GitHub CLI Quick Reference

Repos, issues, PRs, actions, releases, API

Setup

Installation

`brew install gh`	macOS via Homebrew
`sudo apt install gh`	Debian / Ubuntu
`winget install GitHub.cli`	Windows via winget
`conda install gh`	Via conda-forge

Authentication

```
gh auth login                # interactive login
gh auth login --with-token < token.txt
gh auth status               # check auth state
gh auth refresh -s repo,gist   # add scopes
```

Configuration

```
gh config set editor vim
gh config set pager less
gh config set git_protocol ssh
gh config list
```

Repos

Repository Commands

```
gh repo create my-app --public --clone
gh repo clone owner/repo
gh repo fork owner/repo --clone
gh repo view owner/repo --web
```

Repo Options

`--public | --private`	Set repository visibility
`--template owner/repo`	Create from template repo
`--clone`	Clone after creating
`--add-readme`	Initialize with README
`gh repo list owner`	List repos for owner
`gh repo delete owner/repo`	Delete repository (with confirm)
`gh repo rename new-name`	Rename current repo
`gh repo archive owner/repo`	Archive a repository

Issues

Managing Issues

```
gh issue create --title "Bug" --body "Details here"
gh issue list --state open --label bug
gh issue view 42
gh issue close 42 --reason completed
```

Issue Options

`--assignee @me`	Assign to yourself
`--label bug,urgent`	Add labels
`--milestone v2.0`	Set milestone
`--project "Board"`	Add to project
`gh issue edit 42`	Edit issue interactively
`gh issue reopen 42`	Reopen a closed issue
`gh issue comment 42 -b "msg"`	Add comment to issue
`gh issue pin 42`	Pin issue to repo

Pull Requests

Creating & Managing PRs

```
gh pr create --title "feat: add auth" --body "..."
gh pr create --fill          # title/body from commits
gh pr list --state open
gh pr view 123 --web
```

Review & Merge

```
gh pr checkout 123           # check out PR branch
gh pr diff 123               # view PR diff
gh pr review 123 --approve
gh pr merge 123 --squash --delete-branch
```

PR Options

`--draft`	Create as draft PR
`--reviewer user1,user2`	Request reviewers
`--base main`	Set base branch
`--merge | --squash | --rebase`	Merge strategy
`--auto`	Enable auto-merge when checks pass
`--delete-branch`	Delete branch after merge
`gh pr ready 123`	Mark draft PR as ready

Actions

Workflow Commands

```
gh run list                      # recent runs
gh run view 12345                # run details
gh run view 12345 --log-failed   # failed step logs
gh run watch 12345               # live status
```

Triggering & Managing

```
gh workflow run deploy.yml --ref main
gh workflow list
gh workflow view deploy.yml
gh run rerun 12345 --failed   # rerun failed jobs
```

Actions Options

`-f key=value`	Pass input to workflow_dispatch
`--json`	Output as JSON
`-b branch`	Filter by branch
`gh run download 12345`	Download run artifacts
`gh cache list`	List Actions caches
`gh cache delete KEY`	Delete a cache entry

Releases

Managing Releases

```
gh release create v1.0.0 --generate-notes
gh release create v1.0.0 ./dist/*.tar.gz
gh release list
gh release view v1.0.0
```

Release Options

`--title "Release v1.0"`	Set release title
`--notes "Changelog here"`	Set release notes inline
`--notes-file CHANGELOG.md`	Notes from file
`--generate-notes`	Auto-generate from commits
`--draft`	Create as draft
`--prerelease`	Mark as prerelease
`--latest`	Mark as latest release
`gh release download v1.0.0`	Download release assets
`gh release delete v1.0.0`	Delete a release
`gh release edit v1.0.0`	Edit release metadata

Gists

Gist Commands

```
gh gist create file.py -d "My snippet"
gh gist create file1.js file2.js   # multi-file gist
gh gist list
gh gist view <id>
```

Gist Options

`-d "description"`	Set gist description
`--public`	Create public gist (default: secret)
`--web`	Open gist in browser
`gh gist edit <id>`	Edit gist files
`gh gist clone <id>`	Clone gist locally
`gh gist delete <id>`	Delete a gist

API

Making API Calls

```
gh api repos/owner/repo
gh api repos/owner/repo/issues --method POST \
  -f title="Bug" -f body="Details"
gh api graphql -f query='{ viewer { login } }'
```

Formatting Output

```
gh api repos/owner/repo --jq '.stargazers_count'
gh api repos/owner/repo --template '{{.full_name}}'
gh pr list --json number,title --jq '.[].title'
```

API Options

`--method GET|POST|PUT|DELETE`	HTTP method
`-f key=value`	Set string field
`-F key=@file`	Set field from file
`--jq 'expression'`	Filter JSON with jq syntax
`--template 'tmpl'`	Format with Go template
`--paginate`	Fetch all pages
`-H 'Accept: ...'`	Set custom header

Aliases

Managing Aliases

```
gh alias set co 'pr checkout'
gh alias set bugs 'issue list --label bug'
gh alias set last 'run list -L 1'
gh alias list
```

Advanced Aliases

```
# Alias with shell command
gh alias set --shell pv 'gh pr view --json url --jq .url | pbcopy'

# Delete alias
gh alias delete co
```

Alias Tips

`gh alias set name 'cmd'`	Create simple alias
`--shell`	Alias runs via shell (supports pipes)
`gh alias list`	Show all defined aliases
`gh alias delete name`	Remove an alias

Common Patterns

Daily Workflow

```
gh issue list --assignee @me       # my open issues
gh pr status                       # PRs needing attention
gh pr checks 123                   # CI status for PR
gh run list -b main -L 5           # recent CI runs
```

Searching

```
gh search repos --language rust --stars ">1000"
gh search issues --repo owner/repo "memory leak"
gh search prs --state open --review required
```

GitHub CLI Quick Reference

Useful Flags

`--json field1,field2`	Output specific fields as JSON
`--jq 'expr'`	Filter JSON output
`-L N`	Limit results to N items
`--web`	Open result in browser
`-R owner/repo`	Target a specific repository
`GH_TOKEN=xxx`	Auth via environment variable