Lecture 1

Git Introduction

Overview of VCS topic
Introduction to Git
Some examples – related to git.bfh.ch

BTF1230 - Basics of Software Development in C
February 29, 2016

Prof. A. Habegger
Bern University of Applied Sciences

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time
- Not only for source files also binary files can be controlled by a VCS e.g. as a web designer $\rightarrow$ tracking images

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time
- Not only for source files also binary files can be controlled by a VCS e.g. as a web designer $\rightarrow$ tracking images
- Allows to revert to a previous state

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time
- Not only for source files also binary files can be controlled by a VCS e.g. as a web designer $\rightarrow$ tracking images
- Allows to revert to a previous state
- Shows **who** modified **what** and **when**

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time
- Not only for source files also binary files can be controlled by a VCS e.g. as a web designer $\rightarrow$ tracking images
- Allows to revert to a previous state
- Shows **who** modified **what** and **when**
- You can recover easily

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

What means VCS

- VCS : **V**ersion **C**ontrol **S**ystem
- It records chances to a file or file-set over time
- Not only for source files also binary files can be controlled by a VCS e.g. as a web designer $\rightarrow$ tracking images
- Allows to revert to a previous state
- Shows **who** modified **what** and **when**
- You can recover easily
- Very little overhead

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Local Version Control Systems

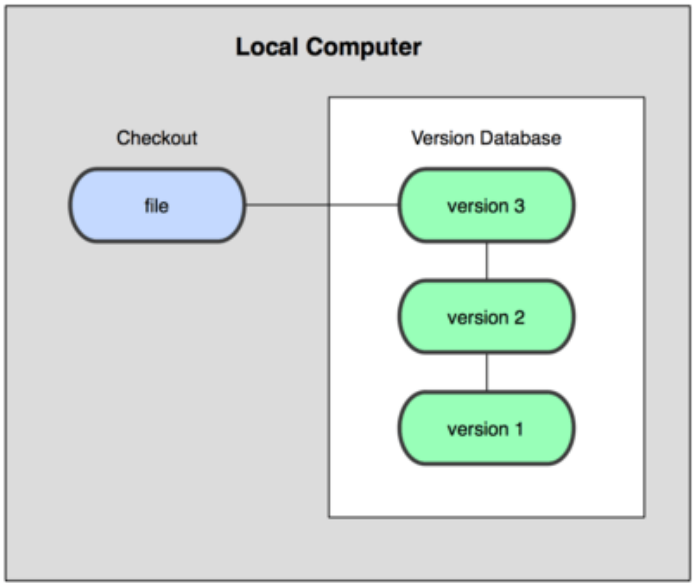
- Copy files into an other directory - timestamped if user is clever
- Simplest method
- Very little control mechanism
- It is easy to forget proper working directory hence wrong file will be accidentally modified

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Local Version Control Systems

- Copy files into an other directory - timestamped if user is clever
- Simplest method
- Very little control mechanism
- It is easy to forget proper working directory hence wrong file will be accidentally modified

- Programmers developed a simple database tracking mechanism to deal with the issues mentioned
- The VCS **RCS** was born that time and is still used nowadays
- RCS (Revision Control System) works on diff patch approach



Img. ref.: `http://git-scm.com`

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Centralized Version Control Systems

- Next issue was that people need to collaborate with developers on other systems.
- A sever based approach was needed (the server acts as a centralized source distributor and repository management unit)

