GFS Informatik | Versionskontrolle

Liam Wachter, TGI 13/2

github.com/sevenmaster/InformatikGFS, dieses Handout in digital und bunt

Versionskontrollsysteme (auch Versionsverwaltungssysteme) sind Systeme zur Erfassung und Verwaltung von Änderungen einzelner Dateien oder ganzer Projekte. Diese ermöglichen es nicht nur jeder Zeit jede erfasste Version wiederherzustellen zu können, sondern auch mehrere Versionen parallel zu entwickeln und zusammenführen zu können. Außerdem können durch Versionskontrollsysteme mehrere Entwickler gleichzeitig an einem Projekt arbeiten.

Grundlegende Konzepte

- **copy - modify - merge**
  git fetch; git merge; git push

  Damit mehrere Entwickler gleichzeitig an der gleichen Datei arbeiten können, kopiert sich jeder Entwickler die aktuelle Version und macht Änderungen daran. Die Änderungen werden später zusammengeführt (Merge). Das geschieht mit einem Merge-Algorithmus. Kann dieser die Änderungen nicht zusammenführen, da sie im Widerspruch zueinander stehen, spricht man von einem Mergekonflikt, der von einem Mensch aufgelöst werden muss.

- **commit**
  git commit

  Mit einem Commit speichert man den aktuellen Zustand von Dateien oder Verzeichnissen. Ein Commit enthält nicht nur eine Referenz auf den eingepflegten Zustand des Projekts (Roottree s.u), sondern enthält auch Autor, eine Beschreibung der vorgenommenen Änderungen, sowie den Namen des vorangegangenen Commit. Commits sind, wie alle Git-Objekte, mit ihrem SHA1-Prüfwert eindeutig benannt.

- **branching**
  git branch; git checkout

  Um parallel an mehreren Versionen arbeiten zu können, gibt es sogenannte Branches (engl. für Zweige). Branches können unterschiedlich genutzt werden, zum Beispiel um eine stabil laufende Version eines Programms zur Verfügung zu haben, während man in einem anderen Branch an neuen Funktionen arbeitet.

Objekte

Git speichert alles in Objekten ($\neq$ Objekte in OOP). Es gibt verschiedene Arten von Objekten.

- **Blob**
  Enthält den Inhalt einzelner Dateien. Blob ist kurz für Binary Large Object.
- **Tree**
  Bildet Ordnerstruktur ab. Kann weitere Trees oder Blobs enthalten
- **Commit**
  Stellt wie oben beschrieben eine Version dar. Die Referenz auf die Änderungen ist ein Tree.
- **Tag**
  Tags zeigen auf einen Commit. Mit Tags können wichtige Versionen markiert und benannt werden.
  (Außerdem zeigen spezielle Tags auf den neuesten Commit eines Brach. So werden Branches intern realisert. Der Tag HEAD markiert die Version auf der momentan gearbeitet wird.)