Most-Used Git/GitHub Commands

Git: learn version control through guided practice

Git is an essential tool for software development, DevOps, SRE, automation,
technical documentation, and collaborative work. It lets you save changes, compare versions,
create branches, work with teams, and connect your local repository with platforms such as
GitHub or GitLab.

View 40 commands

What is Git and why does it matter?

Git records a project’s change history. Each commit works as a checkpoint that helps you review
what changed, who changed it, and why. With branches, you can build new features without affecting
the main version. With remotes, you can synchronize work with servers such as GitHub, GitLab,
Bitbucket, or internal company repositories.

1. Control

Keeps history, supports returning to previous versions, and reduces the risk of losing work.

2. Collaboration

Supports branches, reviews, Pull Requests, Merge Requests, and teamwork.

3. Automation

Forms the foundation for CI/CD, deployments, technical auditing, and modern DevOps practices.

Interactive Git emulator

Practice commands in a safe learning environment. The emulator shows the terminal, result output,
visual canvas, and textual explanation of what happens at each step.

40 Git commands explained

Use this table as a quick guide. Each command can be copied and pasted into the emulator
to practice the Git workflow step by step.

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
1	git --version	Checks the installed Git version.	Confirm Git is installed before starting.	Copy
2	git config --global user.name "Your Name"	Sets the global Git user name.	Initial workstation setup.	Copy

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
3	git config --global user.email "email@domain.com"	Sets the global Git user email.	Associate commits with your identity.	Copy
4	git config --list	Shows the active Git configuration.	Review user, email, and settings.	Copy
5	git init	Creates a Git repository in the current folder.	Start local version control.	Copy
6	git clone https://github.com/user/project.git	Downloads a full copy of a remote repository.	Work on an existing project.	Copy
7	git status	Shows modified, new, or staged files.	Review state before committing.	Copy
8	git add file.txt	Adds a specific file to the staging area.	Prepare a focused change.	Copy
9	git add .	Adds all changes from the current directory.	Stage all local changes.	Copy
10	git restore file.txt	Discards unstaged changes in a file.	Revert unwanted local edits.	Copy

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
11	git restore --staged file.txt	Removes a file from staging without deleting edits.	Fix what was about to be committed.	Copy
12	git commit -m "Clear change message"	Records staged changes in history.	Save a logical unit of work.	Copy
13	git commit --amend -m "New message"	Updates the latest local commit.	Fix a message or include a forgotten file.	Copy
14	git log --oneline	Shows compact commit history.	Quickly review the timeline.	Copy
15	git log --graph --oneline --decorate --all	Shows history with branches and references.	Understand branches and merges visually.	Copy
16	git diff	Shows unstaged differences.	Review changes before git add.	Copy
17	git diff --staged	Shows staged differences.	Validate what will enter the commit.	Copy
18	git branch	Lists local branches.	Know which branch you are working on.	Copy

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
19	git branch feature/login	Creates a new branch without switching to it.	Prepare a parallel line of work.	Copy
20	git checkout -b feature/login	Creates a branch and switches to it.	Start a new feature.	Copy
21	git switch main	Switches to the main branch.	Return to the main branch.	Copy
22	git switch -c feature/api	Creates and switches to a branch using modern syntax.	Modern alternative to checkout -b.	Copy
23	git merge feature/login	Integrates a branch into the current branch.	Join validated changes.	Copy
24	git merge --abort	Cancels a merge with conflicts.	Return to the pre-merge state.	Copy
25	git rebase main	Reapplies current branch commits on top of main.	Keep a linear history before integration.	Copy
26	git rebase --abort	Cancels an in-progress rebase.	Exit a problematic rebase.	Copy

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
27	git remote -v	Shows configured remote repositories.	Verify origin or other remote URLs.	Copy
28	git remote add origin https://github.com/user/project.git	Adds a remote repository named origin.	Connect a local project to a remote server.	Copy
29	git fetch origin	Downloads remote references without merging.	Update information before deciding.	Copy
30	git pull origin main	Downloads and integrates remote main changes.	Update your local branch.	Copy
31	git push origin main	Sends local commits to remote main.	Publish committed changes.	Copy
32	git push -u origin feature/login	Publishes a branch and sets upstream tracking.	First push of a new branch.	Copy
33	git stash	Temporarily saves changes without committing.	Switch branches without losing work.	Copy
34	git stash list	Lists temporarily saved changes.	Review available stashes.	Copy

#	COMMAND	WHAT IT DOES	WHEN TO USE IT	COPY
35	git stash pop	Restores the latest stash and removes it from the list.	Continue paused work.	Copy
36	git tag v1.0.0	Creates a local tag.	Mark an important version.	Copy
37	git push origin v1.0.0	Publishes a tag to the remote.	Share a released version.	Copy
38	git reset --soft HEAD~1	Undoes the last commit while keeping changes staged.	Reorganize a recent commit.	Copy
39	git reset --hard HEAD~1	Deletes the last commit and its local changes.	Use only when sure; it removes changes.	Copy
40	git cherry-pick abc1234	Applies a specific commit to the current branch.	Move one targeted change between branches.	Copy

Learning recommendation

Start with status, add, commit, branch, merge, pull, and push. Then move to rebase, stash, tag, reset, and cherry-pick.

