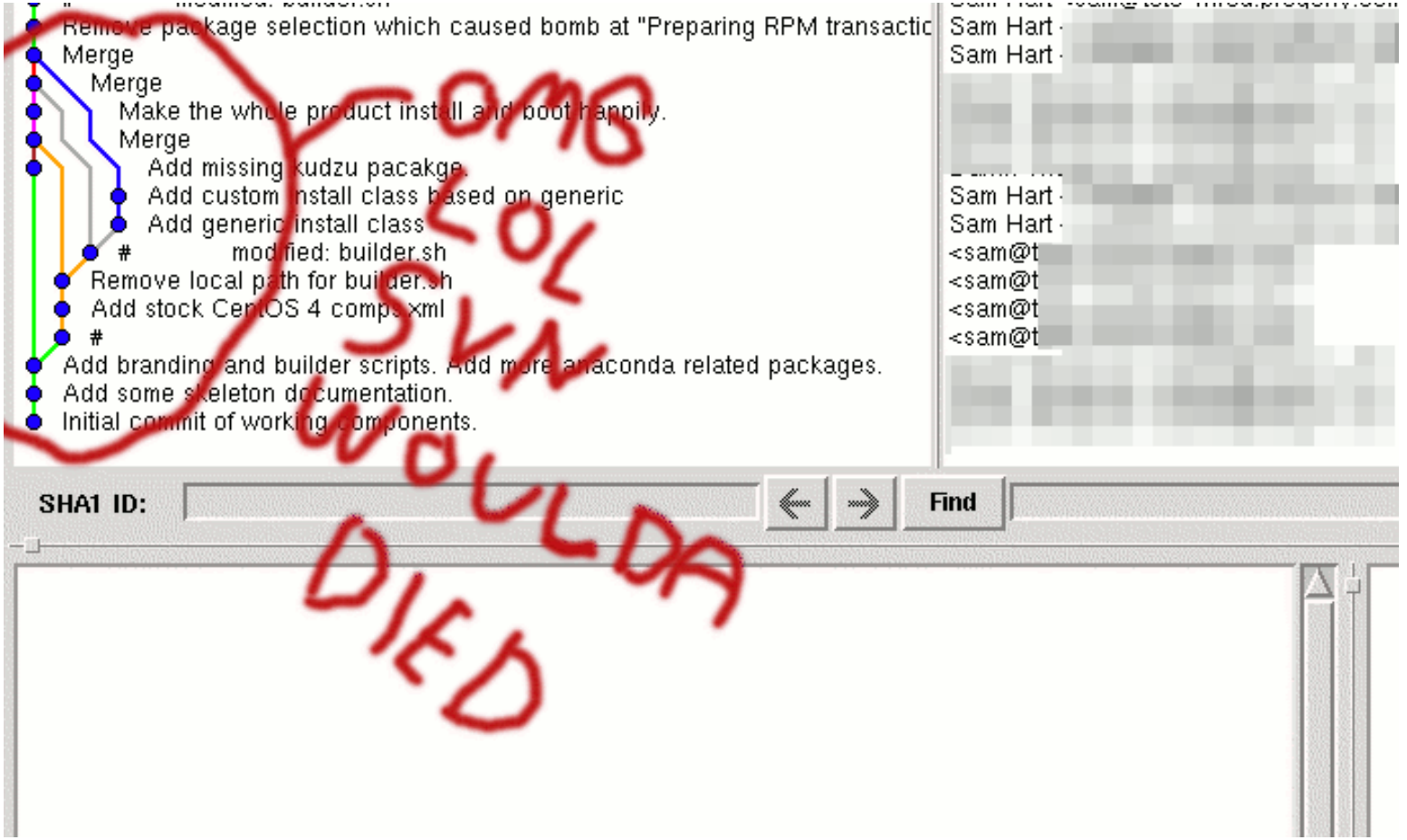
GIT Basics


Remove package selection which caused bomb at "Preparing RPM transactic
Merge
Merge
Make the whole product install and boot happily.
Merge
Add missing kudzu pacakge
Add custom install class based on generic
Add generic install class
modified: builder.sh
Remove local path for builder.sh
Add stock CentOS 4 comps.xml
#
Add branding and builder scripts. Add more anaconda related packages.
Add some skeleton documentation.
Initial commit of working components.
Sam Hart
Sam Hart
Sam Hart
Sam Hart
<sam@t
<sam@t
<sam@t
<sam@t
SHA1 ID:
Find
SVN
DIED

Über mich

