Comandos de Git/GitHub más usados

Git: aprende control de versiones con práctica guiada

Git es una herramienta esencial para desarrollo de software, DevOps, SRE, automatización,
documentación técnica y trabajo colaborativo. Permite guardar cambios, comparar versiones,
crear ramas, trabajar con equipos y conectar tu repositorio local con plataformas como
GitHub o GitLab.

Ver 40 comandos

¿Qué es Git y por qué es importante?

Git registra el historial de cambios de un proyecto. Cada commit funciona como un punto de control
que ayuda a revisar qué cambió, quién lo cambió y por qué. Con ramas puedes crear nuevas funcionalidades
sin afectar la versión principal. Con remotos puedes sincronizar el trabajo con servidores como GitHub,
GitLab, Bitbucket o repositorios internos de una empresa.

1. Control

Conserva el historial, permite volver a versiones anteriores y

reduce el riesgo de perder trabajo.

2. Colaboración

Permite trabajar con ramas, revisiones, Pull Requests, Merge Requests y colaboración en equipo.

3. Automatización

Es la base para CI/CD, despliegues, auditoría técnica y prácticas DevOps modernas.

Emulador interactivo de Git

Practica comandos en un entorno seguro de aprendizaje. El emulador muestra la terminal,
el resultado de salida, el canvas visual y la explicación textual de lo que ocurre en cada paso.

40 comandos de Git explicados

Usa esta tabla como guía rápida. Cada comando se puede copiar y pegar en el emulador
para practicar el flujo de trabajo de Git paso a paso.

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
1	git --version	Verifica la versión instalada de Git.	Confirmar que Git está instalado antes de iniciar.	Copiar

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
2	git config --global user.name "Tu Nombre"	Configura el nombre global del usuario de Git.	Configuración inicial del equipo de trabajo.	Copiar
3	git config --global user.email "correo@dominio.com"	Configura el correo global del usuario de Git.	Asociar los commits con tu identidad.	Copiar
4	git config --list	Muestra la configuración activa de Git.	Revisar usuario, correo y ajustes.	Copiar
5	git init	Crea un repositorio Git en la carpeta actual.	Iniciar control de versiones local.	Copiar
6	git clone https://github.com/usuario/proyecto.git	Descarga una copia completa de un repositorio remoto.	Trabajar en un proyecto existente.	Copiar
7	git status	Muestra archivos modificados, nuevos o preparados.	Revisar el estado antes de hacer commit.	Copiar
8	git add file.txt	Agrega un archivo específico al área de preparación.	Preparar un cambio puntual.	Copiar
9	git add .	Agrega todos los cambios del directorio actual.	Preparar todos los cambios locales.	Copiar

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
10	git restore file.txt	Descarta cambios no preparados en un archivo.	Revertir ediciones locales no deseadas.	Copiar
11	git restore --staged file.txt	Quita un archivo del área de preparación sin borrar los cambios.	Corregir lo que estaba por confirmarse.	Copiar
12	git commit -m "Mensaje claro del cambio"	Registra los cambios preparados en el historial.	Guardar una unidad lógica de trabajo.	Copiar
13	git commit --amend -m "Nuevo mensaje"	Actualiza el último commit local.	Corregir un mensaje o incluir un archivo olvidado.	Copiar
14	git log --oneline	Muestra el historial compacto de commits.	Revisar rápidamente la línea de tiempo.	Copiar
15	git log --graph --oneline --decorate --all	Muestra el historial con ramas y referencias.	Entender visualmente ramas y merges.	Copiar
16	git diff	Muestra diferencias no preparadas.	Revisar cambios antes de usar git add.	Copiar
17	git diff --staged	Muestra diferencias ya preparadas.	Validar lo que entrará al commit.	Copiar

