

Comandos Principales de Git y su Función

Estudiante:

María Paula Ramírez Gómez

Universidad de Cundinamarca, Extensión Chía

Programación II

Docente:

William Alexander Matallana

2025

DOCUMENTO TÉCNICO DE Git EN IntelliJ

Esta guía tiene como propósito mostrar los principales comandos de Git, utilizando el entorno IntelliJ y la plataforma GitHub, para una mejor comprensión comenzaremos explicando qué es cada uno de ellos antes de profundizar en su uso.

IntelliJ: Es un entorno de desarrollo integrado (IDE) creado por JetBrains, diseñado principalmente para el desarrollo en Java, aunque también admite otros lenguajes como Kotlin, Python y JavaScript.

Git: Es un sistema de control de versiones distribuido que permite gestionar cambios en el código fuente de un proyecto.

GitHub: Es una plataforma en la nube que permite almacenar y gestionar repositorios Git de manera remota.

Comandos Principales de Git y su Función: Vamos a dividir los comandos en siete secciones para entender un poco más el orden en el que pueden ser usados:

1. Configuración:

git config --global user.name "Tu Nombre" = Configura el nombre del usuario.

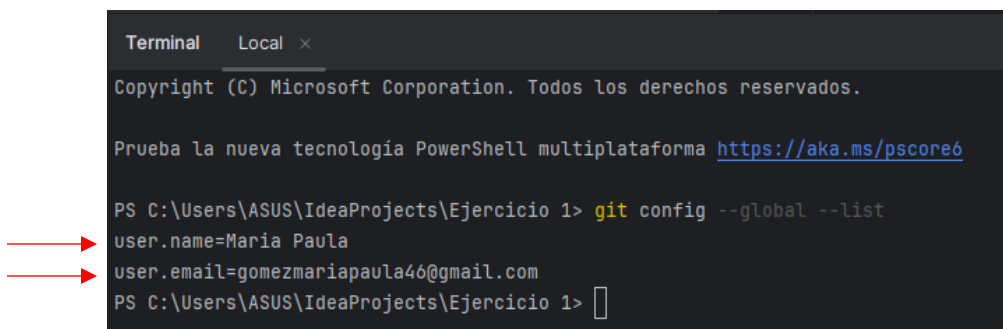
git config --global user.email "tuemail@example.com" Configura el correo electrónico del usuario.

```

PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global user.name "Maria Paula"
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global user.email "gomezmaripaula46@gmail.com"
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global --list
user.name=Maria Paula
user.email=gomezmaripaula46@gmail.com
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales>

```

git config --list → Muestra la configuración actual de Git.



```

Terminal Local x
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/pscore6

PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git config --global --list
user.name=Maria Paula
user.email=gomezmaripaula46@gmail.com
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1>

```

git config --global --unset = Eliminar el nombre del usuario.

git config --global --unset = Eliminar el correo electrónico del usuario.

```

PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global --unset user.name
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global --unset user.email
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales> git config --global --list
PS C:\Users\ASUS\IdeaProjects\Comandos Git Principales>

```

En este espacio vamos a configurar el Github creando un repositorio y vinculándolo con el Git para poder subir cambios a nuestro repositorio remoto.

- Crear repositorio en Github (tener en cuenta que ya se debe tener una cuenta).

Top repositories

Find a repository...

New

- Indicar el nombre que queremos y con el repositorio creado copiar al portapapeles el siguiente link.
- **Copiamos ese código en el entorno IntelliJ damos enter y verificamos que aparezca esta información.**

