redhat.

01 Gitshop

Git - object model

Jiří Jabůrek (jjaburek)
Red Hat 2012

Preprocess part – intro

- Git is git, no git vs xyz
- Git = rich suite, won't show everything
- Git 1.5.x ; 1.6+ = ?
- Porcelain vs plumbing
- The dash in git commands (git-log)
- Strict argument parsing order
  - POSIXLY_CORRECT / POSIX_ME_HARDER

```
git cmd -a arg -b –long-c arg <nonopt> [nonopt] -- file1 file2
```

- Target audience: git users

Git – what is it?

- The {British,American} English dictionary:

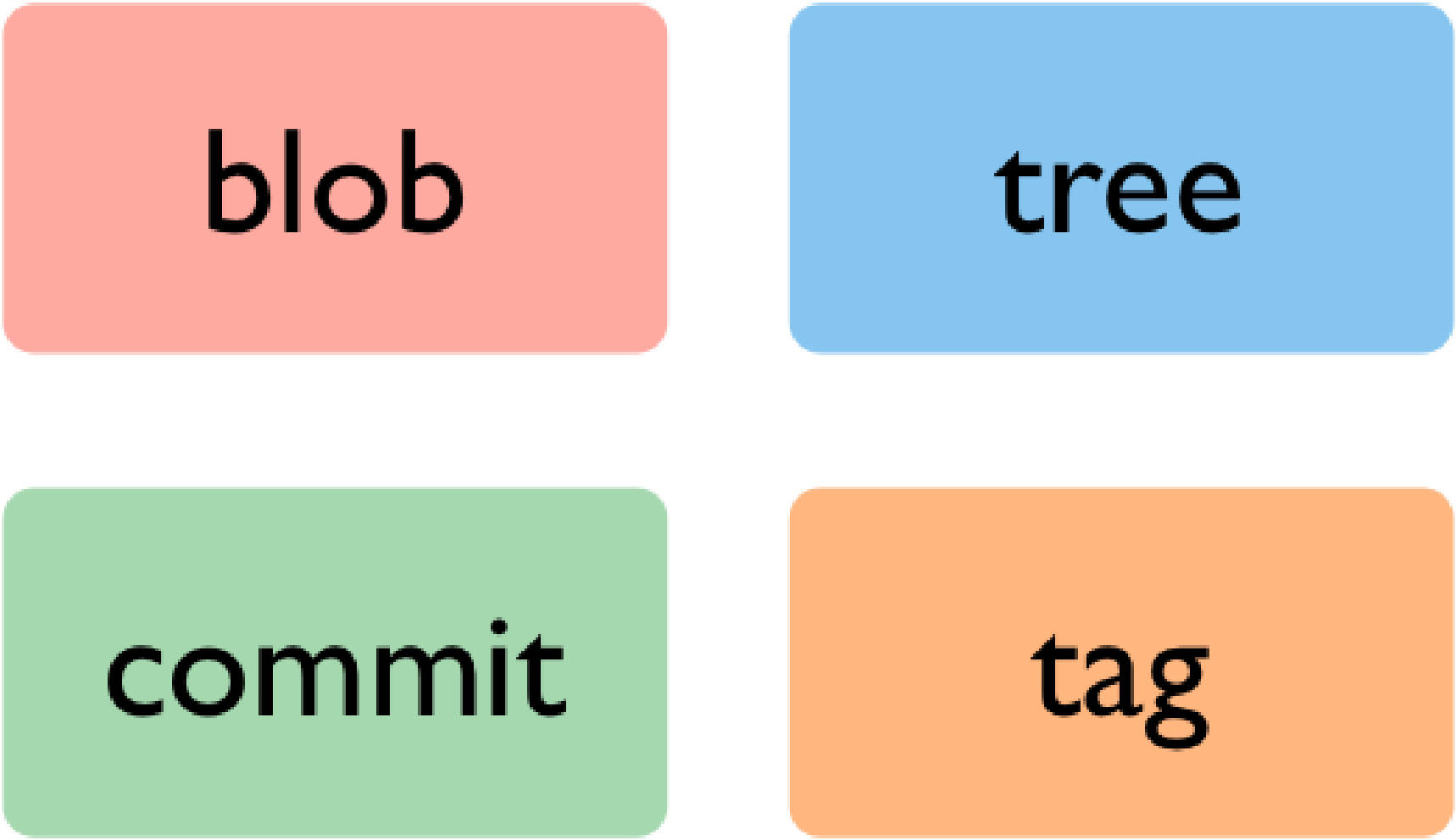
```
noun /gɪt/ [C] UK informal
    a person, especially a man, who is stupid or unpleasant
```