50 Most-Used Azure CLI Commands and Parameters

☁️ 50 Most-Used Azure CLI Commands and Parameters (With Interactive Emulator)

Azure CLI (az) is Microsoft Azure's official command-line tool. It helps you
provision, automate, and manage cloud resources faster than using the portal for repetitive tasks.
This guide gives you a practical reference with 50 high-impact commands/parameters, plus an interactive
emulator so readers can "run" commands and instantly see expected output.

Learning note: Emulator output is simulated for training and onboarding. Real output
can vary by subscription, region, permissions, and Azure CLI version.

Azure CLI Emulator (For Engagement and Hands-On Practice)

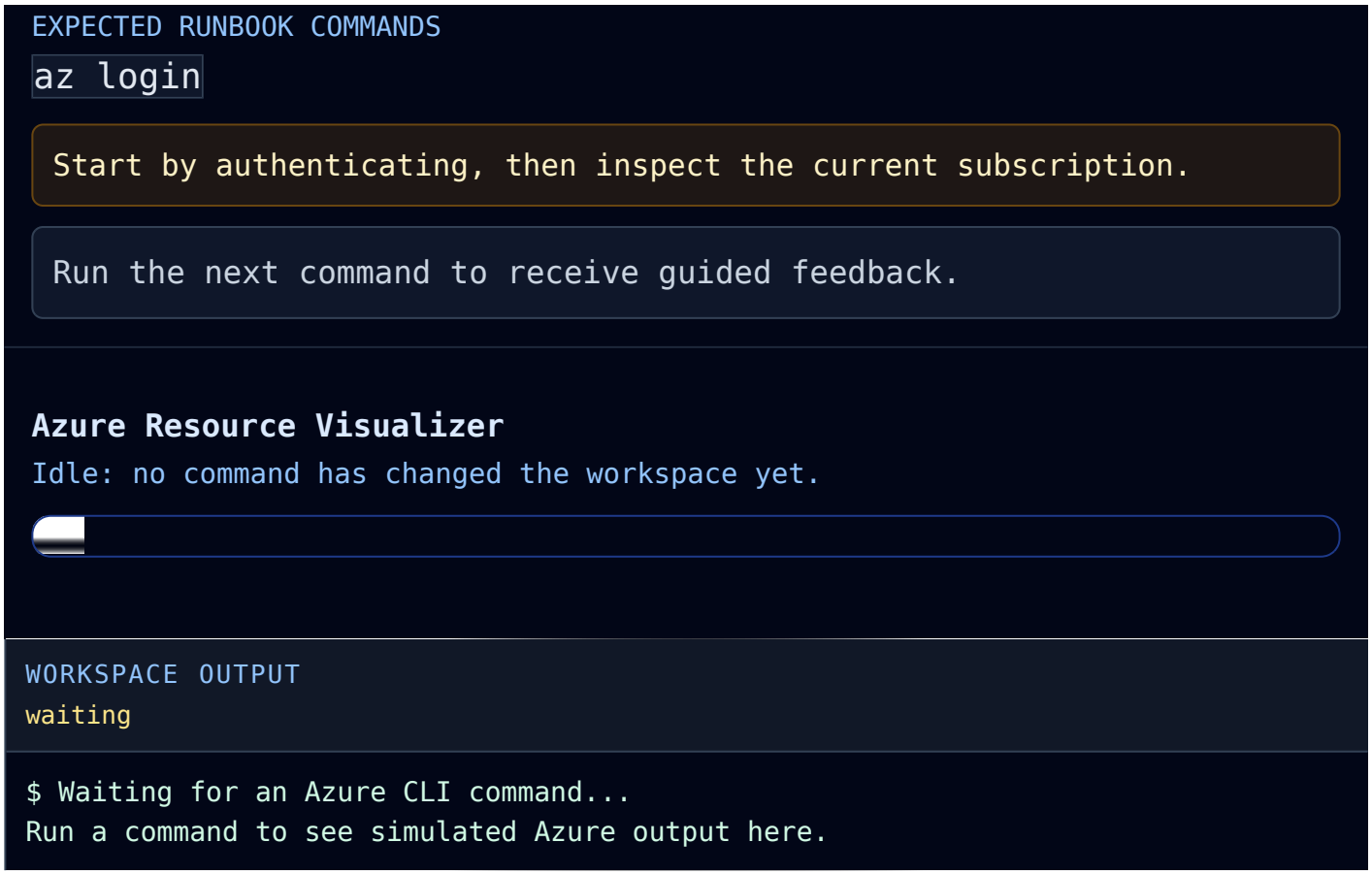
Try a command from the table, click **Run**, and see a simulated result in the terminal.
You can also open the same result in a browser prompt() using **Show Prompt Output**.

Run
Show Prompt Output
Try Challenge
Clear
Challenge: Run az login to start your session.

cloud-runbook.md
0/10

Start Azure session

Authenticate and inspect your active subscription before creating cloud resources.



$ Azure CLI Emulator Ready

Tip: choose a command from the table and click "Use in Emulator".

Expected output (simulated)

Copy Output

Close

Copy Full Table (CSV)

✅ Conclusion

These 50 Azure CLI entries cover the core day-to-day operations: authentication, subscriptions,
resource groups, compute, networking, storage, AKS, SQL, monitoring, and output shaping.
If you practice these consistently, you’ll manage Azure infrastructure faster and with more confidence.