Subversion 1.5 Quickreferenz

Befehl	Bedeutung
Grundlegende Befehle	
`svn` **`help`**	Liefert Hilfe zu allen Befehlen (z.B. `svn help commit`)
Repository-Zugriffs-URLs	• **file:///** - Direkter Repository-Zugriff (auf lokaler Festplatte) • **http://** - Zugriff über das WebDAV-Protokoll auf Apache-Server, die Subversion unterstützen • **https://** - Wie http:// mit SSL-Verschlüsselung • **svn://** - Zutriff über eigenes Protokoll auf einen svnserve-Server • **svn+ssh://** - Wie svn:// jedoch über SSH getunnelt Eine bestimmte Revision kann mit @ adressiert werden z.B. `svn checkout http://svn.irgendwas.de/projekt1/trunk@212`
Revision Schlüsselwörter	Für die Parameter --revision (oder -r), welche in verschiedenen Kommandos verwendet wird, können auch folgende Schlüsselwörter verwendet werden: • **HEAD** - die letzte (jüngste) Revision • **BASE** - Revision in dem aktuellen Arbeitskopie vor lokalen Modifizierungen • **COMITTED** - die aktuellste Revision • **PREV** - Revision vor der aktuellen Revision Anstatt Revisionsnummern und der Schlüsselwörter können auch ISO-8601 Datumangaben angegeben werden (z.B. `-r {2006-02-17}` oder `{2006-02-17 15:30"}`). Beispiel für eine Zeitraumangabe: `svn log -r {2006-11-20}:{2006-11-29}` Es kann bei einigen Kommandos auch ein Revisionsbereich angegeben werde. Dies wird durch einen : gekennzeichnet z.B. `svn log --revision 10:21`
Arbeitskopie anlegen und aktualisieren	
`svn` **`checkout`** *`quelle ziel`*	Erzeugt eine lokale, versionsverwaltete Arbeitskopie vom Repository für das aktuelle Verzeichnis. *quelle* eine gültige Repository-URL (auch Unterverzeichnisse im Repository möglich *ziel* ist optional (es wird sonst das aktuelle Verzeichnis als Zielverzeichnis gewählt. Beispiel: `svn checkout http://svn.collab.net/repos/svn/trunk/lib` **`--revision (oder -r)`** checkt eine explizit angegebene Revision aus.
`svn` **`commit`** *`dateiliste`* **`-m`** *`"aenderungstext"`*	Übermittelt Veränderungen in das Repository und erzeugt eine neue Repository-Revision. **`-m`** ist ein freier Text mit Informationen was an der Arbeitskopie verändert wurde. **`--file (oder -F)`** eine Text-Datei, welche den Informationstext mit den Änderungen enthält (wie -m) **`--revision (oder -r)`** aktualisiert eine Arbeitskopie auf eine explizit angegebene Revision. **`--no-unlock`** Sperren (Locks) die aktuell gehalten werden, werden nicht automatisch freigegeben. **`--keep-changelists`** Commited alle Dateien einer bestimmten Changelist
