Git

Versionskontrolle und Kollaboration

Ivo Blöchliger

git

- Linus Torvalds
  - Brauchte ein gutes Versionskontrollsystem für die Linux-Entwicklung (ca. 5000 Änderungen monatlich)
  - Schrieb selbst git
  - git ist zum de-facto Standard in der Software-Entwicklung geworden

Versionskontrolle mit Bordmitteln

- Immer mal wieder eine zip-Datei vom Projekt machen.
  - Wiederherstellung ist einfach
  - Aber: wo, wann, welche Änderung?
  - Zusammenarbeit?
  - Unterhalt Produktiv- und Entwicklungsversion?

git

- Verwaltet diese «zip-Dateien» im Ordner .git
  - Aber nur die Änderungen ;-)
- git ist lokal!
  - Kann aber mit einem Server synchronisiert werden
  - z.B. GitHub
- Protokolliert alle Änderungen
  - Wer, was, wann, mit Kommentar

Kommandozeile und Setup

- Kommandozeile nützlich und nötig
- GUI praktisch, versteckt aber was läuft

- Setup Name und e-mail
- `git config --global user.name "Rockstar Programmer"`
- `git config --global user.email rockstar@programmer.com`

Initialisierung

- Neues repo
  - `git init` # Im zu versionierenden Verzeichnis
- Repo klonen
  - `git clone` *URL* # Legt neuen Ordner an

Workflow

- Dateien ändern, hinzufügen
- **Staging** (auswählen, was **commit***ed* werden soll)
  - `git add` *DATEI[EN]*
- **Commit***en*
  - `git commit -m ‘sinnvolle Beschreibung der Änderung’`
- Pull / push (mit Server synchronisieren)
  - `git pull` # Importiert Änderung vom Server
  - `git push` # Exportiert Änderung zum Server

Branches

- Entwicklungszweig erstellen
  - `git checkout -b superapp`
- Entwicklungszweig wechseln
  - `git checkout main`
- Entwicklungszweige zusammenführen
  - `git checkout main`
  - `git merge superapp`
  - `git branch -d superapp` # Branch löschen

Empfehlung

- Kleine und häufige commits
  - Z.B. auch Originalbilder vor der Bearbeitung einchecken
- «Gute» commit messages
  - Sehr nützlich, wenn was schief gegangen ist.

Heiklere git Befehle

- Folgende Befehle ändern die history
  - ammend
  - rebase
- Führt zu Problemen, wenn der branch schon publiziert wurde.
- Rein lokal kein Problem

Merge-Konflikte

- Repository muss «clean» sein
  - Keine Änderungen, nichts gestaged.
- Datei wurde in der gleichen Zeile in beiden Branches bearbeitet.
  - Problematische Datei(en) bearbeiten
  - Committen
  - Oder abbrechen (`git merge –abort`)
  - https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts