GIT

rortegag

Configuración

Establecer el nombre y el correo electrónico que se adjuntará a sus confirmaciones y etiquetas.

```
$ git config --global user.name “Joe Doe”
$ git config --global user.email “joedoe@email.com”
```

Iniciar un proyecto

Crear un repositorio local (omitir <directorio> para inicializar el directorio actual como repositorio git).

```
$ git init <directorio>
```

Descargar un repositorio remoto.

```
$ git clone <url>
```

Hacer cambios

Añadir un archivo al stage (etapa).

```
$ git add <archivo>
```

Meter todos los archivos en stage.

```
$ git add .
```

Confirmar todos los archivos en git.

```
$ git commit -m “Mensaje del commit”
```

Añadir todos los cambios realizados en los archivos rastreados y confirmar.

```
$ git commit -am “Mensaje del commit”
```

Conceptos básicos

"main": rama de desarrollo por defecto.
"origin": repositorio por defecto.
"HEAD": rama actual.
"HEAD^": rama principal de HEAD.
"HEAD~4": tatarabuelo de HEAD.

Fusión

Fusionar la rama A en la rama B.

```
$ git checkout b
$ git merge a
```

Fusionar y aplastar todos los commits en un nuevo commit.

```
$ git merge --squash a
```

Ramas

Lista todas las ramas locales.

```
$ git branch
```

-r para ramas remotas.

```
$ git branch -r
```

-a para todas las ramas.

```
$ git branch -a
```

Crear una rama nueva.

```
$ git branch <nueva-rama>
```

Cambiar a una rama y actualizar el directorio de trabajo.

```
$ git checkout <rama>
```

Crear una nueva rama y cambiar a ella.

```
$ git checkout -b <nueva-rama>
```

Eliminar una rama fusionada.

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

Eliminar una rama, fusionada o no.

```
$ git branch -D <rama>
```

Añadir una etiqueta a la confirmación actual (a menudo se utiliza para el lanzamiento de nuevas versiones).

```
$ git tag <nombre-etiqueta>
```

Para más información: https://www.freecodecamp.org/espanol/news/guia-para-principiantes-de-git-y-github/

GIT

rortegag

Rebajar

Rebajar la rama de características a la principal (para incorporar los nuevos cambios realizados en la principal). Evita confirmaciones de fusión innecesarias en la función, manteniendo el historial limpio.

```
$ git checkout feature
$ git rebase main
```

Limpiar interativamente los commits de una rama antes de volver a basar en la principal.

```
$ git rebase -i main
```

Rebajar interativamente los últimos 3 commits en la rama actual.

```
$ git rebase -i HEAD~3
```

Revisar tu repositorio

Lista de archivos nuevos o modificados aún no confirmados.

```
$ git status
```

Lista del historial de commits, con sus respectivos IDs.

```
$ git status
```

Mostrar los cambios en los archivos no clasificados.

```
$ git diff
```

Para mostrar los cambios en los archivos escaneados, añade la opción --cached.

```
$ git diff --cached
```

Mostrar los cambios entre dos commits.

```
$ git diff commit1_ID commit2_ID
```

Deshaciendo cosas

Mover (y/o renombrar) un archivo y mover el escenario.

```
$ git mv <ruta_existente> <nueva_ruta>
```

Retirar sólo del área de preparación.

```
$ git rm --cached <archivo>
```

Ver una confirmación anterior (sólo lectura).

```
$ git checkout <commit_ID>
```

Volver a un commit anterior y borrar todos los commits anteriores a él (revertir es más seguro).

```
$ git reset <commit_ID>
```

Eliminar un archivo del directorio de trabajo y del área de preparación, y luego escenificar la eliminación.

```
$ git rm <archivo>
```

Crear un nuevo commit, revirtiendo los cambios de un commit especificado.

```
$ git revert <commit_ID>
```

Añade el parámetro --hard para eliminar también los cambios del espacio de trabajo (TEN MUCHO CUIDADO).

```
$ git reset --hard <commit_ID>
```

Almacenamiento

Almacenar cambios modificados y escalonados.

```
$ git stash
```

Para incluir los archivos no rastreados, añade el parámetro -u.

```
$ git stash -u
```

Para los archivos no rastreados e ignorados, añade el parámetro -a.

```
$ git stash -a
```

Como en el caso anterior, pero añadiendo un comentario.

```
$ git stash save “Comentario”
```

Almacenamiento parcial. Almacenar sólo un archivo, una colección de archivos o cambios individuales dentro de los archivos.

```
$ git stash -p
```

Lista de todos los almacenes.

```
$ git stash list
```

Volver a aplicar el alijo sin borrarlo.

```
$ git stash apply
```

MÁS

Para más información: https://www.freecodecamp.org/espanol/news/guia-para-principiantes-de-git-y-github/

Git

rortegag

Almacenando

Volver a aplicar el stash en el índice 2, el borrarlo de la lista de stash. Omitir stash@{n} para saltar el stash más reciente.

```
$ git stash pop stash@{2}
```

Mostrar el resumen diff del índice 1.
Usa el parámetro -p para ver el diff completo.

```
$ git stash show stash@{1}
```

Borrar el stash en el índice 1.
Omitir stash@{n} para borrar el último stash realizado.

```
$ git stash drop stash@{1}
```

Borrar todos los almacenes.

```
$ git stash clear
```

Sincronizamiento

Añadir un repositorio remoto.

```
$ git remote add <alias> <url>
```

Ver todas las conexiones remotas.

```
$ git remote
```

Añadir el parámetro -v para ver las urls.

```
$ git remote -v
```

Eliminar una conexión.

```
$ git remote remove <alias>
```

Cambiar el nombre de una conexión.

```
$ git remote rename <viejo> <nuevo>
```

Obtener todas las ramas del repositorio remoto (sin fusión).

```
$ git fetch <alias>
```

Buscar una rama específica.

```
$ git fetch <alias> <branch>
```

Obtener la copia del repositorio remoto de la rama actual, y luego fusionar.

```
$ git pull
```

Mover (rebase) sus cambios locales sobre los nuevos cambios realizados en el repositorio remoto (para un historial limpio y lineal).

```
$ git pull --rebase <alias>
```

Subir el contenido local al repositorio remoto.

```
$ git push <alias>
```

Subir a una rama (luego se puede hacer un pull request).

```
$ git push <alias> <branch>
```

Para más información: https://www.freecodecamp.org/espanol/news/guia-para-principiantes-de-git-y-github/