Spickzettel: GitHub API (Basics) & CLI

Ziel

GitHub-Funktionen über **API oder Kommandozeile** nutzen – für Automatisierung, Skripte oder systematische Abfragen.

GitHub REST API (v3)

Grundlagen

- Basis-URL: `https://api.github.com`
- Authentifizierung: per **Token** (`Authorization: token <TOKEN>`) oder `gh` CLI

Beispiel: Letzte Commits abfragen

```
curl -H "Accept: application/vnd.github.v3+json" \
     -H "Authorization: token $GITHUB_TOKEN" \
     https://api.github.com/repos/<owner>/<repo>/commits
```

Weitere häufige Endpunkte

- Repos: `/repos/:owner/:repo`
- Issues: `/repos/:owner/:repo/issues`
- Pull Requests: `/repos/:owner/:repo/pulls`
- Actions Runs: `/repos/:owner/:repo/actions/runs`

GitHub CLI (`gh`)

Installation

```
# macOS / Linux (mit Brew)
brew install gh

# Windows (Scoop oder MSI)
scoop install gh
```

Login

```
gh auth login
```

- Interaktive Einrichtung (GitHub.com oder Enterprise, Token oder Browser)

Nützliche `gh` Befehle

Repositories

```
gh repo clone owner/repo
gh repo view --web
```

Issues

```
gh issue list
gh issue view 123
gh issue create --title "Fehler" --body "Beschreibung" --label bug
```

Pull Requests

```
gh pr list
gh pr view 42 --web
gh pr create --base main --head feature-xyz --title "Neues Feature"
```

Releases

```
gh release create v1.2.0 build/*.zip --notes "Neue Version mit Fixes"
```

Best Practices

- Tokens über Umgebungsvariable speichern (`$GITHUB_TOKEN`)
- CLI für häufige Aufgaben in Skripten nutzen
- API für komplexere Automatisierung oder Drittintegration
- `gh`-Befehle mit `--json` für maschinenlesbare Ausgabe kombinieren

---

Die GitHub API & CLI bieten volle Kontrolle über Repositories, Releases, PRs und mehr – ideal für DevOps, CI/CD & Admin-Tasks.