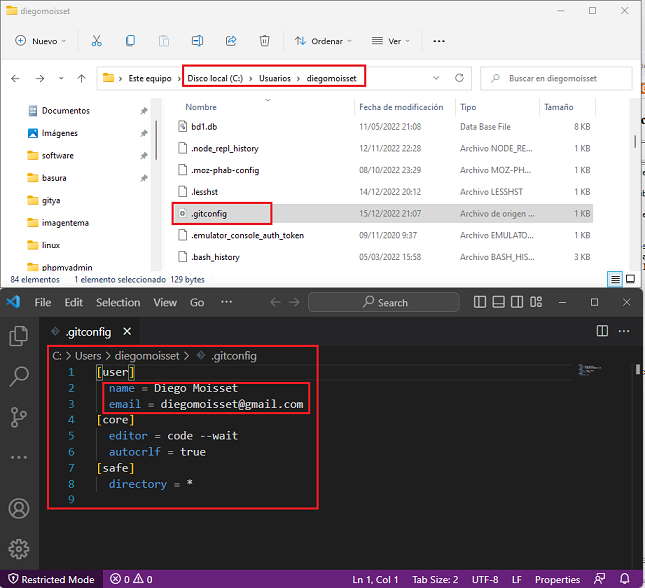
**Git –version**

Git –help o git -h

**git config --global user.name "Diego Moisset"**

**git config --global user.email diegomoisset@gmail.com**

En Windows se modifica el archivo '.gitconfig' en la carpeta del usuario respectivo:



Si queremos ver los datos de configuración que modificamos y otros que se inicializaron por defecto podemos nuevamente ejecutar el comando 'config' pasando el parámetro --list:

git config --list

Otro dato importante a configurar es el editor de text que abre Git cuando hay que introducir datos por teclado. Podemos ver en la imagen anterior que cuando instalamos Git en el equipo identificó 2 editores de texto:

core.editor="C:\\Program Files\\Notepad++\\notepad++.exe" -multiInst -notabbar -nosession -noPlugin

Y el VS Code:

core.editor=code --wait

El último indicado es el que queda configurado por defecto. En este caso el VS Code.

Por el momento vamos a dejar el resto de datos de configuración con los valores fijados por defecto.

Si necesitamos obtener una ayuda de un comando de git debemos llamar al comando "help" y pasar el nombre del comando que necesitamos consultar, luego Git nos abre el navegador con datos de la ayuda que se encuentran en forma local sin la necesidad de acceder a internet:

git help config

INICIANDO NUEVO REPOSITORIO

Git init

CLONANDO REPOSITORIO EXISTENTE

git clone https://github.com/git/git.git

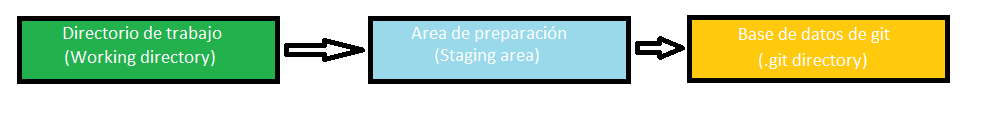
5 - Estados de los archivos cuando trabajamos con Git

Git tiene tres estados principales en los que se pueden encontrar los archivos de un proyecto:

* Modificado: Significa que has modificado el archivo pero todavía no lo has confirmado a la base de datos de Git.
* Preparado: Significa que hemos marcado un archivo modificado en su versión actual para que vaya en tu próxima confirmación.
* Confirmado: Significa que los datos están almacenados de manera segura en la base de datos de git.

Las tres secciones principales de un proyecto Git

El directorio de trabajo (working directory), el área de preparación (staging area) y el directorio de Git (.git directory):



comando status

El comando para determinar qué archivos están en qué estado es el comando status. Si ejecuta este comando inmediatamente después de crear un repositorio, por ejemplo ejecutemos el comando del repositorio creado en proyecto1:

git status

git add \*.py

para agregar todos los archivos con extensión .py

# **6 - Saltear el área de preparación**

**git commit -a -m "agregando y haciendo commit en una sola linea"**

# **7 - Eliminar archivos y cambiar sus nombres**

Por ejemplo si queremos eliminar el archivo programa1.py para que git lo deje de seguir:

git rm programa1.py

El archivo se elimina del área de trabajo y queda en el área de preparación hasta que se ejecute el próximo commit git.

De forma similar para cambiar el nombre de un archivo que estamos siguiendo con git debemos utilizar un comando:

git mv programa2.py programafinal.py

Tener en cuenta que tanto el comando rm y mv requieren un commit para que queden confirmados en el repositorio el borrado y la modificación del nombre de archivo:

# **8 - Consulta del historial de confirmaciones.**

**Git log -3.-** Muestra los últimos 3 commit.

git log -p -2

muestra las diferencias entre los archivos

Para poder ver algunos datos estadísticos podemos utilizar el parámetro --stat:

**Git log -2 –stat**

El parámetro --stat imprime tras cada confirmación una lista de archivos modificados, indicando cuántos han sido modificados y cuántas líneas han sido añadidas y eliminadas para cada uno de ellos, y un resumen de toda esta información.

Otro parámetro es --pretty, al cual le podemos asignar distintos valores (oneline, full, email, short, etc.), veamos con el valor oneline:

git log --pretty=oneline

La opción oneline imprime cada confirmación en una única línea, lo que puede resultar útil si estás analizando gran cantidad de confirmaciones:

También podemos consultar confirmaciones por rango de fechas:

git log --since="2022-12-01" --until="2022-12-16"

Podemos consultar todos los commit de un determinado autor:

git log --author="Diego Moisset"