

# GIT WTF?

Björn Guth und Jörg Behrmann

Fachschaft Mathematik/Physik/Informatik der RWTH Aachen

May 31, 2014



- 1 Show and Tell
  - Git im Vergleich
  - Wie kannst man mit Git arbeiten?
    - ssh
    - Gitolite
    - externe Hoster
- 2 Try and (hopefully not) Fail
  - Installation
  - usage
- 3 Weitere Hilfe



# Git WTF?

- Tool zur Versionskontrolle
- erleichtert kollaboratives Arbeiten
- einfacher Weg zu externen Backups von Projekten
- weiteres Argument für  $\text{\LaTeX}$  und gegen MS-Office / Open Office / Libreoffice



# Vorteile von Git gegenüber SVN

- Git ist schneller als SVN<sup>1</sup>
- Git auch lokal und ohne Verbindung zu einem Server
- Es gibt in Git Branches
- SVN Repositories können in Git engebunden werden
- man kann mit Git genauso arbeiten, wie mit SVN

---

<sup>1</sup>zumindest laut <http://git-scm.com/about/small-and-fast>



# Vorteile von Git gegenüber Dropbox

- Git ist open source, Dropbox nicht.
- Dropbox hat so gut wie keine Versionskontrolle
  - Versionskontrolle nur 30 Tage
  - alles darüber hinaus kostet Geld
- Git hat .gitignore
- Dropbox ist ein zentralisiertes System
- Dropbox ist an einen Ordner gebunden
- viel Spaß mit den unterschiedlichen Quota der Dropboxuser
- Google Drive ist ähnlich einzuordnen wie Dropbox



# Vorteile von Git gegenüber Dropbox



Bildquelle: Wikipedia ([https://upload.wikimedia.org/wikipedia/commons/thumb/0/04/National\\_Security\\_Agency.svg/718px-National\\_Security\\_Agency.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/0/04/National_Security_Agency.svg/718px-National_Security_Agency.svg.png))



# Verschiedene Möglichkeiten mit Git zu arbeiten

- Generell gibt es drei Möglichkeiten, Git in deiner Gruppenarbeit zu nutzen:
  - 1 Plain per ssh von Rechner zu Rechner
  - 2 mit Gitolite
  - 3 mit einen externen Hoster



# ssh direkt von Rechner zu Rechner

- eine Person erstellt mit ein Repository
- alle anderen clonen dieses
- mit `git remote add NAME 'ADRESSE ZUM REPOSITORY'` werden alle anderen als Quellen hinzugefügt
- nun können commits von anderen gepullt werden





# Vor- und Nachteile

## Vorteile:

- dezentrales Netz
- alle Daten nur lokal bei euch

## Nachteile:

- auf jeden Rechner muss ein ssh-Daemon laufen
- sinnvoller Weise müsstet ihr euch dafür einen zusätzlichen User anlegen
- wechselnde Netzwerkadressen bereiten Probleme



# Gitolite zur Rechteverwaltung

- Gitolite erledigt Rechteverwaltung bei Git-Repositories
- muss zusätzlich zu Git installiert und eingerichtet werden
- weitere Verwaltung sehr einfach über ein Git-Repository
- bei Interesse kann ich mehr zeigen



# Vor- und Nachteile

## Vorteile:

- alle Daten sind lokal bei euch
- Rechteverwaltung sinnvoll geregelt

## Nachteile:

- Einrichtung nicht trivial
- als zentralistisches System ausgelegt<sup>a</sup>

---

<sup>a</sup>kann aber auch ähnlich wie plain ssh verwendet werden



# externe Hoster

- es gibt Anbieter, die Git-Repositories zur Verfügung stellen.
- als Beispiel seien hier mal genannt:
  - [github.com](https://github.com)<sup>2</sup>
  - [gitorious.org](https://gitorious.org)
  - [bitbucket.org](https://bitbucket.org)<sup>3</sup>
- sehr leicht zu managen
- sind in der Nutzung genauso wie Gitolite

---

<sup>2</sup>bietet auch einige sehr gute Tutorials zu Git

<sup>3</sup>bietet Git- und Mercurial-Repositories und bei Anmeldung mit einer Universitäts-Mail-Adresse private Repository mit Zugriff durch so viele, wie man will.



# Vor- und Nachteile

## Vorteile:

- leichte Einrichtung
- vergleichsweise geringe Downtime

## Nachteile:

- Daten werden bösen Unternehmen in den Rachen geworfen
- Solange man kein Geld bezahlt, sind die Repositories public



# Installation

- Linux:
  - Mit der Paketverwaltung eurer Distribution das Paket `git` installieren.
  - Debian & Ubuntu: `sudo apt-get install git`
  - Ansonsten: <http://git-scm.com/download/linux>
- Windows:
  - <https://msysgit.github.io/>
- Mac OS:
  - <http://git-scm.com/download/mac>



# Konfiguration

- Globale Konfiguration in `~/.gitconfig`
- wichtige globale Konfigurationen:
  - `git config --global user.name NAME`
  - `git config --global user.email EMAIL@ADRESSE`
- im Projekt in `./.git/config`



# wichtigste Befehle

- basic:

- `git init`
- `git clone 'ADRESSE ZUM REPOSITORY'`
- `git add DATEI`
- `git status`
- `git commit -m 'ÄNDERUNGSBESCHREIBUNG'`
- `git commit --amend`
- `git push ADRESSE BRANCHNAME`
- `git pull`
- `git log`

## ein eigener Versuch

- Clont das Repository dieser Präsi
- `git clone kiss12@137.226.113.201:~/git-vortrag`
- schaut mal nach, wie viele commits ich schon gemacht habe



# weiter wichtige Befehle

- more advanced:
  - `git branch BRANCH`
  - `git checkout BRANCH`
  - `git checkout BRANCH1`  
`git merge BRANCH2`
  - `git diff`
  - `git branch -d BRANCH`

## ein weiterer Versuch

- Erstellt einen neuen Branch
- pusht ihn ins Repository
- hat es geklappt?



# Weitere Hilfe

- <http://git-scm.com/doc>
- <https://help.github.com/>
- <http://wiki.ubuntuusers.de/Git>
- [http://www.markus-gattol.name/misc/mm/si/content/git\\_workflow\\_and\\_cheat\\_sheet.png](http://www.markus-gattol.name/misc/mm/si/content/git_workflow_and_cheat_sheet.png)



## GIT Workflow

Global Git configuration is stored in `$HOME/.gitconfig` (`git config --help`)

## Concepts

\$id : notation used in this sheet to represent either a commit id, branch or a tag name  
\$file : arbitrary file name  
\$branch : arbitrary branch name

gilt gleichwohl)

1. **Introduction**

## Resolve Merge Conflicts

Bildquelle: Markus Gattol ([http://www.markus-gattol.name/misc/mm/si/content/git\\_workflow\\_and\\_cheat\\_sheet.png](http://www.markus-gattol.name/misc/mm/si/content/git_workflow_and_cheat_sheet.png))



# Danke für die Aufmerksamkeit<sup>5</sup>

- Wenn ihr noch Fragen habt, wäre jetzt der richtige Zeitpunkt<sup>4</sup>

---

<sup>4</sup>Später geht aber auch!

<sup>5</sup>auch wenn es sehr kommandozeilenlastig war<sup>6</sup>

<sup>6</sup>Ich hoffe aber ich konnte trotzdem ein paar von euch überzeugen Git zu nutzen