Daniel Hiller

41 Jahre, verheiratet, zwei Kinder, WAF

Java seit 1996

GIT seit 2011

Vorher SVN, HG, CVS, VSS

Ansonsten Clean Code, Refactoring, TDD

Musik (Rock/Metal/Alternative)

Snowboarding, Inlinern

GIT – Warum?

* 2005 von Linus Torvalds als Ablösung von Bitkeeper (kommerzielles DVCS)

Ziele:

- Schnell
- Einfach
- Unterstützung nicht linearer Entwicklung
- Vollständig verteilt
- Fähigkeit zur Verwaltung großer Projekte

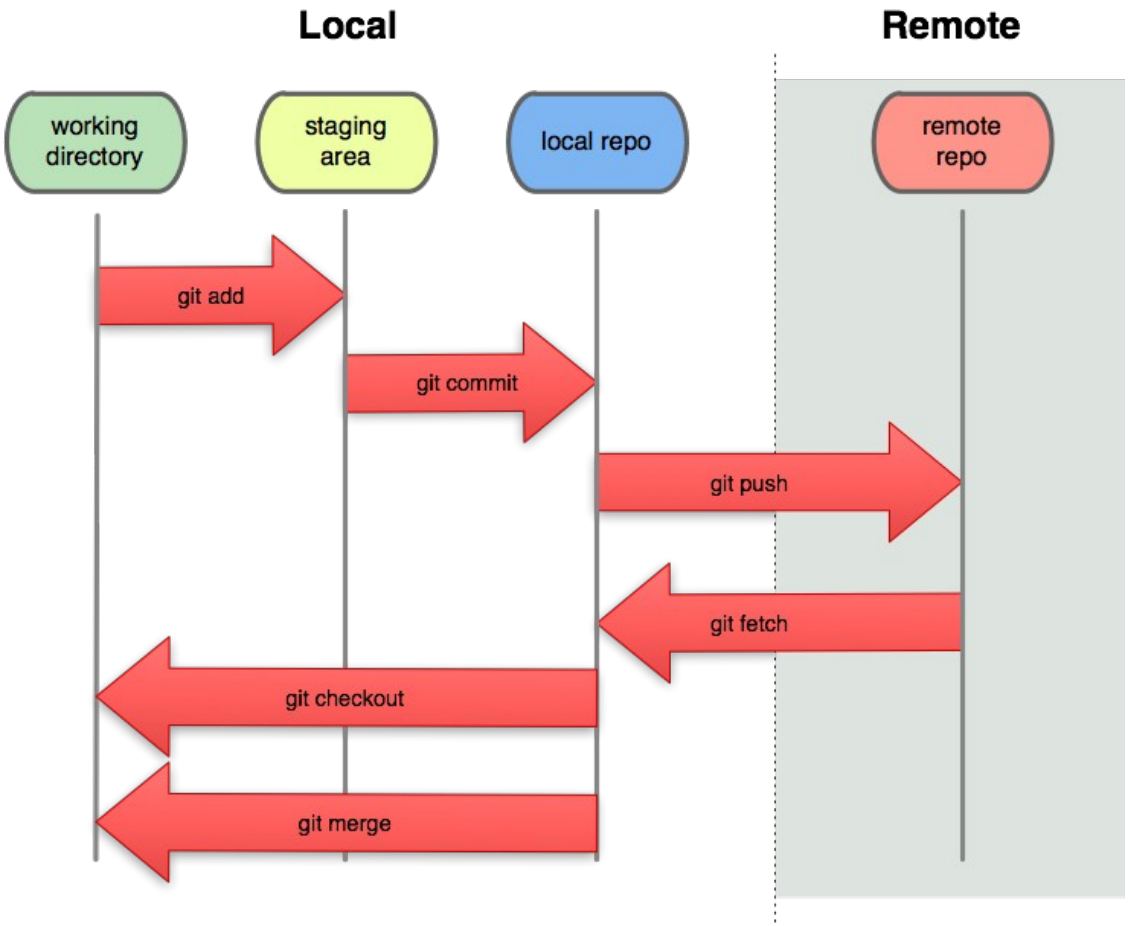
„take Concurrent Versions System (CVS) as an example of what **not** to do; if in doubt, make the **exact opposite decision**“

