# Internals

# Gliederung

- Einleitung
- Git Objekte
- Git Referenzen
- Beispiel
- Zusammenfassung
- Quellen

# Git Objekte

- Git Objectstore ist ein Key-Value Store
  - Key: SHA-1 Hash
  - Value: Git Objekt
- SHA Hash über Header + Inhalt
- Alle Objekte werden gehashed

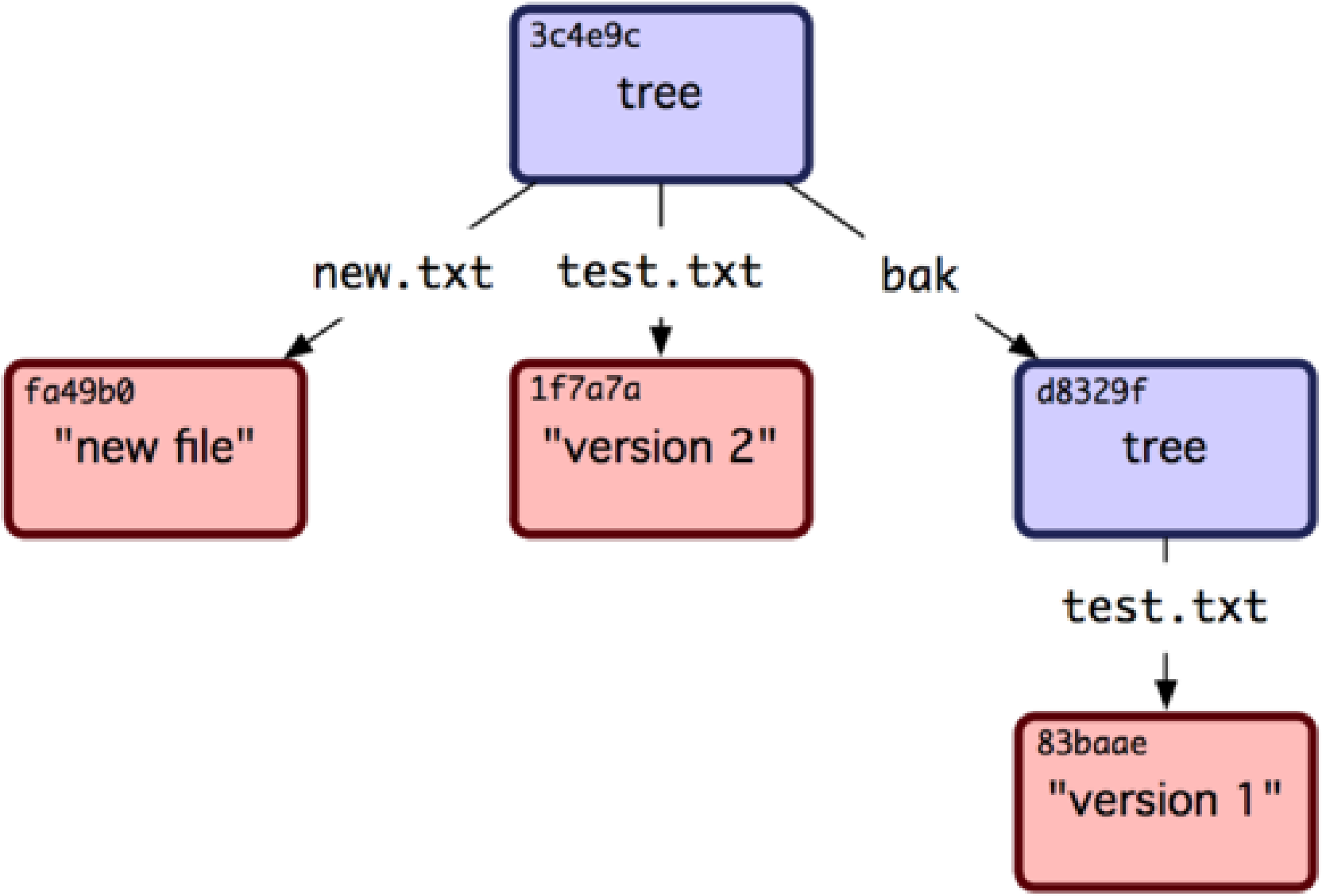
# Blobs

- „just a bunch of bytes“
- Jede Datei wird als Blob gespeichert
- Keine Metadaten, nur Inhalt
- Blob wird bei **git add <file>** erstellt
- Blobs können von mehreren Trees referenziert werden ➜ Speicherefffizienz
- Änderung einer Datei ➜ neuer Blob