Architecture overview

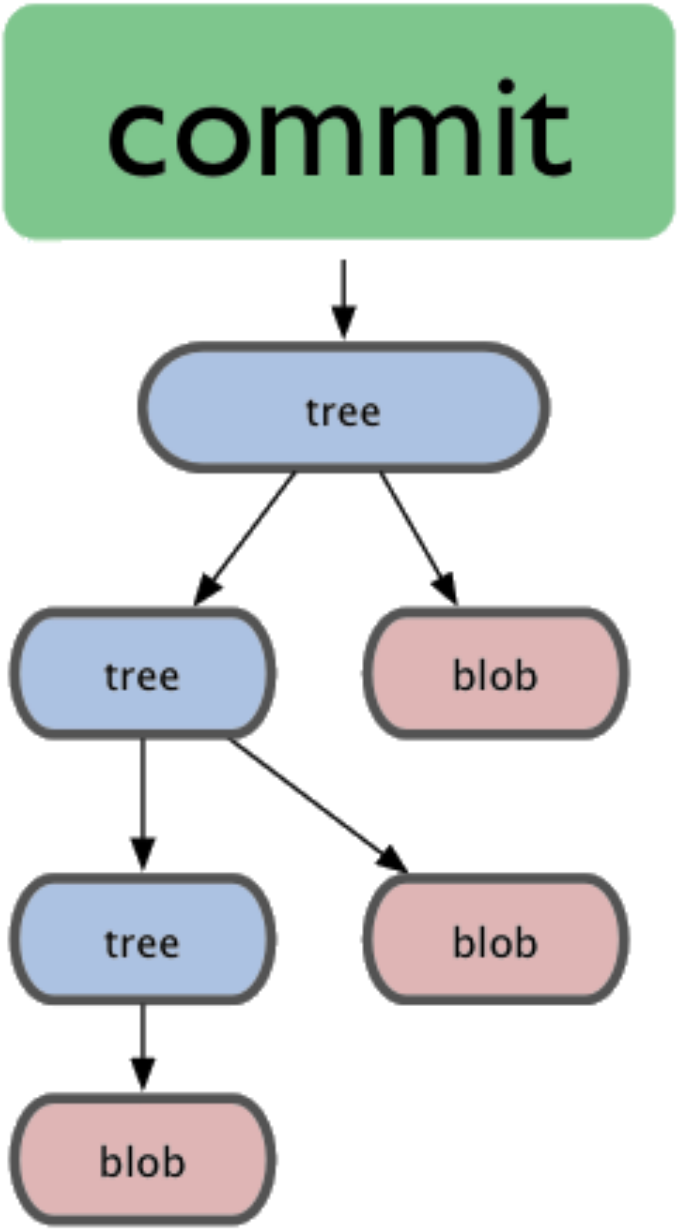
- Objects
  - store data, metadata
  - form the Git object database
  - ex. blob, tree, commit, tag
  - are identified by SHA1 hash
  - no dependency upon / reference --> dangling object
- References
  - provide a way to reach objects
  - ex. branch, tag, symbolic refs (HEAD)
- Many simple ideas form a complex system

