaciones de una rama a otra, a través de reorganizar o cambiar la base de una rama de commit a otra.



Uso:

El motivo principal por el que llevar a cabo una fusión mediante cambio de base es para mantener un historial del proyecto lineal.

Ejemplo:

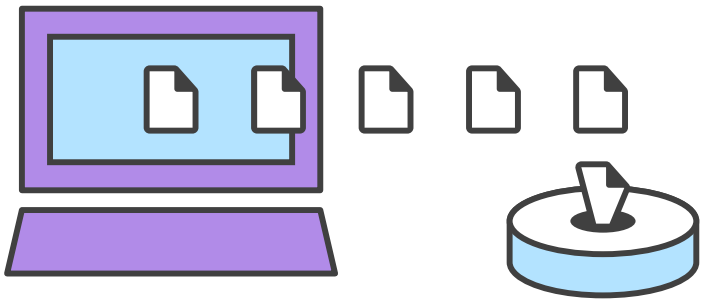
```
git rebase <base>
```

Esto realiza un rebase de la rama actual dentro de la rama <base>

Algunos conceptos git

git Stash

El comando git stash almacena temporalmente (o guarda en un stash) los cambios efectuados en el código en el que se este trabajando para poder trabajar en otra cosa y, más tarde, regresar y volver a aplicar los cambios más tarde.



Uso:

Guardar los cambios en stashes resulta práctico si tienes que cambiar rápidamente de contexto y ponerte con otra cosa

Funcionalidad:

El comando *git stash* selecciona los cambios sin confirmar, los guarda aparte para usarlos más adelante y, acto seguido, los deshace en el código en el que se este trabajando

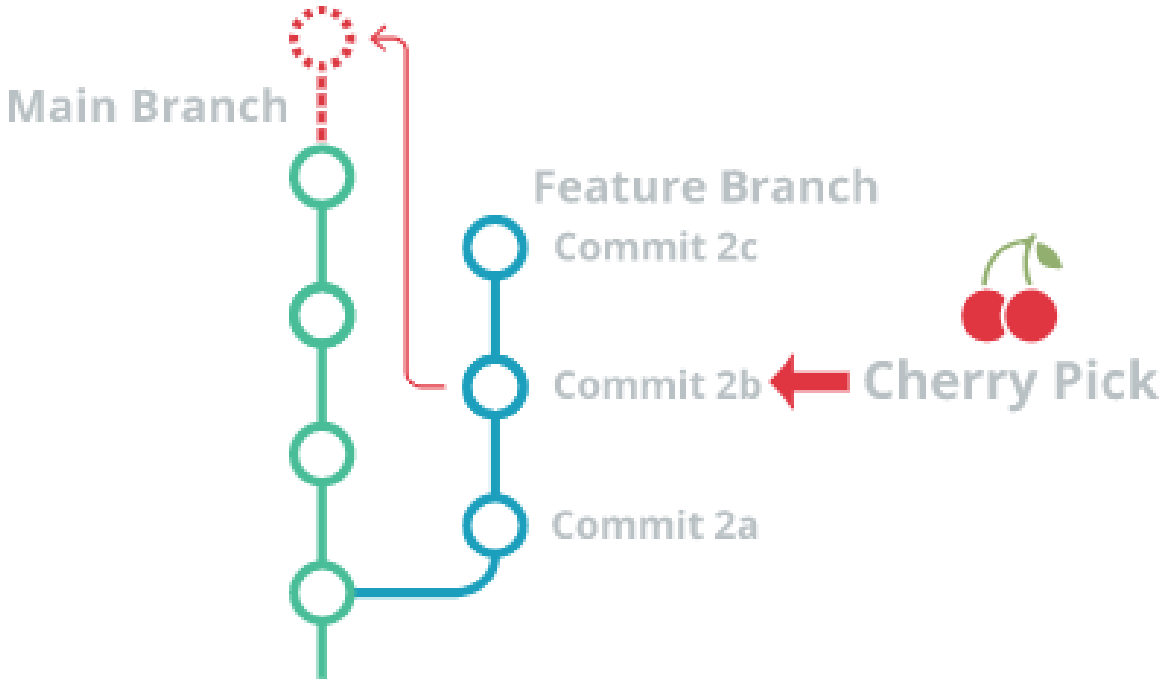
Comando:

```
git stage
```

Algunos conceptos git

Cherry Pick

Git chery-pick es un comando que se encarga de elegir uno o más commits de una rama específica para luego aplicarla a otra rama.



Uso:

Para que el comando cherry-pick funcione correctamente, la rama actual no debe tener cambios (debe estar en “clean status”).

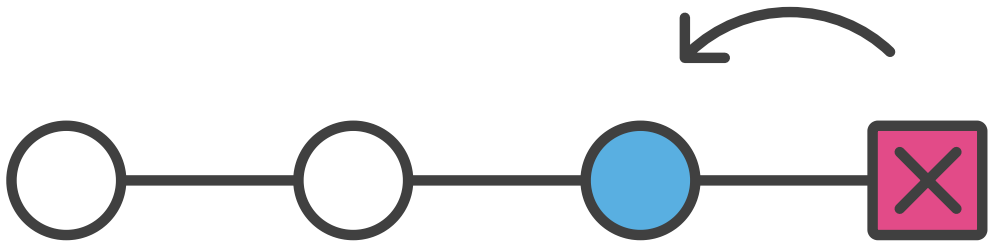
Comando:

```
git cherry-pick Commit2b
```

Algunos conceptos git

Clean

El comando git clean se utiliza para eliminar archivos no deseados de tu directorio de trabajo. Esto podría incluir la eliminación de artefactos de construcción temporal o la fusión de archivos en conflicto.