# Tree Objekte

- Pointer auf Blob oder anderes Tree Objekt
- Metadaten des Blobs
- Baum aus Tree Objekten und Blobs ist Snapshot des Projekts
- Tree Objekte werden bei **git commit** erstellt

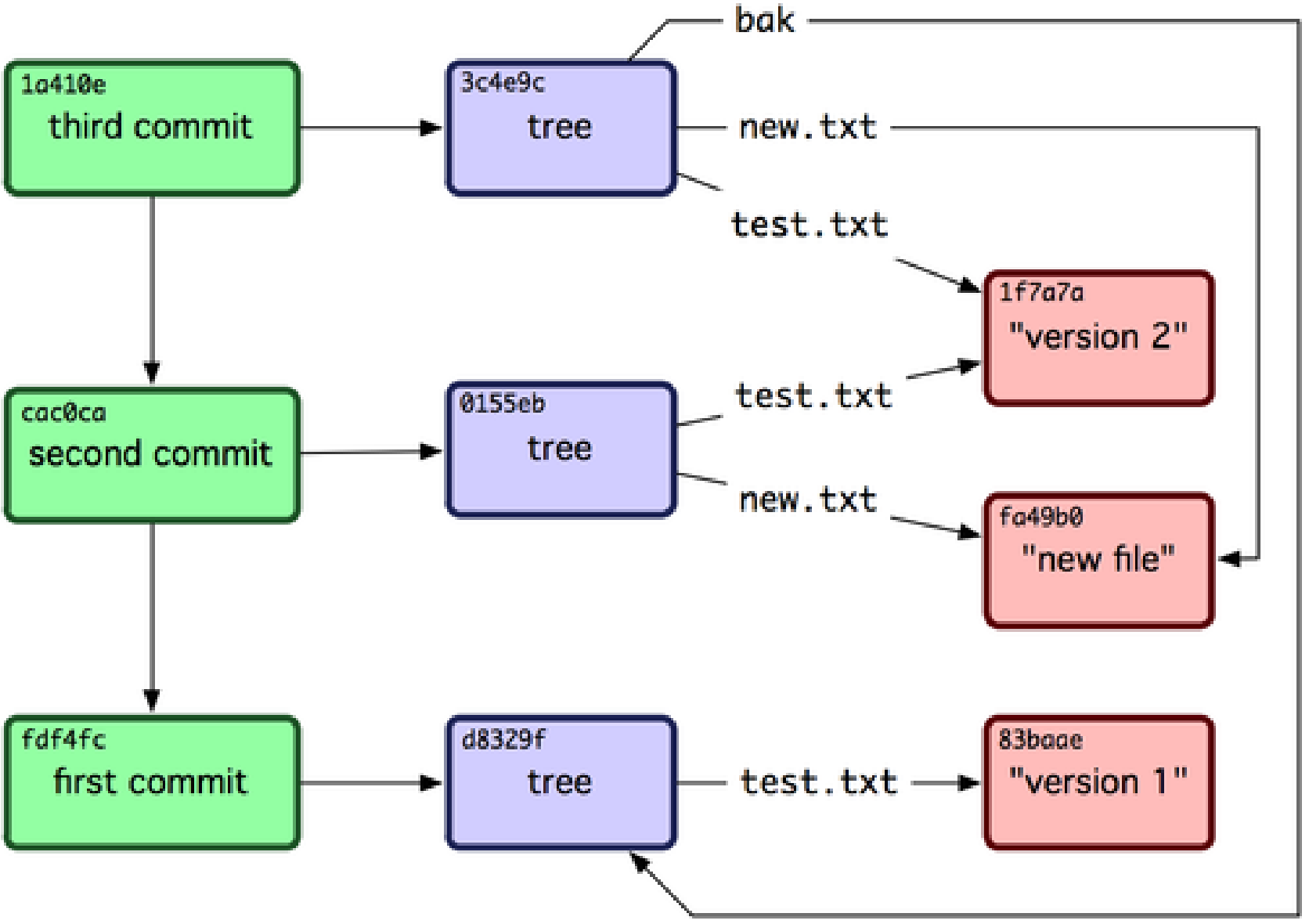
# Baum


3c4e9c
tree
new.txt
test.txt
bak
fa49b0
"new file"
1f7a7a
"version 2"
d8329f
tree
test.txt
83baae
"version 1"

# Commit Objekte

- Commit Objekt
  - Pointer auf einen Baum
  - Pointer auf andere Commit Objekte
  - Author / Commiter (aus Config), Timestamp
  - Commitnachricht
- (fast) jeder Commit hat min. einen Parent
- Repo als DAG aus Commits
- Commit Objekt Hash für Nutzer sichtbar

# Commits


1a410e
third commit
3c4e9c
tree
bak
new.txt
test.txt
1f7a7a
"version 2"
cac0ca
second commit
0155eb
tree
test.txt
new.txt
fa49b0
"new file"
fdf4fc
first commit
d8329f
tree
test.txt
83baae
"version 1"

# Objektspeicherung I

- Kompression: gzib / zlib
- Objekte komprimiert als files in `objects/`
  - Unterordner: Erste 2 Stellen des Hashs
  - Dateiname: Restliche 38 Stellen des Hashs

```
.git/objects
├── 6e
│   └── 1592c16ad49515b09e6b602efd4ab0f7a2472d
├── 11
│   └── f919023e0844a93244c178365d16f8329d32c8
├── info
└── pack
```

# Objektspeicherung II

```
→ git cat-file -p f16c9b2b04
tree 6e1592c16ad49515b09e6b602efd4ab0f7a2472d
parent d97390bf31aa83d7edaffbe5fd84ff9ddf53bc72
author jvf@home <web@jens-fischer.eu> 1372597123 +0200
committer jvf@home <web@jens-fischer.eu> 1372597123 +0200

Update Readme
→
→ git cat-file -p 6e1592c16ad
100644 blob 499c30dacee2f75486fc9225b5aaefa9a3a12814    README.md
→
→ git cat-file -p 499c30dacee
This is a Readme File
```