Objects

Objects

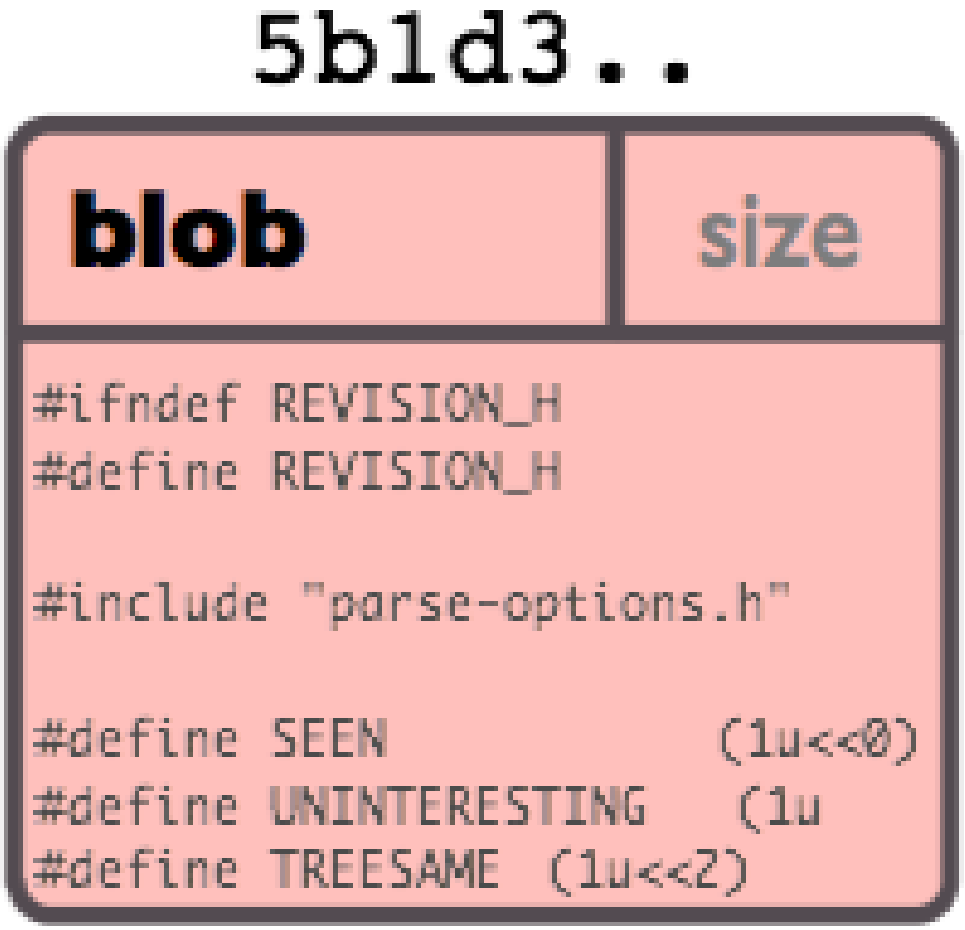


Objects - basic overview



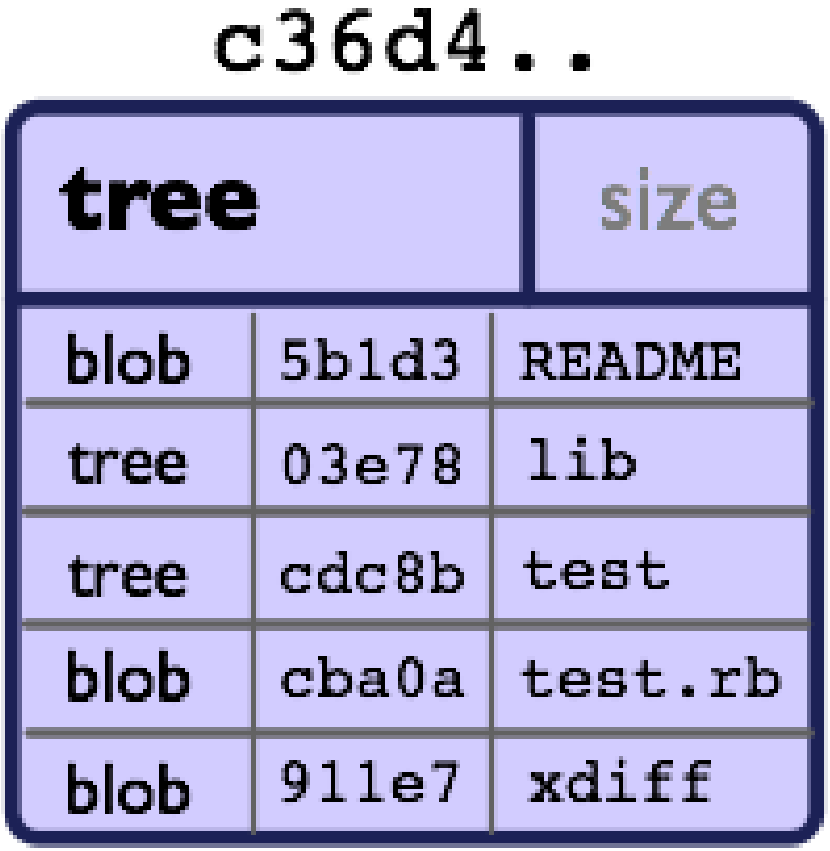
Objects - blob

- Stores regular file contents



Objects - tree

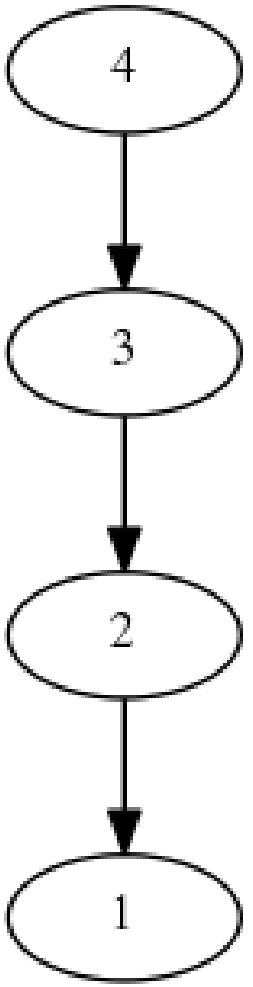
- Hierarchical structure of blobs and (sub)trees



