Записки программиста

Блог о программировании, операционных системах, СУБД, девайсах, сетях, алгоритмах, электронике и пр

Краткая шпаргалка по основным командам Subversion

15 февраля 2016

Не могу сказать, что я большой фанат Subversion. По-моему, Git прекрасен, и никакие другие системы контроля версий не нужны. Тем не менее, работать с Subversion время от времени приходится, потому что нужно сделать checkout какого-то древнего полумертвого проекта или еще почему-то. Так что, в этой заметке мы рассмотрим основы работы с Subversion, ну и заодно почему он иногда может быть даже интереснее, чем Git. Заметка рассчитана на тех, кто уже имеет опыт использования Git или хотя бы Mercurial.

Перекрестная ссылка: Вас также может заинтересовать старенькая заметка Мой первый опыт работы с Subversion. Там речь идет больше про настройку серверной стороны, а также про использование всяких TortoiseSVN и прочей GUI'ни. Данный же пост посвящен работе с уже существующими репозиториями из консоли.

Итак, почему же Subversion иногда может быть интереснее, чем Git:

- Считается, что централизованную систему контроля версий проще объяснить новичкам;
- Используются последовательные номера ревизий, ясно что за чем шло;
- Тут можно чекаутить отдельные каталоги и делать для них свои бранчи;
- Есть поддержка file lock, что иногда, пожалуй, может быть удобно;
- Subversion лучше работает с бинарными файлами и компактно хранит их дифы;

Я, впрочем, не настоящий сварщик. То есть, далеко не гугу SVN. Но в первом приближении вроде все верно.

Теперь перейдем к командам.

Делаем checkout:

svn co --username eax https://example.com/project/trunk/

Подсасываем последние изменения:

svn up

Проверить, в какой ветке мы находимся и на какой сервер смотрим:

svn info

Посмотреть историю изменений:

svn up
svn log | less

История изменений с diff'ами, аналог `git log -p`:

svn log --diff | less

Кто какие строчки когда менял:

svn blame -v test.txt

Посмотреть незакомиченные изменения:

svn diff

Какие файлы были изменены или добавлены:

svn diff --summarize

Изменения в рамках ревизиции, аналог `git show`:

```
# посмотреть комментарий
svn log -c 123456
# посмотреть изменения
svn diff -c 123456
```

Посмотреть измененные в ревизии файлы:

svn diff --summarize -c 123456

Изменения по сравнению с текущей ревизией, аналог `git diff`:

```
svn diff -r 123456
svn diff --summarize -r 123456
```

Применение сохраненного в файл дифа, аналог `git apply`:

patch -p0 -i myfile.diff

Отменить последние изменения, аналог `git reset --hard HEAD`:

svn revert --recursive .

Текущее состояние репозитория, измененные файлы и так далее:

svn status

Удалить неотслеживаемые файлы и каталоги — встроенной команды, увы, нет, но можно прописать алиас в .bashrc:

svn status | perl -lne 'if(/^\?\s+(.*?)$/){ print $1 }' | xargs rm -r

Получение списка бранчей:

svn ls https://example.com/project/branches/

Создание нового бранча или тэга:

```
svn copy https://example.com/project/trunk/ \
  https://example.com/project/branches/test-branch

svn copy https://example.com/project/trunk/ \
  https://example.com/project/tags/1.0 \
  -m "Release 1.0"
```

Переключение на бранч:

```
cd path/to/trunk
cd ..
mkdir branches
cd branches
svn co https://example.com/project/branches/test-branch
cd test-branch
```

Мерж бранча:

```
svn merge http://example.ru/project/branches/test-branch
```

Удаление бранча:

```
svn delete http://example.ru/project/branches/test-branch \
  -m "Removing test-branch"
```

Примечание: Примите также во внимание, что если вы сделали checkout самого корня репозитория, в котором находятся каталоги trunk, branches и tags, то можете просматривать бранчи обычным `ls`, удалять обычным `svn rm` с последующим коммитом, и так далее. Впрочем, в больших проектах вы вряд ли захотите делать checkout вот прямо всего репозитория целиком.

Добавить файл:

```
svn add text.txt
```

Переименовать файл:

```
svn mv from.txt to.txt
```

Удалить файл:

```
svn del file.txt
```

Lock/unlock, чтобы файл никто не мог менять кроме нас:

```
svn lock file.txt
svn unlock file.txt
```

Коммит и сразу пуш, потому что это SVN:

```
svn commit -m 'Your comment here'
```

В общем-то, описанных команд вам хватит в 95% всех случаев. Дополнительную информацию рекомендую искать в man'ах и help'ах. Еще можно порекомандовать книгу Pragmatic Version Control using Subversion.

А пользуетесь ли вы Subversion и если да, то какие команды советовали бы добавить к приведенному выше списку?

Дополнение: Практика работы с системами контроля версий

Метки: Разработка.

Anon • 3 года назад

git svn - снимет всю боль

1 • Поделиться ›

Timoshka Totoshka ➚ Anon • год назад

Не всегда. Порой git svn - только добавляет боль

• Поделиться ›

TheDoctor ➚ Anon • 3 года назад

Аминь. В силу непреодолимых обстоятельств на проекте используется svn, для себя пользуюсь git svn, но грядуд времена, когда мне это запретят. Как жить тогда не представляю.

