git - la guía sencilla

rogerdudler.github.io/git-guide/index.es.html

una guía sencilla para comenzar con git. sin complicaciones ;)

crea un repositorio nuevo

Crea un directorio nuevo, ábrelo y ejecuta
`git init`

para crear un nuevo repositorio de git.

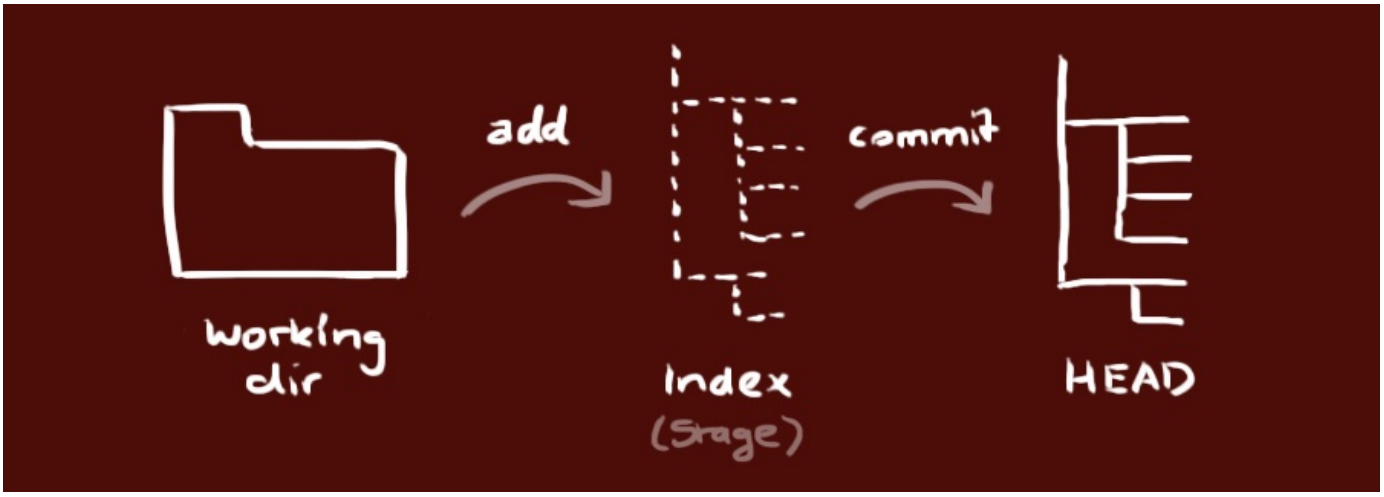
hacer checkout a un repositorio

Crea una copia local del repositorio ejecutando
`git clone /path/to/repository`

Si utilizas un servidor remoto, ejecuta
`git clone username@host:/path/to/repository`

flujo de trabajo

Tu repositorio local esta compuesto por tres "árboles" administrados por git.
El primero es tu `Directorio de trabajo` que contiene los archivos,
el segundo es el `Index` que actua como una zona intermedia, y el último es el
`HEAD` que apunta al último commit realizado.



add & commit

Puedes registrar cambios (añadirlos al **Index**) usando
`git add <filename>`
`git add .`

Este es el primer paso en el flujo de trabajo básico. Para hacer commit a estos cambios usa
`git commit -m "Commit message"`

Ahora el archivo esta incluído en el **HEAD**, pero aún no en tu repositorio remoto.

envío de cambios

Tus cambios están ahora en el **HEAD** de tu copia local. Para enviar estos cambios a tu repositorio remoto ejecuta
`git push origin master`

