Git Schulung: Theorie

2015-10-19

Andreas Fett & Till Salzer
iSAX

Ziel

Ihr kommt beim Andi ins Projekt und sollt dort produktiv arbeiten und nix kaputt machen.

Was bedeutet das?

- Ihr arbeitet per **ssh** auf der *Kommandozeile*.
- Ihr arbeitet im **vim**
- Ihr checked Eure Arbeit in viele geteilte git **Repos** ein.

Geteilte Repos sind wie SVN Repos: Der Admin hat das Backup.

Ziel (weiter...)

Ihr kommt beim Andi ins Projekt und sollt dort produktiv arbeiten und nix kaputt machen.

Was bedeutet das **nicht**?

- Wir erklären euch die IDE Integration (eclipse, netbeans, github, emacs, vim)
- Wir erklären euch toolgebunden Workflows (z.B. *gitflow*)
- Wir erklären euch Repo-Administration

Das wäre zu viel. Ihr lernt die Grundlagen kennen und versteht, wo euch die Tools helfen.

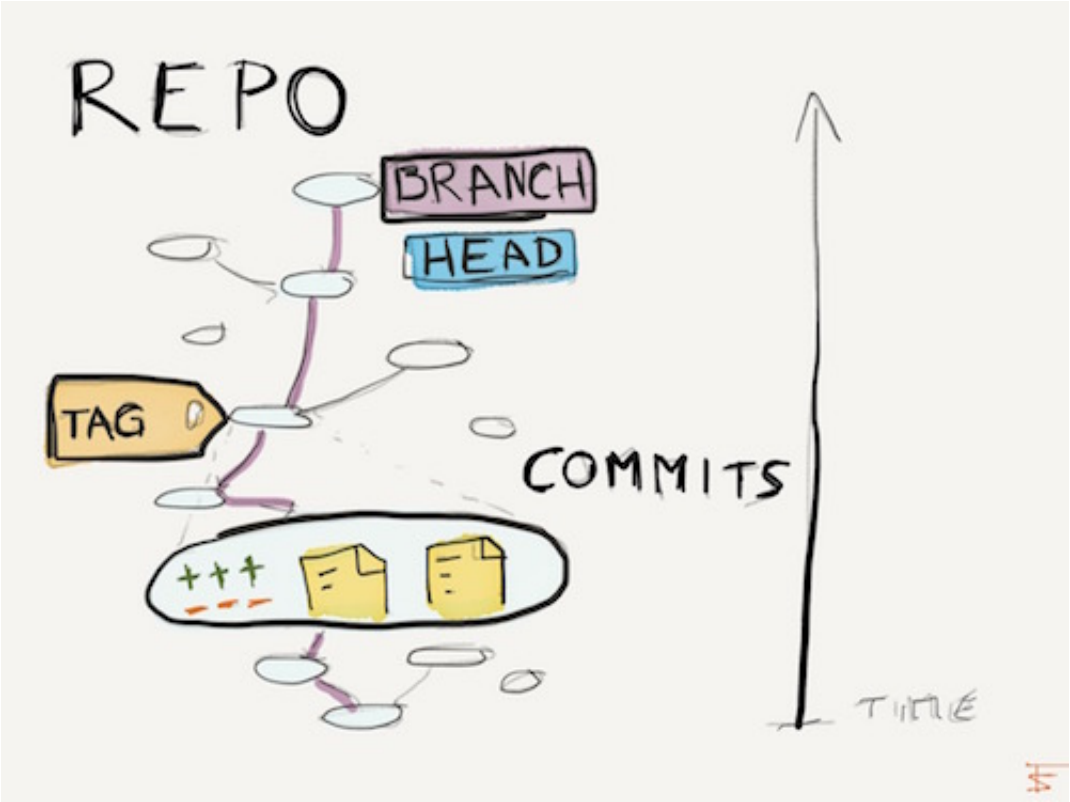
Ich komme gar nicht zum Andi in Projekt!

Ihr wollt das trotzdem wissen, weil:

- Git erleichtert Zusammenarbeit.
- Git erklärt Geschichte (Fehlersuche, Dokumentation, wer hat wann was warum und wie gemacht?)
- Git erleichtert Featureentwicklung (Isolation: ich mach was, was ansonsten den Code kaputt macht).
- Git *kann* eine Art Backup sein -- aber kein Mülleimer!
- Git entlastet Anwalt und Debugger (Was haben wir dem Kunden wann geliefert, was läuft da?)

