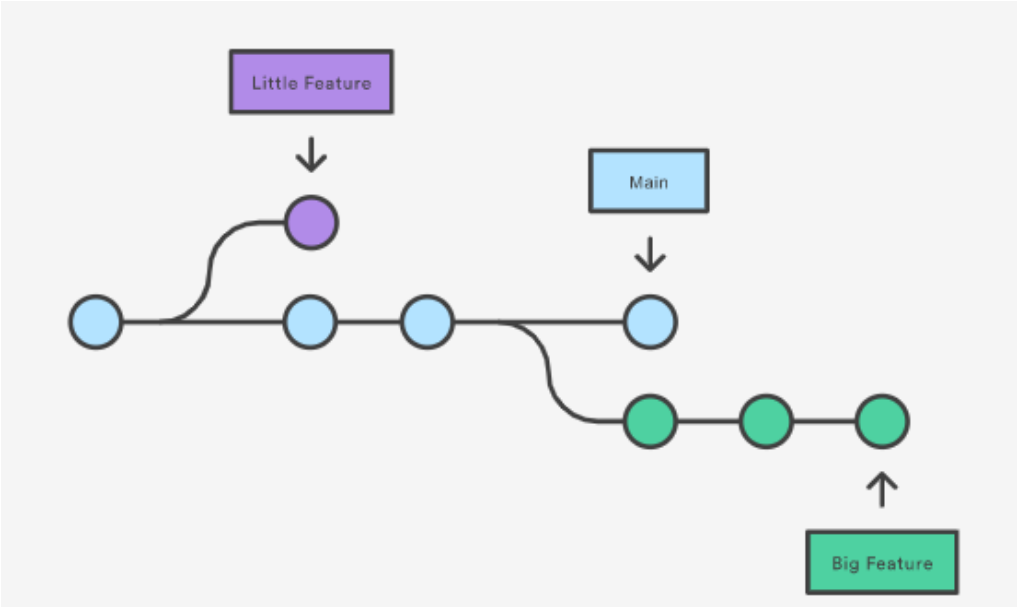
GitHub (branches, merge & conflicts)

Branches

Los branches, ramas traducido al español, son espacios en donde el programador puede trabajar sobre un mismo proyecto sin modificar los archivos originales del proyecto, para así evitar que el código original sufra modificaciones que puedan ser erróneas. Las ramas te permiten desarrollar características, corregir errores, o experimentar con seguridad las ideas nuevas en un área contenida del repositorio. Además de todas las funciones anteriores, también sirve para dar flexibilidad al proceso de desarrollo del proyecto y llevar este de una manera más organizada.

La rama principal que es en donde se encuentra el proyecto original por lo regular recibe el nombre de main y a partir de esta se crean otras ramas que sean necesarias, como por ejemplo la rama de development que es una copia de la rama main en donde se pueden hacer cambios para experimentar y otra rama seria hotfix que es una rama en donde se hacen correcciones al código de la rama principal.

Ejemplo grafico de branches en git.



Algunos de los comandos de branches en git son:

(crear una nueva rama)

git branch <nombre-de-la-rama>

(visualización de las ramas existentes)

git branch

(eliminar una rama)

git branch -d <nombre-de-la-rama>

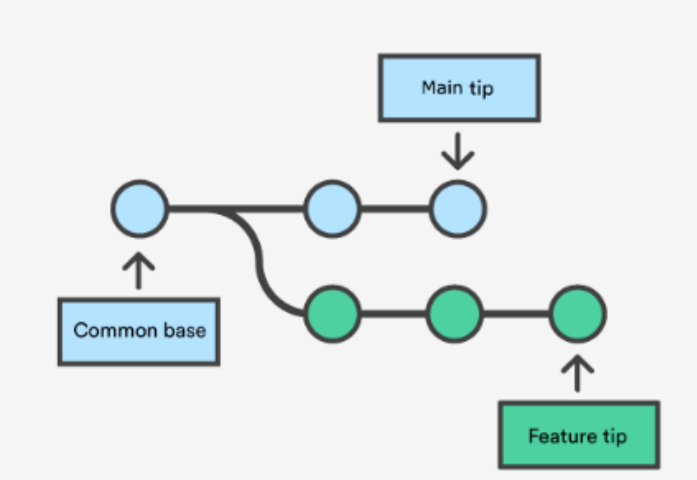
(cambiar a otra rama)