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
18	git branch	Lista las ramas locales.	Saber en qué rama estás trabajando.	Copiar
19	git branch feature/login	Crea una nueva rama sin cambiarte a ella.	Preparar una línea de trabajo paralela.	Copiar
20	git checkout -b feature/login	Crea una rama y cambia a ella.	Iniciar una nueva funcionalidad.	Copiar
21	git switch main	Cambia a la rama main.	Regresar a la rama principal.	Copiar
22	git switch -c feature/api	Crea y cambia a una rama usando sintaxis moderna.	Alternativa moderna a checkout -b.	Copiar
23	git merge feature/login	Integra una rama dentro de la rama actual.	Unir cambios ya validados.	Copiar
24	git merge --abort	Cancela un merge con conflictos.	Volver al estado previo al merge.	Copiar
25	git rebase main	Reaplica los commits de la rama actual sobre main.	Mantener un historial lineal antes de integrar.	Copiar
26	git rebase --abort	Cancela un rebase en progreso.	Salir de un rebase problemático.	Copiar

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
27	git remote -v	Muestra los repositorios remotos configurados.	Verificar origin u otras URLs remotas.	Copiar
28	git remote add origin https://github.com/usuario/proyecto.git	Agrega un repositorio remoto llamado origin.	Conectar un proyecto local con un servidor remoto.	Copiar
29	git fetch origin	Descarga referencias remotas sin mezclar cambios.	Actualizar información antes de decidir.	Copiar
30	git pull origin main	Descarga e integra cambios remotos de main.	Actualizar tu rama local.	Copiar
31	git push origin main	Envía commits locales a main remoto.	Publicar cambios confirmados.	Copiar
32	git push -u origin feature/login	Publica una rama y configura el seguimiento upstream.	Primera subida de una rama nueva.	Copiar
33	git stash	Guarda cambios temporalmente sin hacer commit.	Cambiar de rama sin perder trabajo.	Copiar
34	git stash list	Lista los cambios guardados temporalmente.	Revisar stashes disponibles.	Copiar

#	COMANDO	QUÉ HACE	CUÁNDO USARLO	COPIAR
35	git stash pop	Restaura el último stash y lo elimina de la lista.	Continuar trabajo pausado.	Copiar
36	git tag v1.0.0	Crea una etiqueta local.	Marcar una versión importante.	Copiar
37	git push origin v1.0.0	Publica una etiqueta en el remoto.	Compartir una versión liberada.	Copiar
38	git reset --soft HEAD~1	Deshace el último commit conservando los cambios preparados.	Reorganizar un commit reciente.	Copiar
39	git reset --hard HEAD~1	Elimina el último commit y sus cambios locales.	Usar solo con seguridad; elimina cambios.	Copiar
40	git cherry-pick abc1234	Aplica un commit específico en la rama actual.	Mover un cambio puntual entre ramas.	Copiar

Recomendación de aprendizaje

Comienza con status, add, commit, branch, merge, pull y push. Después avanza con rebase, stash, tag, reset y cherry-pick.

50 Comandos esenciales de Linux que debes conocer

LINUX CLI LAB

Linux: 50 comandos esenciales que debes conocer

Linux es uno de los sistemas operativos mas usados en servidores, programacion, nube, DevOps y administracion de sistemas. Esta guia combina una tabla practica con un emulador interactivo para practicar comandos, reforzar conceptos y visualizar cambios simulados sin tocar un servidor real.

Nota de aprendizaje: la salida del emulador es simulada para practica, capacitacion y onboarding.
En un sistema real puede variar por distribucion, permisos, paquetes instalados y configuracion.

Emulador Linux CLI interactivo

Escribe un comando, usa la tabla o ejecuta un reto guiado. El emulador mostrara salida simulada, retroalimentacion y cambios visuales del sistema Linux.

Ejecutar
Ver salida grande
Probar reto
Limpiar
Reto: ejecuta pwd para revisar la ruta actual.
LX
[]
::
..
LINUX WORKSPACE
Pack
Free ▼
ESTADO DEL SISTEMA 0

linux-runbook.md
0/10

Linux CLI

Ejecuta el siguiente comando para recibir retroalimentacion guiada.