Git Objekte: Logische Objekte

- Repository: Container für Git-Objekte
- Commit: Änderung an den Nutzdaten + Metadaten (Autor, Zeit, Message, ...)
- Tag: Name für einen spezifischen Commit
- Branch: Reihe lineare Folge von Commits (svn trunk -> git master)



Git Objekte: Filesystem Objekte

- Repository (Verzeichnis *.git*)
- working directory: Verzeichnisbaum, in dem *.git* liegt
- *.gitignore* : Alles, was nicht ins Git soll

Beachte: Das working directory enthält immer nur eine Revision (Commit/Tag/Branch) und kann frei editiert werden.

Was gehört ins Repo?

checke ein:

- Alles was andere brauchen um mit den Projekt arbeiten zu können. (Source, Makefiles, Doku, ...)

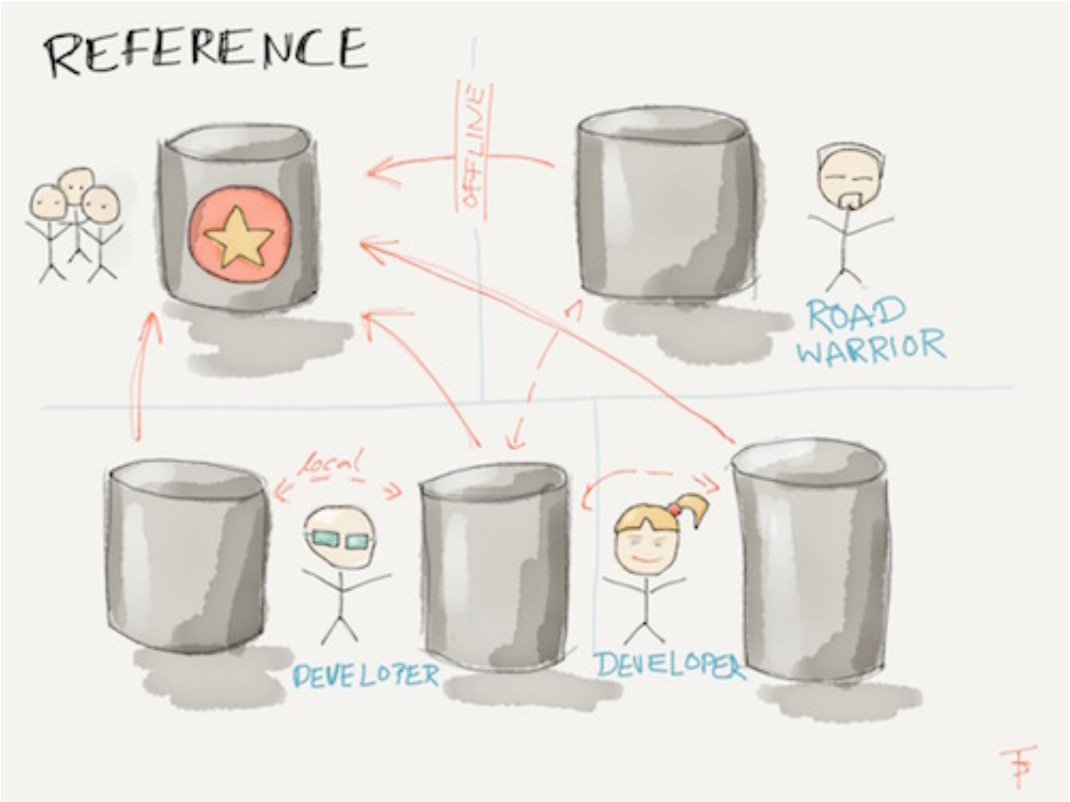
checke **nicht** ein was:

- niemand sehen soll (Password, SSH-Key, vertrauliche Daten, ...)
- niemand sehen muss (abgeleitete Daten wie Compilate, temporäre Dateien, persönliche Konfiguartionen, ...)
- niemand sehen will (kaputte Unit tests, nicht kompilierender Code ...)

Beachte: Es gibt keinen Zugriffsschutz. *(D'oh!)*

Dezentral verteilte Versionskontrollsysteme (DRCS)

- Jeder arbeitet immer lokal mit einem vollständigen Repository
- Alle Repos sind gleichwertig und untereinander synchronisierbar
- Keine (zentralen) Versionsnummern sondern Indizierung über den Inhalt
- Alle SCM Operationen offline verfügbar (Commit, Diff, Tag, Branch)



Und jetzt...

Weiter geht's mit der Praxis! (git-schulung-2-praxis.slide)

Thank you

Andreas Fett & Till Salzer
iSAX
andreas.fett@isax.com, till.salzer@isax.com (mailto:andreas.fett@isax.com,%20till.salzer@isax.com)
http://isax.com (http://isax.com)