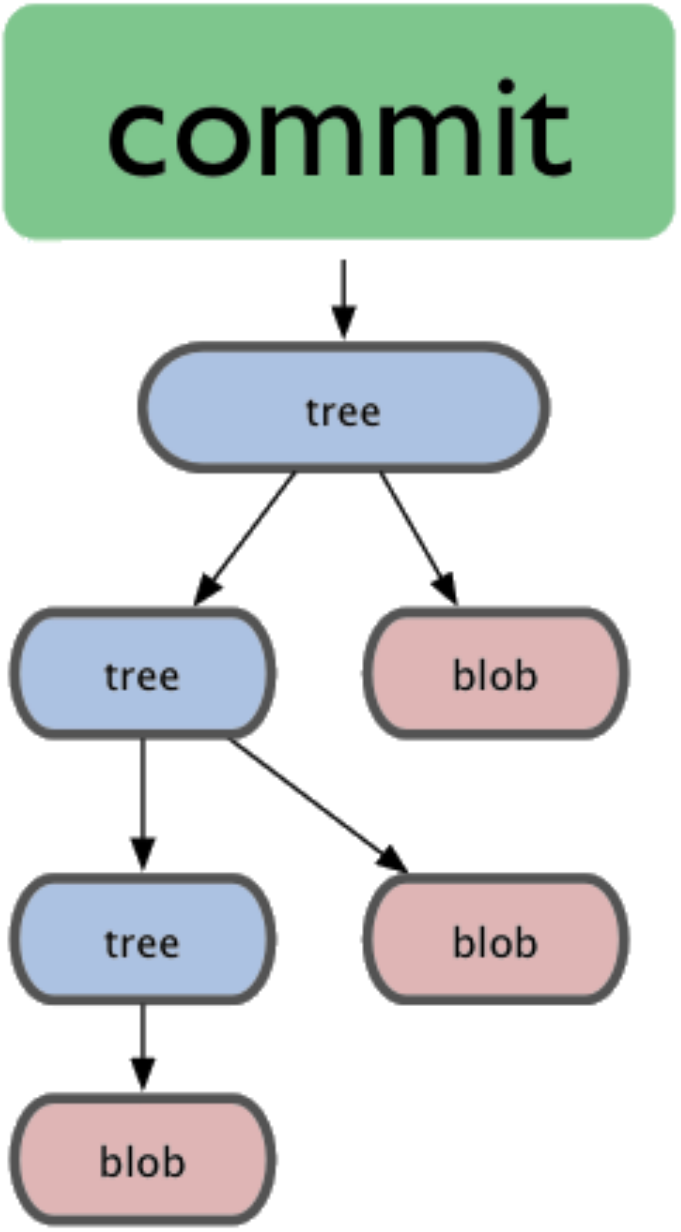
+ file modes

Objects - commit

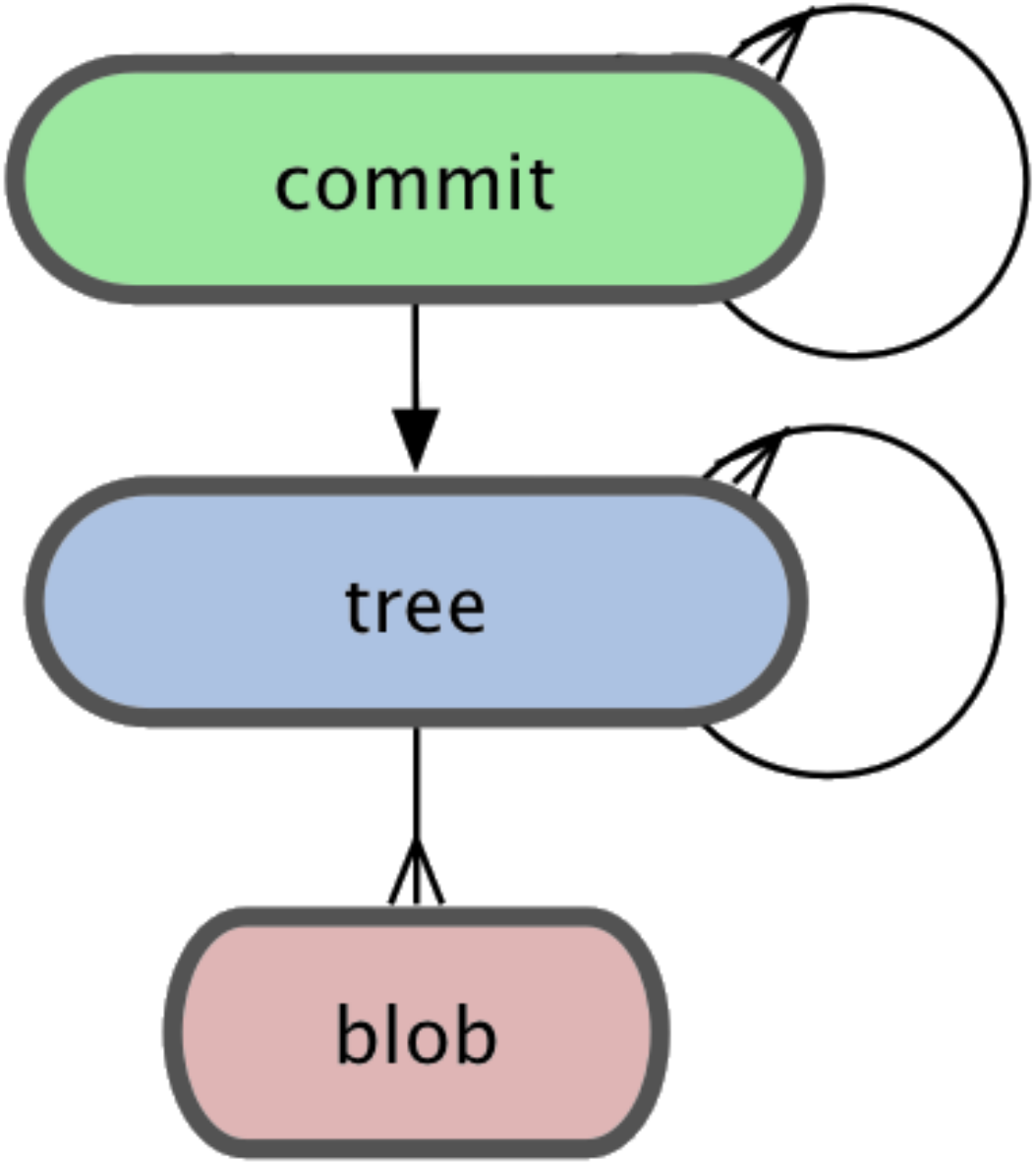
- Forms project history, holds commit msg