http://en.wikipedia.org/wiki/Git_(software)

GIT - Features

- Verteiltes Versionskontrollsystem (=DVCS)
- Commits = Snapshots
- Commits haben Checksummen
- Branches sind billig
- Staging Area bestimmt, was commited wird

GIT – DVCS?


Local
Remote
working directory
staging area
local repo
remote repo
git add
git commit
git push
git fetch
git checkout
git merge

GIT - Vorteile

- Geschwindigkeit
- Grundsätzlich wird immer lokal gearbeitet
  - Committing
  - Branching
  - Historie lokal
- Änderungen werden nur explizit im entfernten Repository veröffentlicht bzw. von diesem geholt
- Mehrere entfernte Repositories möglich

GIT vs. SVN

GIT	SVN
Jeder Benutzer hat eine vollständige Kopie des Repositories (incl. aller ausgecheckten Branches und **der Historie**)	Jeder Benutzer hat die Arbeitskopie der aktuell ausgecheckten Revision
Jeder Commit wird durch eine **eindeutige** hashbasierte ID identifiziert	Jeder Commit wird durch eine fortlaufende Nummer identifiziert
Jeder Commit identifiziert den Zustand einer Menge von Dateien („Snapshot des Teil-Dateisystems“)	Jeder Commit identifiziert eine Menge von Änderungen (Diffs)
Der Benutzer macht seine Commits **lokal** und veröffentlicht (pusht) die Commits später auf dem entfernten Repository	Jeder Commit des Benutzers wird sofort an den Server gesendet

GIT

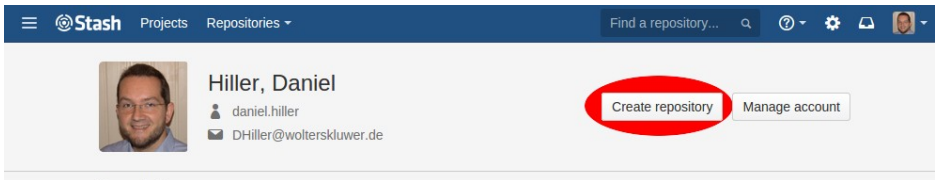
DEMO!