`svn` **`update`**	Aktualisiert die Arbeitskopie und überträgt Veränderungen aus dem Repository in die eigene Arbeitskopie (wird dieser Befehl von einem Unterverzeichnis aus aufgerufen, so wird nur das Verzeichniss und die Unterverzeichnisse aktualisiert). Treten Konflikte auf, die nicht automatisch gelöst werden können, so sind folgende Eingaben möglich: • **(p)ostpose** - Datei in Konfliktzustand lassen und später manuell Konflicktauflösung durchführen • **(d)iff** - Unterschiede anzeigen • **(e)dit** - Konfliktdatei in bevorzugten Editor (in SVN Config setzen) öffnen • **(r)esolved** - nach Bearbeiten (e) aktuellen Inhalt übernehmen • **(m)ine-(f)ull** - lokale Datei übernehmen • **(t)heirs-(f)ull** - Datei vom Server übernehmen • **(l)aunch** - externes Programm zur Konfliktauflösung verwenden • **(h)elp** - diese Liste **`--non-interactive`** keine interaktive Konfliktauflösung (muss manuell mit svn resolve im Anschluss dann gemacht werden). Es werden die Dateien `filename.mine, filename.rOLDREV und filename.rNEWREV` angelegt.
`svn` **`resolve`** `--accept` *`option dateiname`*	Als *`option`* sind folgende Werte möglich: • **base** - Datei vor eigenen Änderungen • **mine-full** - die eigenen Änderungen übernehmen • **theirs-full** - die Version im Repository übernehmen • **working** - angebene Datei als neue Datei übernehmen Die temporären Dateien (filename.mine etc.) werden dann automatisch gelöscht
`svn` **`import`** *`quelle ziel`* **`-m`** *`"ein freier text"`*	Importiert neue Dateien und Verzeichnisse in das Repository *quelle* ist der Quellpfad *ziel* ist die Repository-URL Beispiel: `svn import mytree file:///var/svn/newrepos/some/project -m "Erstimport"`
`svn` **`add`** *`datei`*	Markiert eine unversionierte Datei oder ein Verzeichnis für eine Aufnahme in das Repository
`svn` **`delete`** *`datei`*	Löscht eine Datei oder Verzeichnis
`svn` **`copy`** *`quelle ziel`*	Kopiert eine Datei oder ein Verzeichnis
`svn` **`move`** *`quelle ziel`*	Verschiebt eine Datei oder ein Verzeichnis
`svn` **`mkdir`** *`neues-verzeichnis`*	Erstellt ein neues Verzeichnis
`svn` **`revert`** *`datei`*	Macht alle Änderungen (Inhalt und Eigenschaften/Properties), die seit dem letzten Commit gemacht wurden wieder rückgängig. Es kann auch ein Verzeichnis angegeben werden. Auch ein `svn delete` kann so wieder rückgängig gemacht werden.
`svn` **`merge`** *`von nach ziel`*	Ermittelt alle Änderungen die zwischen *von* und *nach* durchgeführt wurden und wendet diese auf *ziel* an. Kann verwendet werden um Änderunen im Repository wieder Rückgängig zu machen: `svn merge -c 303 http://svn.example.-com/repos/calc/trunk` oder `svn merge -r 303:302 http://svn.example.-com/repos/calc/trunk` oder `svn merge http://svn.example.com/repos/branch1@150 http://svn.example.com/repos/branch2@212 my-working-copy`
`svn` **`copy`** *`quell-url@revision ziel`*	Stellt ein Ziel wieder her. `svn copy http://svn.example.com/repos/calc/trunk/real.c@807 ./real.c`