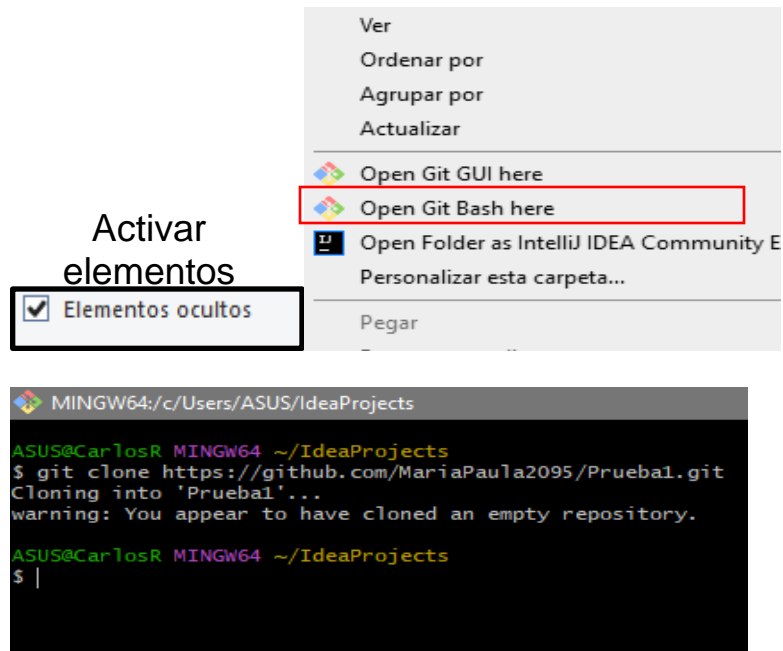
```
Enumerating objects: 3, done.  
Delta compression using up to 4 threads  
Writing objects: 100% (3/3), 268 bytes | 268.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
To https://github.com/MariaPaula2095/Comandos-Git-Principales.git  
* [new branch]      main -> main  
branch 'main' set up to track 'origin/main'.
```

2. Inicialización y Clonación:

git init → Inicializa un nuevo repositorio en el directorio actual.

```
PS C:\Users\ASUS\IdeaProjects\Comandos Git programacion> git init  
Reinitialized existing Git repository in C:/Users/ASUS/IdeaProjects/Comandos Git programacion/.git/  
PS C:\Users\ASUS\IdeaProjects\Comandos Git programacion> █
```

git clone <URL> → Clona un repositorio remoto en tu máquina.



3. Gestión de cambios:

git status → Muestra el estado actual del repositorio (archivos modificados, en seguimiento, etc.)

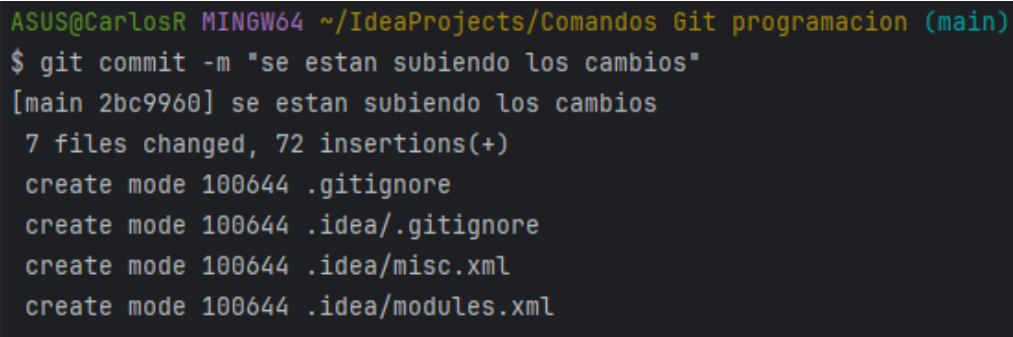
```
Untracked files:
(use "git add <file>..." to include in what will be committed)
.gitignore
.idea/
Comandos Git programacion.iml
src/

nothing added to commit but untracked files present (use "git add" to track)
```

git add <archivo> → Agrega un archivo al área de preparación (staging).

Para este caso se escribe específicamente que archivo queremos añadir.

git commit -m "Mensaje" → Guarda los cambios con un mensaje descriptivo.



A terminal window with a dark background. The prompt is 'ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)'. A red arrow points to the command '\$ git commit -m "se estan subiendo los cambios"'. The output shows the commit message '[main 2bc9960] se estan subiendo los cambios', the number of files changed ('7 files changed, 72 insertions(+)'), and a list of files created: '.gitignore', '.idea/.gitignore', '.idea/misc.xml', and '.idea/modules.xml'.