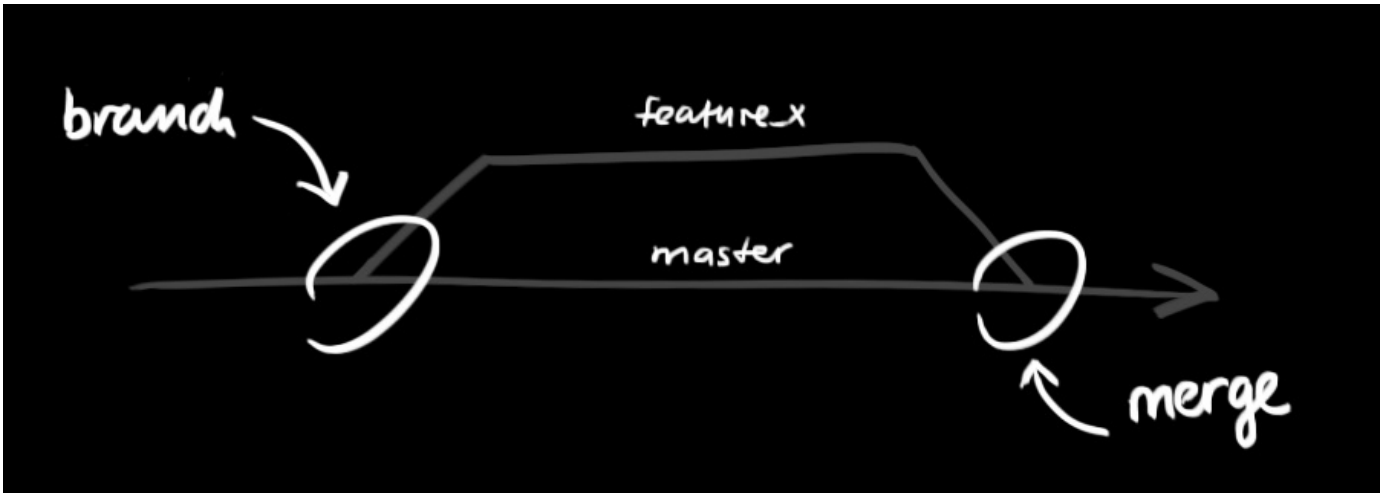
Reemplaza *master* por la rama a la que quieres enviar tus cambios.

Si no has clonado un repositorio ya existente y quieres conectar tu repositorio local a un repositorio remoto, usa
`git remote add origin <server>`

Ahora podrás subir tus cambios al repositorio remoto seleccionado.

ramas

Las ramas son utilizadas para desarrollar funcionalidades aisladas unas de otras. La rama *master* es la rama "por defecto" cuando creas un repositorio. Crea nuevas ramas durante el desarrollo y fusiónalas a la rama principal cuando termines.



Crea una nueva rama llamada "feature_x" y cámbiate a ella usando
`git checkout -b feature_x`

vuelve a la rama principal
`git checkout master`

y borra la rama
`git branch -d feature_x`

Una rama nueva *no estará disponible para los demás* a menos que subas (push) la rama a tu repositorio remoto
`git push origin <branch>`

actualiza & fusiona

Para actualizar tu repositorio local al commit más nuevo, ejecuta
`git pull`

en tu directorio de trabajo para *bajar* y *fusionar* los cambios remotos.

Para fusionar otra rama a tu rama activa (por ejemplo master), utiliza
`git merge <branch>`

en ambos casos git intentará fusionar automáticamente los cambios. Desafortunadamente, no siempre será posible y se podrán producir *conflictos*.
Tú eres responsable de fusionar esos *conflictos*
manualmente al editar los archivos mostrados por git. Después de modificarlos, necesitas marcarlos como fusionados con
`git add <filename>`

Antes de fusionar los cambios, puedes revisarlos usando
`git diff <source_branch> <target_branch>`

etiquetas

Se recomienda crear etiquetas para cada nueva versión publicada de un software. Este concepto no es nuevo, ya que estaba disponible en SVN. Puedes crear una nueva etiqueta llamada *1.0.0* ejecutando
`git tag 1.0.0 1b2e1d63ff`
1b2e1d63ff se refiere a los 10 caracteres del commit id al cual quieres referirte con tu etiqueta. Puedes obtener el commit id con
`git log`

también puedes usar menos caracteres que el commit id, pero debe ser un valor único.

reemplaza cambios locales

En caso de que hagas algo mal (lo que seguramente nunca suceda ;) puedes reemplazar cambios locales usando el comando
`git checkout -- <filename>`

Este comando reemplaza los cambios en tu directorio de trabajo con el último contenido de HEAD. Los cambios que ya han sido agregados al Index, así como también
los nuevos archivos, se mantendrán sin cambio.

Por otro lado, si quieres deshacer todos los cambios locales y commits, puedes traer la última versión del servidor y apuntar a tu copia local principal de esta forma
`git fetch origin`
`git reset --hard origin/master`

datos útiles

Interfaz gráfica por defecto
`gitk`

Colores especiales para la consola
`git config color.ui true`

Mostrar sólo una línea por cada commit en la traza

`git config format.pretty oneline`

Agregar archivos de forma interactiva
`git add -i`