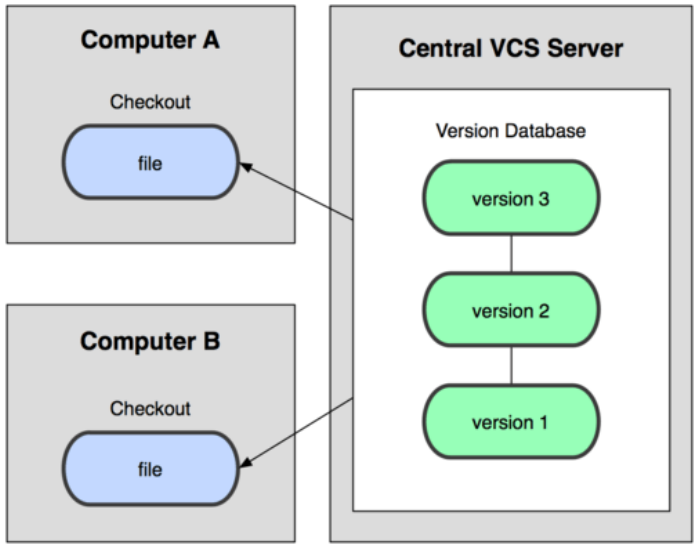
- Drawbacks are single point of failure, nobody can collaborate without server, risky to loos everything

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Centralized Version Control Systems

- Next issue was that people need to collaborate with developers on other systems.
- A sever based approach was needed (the server acts as a centralized source distributor and repository management unit)

- Therefore a Centralized Version Control System (CVCS) were developed
- CVC Systems, such as CVS, Perforce, and Subversion, have a single central place - server.
- Advantages are, everyone knows what everyone else is doing
- Administrators have fine-grained permission control



Img. ref.: `http://git-scm.com`

- Drawbacks are single point of failure, nobody can collaborate without server, risky to loos everything

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

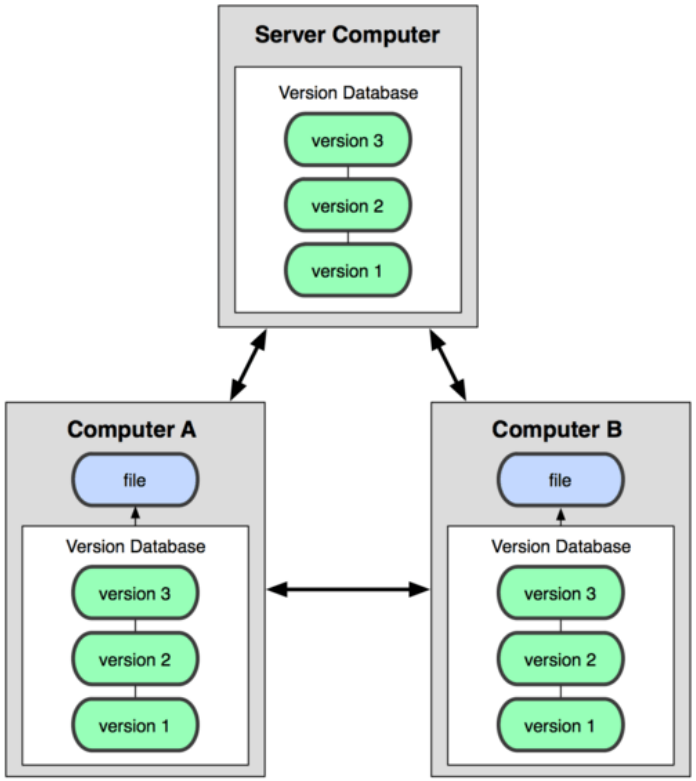
Distributed Version Control Systems

- Distributed Version Control Systems (DVCSs) stepped in
- DVC Systems, such as Git, Mercurial, Bazaar or Darcs, are nowadays standard
- The major difference is clients not only checkout a work-tree copy also they fully mirror the repository

Distributed Version Control Systems

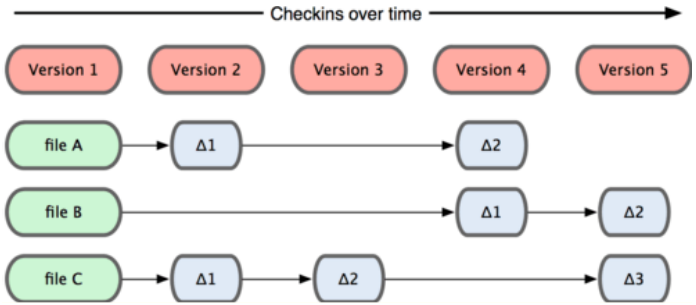
- Distributed Version Control Systems (DVCSs) stepped in
- DVC Systems, such as Git, Mercurial, Bazaar or Darcs, are nowadays standard
- The major difference is clients not only checkout a work-tree copy also they fully mirror the repository