Actúa en archivos sin seguimiento, este tipo de archivos son aquellos que se encuentran en el directorio de trabajo, pero que aún no se han añadido al índice de seguimiento de repositorio con el comando *add*.

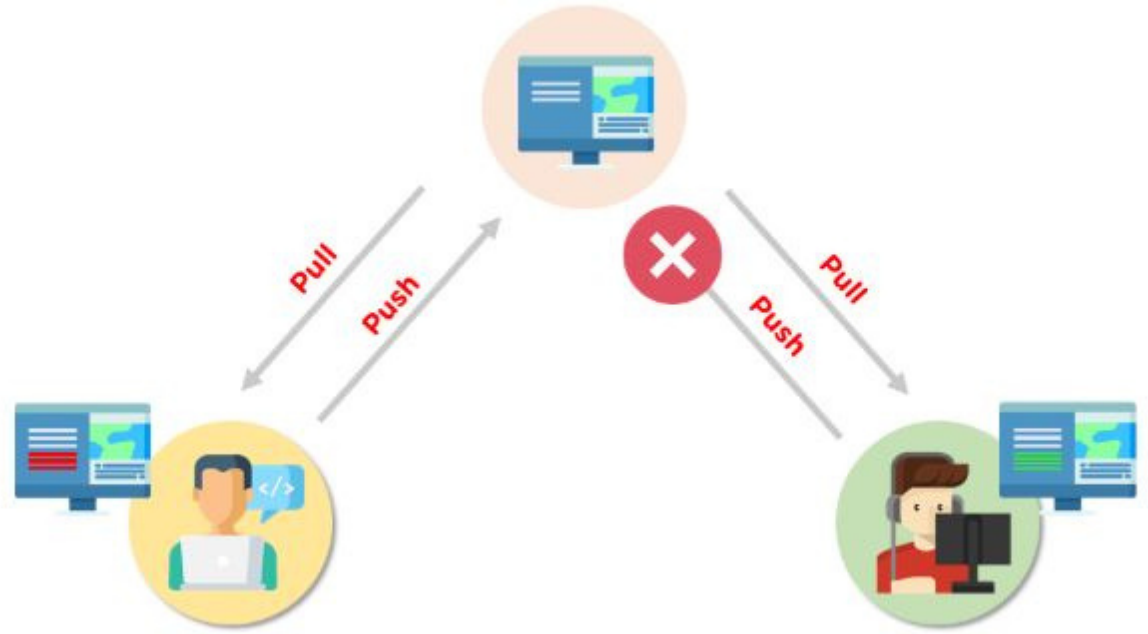
Comando:

```
git clean
```

Algunos conceptos git

Conflicts

A veces, se puede dar el caso de que varios desarrolladores intenten editar el mismo contenido. Si el desarrollador A intenta editar código que el desarrollador B está editando, podría producirse un conflicto.



Comúnmente, los conflictos surgen cuando dos personas han cambiado las mismas líneas de un archivo o si un desarrollador ha eliminado un archivo mientras otro lo estaba modificando.

Tipos de conflictos de fusión:

Una fusión puede entrar en un estado conflictivo en dos momentos diferentes. Al inicio y durante un proceso de fusión.

Algunos conceptos git

Conflicts

Al inicio:

Una fusión no se iniciará si Git detecta que hay cambios en el directorio de trabajo o el entorno de ensayo del proyecto actual.

Git no inicia la fusión porque las confirmaciones que se están fusionando podrían sobrescribir estos cambios pendientes.

Cuando esto sucede, no se debe a conflictos con otros desarrolladores, sino con cambios locales pendientes. El estado local tendrá que estabilizarse mediante *git stash*, *git checkout*, *git commit o git reset*. Un fallo de fusión durante el inicio provocará que aparezca el siguiente mensaje de error:

Algunos conceptos git

Conflicts

Durante:

Un fallo DURANTE una fusión indica un conflicto entre la rama local actual y la rama que se está fusionando.

Esto indica un conflicto con el código de otro desarrollador.

Git hará lo posible para fusionar los archivos, pero dejará cosas para que las resuelva el usuario manualmente en los archivos con conflictos.

Un fallo a la mitad de la fusión provocará que aparezca el siguiente mensaje de error:

error: Entry '<archivo.txt>' would be overwritten by merg e. Cannot merge. (Changes in staging area)

Algunos conceptos git

Conflicts

¿Cómo Resolverlos?

La forma más directa de resolver un conflicto de fusión es **editar el archivo conflictivo.**

Abriendo el archivo corrupto en el editor preferido, simplemente se va a identificar las lineas de código que están causando conflicto para poder editarlas y ajustarlas a lo que se desea.

Cuando se haya editado el archivo, se utiliza *git add* <archivo.txt> para preparar el nuevo contenido fusionado. Para finalizar la fusión, se crea una nueva confirmación ejecutando lo siguiente:

git commit -m "merged and resolved the conflict in merge.txt"

Algunos conceptos git

Conflicts

Algunos comandos utiles para la identificcaión y solucion de conflictos son:

- git status
- git log --merge
- git diff
- git checkout
- git reset --mixed
- git merge --abort
- git reset

Fuente: https://www.atlassian.com/es/git