ENTWICKLER CLI, GIT & ARCHITEKTUR-SPICKZETTEL

Open-Source-Werkzeuge, Terminal-Kürzel & System-Referenz (2026)

1. Moderner Git & Versionskontroll-Workflow

VCS-Kern

- `git switch -c <branch>` — Neuen Zweig anlegen und direkt wechseln.
- `git commit --amend --no-edit` — Vergessene Dateien zum letzten Commit hinzufügen.
- `git rebase -i HEAD~3` — Interaktives Rebase zum Zusammenführen (Squash) der Commits.
- `git cherry-pick <hash>` — Einzelnen Commit aus anderem Zweig übernehmen.
- `git stash push -m "notiz"` — Änderungen sichern; mit `git stash pop` anwenden.
- `git restore --staged <datei>` — Datei aus der Staging-Area entfernen.
- `git reset --soft HEAD~1` — Commit rückgängig machen, Änderungen behalten.
- `git log --oneline --graph --all` — Saubere visuelle ASCII-Zweigübersicht anzeigen.

2. GitHub CLI (gh) & Lazygit Tastenkürzel

TUI & CLI

Werkzeug	Befehl / Taste	Funktion und Produktivitätsvorteil
gh	`gh pr create --web`	Browser öffnen, um Pull Request mit Vorlage einzureichen.
gh	`gh pr checkout <nr>`	PR-Zweig lokal abrufen und sofort aktivieren.
gh	`gh run watch`	GitHub Actions CI/CD-Pipelines live verfolgen.
lazygit	`Space` / `c` / `P`	Datei stagen / Commit verfassen / Upstream pushen.
lazygit	`z` / `e` / `[ ]`	Aktion rückgängig machen / Editor öffnen / Bereich wechseln.

3. Hochgeschwindigkeits-Paketmanager

uv • pnpm • fnm

```
# uv (Python - Rust-basiert, 10-100x schneller als pip)
uv venv && source .venv/bin/activate # Virtuelle Umgebung in Millisekunden
uv pip install -r req.txt # Globales Caching für extrem schnelle Installation
uv run script.py # Führt Skripte in isolierter Umgebung aus
```

```
# pnpm (Node.js - Hardlinks & Symlinks sparen 80% Speicher)
pnpm install && pnpm add -D typescript # Inhaltsadressierter Speicher
pnpm dlx degit <repo> <ziel> # Sofortige Ausführung ohne Installation
```

```
# fnm (Fast Node Manager - Rust-basierter Sofortwechsel)
fnm install --lts && fnm use 22 # Keine Startverzögerung in der Shell
```

4. Terminal-Navigation & Readline-Kürzel

Tastenkürzel

Taste	Aktion	Taste	Aktion
`Ctrl + A`	Zum Zeilenanfang springen	`Ctrl + E`	Ans Zeilenende springen
`Ctrl + U`	Vom Cursor zum Anfang löschen	`Ctrl + K`	Vom Cursor zum Ende löschen
`Ctrl + W`	Vorheriges Wort löschen	`Ctrl + Y`	Gelöschten Text einfügen (Yank)
`Ctrl + R`	Befehlsverlauf rückwärts suchen	`Ctrl + L`	Bildschirm leeren (Clear)
`Alt + .`	Letztes Befehlsargument einfügen	`Ctrl + C`	Laufenden Prozess abbrechen

5. HTTP-Statuscodes & REST-Architektur

API-Standards

Code	Bedeutung	Typische Verwendung & Architekturnotiz
200 OK	Erfolgreich	GET/PUT/PATCH erfolgreich mit Antwortkörper abgeschlossen.
201 Created	Ressource Produziert	POST hat neue Ressource bereitgestellt (Header: Location).
204 No Content	Kein Inhalt	Löschung/Änderung erfolgreich; leerer Antwortkörper.
304 Not Mod.	Unverändert	ETag-Cache gültig; kein Datenverkehr übertragen.
400 Bad Req.	Fehlerhafte Anfrage	Syntax ungültig oder Schema-Validierung fehlgeschlagen.
401 Unauth.	Nicht authentifiziert	Fehlende oder ungültige Zugangsdaten (Token).
403 Forbidden	Zugriff verweigert	Identität bekannt, aber keine RBAC-Berechtigung vorhanden.
404 Not Found	Nicht gefunden	Angefordertes URI-Ziel existiert auf dem Server nicht.
409 Conflict	Konflikt	Eindeutiger Schlüssel oder Versionskollision.
422 Unproc.	Nicht verarbeitbar	Gültiges JSON, aber Geschäftsregeln verletzt.
429 Rate Limit	Zu viele Anfragen	Quote überschritten; Header Retry-After beachten.
500 / 502 / 503	Serverfehler	Interner Absturz / Bad Gateway / Server überlastet.

6. Reguläre Ausdrücke (Regex) Spickzettel

Validierung

E-Mail: `^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$`

URL (HTTP/S): `^https?:\/\/[\w.-]+(?:\.[\w\.-]+)+[\w\-\._~:/?#[\]@!\$&'\(\)\*\+,;=.]+$`

IPv4-Adresse: `^(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$`

UUID v4: `^[0-9a-f]{8}-[0-9a-f]{4}-4[0-9a-f]{3}-[89ab][0-9a-f]{3}-[0-9a-f]{12}$`

ISO 8601 Datum: `^\d{4}-(?:0[1-9]|1[0-2])-(?:0[1-9]|[12]\d|3[01])$`

Semantische Version: `^v?(0|[1-9]\d*)\.(0|[1-9]\d*)\.(0|[1-9]\d*)(?:-[0-9A-Za-z.-]+)?$`

7. SQL-Indexierung & Relationale Architektur

RDBMS-Regeln

Leftmost-Prefix-Regel: `INDEX(a, b, c)` beschleunigt nur Abfragen auf `(a)`, `(a, b)` oder `(a, b, c)`. Ein einzelnes `WHERE b = 5` führt zu einem Full Table Scan.

JOIN-Logik: `INNER` liefert die Schnittmenge. `LEFT JOIN` behält alle Zeilen der linken Tabelle. `SELECT *` in Produktionsumgebungen vermeiden.

ACID-Garantien: Atomicity (Alles oder nichts), Consistency (Datenintegrität), Isolation (Transaktionsisolierung), Durability (Persistente Festplattenspeicherung).

Abfrage-Analyse: Langsame Endpunkte stets vorab mit `EXPLAIN ANALYZE` auf teure Seq Scans überprüfen.

8. Moderne CLI-Werkzeuge (Rust & Go Ökosystem)

CLI-Booster

Klassisch	Modern	Leistungs- und Funktionsvorsprung
`cd`	**zoxide (z)**	Lernt Pfadgewohnheiten; Direktaufruf mit `z projekt`.
`cat`	**bat**	Syntaxhervorhebung, Zeilennummern und Git-Integration.
`grep`	**ripgrep (rg)**	Rust-basiert, respektiert .gitignore, extrem schnelle Suche.
`find`	**fd**	Intuitive Syntax mit paralleler Verzeichnisdurchsuchung.
`ls`	**eza**	Symbole, Dateirechte und differenzierter Git-Status.