# Referenzen

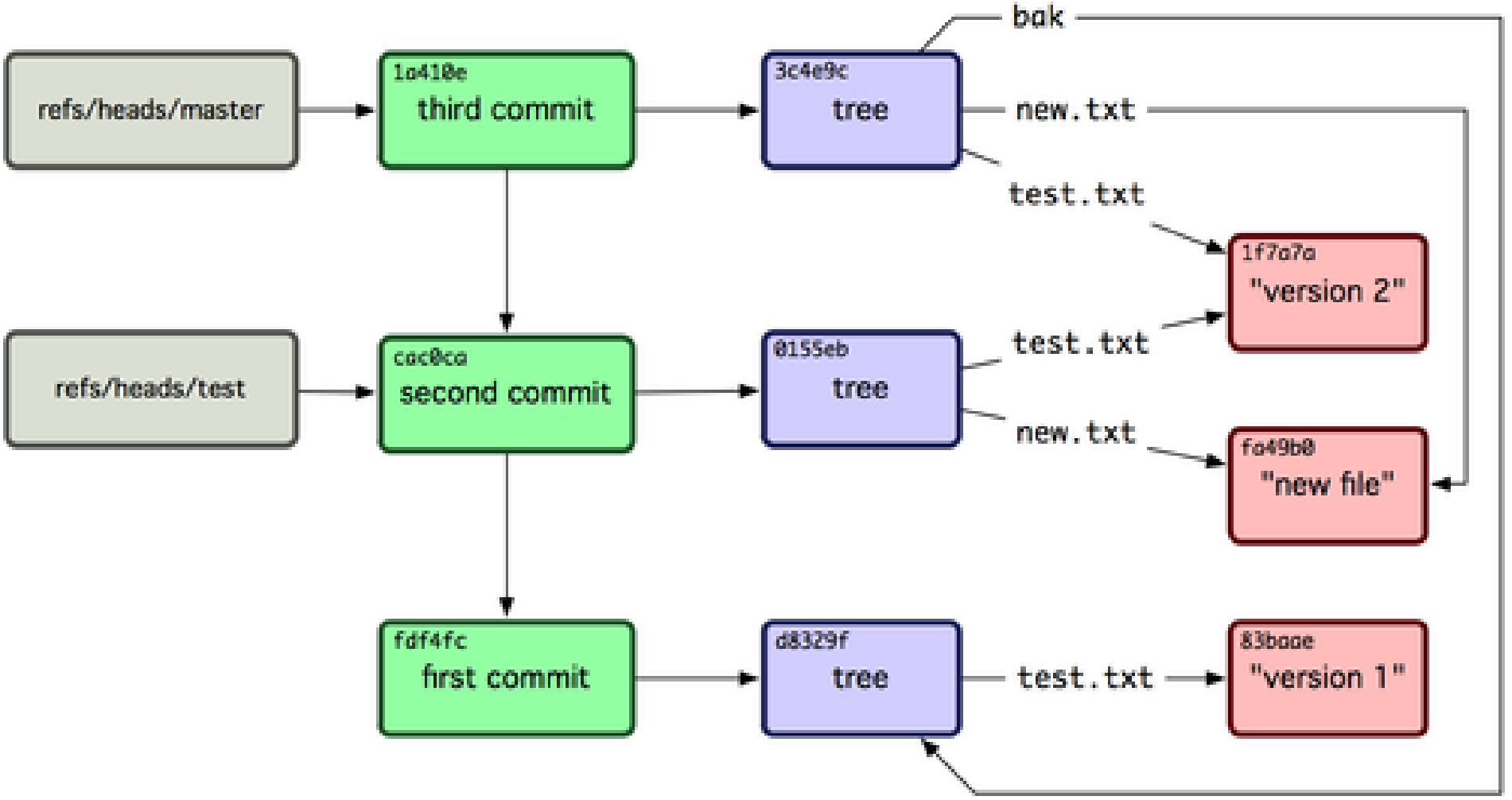
- Nächste Abstraktionsebene: Referenzen
- Basisblock: Commits
- Namenssystem von Git
- Verweis auf einen Commit
- .git/refs

# Referenzen - Heads

- Heads aka Branches
- Pointer auf einen Commit
- Branch: Commit hat mehrere Vorgänger
- .git/refs/heads

```
→ cat .git/refs/heads/master
f16c9b2b04350af1f79647df03e8913388808b1b
```

# Referenzen - Heads



# Referenzen – The HEAD

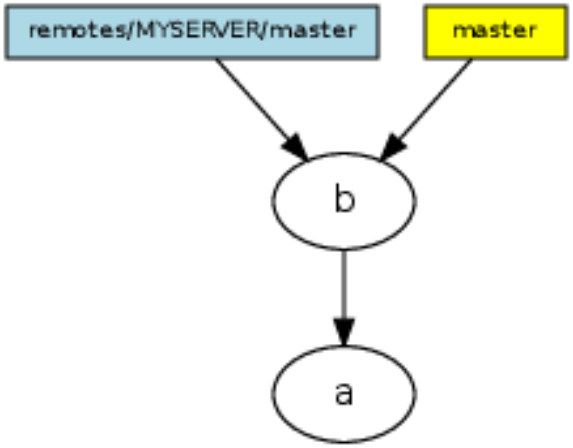
- Spezielle (symbolische) Referenz
- Zeigt auf den aktuellen Branch
- aka „current branch“ / „checked out branch“