- No longer reliable on an online available central server at anytime
- Off-site full backups of all the data
- Those system deal pretty well with several remote repository e.g. for libraries
- Several new smart workflows are possible

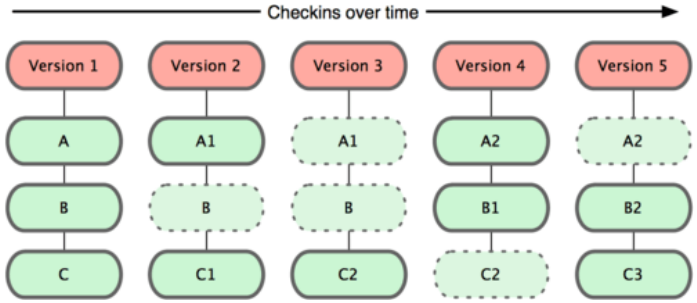


Img. ref.: `http://git-scm.com`

Git concept



Img. ref.:
http://git-scm.com



Img. ref.:
http://git-scm.com

- List of file based changes
- Information is keep as a set of files

- Set of snapshots of a mini filesystem
- No changes means no snapshot of that part

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Git concept

Local operations : Most operations in Git only need local resources and files to operate. Due to local file operation, Git commands don't suffer from network latency. If you browse history Git doesn't need a server connection. An other advantage is the offline-tracking capability

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Git concept

Local operations : Most operations in Git only need local resources and files to operate. Due to local file operation, Git commands don't suffer from network latency. If you browse history Git doesn't need a server connection. An other advantage is the offline-tracking capability

Data integrity : Git is using SHA-1 hash numbers for integrity check. Every file will be check-summed before stored. Hence no modification can be made without letting Git know of the change.

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Git concept

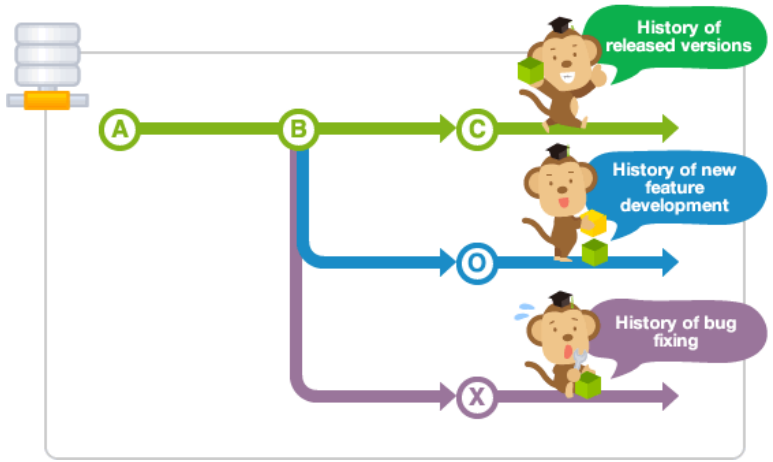
Local operations : Most operations in Git only need local resources and files to operate. Due to local file operation, Git commands don't suffer from network latency. If you browse history Git doesn't need a server connection. An other advantage is the offline-tracking capability

Data integrity : Git is using SHA-1 hash numbers for integrity check. Every file will be check-summed before stored. Hence no modification can be made without letting Git know of the change.

Three states : Git has three stages where files can reside in. Committed, modified and staged.