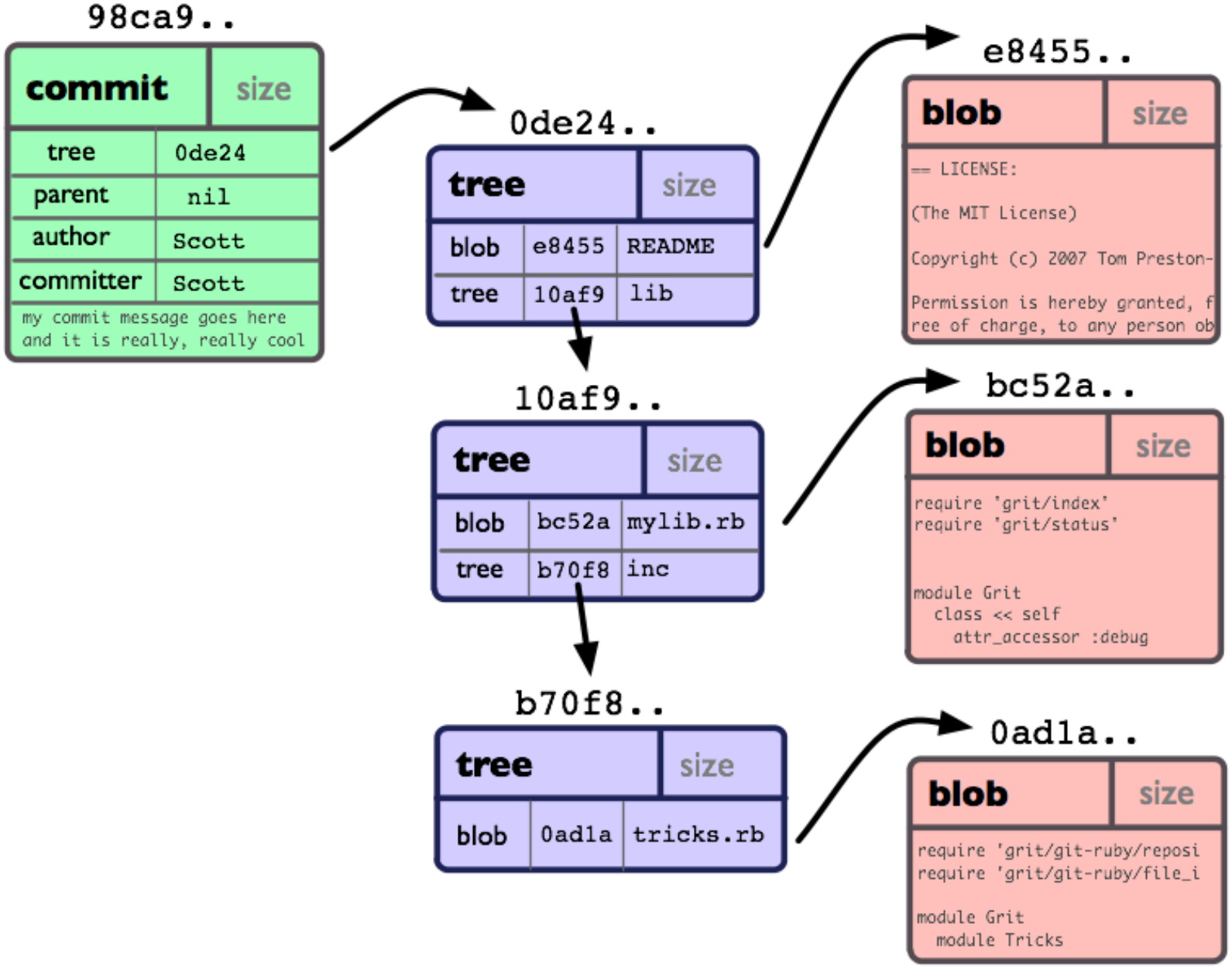
Objects - basic overview



Objects - basic overview



Objects - basic overview

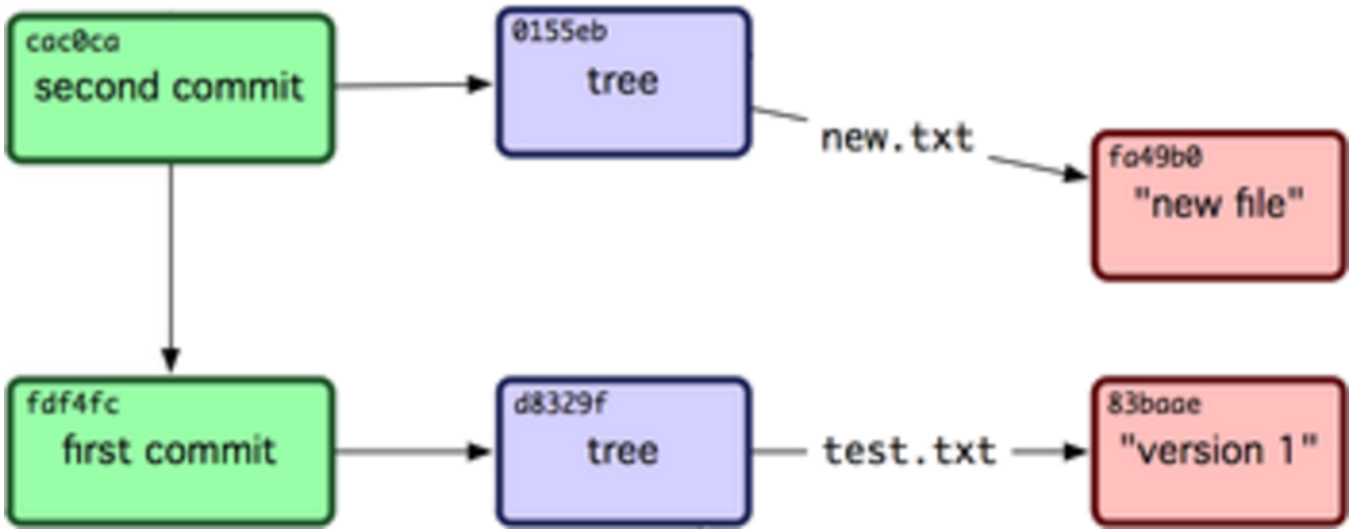