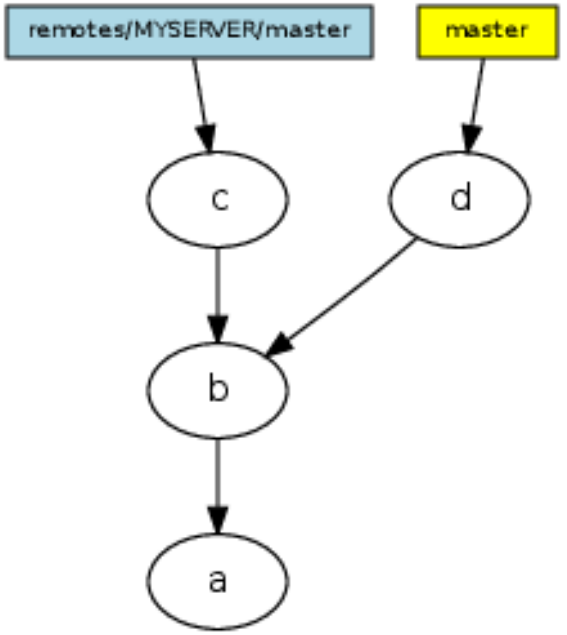
```
→ cat .git/HEAD
ref: refs/heads/master
```

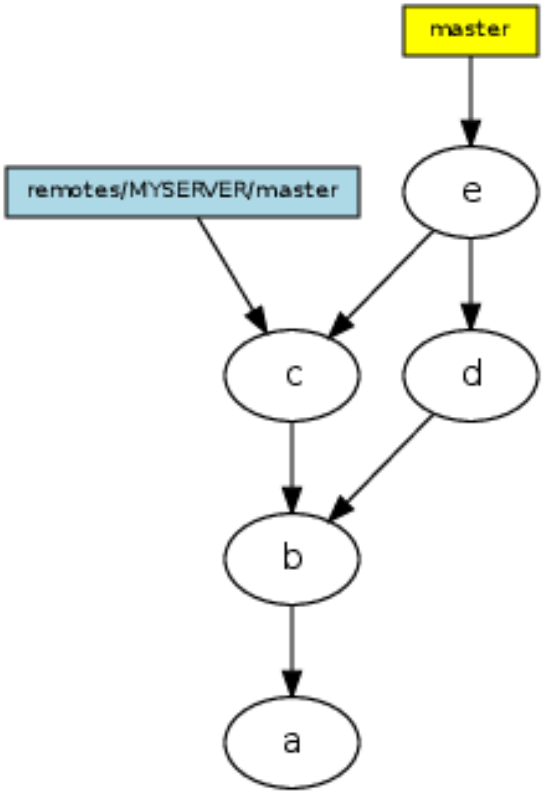
# Referenzen - Remotes

- Referenziert den letzten gesehen Commit eines Remote Repositories
- Vergleichbar mit Branches
- Können nicht ausgechecked werden
- .git/refs/remotes

# Beispiel: Merge



# Zusammenfassung

- „Collection of Objekts and a system for naming those objects“ (Berglund)
- Repository als DAG von Commits
- Jeder Commit verweist auf Baum
- Referenzen zum benennen und ordnen

# Quellen


Tim Berglund (2013): Git From the Bits Up, Talk at JAXConf 4-5.6.2013, (55 Min).

Online: http://www.youtube.com/watch?v=MYP56QJpDr4&hd=1.

Scott Chacon (2009): Git Internals, in: Pro Git, Apress.

Online: http://git-scm.com/book, Chapter 9.

Sitaram Chamarty: Git Concepts Simplified. Online: gitolite.com/gcs.

Chris Johnsen (2011): Packfiles, Stack Overflow.

Online: http://stackoverflow.com/a/5576688

Matthew McCullough (2012): Advanced Git: Graphs, Hashes, and Compression, Oh My, Talk at The San Francisco Java User Group, 9.11.2012 (70 Min).

Online: http://www.youtube.com/watch?v=ig5E8CcdM9g&hd=1.

