Git

Präsentiert von Christian Ehringfeld

Inhaltsverzeichnis

Versionsverwaltung

- Protokollierung von Änderungen
- Versionierung
- Benutzerkennung und Zeitstempel
- Wiederherstellung von älteren Versionen
- Archivierung von Projektständen

Arten

- Lokale Versionsverwaltung
- Zentrale Versionverwaltung
- Verteilte Versionsverwaltung
- Vergleich

Lokale Versionsverwaltung

- Lokale, persönliche Speicherung von „Versionen“
- Beispiel:
  - Auto-Save Funktion
  - (Strg+Z)

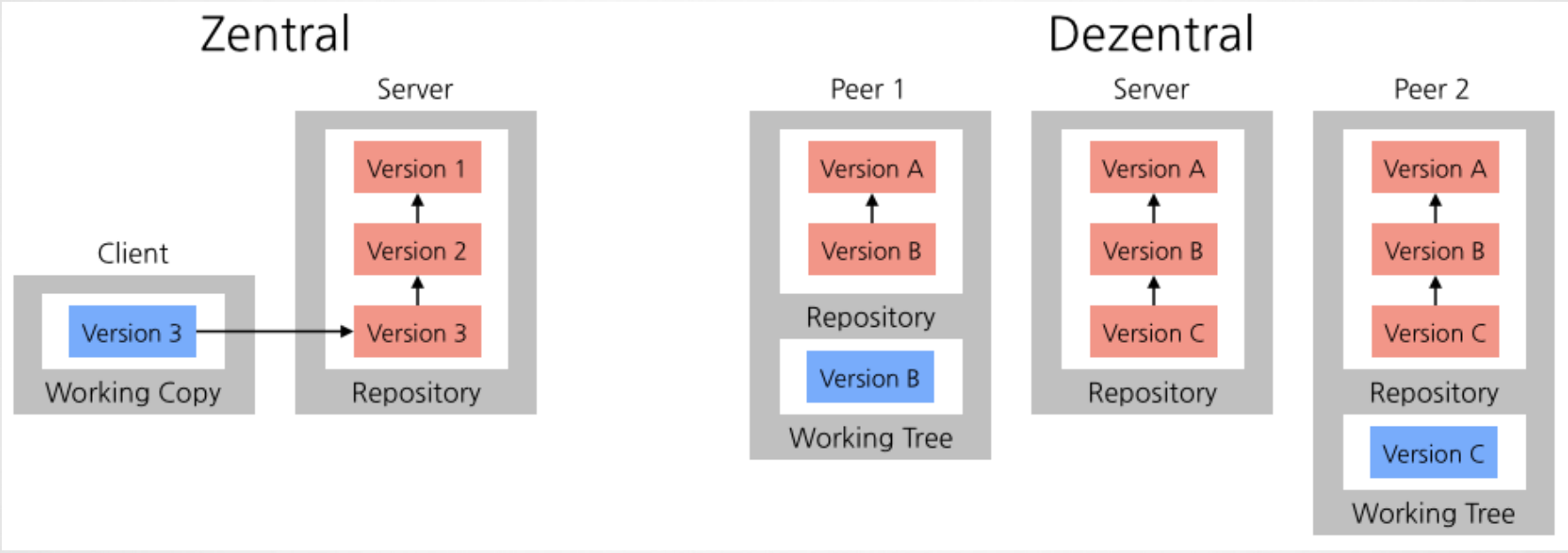
Zentrale Versionsverwaltung

- Speicherung auf EINEM zentralem Server
- Meist mit Benutzer-/Rechteverwaltung
- Beispiele:
  - SVN
  - CVS
  - Team Foundation Server

Verteilte Versionsverwaltung

- Versionen zunächst lokal gespeichert
- Enthält üblicherweise alle Versionen
- Kann mit anderen geteilt/verteilt werden
- Beispiele:
  - Git
  - Bazaar
  - Bitkeeper
  - Mercurial

Vergleich


Zentral
Server
Version 1
Version 2
Version 3
Repository
Client
Version 3
Working Copy
Dezentral
Peer 1
Version A
Version B
Repository
Version B
Working Tree
Server
Version A
Version B
Version C
Repository
Peer 2
Version A
Version B
Version C
Repository
Version C
Working Tree

Vergleich

Zentrale Versionsverwaltung	Verteilte Versionsverwaltung
Single-Point-of-Failure	Offline-Versionierung Kein Single-Point-of-Failure
übersichtlich	Änderungen können ohne Server geteilt werden
Kleine Versionsnummern	Zu jedem Zeitpunkt konfliktfreie Version vorhanden
Wenig Redundanz	Mehr Redundanzen möglich
	schnell
	Branches komfortabler

Begriffe

- Repository
- Branch
- Tag
- Commit
- Working Copy
- Diff
- Conflict
- HEAD

Git

- Verteilte Versionsverwaltung
- Ursprünglich für den Linux-Kernel
- Breite Plattformunterstützung
- Wer nutzt Git?
  - Android, Debian, Gnome, jQuery, LibreOffice, Linux, PHP, Samba, Qt, TYPO3, VLC, …
  - Siehe github

Git – Die wichtigsten Befehle

- *git add DATEI ORDNER*
- *git commit [-m “fixes #42“]*
- *git init [name]*
- *git clone http://YOURDOMAIN/REPO*
- *git pull [origin] [branch]*
- *git push [origin] [branch]*
- *git merge [branch/commit]*
- *git checkout [-b] [branch/commit]*

Git

Demo

Git Konfiguration

git config --global user.name "Max Muster"

git config --global user.email "max.muster@yourdomain.de"

git config --global color.ui auto

- *Speicherung der Anmeldedaten:*

git config --global credential.helper store

.gitignore

- (Temporäre) Dateien und Pfade exkludieren
- *vim .gitignore*

  **.log*

  build/

  *temp-**

  dist/
- *git add .gitignore & git commit -m “added .gitignore“*

Redmine

- Bei vorhandenem Git Repository:

  git remote add origin http://YOURDOMAIN/REPO
- *Kontrolle mit git remote -v*
- *git pull origin*
- *git push origin #branch/tag angeben*

SVN 2 Git

- *git svn clone http://YOURDOMAIN/svn/PROJEKT*
- Für „advanced“ Migration siehe Quellen

Aufgabe

- Installiere Git
- Importiere das Projekt von Git (http://YOURDOMAIN/git-test)
- Erstelle eine Datei mit willkürlichem Inhalt
- Stell die Datei deinen Kollegen über die Versionsverwaltung zur Verfügung
- Schau über die Versionsverwaltung, was die anderen hochgeladen haben

Git

- Fragen?

Hilfen

- https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf
- http://www.cheat-sheets.org/saved-copy/git-cheat-sheet.pdf
- https://rogerdudler.github.io/git-guide/index.de.html
- http://justinhileman.info/article/git-pretty/
- http://git-scm.com/book/de/v1

Git als „Spiel“:

- http://pcottle.github.io/learnGitBranching/
- https://try.github.io/levels/1/challenges/1

Quellen

- https://www.mittwald.de/blog/wp-content/uploads/2013/05/Bildschirmfoto-2013-05-13-um-09.53.36.png
- http://www.itworld.com/article/2885956/migrating-from-svn-to-git-version-control-part-2.html
- Präsentation „Versionsverwaltung“ von Daniel Failing