

# Git Tahák Pro (Bash)

© 2025 Vaňha Jiří (®). Distribuce povolena pro studijní účely.

---

## Základní práce s repositářem

```
git init
git clone <url>
git status
git add <soubor>
git add .
git commit -m "popis"
git log -oneline
git diff
git show <hash>
```

Inicializuje nový lokální repositář.  
Naklonuje vzdálený repositář (např. GitHub).  
Zobrazí aktuální stav pracovního adresáře.  
Přidá konkrétní soubor do stagingu.  
Přidá všechny nové a změněné soubory.  
Uloží změny do historie s komentářem.  
Stručný výpis commitů.  
Zobrazí rozdíly oproti poslednímu commitu.  
Zobrazí detailní informace o commitu.

## Práce s větvemi

```
git branch
git branch <název>
git checkout <název>
git checkout -b <název>
git branch -m <nový>
git merge <větev>
git merge -abort
git branch -d <větev>
git branch -D <větev>
git log master..bug -oneline
git cherry-pick <hash>
```

Vypíše všechny lokální větve.  
Vytvoří novou větev.  
Přepne se do dané větve.  
Vytvoří novou větev a přepne se do ní.  
Přejmenuje aktuální větev.  
Sloučí danou větev do aktuální.  
Zruší probíhající merge (např. při konfliktu).  
Smaže větev (pokud je sloučená).  
Násilně smaže větev (i když není sloučená).  
Zobrazí commity pouze ve větvi bug.  
Přidá konkrétní commit z jiné větve.

## Vzdálený repositář (GitHub, GitLab)

```
git remote -v
git remote add origin <url>
git push -u origin master
git pull
git fetch
git push origin -delete <větev>
git push origin -tags
```

Zobrazí připojené vzdálené repositáře.  
Přidá vzdálený repositář.  
Nahraje větev **master** na server.  
Stáhne a sloučí změny z remote repositáře.  
Stáhne změny bez sloučení.  
Smaže vzdálenou větev.  
Nahraje všechny lokální tagy.

## Práce se změnami a zálohování

```
git restore <soubor>
git restore -staged <soubor>
```

Vrátí soubor do posledního commitu.  
Odebere soubor ze stagingu.

```
git reset -soft HEAD~1  
git reset -hard HEAD~1  
git stash push -m "záloha"  
git stash list  
git stash apply stash@{0}  
git diff master..bug
```

Vrátí poslední commit, ale ponechá změny.

Vrátí stav repozitáře o jeden commit zpět ( destruktivní).

Uloží rozpracované změny bokem.

Zobrazí uložené stash zálohy.

Obnoví konkrétní stash.

Porovná rozdíly mezi dvěma větvemi.

## Reverze, obnova a historie

```
git revert <hash>
```

Vytvoří nový commit, který vrátí změny z daného commitu.

```
git reflog
```

Zobrazí všechny pohyby HEADu (i smazané commity).

```
git log -graph -oneline -all
```

Zobrazí historii commitů s větvemi jako graf.

```
git blame <soubor>
```

Zobrazí, kdo upravil který řádek souboru.

```
git shortlog -sn
```

Statistika commitů podle autora.

```
git checkout <hash>
```

Přepne se do konkrétního commitu (detached HEAD).

## Tagy a verze

```
git tag
```

Zobrazí všechny tagy.

```
git tag <název>
```

Vytvoří lehký tag na aktuální commit.

```
git tag -a v1.0 -m "verze 1.0"
```

Vytvoří anotovaný tag s popisem.

```
git show <tag>
```

Zobrazí commit a info tagu.

```
git push origin <tag>
```

Nahraje tag na server.

```
git push origin -tags
```

Nahraje všechny lokální tagy.

```
git tag -d <tag>
```

Smaže lokální tag.

```
git describe -tags
```

Zobrazí nejbližší tag k aktuálnímu commitu.

## Konfigurace a uživatelské nastavení

```
git config --list
```

Zobrazí aktuální konfiguraci.

```
git config user.name "Jméno"
```

Nastaví jméno pro commity.

```
git config user.email "email@example.com"
```

Nastaví e-mail.

```
git config --global user.name "Jméno"
```

Nastaví jméno globálně pro všechny repozitáře.

```
git config --global core.editor "code -w"
```

Nastaví výchozí editor (např. VS Code).

```
git config --global color.ui auto
```

Zapne barevné výpisy.

```
git config --list --show-origin
```

Ukáže konfiguraci s cestami k souborům.

## Další užitečné příkazy

```
git clean -n
```

Ukáže, které nesledované soubory by byly smazány.

```
git clean -f
```

Smaže nesledované (untracked) soubory.

```
git mv <starý> <nový>
```

Přejmenuje nebo přesune soubor.

```
git rm <soubor>
```

Smaže soubor ze sledování i z disku.

```
git grep <text>
```

Vyhledá text v historii commitů nebo souborech.

```
git archive -o verze1.zip HEAD
```

Vytvoří ZIP archiv aktuálního stavu.

```
git bisect start
```

Zahájí binární hledání chyby mezi commity.

---

*Poznámka:* Tento tahák obsahuje nejpoužívanější i rozšířené příkazy Gitu vhodné pro vývojáře, správce kódu a pokročilé uživatele. Kompiluj pomocí `pdflatex -shell-escape Git_tahak_pro.tex` nebo v Overleafu s povoleným **shell escape**.