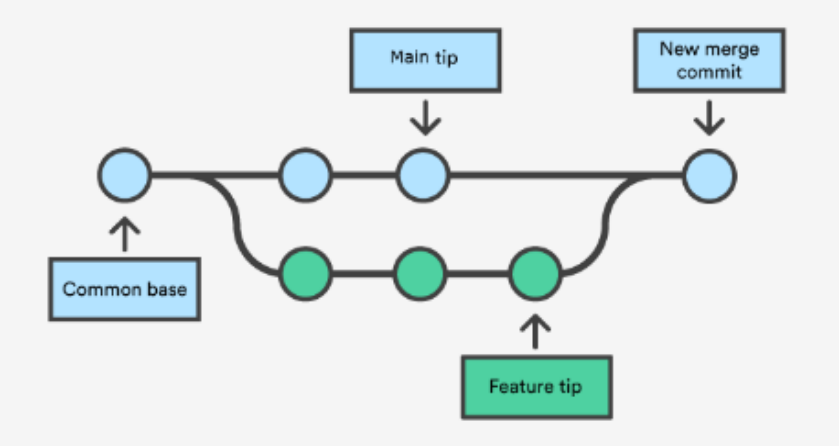
git checkout <nombre-de-la-rama>

Merge

El concepto merge se refiere a la unión de dos branches diferentes para unificar un solo branch con sus respectivos cambios ya realizados en cada una de las ramas cuando el trabajo en ellas ya ha sido completado. El merge fusionará cualquier cambio que se haya hecho en la base de código en una rama separada de la rama actual como un nuevo commit. El proceso de merge puede ser manual o automático, cuando el proceso es manual entonces existe un supervisor que se encarga de asegurarse que los cambios hechos sean correctos y que no causen problemas en la rama a la cual se le harán cambios.

Ejemplo grafico de branches en git.





Algunos de los comandos de merge en git son:

(comando principal para hacer un merge)

git merge <nombre-de-la-rama-remota>

(comando que descarga los cambios de la rama remota y después realiza el merge de los cambios realizados a la rama local)

git pull <nombre-de-la-rama-local> <nombre-de-la-rama-remota>

(comando que aplica los cambios de una rama sobre la otra)

git rebase <nombre-de-la-rama-remota> <nombre-de-la-rama-local>

Conflicts

Los conflicts en git son problemas que se presentan cuando se realiza un merge, estos conflictos suceden cuando fusionas ramas que tienen confirmaciones de cambios contrapuestas, y Git necesita ayuda para decidir qué cambios incorporar en la fusión final. Por lo general, estos conflictos de fusión suceden cuando las personas realizan diferentes cambios en la misma línea en el mismo archivo o cuando una persona edita un archivo y otra persona elimina el mismo archivo. Cuando ocurre un conflicto en la fusión, git notifica la existencia de dicha inconsistencia ya que no puede ser arreglado automáticamente por el propio git y se detiene el proceso de merge, por ese motivo es necesario que una persona lo resuelva manualmente.

Cuando se encuentra un conflicto en el proceso de merge se editará el contenido de los archivos afectados con indicadores visuales que marquen el final y el inicio del código que esté causando problemas. Estos marcadores visuales son: <<<<<<<, ======= y >>>>>>>. Por lo regular, el contenido que se encuentra delante del marcador ======= es la rama de recepción, es decir, la rama remota y el de la parte de detrás es la rama de fusión o principal.

Algunos de los errores de fusión más comunes son:

- Git no inicia la fusión:

  No se iniciará la fusión si git detecta que hay cambios que no se han guardado en la rama actual.

```
error: Entry '<fileName>' not uptodate. Cannot merge. (Changes in working directory)
```

- Git falla durante la fusión

  No se iniciará la fusión porque existe una inconsistencia entre la rama actual y remota, por lo que puede ser que sea un error de otro programador y esa persona encargada lo debe de resolver

```
error: Entry '<fileName>' would be overwritten by merge. Cannot merge. (Changes in staging a
rea)
```

Algunos comandos de git que nos ayudarán a resolver un conflicto son:

(comando que nos proporcionará información acerca de cambios realizados en la rama)

git status

(comando que creará un registro con una lista de confirmaciones que entran en conflicto entre las ramas que se van a fusionar.)

git log - - merge

(comando que ayuda a encontrar diferencias entre los estados de un repositorio/ archivos y es útil para evitar los conflictos.)

git diff

(comando que sirve para deshacer cambios)

git checkout

(comando que sirve para salirse del proceso de fusion)

git merge --abort