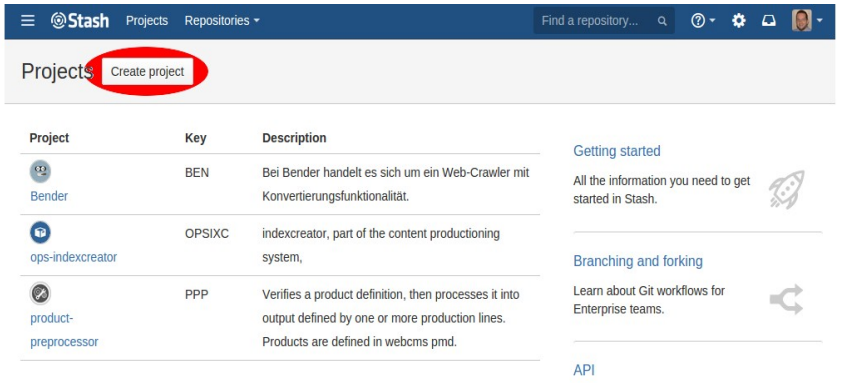
GIT – Neues lokales Repo

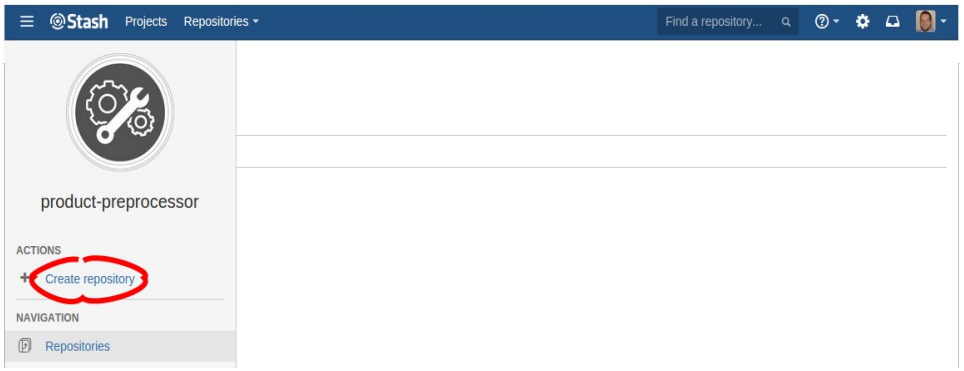
```
➜  mkdir git-demo
➜  cd git-demo
➜  git init  # Neues lokales GIT repository erstellt
Initialisierte leeres Git-Repository in /home/dhiller/projects/git.wkd.wolterskluwer.de/git-demo/.git/
➜  git:(master) touch file1
➜  git:(master) ✗ git add file1  # erste Datei zur Staging Area hinzugefügt
➜  git:(master) ✗ git commit -m "Erster Commit"
[master (Basis-Commit) 4712b04] Erster Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file1
➜  git:(master) touch file2
➜  git:(master) ✗ git add file2  # Zweite Datei zur Staging Area hinzugefügt
➜  git:(master) ✗ touch file3
➜  git:(master) ✗ git commit -m "Zweiter Commit (nur file1 in staging area eingefügt)"
[master a80a9b3] Zweiter Commit (nur file1 in staging area eingefügt)
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file2
➜  git:(master) ✗ git status  # zeigt, dass die dritte Datei nicht Teil des Commits war, da kein „git add file3“ erfolgte
Auf Branch master
Unbeobachtete Dateien:
  (benutzen Sie "git add <Datei>..." um die Änderungen zum Commit vorzumerken)

file3

nichts zum Commit vorgemerkt, aber es gibt unbeobachtete Dateien (benutzen Sie "git add" zum Beobachten)
```

```
➜  git:(master) ✗ git --no-pager log -2  # --no-pager = "less" deaktivieren
commit a80a9b3c5789ef65ca16eb2653a973cabcc9cf85
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 08:05:14 2014 +0200

    Zweiter Commit (nur file1 in staging area eingefügt)

commit 4712b04f8f77113f1fa7d0116f51681836215360
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 07:59:39 2014 +0200

    Erster Commit
➜  git:(master) ✗ git remote add origin ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git # URL des entfernten Repositories hinzufügen
➜  git:(master) ✗ git push --set-upstream origin master  # Änderungen veröffentlichen
Zähle Objekte: 5, Fertig.
Delta compression using up to 4 threads.
Komprimiere Objekte: 100% (3/3), Fertig.
Schreibe Objekte: 100% (5/5), 461 bytes | 0 bytes/s, Fertig.
Total 5 (delta 0), reused 0 (delta 0)
To ssh://git@eulnxd272023:7999/~daniel.hiller/.git
 * [new branch]      master -> master
Branch master konfiguriert zum Folgen von Remote-Branch master von origin.
➜  git:(master) ✗ git ls-remote --heads # Heads des entfernten Repositories
From ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
a80a9b3c5789ef65ca16eb2653a973cabcc9cf85  refs/heads/master
➜  git:(master) ✗ git log -1
➜  git:(master) ✗ git --no-pager log -1
commit a80a9b3c5789ef65ca16eb2653a973cabcc9cf85
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 08:05:14 2014 +0200

    Zweiter Commit (nur file1 in staging area eingefügt)
```

