# GitHub CLI

GitHub from the terminal

**START HERE**

**`gh auth login`**
Sign in to GitHub

**`gh repo clone <owner>/<repo>`**
Clone a repository

**`gh pr create --fill`**
Open a pull request

**`gh pr checkout <number>`**
Try out a pull request

**`gh pr merge --squash`**
Merge a pull request

**`gh issue list`**
See open issues

**`gh run watch`**
Follow a CI run

**`gh api <endpoint>`**
Call the REST API

**Branch** local commits → `gh pr create` → **Pull request** review and checks → `gh pr merge` → **Merged** base branch → `gh run watch` → **Actions run** CI and deploy

## CONCEPTS

Inside a git clone, gh picks the repo from the remotes. Elsewhere add -R <owner>/<repo>.

Most list and view commands take --json <fields> with --jq <expr> for scripting.

Without a terminal (pipes, scripts) set GH_TOKEN instead of running gh auth login.

## AUTH AND SETUP

**`gh auth login`**
Sign in interactively

**`gh auth status`**
Show who you are signed in as

**`gh auth login --with-token < <file>`**
Sign in with a token from stdin

**`gh auth setup-git`**
Use gh as git credential helper

## REPOSITORIES

**`gh repo clone <owner>/<repo>`**
Clone a repository

**`gh repo create <name> --public --clone`**
Create a repo and clone it

**`gh repo create <name> --private --source=. --push`**
Publish the current folder

**`gh repo view --web`**
Open the repo in the browser

**`gh repo fork --clone`**
Fork and clone

**`gh repo list <owner>`**
List an owner's repos

**`gh repo set-default`**
Pick the default repo for gh

**`gh browse`**
Open the repo, file or PR in the browser

## PULL REQUESTS

**`gh pr create --fill`**
Open a PR with title from commits

**`gh pr create --draft --title "<t>" --body "<b>"`**
Open a draft PR

**`gh pr list`**
List open PRs

**`gh pr view <number>`**
Show a PR

## PULL REQUESTS (cont.)

**`gh pr checkout <number>`**
Check out a PR branch

**`gh pr status`**
PRs that involve you

**`gh pr diff <number>`**
Show the PR diff

**`gh pr checks --watch`**
Watch CI checks on this PR

**`gh pr review --approve`**
Approve a PR

**`gh pr review --request-changes -b "<text>"`**
Request changes

**`gh pr comment <number> -b "<text>"`**
Comment on a PR

**`gh pr merge --squash --delete-branch`**
Squash-merge and clean up

**`gh pr merge --auto --squash`**
Merge when checks pass

**`gh pr ready`**
Mark a draft as ready

**`gh pr edit <number> --add-label <name>`**
Change labels, reviewers, base

**`gh pr list --author @me --state all`**
Your PRs in any state

**`gh pr create --reviewer <user> --label <name>`**
Request review while opening

**`gh pr close <number>`**
Close without merging

## ISSUES

**`gh issue list`**
List open issues

**`gh issue create --title "<t>" --body "<b>"`**
File an issue

**`gh issue view <number>`**
Show an issue

**`gh issue comment <number> -b "<text>"`**
Comment on an issue

**`gh issue close <number>`**
Close an issue

**`gh issue list --assignee @me`**
Issues assigned to you

**`gh issue list --label <name> --state all`**
Filter by label and state

**`gh issue create --assignee @me --label <name>`**
File and self-assign

## ISSUES (cont.)

**`gh issue develop <number> --checkout`**
Create a linked branch for an issue

**`gh issue edit <number> --add-label <name>`**
Change labels or assignees

**`gh issue reopen <number>`**
Reopen an issue

## ACTIONS

**`gh run list`**
Recent workflow runs

**`gh run view <run-id>`**
Summary of a run

**`gh run watch`**
Follow a run live

**`gh run view <run-id> --log-failed`**
Logs of failed steps only

**`gh run rerun <run-id> --failed`**
Rerun only failed jobs

**`gh run list --branch <branch> --workflow <name>`**
Filter runs

**`gh run download <run-id>`**
Download run artifacts

**`gh workflow list`**
List workflows

**`gh workflow run <workflow> -f <key>=<value>`**
Trigger a workflow_dispatch

## RELEASES AND GISTS

**`gh release list`**
List releases

**`gh release create <tag> --generate-notes`**
Publish a release

**`gh release create <tag> <file>...`**
Release with attached assets

**`gh release download <tag>`**
Download release assets

**`gh gist create <file>`**
Share a file as a gist

## SEARCH AND STATUS

**`gh search repos <query>`**
Search repositories

**`gh search issues <query>`**
Search issues across GitHub

**`gh search prs --author @me --state open`**
Your open PRs everywhere

## SEARCH AND STATUS (cont.)

**`gh search code <query>`**
Search code

**`gh status`**
Your PRs, issues and notifications

**`gh secret set <name>`**
Set an Actions secret

## API AND SCRIPTING

**`gh api repos/{owner}/{repo}/issues`**
GET a REST endpoint
*{owner} and {repo} are filled in from the current repo*

**`gh api <endpoint> --jq '<expr>'`**
Filter the response with jq

**`gh api <endpoint> --paginate`**
Fetch every page

**`gh api <endpoint> -f <key>=<value>`**
Send a string field (POST)

**`gh api <endpoint> -F <key>=<value>`**
Typed field: numbers, bools, @file

**`gh api -X PATCH <endpoint> -f <key>=<value>`**
Choose the HTTP method

**`gh api graphql -f query='<query>'`**
Run a GraphQL query

**`gh pr list --json number,title --jq '.[].title'`**
JSON fields, then jq

## RECIPES

**Feature branch to merged PR**

```
git switch -c feature/x
# commit your work
git push -u origin feature/x
gh pr create --fill
gh pr checks --watch
gh pr merge --squash --delete-branch
```

**Review someone's PR**

```
gh pr checkout <number>
gh pr diff
gh pr review --approve -b "LGTM"
```

**Debug a failed CI run**

```
gh run list --branch <branch> --limit 5
gh run view <run-id> --log-failed
gh run rerun <run-id> --failed
```

**Publish a local project**

```
git init
git add . && git commit -m "Initial commit"
gh repo create <name> --private --source=. --push
```