Why Git

- It is fast
- It has a simple structure
- It is well suited for nonlinear development (branching model)
- Easy to revert
- It is distributed
- It is one of the best systems for huge project (i.e. Linux Kernel)



VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

GUIs for everyone

Platform	Tool Name
plain	git gui / gitk
CLI	tig
Eclipse	eGit
QTcreator	(integrated)
Netbeans	(integrated)
Java standalone	**SmartGit** (free for non-commercial)
KDE (dolphin)	kdesdk-dolphin-plugins
Gnome (nautilus) / Xfce (Thunar)	RabbitVCS
Windows	**SourceTree**, GitEye, Git Extensions, github for Win, Git Bash/GUI

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

First Steps

- Create Repository (local)

  `git init`

- Add a file

  `git add`

- Send a file into the repository

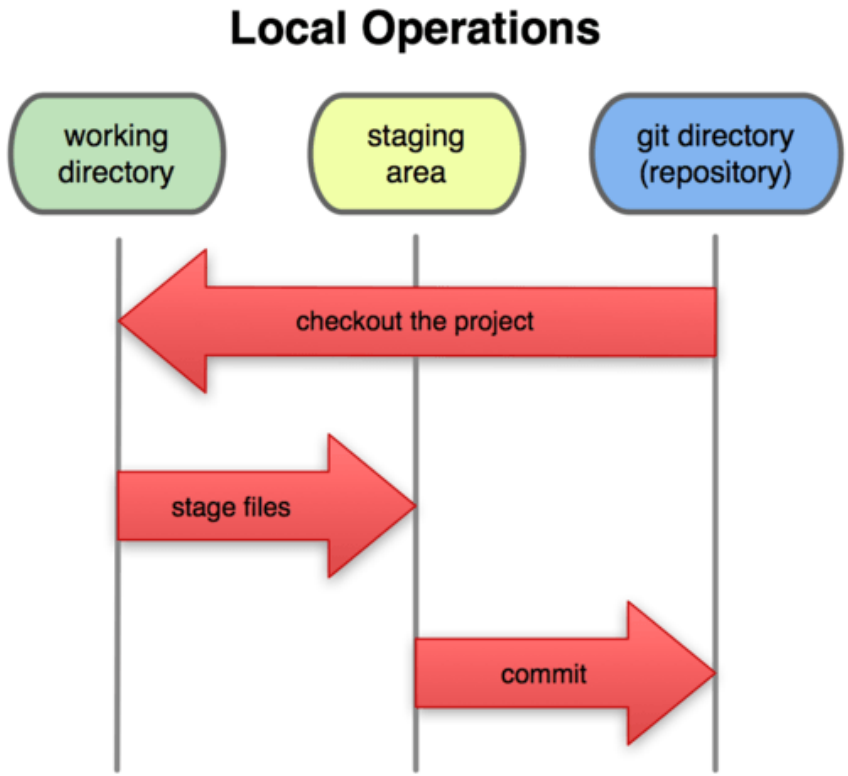
  `git commit`

- Delete a file

  `git rm`

- Rename a file

  `git mv`



Img. ref.: `http://git-scm.com`

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Exclude Files from Tracking

- The file ".gitignore" consists exclude patterns
- Empty lines and characters after # are ignored
- The asterisk (*) is the wild-chart character
- The slash (/) at the end of a pattern indicates a directory name
- By using (!) in-front of a pattern we invert it's meaning (negation)

```
# .gitignore file
*.a
!lib.a
_*
build/
doc/*.txt
```

Remote Repository

Repositories on “git.bfh.ch” are created/configured by ssh commands

- Show permissions granted to you

```
ssh git@git.bfh.ch
```

- Create a new repository

```
ssh git@git.bfh.ch create <REPO_NAME>
```

- Clone the repository

```
git clone git@git.bfh.ch:<REPO_NAME>
```

- Give write access

```
ssh git@git.bfh.ch perms <REPO_NAME> + WRITE <GIT_USER>
```