Informationen auslesen

`svn` **`status`**	Gibt alle Veränderungen seit dem letzten commit aus. Die Buchstaben vor den Dateien bedeuten: • **?** - Datei ist nicht versionskontrolliert • **A** - Datei ist zum Hinzufügen vorgemerkt • **C** - Datei hat Konflikte durch einen Update • **D** - Datei ist zum löschen vorgemerkt • **M** - Der Inhalt von der Datei hat lokale Änderungen • **I** - Dateien die ignoriert werden • **K** - Datei ist gesperrt (siehe Locks) Optional kann ein Ziel angegeben werden (dann werden nur dessen Veränderung(en) angezeigt). **`--verbose`** (oder `-v`) = es jede Datei gelistet (egal ob Änderungen da sind oder nicht) **`--show-updates`** (oder `-u`) = es wird eine Verbindung mit dem Repository hergestellt und alle Dateien, die nicht mehr aktuell sind mit einem * markiert, alle Dateien die gesperrt sind werden mit einem O markiert. **`--no-ignore`** = es werden auch Dateien angezeigt, die in der Verzeichnis Eigenschaft (svn:ignore) auf ignoriert gesetzt sind.
`svn` **`diff`**	Zeigt alle Änderungen die innerhalb von Dateien durchgeführt wurden im unified-diff-Format. Optional kann auch eine Datei oder ein Verzeichnis angegeben werden. **`--revision (oder -r)`** zeigt Unterschiede zwischen zwei gegebene Revisionen an (z.B. `-r 22`) oder Revisionsinterval (z.B. `-r 5:19`) **`--change (oder -c)`** zeigt Unteschiede zwischen aktueller Revision und einer gegebenen Revision an (z.B. `-c 5`)
`svn` **`log`**	Zeigt alle Änderungen die an dem aktuellen Verzeichnis oder einem/einer übergebenen Verzeichnis/Datei durchgeführt wurden. **`--verbose (oder -v)`** zeigt eine Liste geändertenr Dateien und Verzeichnisse **`--revision (oder -r)`** zeigt eine gegebene Revision an (z.B. `-r 22`) oder Revisionsinterval (z.B. `-r 5:19`) **`--quiet (oder -q)`** unterdrückt die Ausgabe von Protokollausgaben
`svn` **`cat -r`** *`revisionsnummer datei`*	Zeigt den Inhalt einer Datei für einen bestimmten Revisionsstand an
`svn` **`list`** *`repository-url`*	Zeigt den Inhalt eines Repositories an, ohne ihn auschecken zu müssen **`--verbose (oder -v)`** zeigt eine detailierte Auflistung
`svn` **`export`** *`repository-url`*	Lädt den Inhalt eines Repositories ohne .svn Verzeichnisse herunter. **`--revision (oder -r)`** checkt eine explizit angegebene Revision aus

Eigenschaften (Properties)

Grundlagen	Eigenschaften/Properties sind Metadaten, die an Dateien und Verzeichnisse, sowie dem Repository angefügt werden können. Diese werden (mit Ausnahme der Repository-Eigenschaften) ebenfalls versioniert verwaltet. Eigenschaften dürfen nicht mit *svn:* beginnen. Es gibt automatisch von Subversion gesetzte Eigenschaften: • **svn:executable**: markiert ob eine Datei automatisch beim checkout im Dateisystem auf ausführbar gesetzt wird (nur falls Dateisystem das unterstützt z.B. ext3) • **svn:mime-type**: MIME type, legt fest ob Subversion den Inhalt als Text oder Binary behandelt • **svn:eol-style**: End-Of-Line Zeichen der Datei (DOS oder UNIX Style) • **svn:ignore**: Liste von File Patterns in Verzeichnissen, die von SVN ignoriert werden (komplett) • **svn:needs-lock**: Setzt eine Datei automatisch auf Read-Only (Dateisystemattribut), so lange man kein Lock besitzt • **svn:mergeinfo**: Enthält Informationen welche Revisionen mit dieser Datei bereits gemerged wurden (nicht verändern)
`svn` **`propset`** *`name value ziel`*	Setzt oder fügt eine neue, versionierte Eigenschaft für das Verzeichnis, die Datei oder mehrere Objekte (Wildcards erlaubt) *ziel* ein.
`svn` **`propedit`** *`name value ziel`*	Setzt oder fügt eine neue, versionierte Eigenschaft für das Verzeichnis, die Datei oder mehrere Objekte (Wildcards erlaubt) *ziel* ein, greift hierfür aber auf einen externen Editor zurück (Konfiguration).
`svn` **`proplist`** *`ziel`*	Listet alle Eigenschaften von *ziel* auf. **--verbose (oder -v)**: es werden auch die Werte, nicht nur die Namen angezeigt
`svn` **`propdel`** *`name ziel`*	Löscht die Eigenschaft *name*
`svn` **`propset`** *`name wert`* `--revprop`	Setzt eine Repository-Eigenschaft. Achtung: diese sind nicht versioniert und gelten für das gesamte Repository.

Sperren (Locks)

`svn` **`lock`** *`ziel`* **`-m`** `"message"`	Setzt eine Sperre auf ein Objekt. Niemand anders darf dieses commiten. **Achtung: alle Locks werden nach einem Commit automatisch wieder freigegeben, sofern nicht --no-unlock beim Commit gesetzt ist**
`svn` **`unlock`** *`ziel`*	Entfernt eine Sperre wieder.
`svn` **`unlock`** `--force` *`ziel-repository-url`*	Entfernt eine Sperre wieder ohne das der Benutzer selbst die Sperre hält (falls ein Benutzer vergessen hat ein Lock freizugeben).