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)
$ git commit -m "se estan subiendo los cambios"
[main 2bc9960] se estan subiendo los cambios
7 files changed, 72 insertions(+)
create mode 100644 .gitignore
create mode 100644 .idea/.gitignore
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
```

git add . → Agrega todos los archivos modificados al área de preparación.

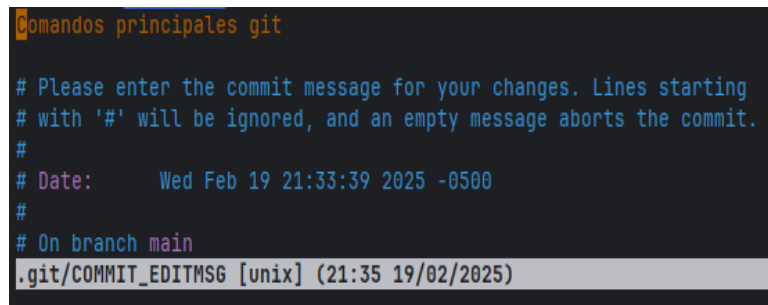
(Cabe aclarar que los cambios se pueden hacer desde la terminal local de IntelliJ o desde la terminal de Git Bash)



A terminal window with a dark background. The prompt is 'ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)'. The command '\$ git add .' is entered.

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)
$ git add .
```

git commit --amend → Modifica el último commit.



A terminal window showing the 'git commit' command in a text editor. The title bar is 'Comandos principales git'. The editor content shows the commit message prompt, a date, and the branch name. The cursor is at the end of the line '# On branch main'.

```
Comandos principales git
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Wed Feb 19 21:33:39 2025 -0500
#
# On branch main
.git/COMMIT_EDITMSG [unix] (21:35 19/02/2025)
```

4. Historial y seguimiento:

git log → Muestra el historial de commits.

```
commit af34cb92c65c56f208f59f654cc9f169980e7bc9 (HEAD -> main)
Author: Maria Paula <gomezmaripaula46@gmail.com>
Date:   Wed Feb 19 21:33:39 2025 -0500
```

Comandos principales git

```
commit 2fdc2db528388cd5465918c9bd6aa2b542c6ba89 (origin/main)
:
```

git log --oneline → Muestra el historial en una línea por commit.

```
Prueba la nueva tecnologia PowerShell multiplataforma https://aka.ms/pscore6
```

```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git log --oneline
af34cb9 (HEAD -> main) Comandos principales git
2fdc2db (origin/main) primer ejercicio
62b1444 first commit
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> 
```

5. Ramas (Branches):

git branch → Lista las ramas existentes.

```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git branch
→ * main
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> 
```

git switch -c → Crea una nueva rama.

```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git switch -c comandos
Switched to a new branch 'comandos'
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> █
```

git branch -D → Elimina una rama existente.

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)
$ git branch
  Comandos
* main
```

Tenemos dos ramas

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)
$ git branch -D comandos
Deleted branch comandos (was 2bc9960).
```

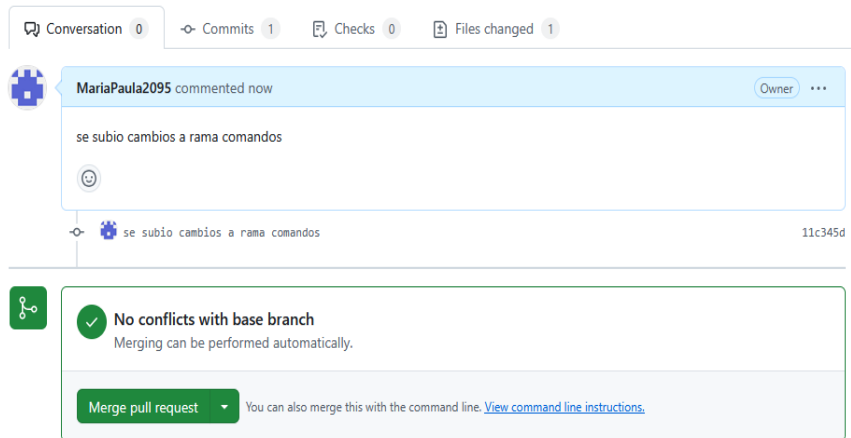
Se elimino rama comandos

git switch <nombre_rama> → Cambia a una rama existente.

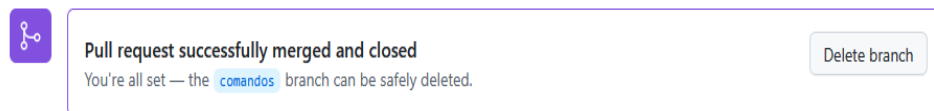
```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git switch comandos
Already on 'comandos'
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> █
```

git merge <rama> → Fusiona una rama en la actual.

git rebase <rama> → Reaplica los commits de una rama sobre otra.