Objects - commit history

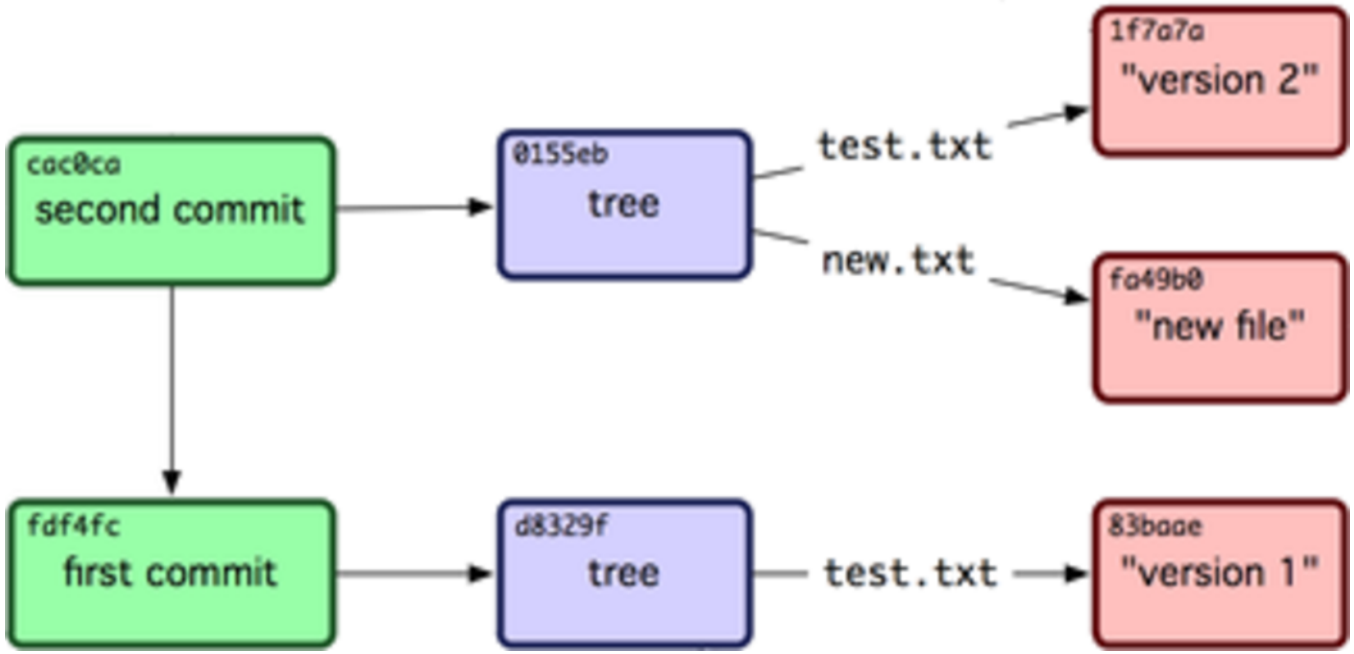
Objects - commit history

Objects - commit history

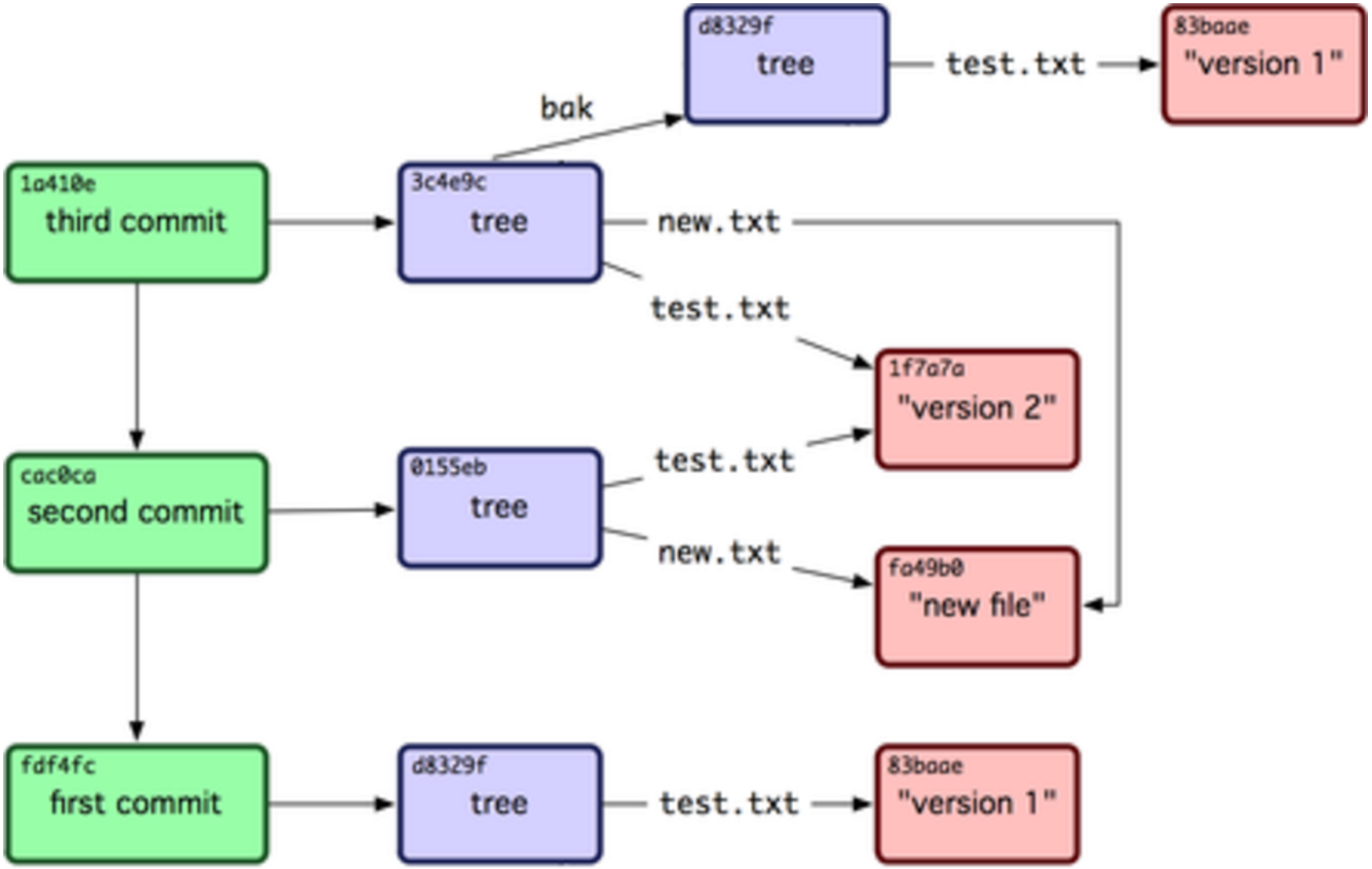
Objects - commit history