- Remove write access

```
ssh git@git.bfh.ch perms <REPO_NAME> - WRITE <GIT_USER>
```

- List permissions given by you

```
ssh git@git.bfh.ch perms <REPO_NAME> -l
```

Remote Repository

- Upload modifications

  `git push`

- Pull modifications in repository only

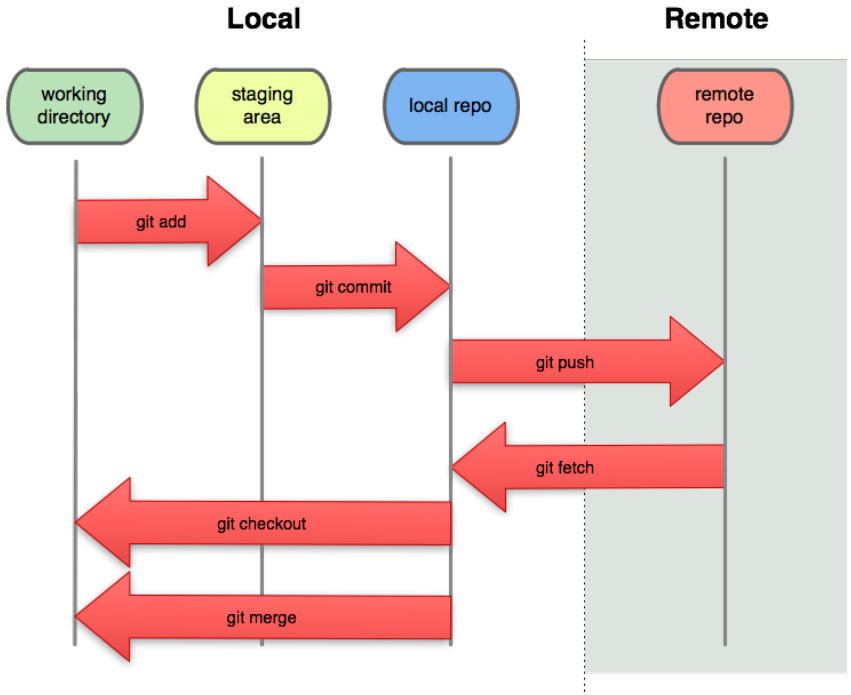
  `git fetch`

- Pull modifications and check-out by ff-merge

  `git pull`

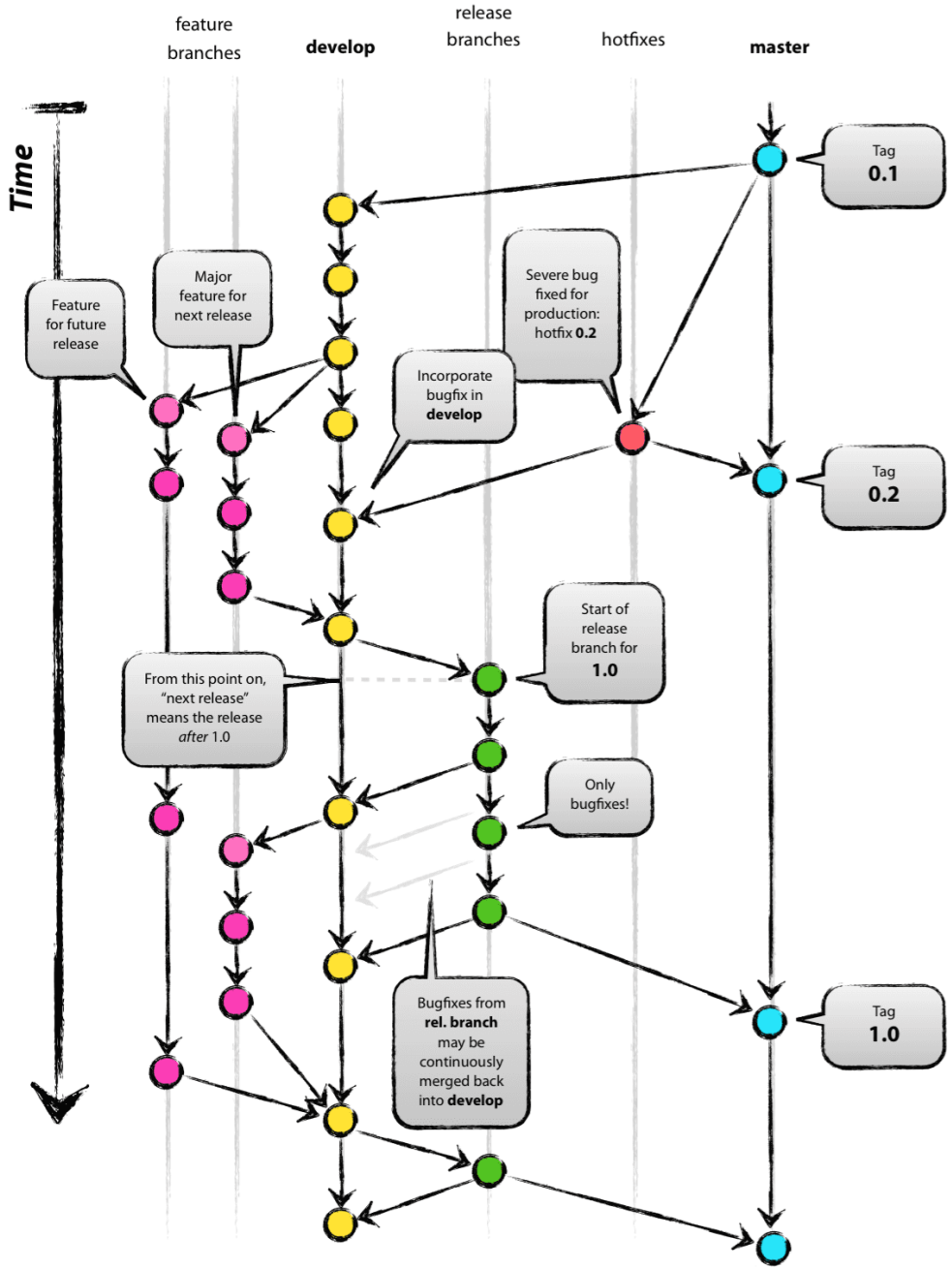
- Check-out data

  `git checkout`



VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

Git Branching Model



Useful links

It was quite a bit of information and there is a big chance to get lost. So, don't waste your time...learn Git!

- Official Git book `http://git-scm.com/book`
- Git wiki on kernel.org `https://git.wiki.kernel.org/index.php/GitFaq`
- Take a tour `http://cworth.org/hgbook-git/tour`
- Learn Git visually `http://marklodato.github.io/visual-git-guide/index-en.html`
- Git for beginners `http://backlogtool.com/git-guide/en`
- Git online reference `http://gitref.org/`
- Upstream: `https://git-scm.com/`
- Overview: `https://linux.bfh.ch/software/git/`
- Git@BFH: `https://git.bfh.ch/`

VCS Introduction
CVS
DVCS
Git
Intro
Why Git
Git GUIs
Usage
Best current practice
Links
List of Literature

List of Literature

- Chacon, Scott and Straub, Ben: *Pro Git*. 2nd Edition. New York, N.Y: Apress. 2014. ISBN 978-1-484200-77-3; available at `http://git-scm.com/book/en/v2`.
- Haenel, Valentin and Plenz, Julius: *Git*. Verteilte Versionsverwaltung fuer Code und Dokumente. München: Open Source Press. 2011. ISBN 978-3-941841-42-0.
- Atlassian Inc.: *Git Tutorial*. An interactive online tutorial to Git `https://www.atlassian.com/git/tutorials`

VCS Introduction
- CVS
- DVCS

Git

Intro
- Why Git
- Git GUIs
- Usage

Best current practice

Links

List of Literature