GIT

© Stefan Edlich

Basics

```
git init   => Frisches Repo erstellen
git clone  => Repo kopieren (Verzeichnis prüfen!)

git pull   => So startet man den Tag! (ggf. mergen!)
git add [.|file|pattern] => nur staging auf Rampe

git status -s => globale Änderungen (short)
git log --pretty=oneline => Historie in Github
git diff    => ist passiert? --cached besser --HEAD
gitk        => Tool für visuelle History & Diff

git commit [.|file|fpattern] => -a für all (emoji!)
git push    => wie upload

git rm/mv file => Aus dem Index raus!
    Danach (!) ggf. echtes remove. Hilfreich dann:
    git commit -m "removed file CNAME weil Mist"
```

Reverting (committing an undo of previous)

```
git reset -- hard (no undo, throw away all changes!)
git revert -ref
git commit -a --amend
git checkout ref file
```

Links

Github, Reference Manual, use **emoji's**! Apress Buch (D), Kurzreferenz!, **Git-Magic!**, erfolgreich Branchen, CheatSheet, InterAct, CS1, CS2, CS3, Semantic Versioning! Internal-Cheat-Sheet, Einfach

Advanced

```
git branch [-av] => list all
git branch testlib => I want a new branch!
git checkout testlib => now commit + push here
git push origin testlib => Push branch to source=origin
git checkout master => all files dissapear! Used to switch!

Jetzt den neuen Branch holen: (visuell löschen)
clone + git checkout -b testlib origin/testlib oder
git clone git@github.com:name/app.git -b master

git tag => lists all tags = revisions!
git tag [-a] v0.1 -m "can play one move"
git push --tags => sonst gehen die nicht mit :-(
```

Global Configurations

```
git config --global color.ui always
git config credential.helper cache --timeout=2592000 //1Mon
git config credential.helper store //plaintext :-(
git-credential-[store|wincred].exe
$ git config --global core.editor vim
```

- [ ] GitHub Task

Tipps & Tricks

- consider working in a **branch** and merge later if unsure
- make one **commit per file** with a short message (this reads better in GitHub)
- Refer to lines by **selecting** the **line** and get the **link** to it
- Regelmäßig **versionieren**! In Code, in Project File, in Github and then savecopy!

Branch mit Comments:

```
git diff --name-status master dev
> gitk branch1 branch2
```

First click on the tip of branch1. Now right-click on the tip of branch2 and select Diff this->selected.

In GitHub in Branch gehen und dann commits anschauen = diff

```
git checkout better_branch
git merge --strategy=ours --no-commit master # keep the content of this branch, but record a merge
git commit # add information to the template merge message
git checkout master
git merge better_branch # fast-forward master up to the merge
```