Comandos esperados del runbook
pwd
Ejecuta el siguiente comando para recibir retroalimentacion guiada.
Ejecuta el siguiente comando para recibir retroalimentacion guiada.

Visualizador del sistema Linux
Entorno listo: sin cambios todavia.

SALIDA DEL WORKSPACE
esperando

```
$ Emulador Linux listo
Elige un comando de la tabla o escribe uno en la consola.
```


```
$ Emulador Linux listo
Elige un comando de la tabla o escribe uno en la consola.
```


Salida esperada simulada

Copiar salida
Cerrar
Copiar tabla completa (CSV)

En pantallas pequenas, la tabla se adapta a tarjetas para evitar cortes de contenido.

#	Comando	Descripcion	Ejemplo	Acciones

Conclusion

Estos 50 comandos de Linux cubren navegacion, archivos, permisos, procesos, red, paquetes y soporte operativo. Practicarlos en un entorno simulado permite aprender con seguridad antes de trabajar en infraestructura real.

Los 10 comandos de Linux más usados para administración de

sistemas

🚀 Los 10 comandos de Linux más usados para administración de sistemas

Linux es el corazón de muchos servidores y sistemas críticos. Como administrador, dominar ciertos comandos te permite mantener la seguridad, el rendimiento y la estabilidad de tu infraestructura. Aquí tienes una guía práctica con los 10 comandos más usados en la administración de sistemas Linux, explicados de forma sencilla.

1. 🖥️ top – Monitorear procesos en tiempo real

Muestra los procesos en ejecución, su consumo de CPU y memoria.

Copiar

```
top
top -u usuario   # Procesos de un usuario específico
```

2. 📊 htop – Monitor interactivo

Una versión mejorada y visual de top, con navegación sencilla.

Copiar

```
htop
```

3. ⚙ systemctl – Administrar servicios

Controla el arranque, detención y estado de los servicios en sistemas con systemd.

Copiar

```
systemctl status nginx
systemctl restart ssh
systemctl enable apache2
```

4. df – Espacio en disco

Permite conocer el espacio libre y ocupado en discos y particiones.

Copiar

```
df -h     # Espacio en formato legible
df -i     # Inodos disponibles
```

5. du – Tamaño de directorios

Analiza cuánto ocupan directorios y archivos.

Copiar

```
du -sh *   # Tamaño resumido por carpeta
du -sh /var/log
```

6. chmod y chown – Permisos y

propietarios

Controlan quién puede leer, escribir o ejecutar archivos y a quién pertenecen.

Copiar

```
chmod 755 script.sh
chown usuario:grupo archivo.txt
```

7. □ netstat / ss – Conexiones de red

Permiten revisar puertos abiertos y conexiones activas.

Copiar

```
netstat -tulnp
ss -ltnp
```

8. □ tail – Ver logs en vivo

Muestra las últimas líneas de un archivo, útil para monitorear logs.

Copiar

```
tail -f /var/log/syslog
tail -n 100 /var/log/auth.log
```

9. □□ ufw – Cortafuegos sencillo

Configura reglas básicas de firewall en sistemas basados en Ubuntu/Debian.

Copiar

```
ufw status
ufw allow 22/tcp
ufw deny 80/tcp
```

10. tar – Comprimir y descomprimir

El comando estándar para empaquetar y respaldar archivos.

Copiar

```
tar -czvf backup.tar.gz /carpeta
tar -xzvf backup.tar.gz
```

Conclusión

Estos 10 comandos forman la base del día a día en la administración de Linux. Aprender a dominarlos te permitirá monitorear, proteger y mantener sistemas de forma más ágil y segura. Como todo en Linux, mientras más los uses, más natural será tu trabajo como administrador.