GIT – Remote Repository








```
➜   git:(master) ✗ git remote add origin ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git # URL des entfernten Repositories hinzufügen
➜   git:(master) ✗ git push --set-upstream origin master  # Änderungen veröffentlichen
Zähle Objekte: 5, Fertig.
Delta compression using up to 4 threads.
Komprimiere Objekte: 100% (3/3), Fertig.
Schreibe Objekte: 100% (5/5), 461 bytes | 0 bytes/s, Fertig.
Total 5 (delta 0), reused 0 (delta 0)
To ssh://git@eulnxd272023:7999/~daniel.hiller/.git
 * [new branch]      master -> master
Branch master konfiguriert zum Folgen von Remote-Branch master von origin.
➜   git:(master) ✗ git ls-remote --heads # Heads des entfernten Repositories
From ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
a80a9b3c5789ef65ca16eb2653a973cabcc9cf85 refs/heads/master
➜   git:(master) ✗ git log -1
➜   git:(master) ✗ git --no-pager log -1
commit a80a9b3c5789ef65ca16eb2653a973cabcc9cf85
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 08:05:14 2014 +0200

    Zweiter Commit (nur file1 in staging area eingefügt)
```

GIT - Branching

```
→ git:(master) ✗ git checkout -b my-feature-branch
Gewechselt zu einem neuem Branch 'my-feature-branch'
→ git:(my-feature-branch) ✗ git status
Auf Branch my-feature-branch
Unbeobachtete Dateien:
  (benutzen Sie "git add <Datei>..." um die Änderungen zum Commit vorzumerken)

file3

nichts zum Commit vorgemerkt, aber es gibt unbeobachtete Dateien (benutzen Sie "git add" zum Beobachten)
→ git:(my-feature-branch) ✗ joe file3 # "Dies ist ein Test" in file3 einfügen
Bearbeite '/home/dhiller/.joerc'...Bearbeite '/etc/joe/ftyperc'...fertig
    IW   file3 (Geändert)                                                Row 1
  Col 18   9:21  Ctrl-K H for help
Dies ist ein Test
fertig
Datei file3 gespeichert
→ git:(my-feature-branch) ✗ git add --all
→ git:(my-feature-branch) ✗ git status
Auf Branch my-feature-branch
zum Commit vorgemerkte Änderungen:
  (benutzen Sie "git reset HEAD <Datei>..." zum Entfernen aus der Staging-Area)

neue Datei:     file3
```