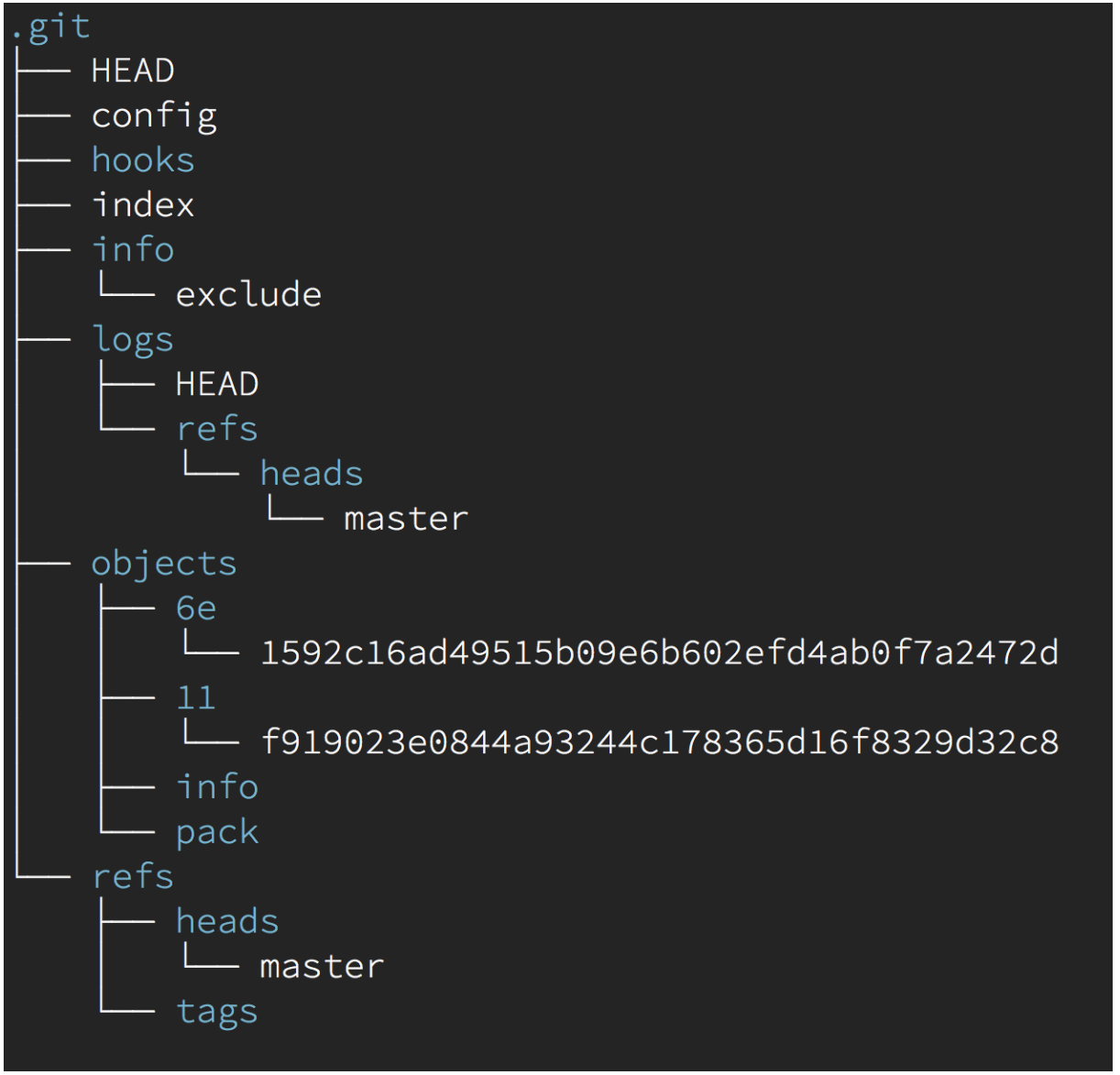
Tommi Virtanen: Git for Computer Scientists.

Online:eagain.net/articles/git-for-computer-scientists.

Fragen?

# .git Ordner

.git Ordner



# Objektspeicherung - Packfiles

- Bisherige Objektspeicherung: „loose object format“
- „pack file format“
  - Optimiert auf Speichereffizienz und Zugriffsgeschwindigkeit
- z.B. werden mehrere ähnliche Blobs zu Deltas zusammen gefasst
- Wird periodisch ausgeführt, sowie bei **git push** und durch **git gc**

# Beispiel - Commit

- **git commit**
- Commit Objekt erstellen
- Aktuellen HEAD als Parent eintragen
- Neuen Commit in Branch Referenz (head) eintragen