Changelists

`svn` **`changelist`** *`name dateiliste`*	Fügt Dateien zu einer Changelist hinzu. Changelists können dann gezielt commited werden oder deren Änderungen mit svn diff aufgelistet werden:
	`svn diff --changelist bugfix32`
	`svn commit --changelist bugfix32`
	Changelists können nicht mit anderen Benutzern geteilt werden und werden bei einem Commit wieder entfernt (sofern nicht --keep-changelists angegeben wurde).

Zweige (Branches)

`svn` **`copy`** *`http://svn.example.com/project/trunk http://svn.example.com/project/branch/my-new-branch`* **`-m`** *`"message"`*	Legt einen neuen Entwicklungszweig an (immer direkt im Repository kopieren, über URLs, da SVN die Daten intern nicht sofort kopiert, sondern erst einmal nur Verweise anlegt).
`svn` **`update`** `svn` **`merge`** *`http://svn.example.com/project/trunk`*	Entwicklungszweig aktuell halten. Wird dieses Kommando vom aktuellen Zweig aus aufgerufen, so werden die Änderungen des Hauptzweiges in den aktuellen Zweig eingepflegt. Mittels `svn revert . -R` können alle Änderungen wieder rückgängig gemacht werden.
`current path = /home/user/project/trunk` `svn` **`update`** `svn` **`merge`** `--reintegrate` *`http://svn.example.com/project/branch/my-new-branch`* `svn` **`commit`** `-m` *`"my-new-branch merged"`* `svn` **`delete`** *`http://svn.example.com/project/branch/my-new-branch`*	Zusammenführen der Hauptzweiges mit einem Entwicklungszweig. Eine aktuelle Kopie des Hauptzweiges auschecken (wichtig: unbedingt sicherstellen das dies die aktuellste Version ist - svn update). `svn merge --reintegrate` übernimmt alle Änderungen in den Hauptzweig. **Wichtig: den Entwicklungszweig nicht mehr verwenden und ein weiteres mal mit dem Hauptzweig mergen. Wenn dann wieder neu mittels copy einen neuen Zweig erstellen** `svn merge url --dry-run` = zeigt an, was durchgeführt wird, ohne etwas zu ändern

`svn` **`mergeinfo`** *`quelle ziel`*	*ziel* ist optional (wenn nicht angegeben dann aktuelles Verzeichnis) Zeigt alle Revisionen an, die mit dem aktuellen Ziel bereits gemerged wurden. **--show-revs eligible** = es werden zusätzlich Revisionen angezeigt, die für ein merge in Frage kommen
`svn` **`switch`** *`new-repository-url`*	Aktuelle Arbeitskopie wird auf eine andere Repository-URL umgestellt. Nur Unterschiede werden neu heruntergeladen (=schnell). Kann auch auf Unterverzeichnisse angewendet werden (so können diese auf verschiedene Zweige verweisen)

Allgemeine Parameter

`--no-auth-cache`	Unterbindet einmalig das Passwort-Caching für den Repository Zugriff
`--username`	Setzt einen Benutzernamen für die aktuelle SVN Operation

`--no-recursive (-N)`
`--revursive (-R)`
`--depth`
	Legt die Rekursionstiefe eines SVN Kommandos fest. Per default arbeiten die meisten Befehle rekursiv.
	--depth empty: es wird nur das aktuelle Ziel behandelt (keine Dateien oder Verzeichnisse darunter)
	--depth files: es wird nur das aktuelle Ziel und dessen unmittelbaren Kinder, die Dateien sind behandelt
	--depth immediates: es wird nur das aktuelle Ziel und dessen unmittelbaren Kinder behandelt
	--depth infinity: es werden alle Kinder behandelt