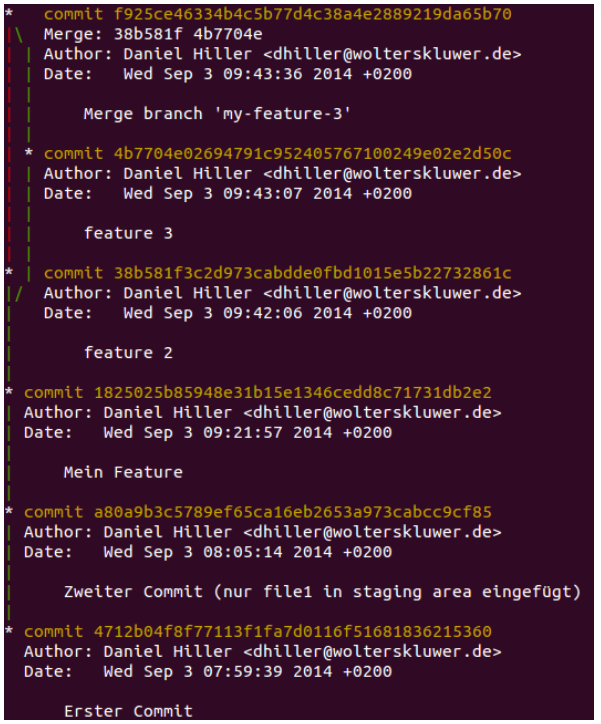
```
→ git:(my-feature-branch) ✗ git commit -m "Mein Feature"
[my-feature-branch 1825025] Mein Feature
 1 file changed, 1 insertion(+)
 create mode 100644 file3
→ git:(my-feature-branch) git push --set-upstream origin my-feature-branch
Zähle Objekte: 3, Fertig.
Delta compression using up to 4 threads.
Komprimiere Objekte: 100% (2/2), Fertig.
Schreibe Objekte: 100% (3/3), 296 bytes | 0 bytes/s, Fertig.
Total 3 (delta 0), reused 0 (delta 0)
To ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
 * [new branch]      my-feature-branch -> my-feature-branch
Branch my-feature-branch konfiguriert zum Folgen von Remote-Branch my-feature-branch von origin.
→ git:(my-feature-branch) git --no-pager diff master..HEAD
diff --git a/file3 b/file3
new file mode 100644
index 0000000..6bb8fa4
--- /dev/null
+++ b/file3
@@ -0,0 +1 @@
+Dies ist ein Test
\ No newline at end of file
→ git:(my-feature-branch) git ls-remote --heads
From ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
a80a9b3c5789ef65ca16eb2653a973cabcc9cf85  refs/heads/master
1825025b85948e31b15e1346cedd8c71731db2e2  refs/heads/my-feature-branch
```

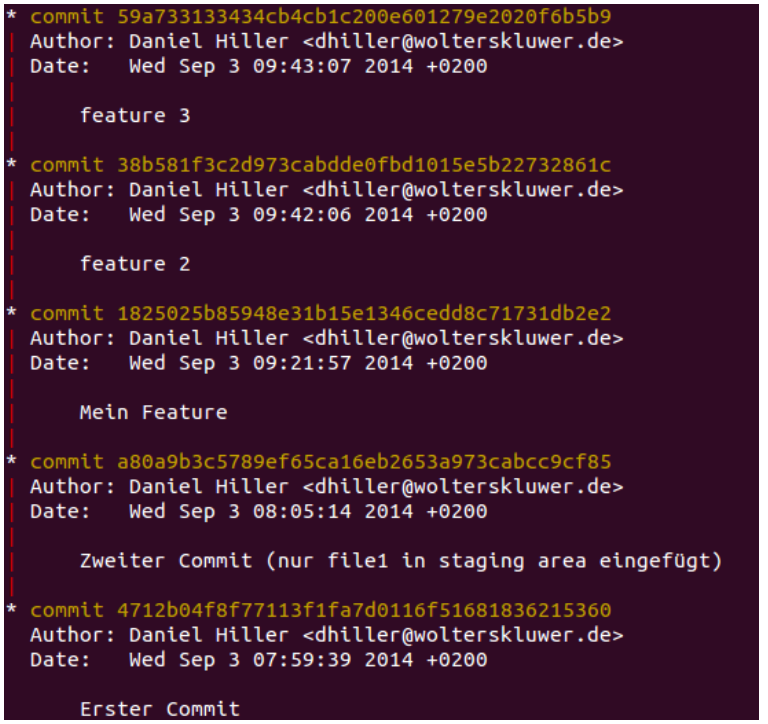
GIT - Merging

```
➜  git-demo git:(my-feature-branch) git checkout master
Gewechselt zu Branch 'master'
Ihr Branch ist auf dem selben Stand wie 'origin/master'.
➜  git-demo git:(master) git pull # Änderungen von remote holen
Already up-to-date.
➜  git-demo git:(master) git merge my-feature-branch
Aktualisiere a80a9b3..1825025
Fast-forward
 file3 | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 file3
➜  git-demo git:(master) git push
Total 0 (delta 0), reused 0 (delta 0)
To ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
   a80a9b3..1825025  master -> master
➜  git-demo git:(master) git ls-remote --heads
From ssh://git@eulnxd272023:7999/~daniel.hiller/git-demo.git
1825025b85948e31b15e1346cedd8c71731db2e2 refs/heads/master
1825025b85948e31b15e1346cedd8c71731db2e2 refs/heads/my-feature-branch
```

```
➜  git-demo git:(master) git --no-pager log
commit 1825025b85948e31b15e1346cedd8c71731db2e2
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 09:21:57 2014 +0200

    Mein Feature

commit a80a9b3c5789ef65ca16eb2653a973cabcc9cf85
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 08:05:14 2014 +0200

    Zweiter Commit (nur file1 in staging area eingefügt)

commit 4712b04f8f77113f1fa7d0116f51681836215360
Author: Daniel Hiller <dhiller@wolterskluwer.de>
Date:   Wed Sep 3 07:59:39 2014 +0200

    Erster Commit
```

