CIPFP Mislata

Centre Integrat Públic
Formació Professional Superior

Resumen comandos de Git

Autor: Joan Puigcerver Ibáñez
Correo electrónico: j.puigcerveribanez@edu.gva.es
Curso: 2022/2023

1.Introducción

En este material se recopila un resumen de los archivos, pedidos y opciones más utilizados de Git.

2.Archivos

- `~/.gitconfig`: Archivo de configuración global de tu usuario. Se almacena en la carpeta de usuario. Todas las configuraciones hechas con `git config --global`se almacenan en este archivo. También permite definir **alias**.

Ejemplo de `.gitconfig` con alias que permiten consultar el `log` de una forma más visual.

```
[user]
    name = <Your Name>
    email = <your email>
[alias]
    lg1 = log --graph --abbrev-commit --decorate --
format=format:'%C(bold blue)%h%C(reset) - %C(bold green)(%ar)%C(reset)
%C(white)%s%C(reset) %C(dim white)- %an%C(reset)%C(bold
yellow)%d%C(reset)'
    lg2 = log --graph --abbrev-commit --decorate --
format=format:'%C(bold blue)%h%C(reset) - %C(bold cyan)%aD%C(reset)
%C(bold green)(%ar)%C(reset)%C(bold yellow)%d%C(reset)%n''
%C(white)%s%C(reset) %C(dim white)- %an%C(reset)'
    lg = !"git lg1"
    lga = !"git lg --all"
```

- `.git/`: Directorio que contiene el *Repositorio Local* . Esta carpeta está presente en todos los repositorios de Git.
- `.gitignore`: Archivo que especifica qué archivos no queremos que Git considere, es decir, que Git debe ignorar.

3.Comandos básicos

- `git init`: Inicializa un repositorio vacío en la carpeta donde estés situado.
- `git clone <url> <destination>`: Clona un repositorio remoto, especificado por una `url`en el directorio `destination`. Si no se especifica ningún directorio, se clonará en un directorio nuevo con el nombre del repositorio.
- `git status`: Muestra el estado actual del repositorio, como los cambios en el *Directorio de trabajo* o el *Staging Area*.
- `git add <files or directory>`: Añade los archivos o directorios especificados al *Staging Area*.
- `git commit`: Crea un *commit* que hace efectivos los cambios del *Staging Area*. Por defecto, abra un editor para especificar el mensaje del *commit*.
    - Opción `-m <missatge>`: Permite especificar un mensaje en una línea de una forma rápida, sin necesidad de un editor de texto.
- `git restore <files or directory>`: Descarta los cambios de los archivos o directorios especificados del *Directorio de Trabajo*. **Una vez descartados NO se pueden volver a recuperar**.
- `git restore --staged <files or directory>`: Descarta los cambios de los archivos o directorios especificados del *Staging Area* y los devuelve al *Directorio de Trabajo*.
- `git log`: Permite consultar los commits realizados en la rama actual.
- `git diff <files or directory>`: Permite mostrar los cambios realizados en el *Directorio de Trabajo*. Se permite especificar el archivo o directorio del que se desea consultar los cambios. Si no se especifica, se muestran todos los cambios.
    - Opción `--staged`: Permite mostrar los cambios realizados en el *Staging Area*.
- `git diff <ref>`: Permite mostrar los cambios en una referencia, como un *commit* o una *etiqueta*.

4.Sincronización con el *Repositorio Remoto*

- `git push`: Publica y sincroniza los cambios realizas en el *Repositorio Local* en el *Repositorio Remoto*.
    - Opción `--tags`: Publica todas las etiquetas en el *Repositorio Remoto*.
- `git pull`: Incorpora y actualiza los cambios del *Repositorio Remoto* en el *Repositorio Local*. Este pedido puede dar lugar a conflictos.
- `git fetch`: Obtiene los cambios del *Repositorio Remoto* en el *Repositorio Local*, pero no actualiza los archivos.

5.Trabajar en ramas

- `git branch`: Muestra las ramas existentes en el repositorio.
- `git branch <branch name>`: Crea una nueva rama con el nombre especificado.
- `git branch -d <branch name>`: Elimina la rama con el nombre especificado.
- `git checkout <ref>`: Mueve el `HEAD` (estado actual del repositorio) a la referencia especificada. La referencia puede ser una rama, una etiqueta o un *commit hash*.
- `git checkout -b <branch name>`: Crea una nueva rama con el nombre especificado y mueve el `HEAD`a esta rama.

6.Fusionar ramas

- `git merge <branch name>`: Fusiona e incorpora los cambios de la rama especificada en la rama actual (donde está situal el `HEAD`).
    - Si la fusión es *fast forward*, se producirá sin necesidad de ninguna acción más.
    - Si la fusión no es *fast forward*, será necesario especificar el mensaje del *merge commit*.
    - Si hay conflictos, será necesario resolverlos.
    - Opción `--abort`: Aborta la fusión que se está produciendo.
- TODO: `git rebase`.

7.Etiquetas

- `git tag`: Muestra las etiquetas existentes en el repositorio.
- `git tag <tag name>`: Crea una nueva etiqueta *ligera* con el nombre especificado.
- `git tag -a <tag name>`: Crea una nueva etiqueta *anotada* con el nombre especificado. Es necesario indicar un mensaje.
    - Opción `-m`: Permite especificar un mensaje en una línea de una forma rápida, sin necesidad de un editor de texto.
- `git tag -d <tag name>`: Elimina la etiqueta con el nombre especificado.