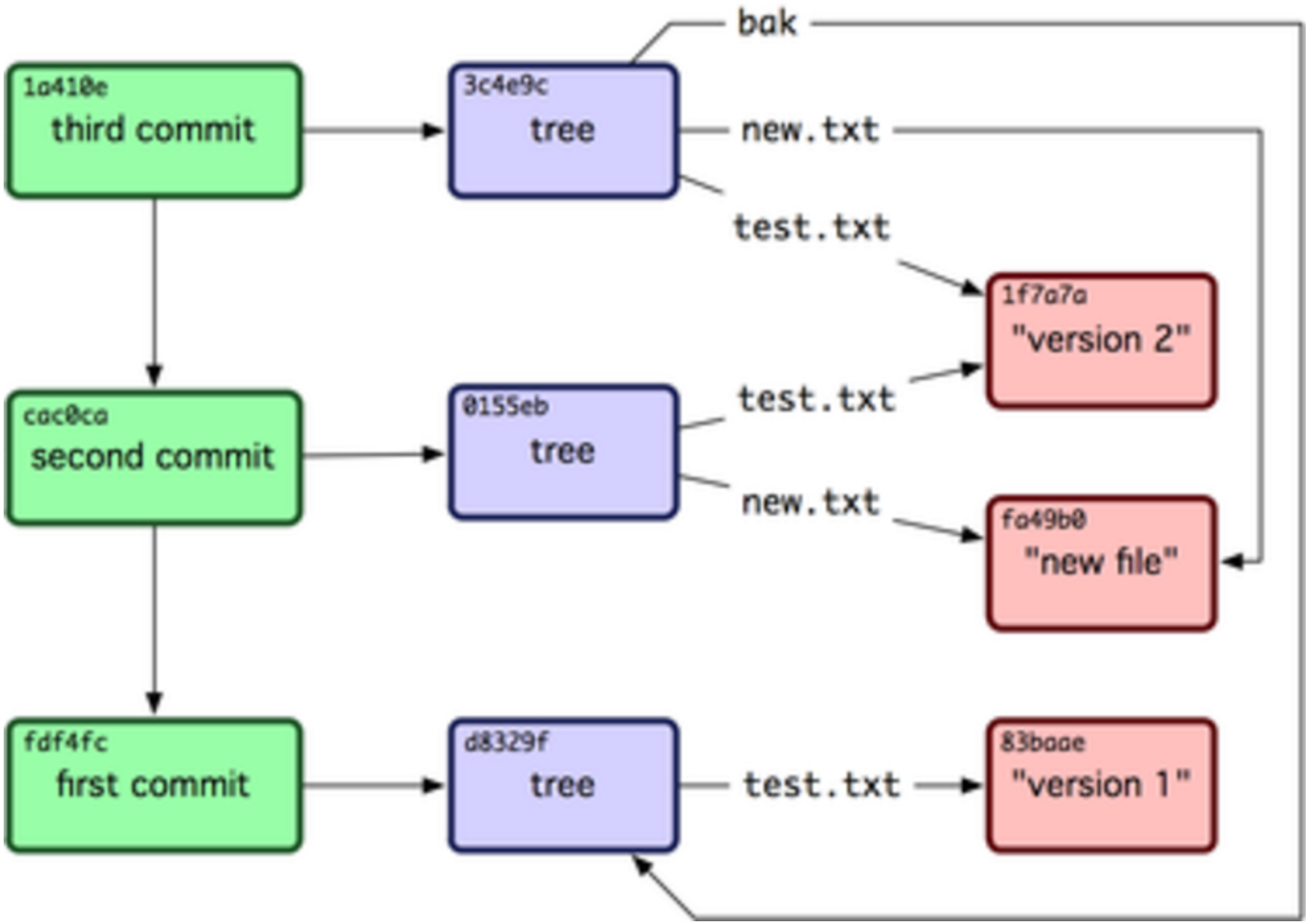
Objects - commit history



Objects - commit history



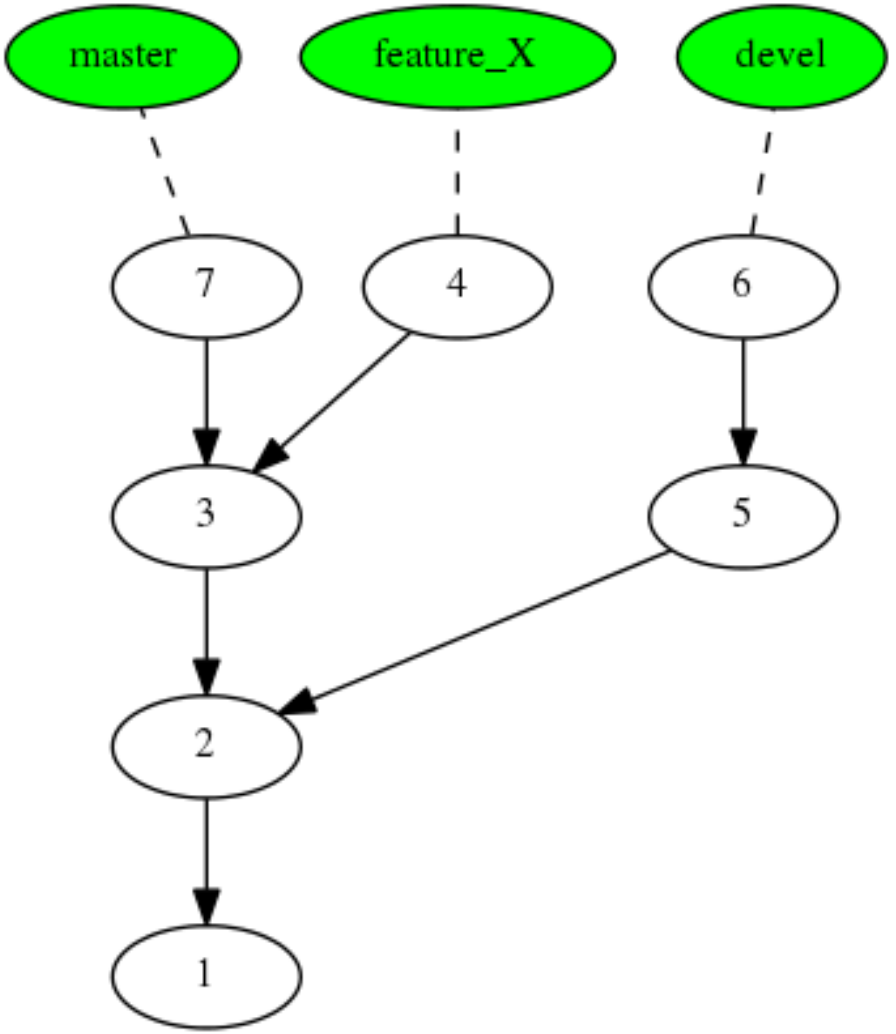
Objects - commit history