• Поделиться ›

dmytrish ➚ Anon • 3 года назад

Лучше, чем ничего, конечно, но лишняя прослойка иногда добавляет головной боли.

• Поделиться ›

Shchvova • 3 года назад

Во всех игровых компаниях которые я видел использовался SVN. Причина банальна - там были репозитории на петабайты буквально. Типа исходники ассетов, экспортированные ассеты, видео и т.п.

• Поделиться ›

Konstantin Makarov • 3 года назад

diff и merge принимают опции
-c N -- конкретная ревизия
-r M:N -- диапазон правок для сравнения
Здесь есть нюанс. Git и Mercurial работают с наборами изменений (changesets) и "нумеруют" их.
Svn же нумерует ревизии - замороженные состояния между изменениями.
Поэтому, обращаю внимание людей пришедших из Git, чтобы слить в trunk из ветки изменения с M по N, нужно указывать первую ревизию на единицу меньше:
svn merge -r M-1:N branch-url

Для отмены закоммиченных изменений нужно мержить диапазон правок, указанных в обратном порядке:
-r N:M
или указать отрицательный номер ревизии:
-c -N

URL-ы принимают параметр @N - номер правки. Бывает полезно, чтобы сослаться, например, на удаленную ветку, по URL-у которой репозиторий теперь отвечает 404

NotFound

svn merge -r 12:34 https://svn.my.domain/repo/branches/deleted@34

^ v • Поделиться ›

Yauheni Akhotnikau • 3 года назад

К use-cases, в которых использование Svn может быть оправдано, можно добавить еще и возможность назначать права доступа к частям репозитория (т.е. кто-то может только читать часть репозитория, а кто-то вообще не может из этой части ничего взять).

Так же имеет смысл сказать, что svn rm (он же svn del) может удалять не только файлы, но и ветки (когда они больше не нужны после слияния).

Еще очень полезна команда svn export, которая позволяет взять содержимое ветки (или части ветки) из репозитория к себе без создания рабочей копии (т.е. без создания .svn со всей внутренней кухней).

^ v • Поделиться ›

Konstantin Makarov • 3 года назад

После svn update бывает нелишним прибраться в освободившихся каталогах, например, в python-проекте прибить *.pyc-файлы, оставшиеся без исходника после переименования модуля или пакета:
find -name '*.pyc' -delete

Переключить рабочий каталог на другую ветку (по дефолту обновит до последней правки HEAD):
svn switch https://example.com/project...

Посмотреть, на какой URL/ветку нацелен рабочий каталог:
svn info

Со свойствами (properties) работать приходится не часто, но они полезны для:
* заигнорить файлы/каталоги;
* подключить "внешний" svn-url в подкаталог;
* автоматически добавлять mime-type;
* автоконвертация EOL, ...

У всех команд есть дополнительные ключи, самые полезные запоминаются быстро.
svn help
svn help <команда>

1 ^ v • Поделиться ›

- # Коротко о себе

  Всем привет! Меня зовут Александр, позывной любительского радио R2AUK. Здесь я пишу об интересующих меня вещах и временами — просто о жизни.

  Вы можете следить за обновлениями этого блога с помощью RSS, Facebook, ВКонтакте, Twitter или Telegram. Если вам нравится данный сайт, возможно, вы захотите поддержать его на Patreon.

  Мой контактный e-mail — mail@eax.me. Если вы хотите мне написать, прошу предварительно ознакомиться с этим FAQ. Если у вас технический вопрос, просьба адресовать его на форум forum.devzen.ru.

- Поиск

- # Популярные заметки

  - Моя шпаргалка по работе с Git, 11964 просмотра за месяц
  - Краткая шпаргалка по сочетаниям клавиш в IntelliJ IDEA, 7185 просмотров за месяц
  - Моя шпаргалка по работе в Vim, 6407 просмотров за месяц
  - Начало работы с PostgreSQL, 5361 просмотр за месяц
  - Настройка фаервола с помощью iptables за пять минут, 3546 просмотров за месяц
  - Памятка по virtualenv и изолированным проектам на Python, 2875 просмотров за месяц
  - Установки и настройка OpenVPN в Ubuntu Linux за 5 минут, 2637 просмотров за месяц
  - Зачем нужен Docker и практика работы с ним, 2621 просмотр за месяц
  - Начало работы с Vagrant и зачем он вообще нужен, 2584 просмотра за месяц
  - Пишем под микроконтроллеры STM32 в Arduino IDE, 1907 просмотров за месяц
  - Потоковая репликация в PostgreSQL и пример фейловера, 1831 просмотр за месяц
  - Советы и примеры задач, которые помогут вам в освоении нового языка программирования, 1809 просмотров за месяц
  - Памятка по регулярным выражениям, 1780 просмотров за месяц
  - Туториал по контейнеризации при помощи LXC, 1755 просмотров за месяц
  - Как спроектировать схему базы данных, 1672 просмотра за месяц
  - Redis и области его применения, 1671 просмотр за месяц
  - Непрерывная интеграция с Jenkins, 1649 просмотров за месяц
  - Простой пример работы с PostgreSQL на Python, 1617 просмотров за месяц
  - Перестаем бояться виртуализации при помощи KVM, 1598 просмотров за месяц
  - Шпаргалка по основным инструкциям ассемблера x86/x64, 1533 просмотра за месяц