

Subversion

Sourcecodeverwaltung

zentrale Verwaltung des Sourcecodes unablässig bei Gruppenarbeit

offensichtlich schlecht:

- Code per Email herumschicken: Aktualität, Vollständigkeit
- Code in Verzeichnis mit Zugriff für alle: Überschreiben von Änderungen anderer

stattdessen professionelle Verwaltung des Sourcecodes mit Versionskontrolle: SVN

SVN – Server/Client-Modell

SVN-Server hält Daten: *Repository*

SVN-Clients können sich lokale Kopie davon machen, editieren und ins Repository einspielen

Transfer erfolgt über HTTP

hier: Repositories werden auf Server des Lehrstuhls TCS bereitgestellt; Zugriff über

`http://svn.tcs.ifi.lmu.de/⟨Name des Repositories⟩`

Repository

ist mehr als nur Ansammlung von Dateien, sondern auch Meta-Information

- Geschichte der Daten wird protokolliert: Revisionen 1,2,...
- Wer hat welche Änderungen gemacht?
- Nachrichten zu Änderungen

Revision = aktueller Zustand der Dateien zu gewissem Zeitpunkt

Operationen auf/mit Repository:

- checkout, lokale Kopie mit letzter Revision erzeugen
- update, lokale Kopie auf aktuellen Stand bringen
- commit, Änderungen an lokaler Kopie ins Repository einspielen

Checkout

lokale Kopie wird so erzeugt

```
mlange@pozzuoli:~/tmp> svn checkout --username lange http://svn.tcs.ifi.lmu.de/  
Authentication realm: <http://svn.tcs.ifi.lmu.de:80> Repository SEP Project  
Password for 'lange':  
A    SEP/MeineKlasse.java  
...  
Checked out revision 117.
```

erzeugt lokale Kopie, die editiert werden kann

Usernamen und Passwörter werden von uns vergeben und per Email verschickt

Ausgabe von `svn checkout` ist Liste von Dateien mit Flag für Aktionen

Update

lokale Kopie auf aktuellen Stand bringen

```
mlange@pozzuoli:~/tmp/SEP> svn update
A    MeineKlasse.java
...
Updated to revision 118.
```

lokale Änderungen gehen dabei nicht verloren

Statusmeldungen:

A	Datei.java	Datei wurde neu angelegt
D	Datei.java	Datei wurde gelöscht
U	Datei.java	Datei wurde auf aktuellen Stand gebracht
M	Datei.java	... und dabei wurden Änderungen im Repository mit lokalen Änderungen vereint
C	Datei.java	... und dies führte zu einem Konflikt

Commit

lokale Änderungen ins Repository einspielen, um sie den anderen zugänglich zu machen

```
mlange@pozzuoli:~/tmp/SEP> svn commit
```

öffnet Editor, um Nachricht einzutragen; diese sollte Änderungen für die Teammitglieder beschreiben

Commit nicht immer möglich, z.B. falls sich Repository mittlerweile ebenfalls geändert hat; dann zuerst nochmal Update

Merging und Konflikte

betrachte folgenden Ablauf, Datei.java ist im Repository

- ① User A ändert Datei.java bei sich
- ② User B ändert Datei.java bei sich
- ③ User A führt `svn commit` aus

User B kann dann kein Commit durchführen, aber Update von User B kann zu Problemen führen

SVN versucht, Änderungen im Repository und lokale zu vereinen; mehr oder weniger zeilenweise

Vereinigung nicht möglich: Konflikt; wird in Datei markiert

```
<<<<<<< .mine
lokale Änderung von User B
=====
Änderung im Repository
>>>>>>> .r118
```

Lösung: Änderungen per Hand vereinen, dann

```
mlange@pozzuoli:~/tmp/SEP> svn resolved Datei.java
```

Sonstiges

Update kann Datei aus bestimmter Revision wiederherstellen
(Option `-r`); keine Backups in lokaler Kopie nötig

`svn move` verschiebt Dateien lokal und bei nächstem Commit auch
im Repository

`svn delete` genauso für Löschen

Verhaltensregeln

was SVN nicht erzwingt, wir aber trotzdem verlangen

- aussagekräftige Nachrichten bei Commit angeben
- keine Binärdateien (z.B. `.class`) im Repository (Ausnahme: unveränderliche und nicht erzeugbare wie z.B. Bilder, Sounds)
- keine Backup-Dateien im Repository
- nur syntaktisch korrekte Java-Dateien einchecken
- regelmäßige Updates durchführen

Hilfe

```
mlange@pozzuoli:~/tmp/SEP> svn help
```

```
mlange@pozzuoli:~/tmp/SEP> svn help <Kommando>
```

```
http://svnbook.red-bean.com/
```