Confirmamos el merge



6. Trabajo con remotos:

git remote add origin <URL> → Vincula el repositorio local con un repositorio remoto.

```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git remote add prueba1 https://github.com/MariaPaula2095/Prueba1.git
```

git remote -v → Muestra las URLs de los repositorios remotos configurados.

```
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1> git remote -v
origin https://github.com/MariaPaula2095/Ejercicio1Java.git (fetch)
origin https://github.com/MariaPaula2095/Ejercicio1Java.git (push)
prueba1 https://github.com/MariaPaula2095/Prueba1.git (fetch)
prueba1 https://github.com/MariaPaula2095/Prueba1.git (push)
PS C:\Users\ASUS\IdeaProjects\Ejercicio 1>
```

git push origin <rama> → Envía los commits al repositorio remoto.

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (main)
$ git push
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 1.84 KiB | 627.00 KiB/s, done.
```

→ Cuando tengo 1 sola rama como en este caso es suficiente con <git push>

git pull origin <rama> → Obtiene y fusiona los cambios del repositorio remoto.

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/Comandos Git programacion (comandos)
$ git pull origin main
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 907 bytes | 129.00 KiB/s, done.
From https://github.com/MariaPaula2095/Comandos-Git-Principales
* branch          main          -> FETCH_HEAD
```

7. Restauración y deshacer cambios:

git revert → Quita un archivo del área de preparación.

```
B
Revert "se creo archivo1"

This reverts commit 4e28f0101d14b31ce07bc4f3fa1636de6549ea2b.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
.git/COMMIT_EDITMSG[+] [unix] (15:06 21/02/2025)
```

```
#
# On branch comandos
# Your branch is ahead of 'origin/main' by 1 commit.
#   (use "git push" to publish your local commits)
#
# Changes to be committed:
#   deleted:    archivo.txt
#
.git/COMMIT_EDITMSG[+] [unix] (15:06 21/02/2025)
-- INSERT --
```

- Como asociar un archivo a un repositorio existente:

En la carpeta que vamos a asociar creo un archivo

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/carpeta-repositorio
$ touch archivocarpeta.txt
```

Asocio la carpeta con el repositorio y git status

```
[master (root-commit) 653c952] first commit
1 file changed, 1 insertion(+)
 create mode 100644 README.md
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 241 bytes | 241.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MariaPaula2095/carpeta-repositorio.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

ASUS@CarlosR MINGW64 ~/IdeaProjects/carpeta-repositorio (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

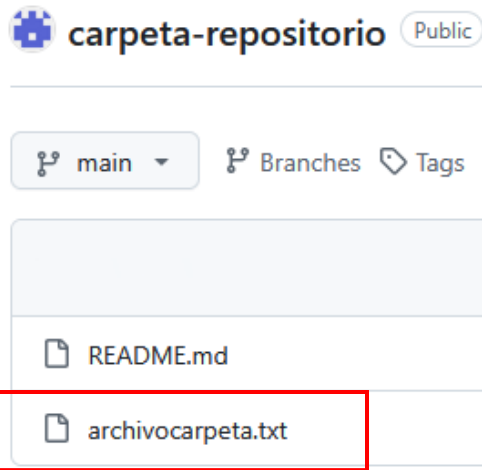
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    archivocarpeta.txt
```

Sigo subiendo el archivo hasta que sea exitoso

```
ASUS@CarlosR MINGW64 ~/IdeaProjects/carpeta-repositorio (main)
$ git commit -m "Se sube archivo a repositorio"
[main d09b838] Se sube archivo a repositorio
1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 archivocarpeta.txt

ASUS@CarlosR MINGW64 ~/IdeaProjects/carpeta-repositorio (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 306 bytes | 153.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/MariaPaula2095/carpeta-repositorio.git
 653c952..d09b838  main -> main
```

Verifico en el GitHub:



Conclusiones:

Git es una herramienta esencial para el control de versiones, ya que permite gestionar cambios en el código de manera eficiente, facilitando la colaboración entre desarrolladores y asegurando la integridad del proyecto. Su integración con plataformas como GitHub y entornos de desarrollo como IntelliJ mejora aún más su utilidad en el desarrollo de software.

El uso de comandos Git junto con ejemplos prácticos y capturas facilita el aprendizaje, permitiendo comprender mejor su funcionamiento. Documentar estos procesos no solo ayuda a reforzar conocimientos, sino que también sirve como una referencia útil para futuras implementaciones.