GIT – Rebasing (lokal!)





Warum?

- Um die Commit Historie möglichst linear zu halten
- Vorsicht: Historie wird verändert (= Commit Ids!)

GIT - Rebasing

```
➜  git-demo git:(master) git checkout my-feature-3
Gewechselt zu Branch 'my-feature-3'
➜  git-demo git:(my-feature-3) git rebase master
Zunächst wird der Branch zurückgespult, um Ihre Änderungen
darauf neu anzuwenden...
Wende an: feature 3
➜  git-demo git:(my-feature-3) git checkout master
Gewechselt zu Branch 'master'
...
➜  git-demo git:(master) git merge my-feature-3
Aktualisiere 38b581f..59a7331
Fast-forward
 file5 | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file5
```

GIT SVN – Die GIT Einstiegsdroge

```
> git svn clone <svn-url> -s # -s = Standard Repo Layout

… changes …

> git commit -am „My commit message“

> git svn dcommit # Änderungen veröffentlichen

Committing to …

Merge conflict during commit: Your file or directory '…' is
probably out-of-date: resource out of date; try updating …

> git svn rebase # Entfernte Änderungen holen

…

> git svn dcommit # Änderungen veröffentlichen
```

GIT – In Short

Kommando	Bedeutung
git help <command>	Manual öffnen
git clone <url>	Remote Repository holen (vgl. svn co)
git checkout <branch>	Auf branch wechseln
git checkout -b <branch>	Neuen Branch erstellen
git pull <remote>	Änderungen holen
git push <remote>	Änderungen veröffentlichen
git merge <branch>	Änderungen des Branches in aktuellen einfügen
git add <file>	Datei zur Staging Area hinzufügen
git commit -m „Nachricht“	Dateien in Staging Area committen
git status	Derzeitiger Zustand der Arbeitskopie
git diff	Änderungen zum letzten Commit
git stash	Änderungen zwischenspeichern
git stash pop	Zwischengespeicherte Änderungen anwenden

GIT - Q&A

GIT – C'est fini

Repos:

- Try Git: http://try.github.com/
- Lokaler Stash: http://eulnxd272023:7990/
- github.com

Referenzen:

- http://gitref.org/index.html
- GIT: http://git-scm.com/book/de
- GIT-SVN: http://git-scm.com/book/de/Git-und-andere-Versionsverwaltungen-Git-und-Subversion

Weiterführende Literatur

- https://www.atlassian.com/git/workflows