References

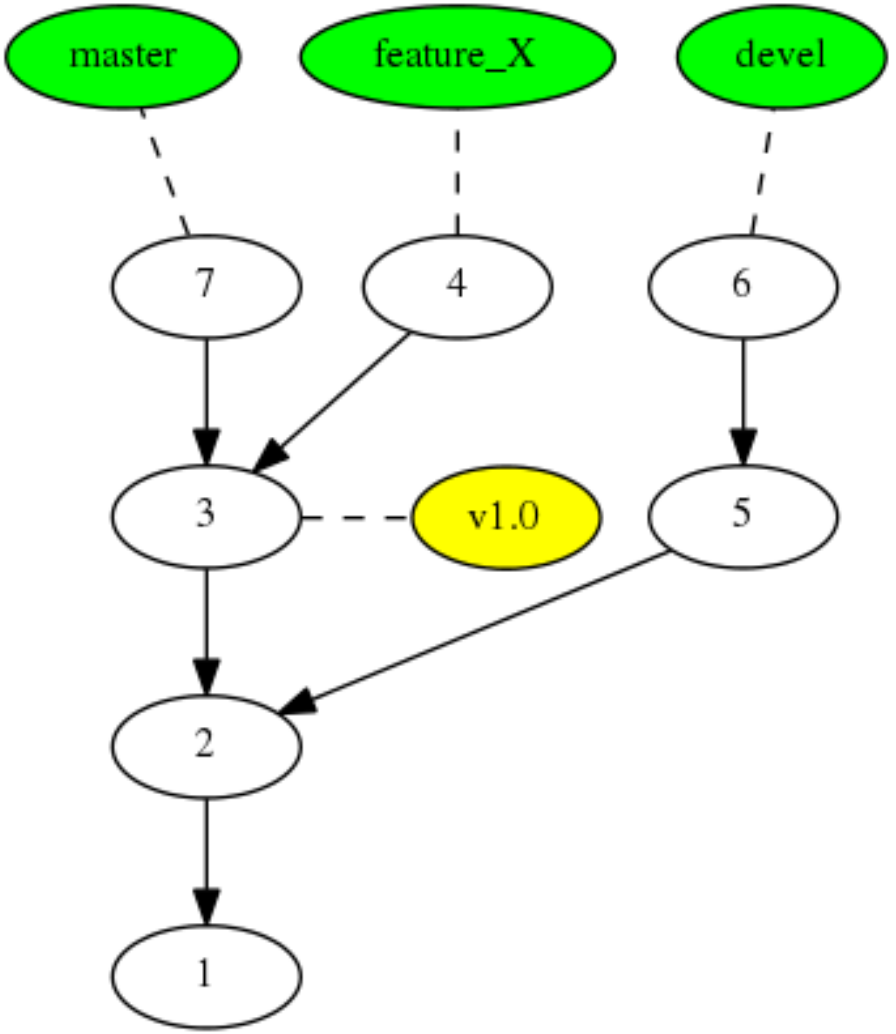
References - branch

- Commit "pointer", moves with new commit



References - tag

- Commit "pointer", stays in place

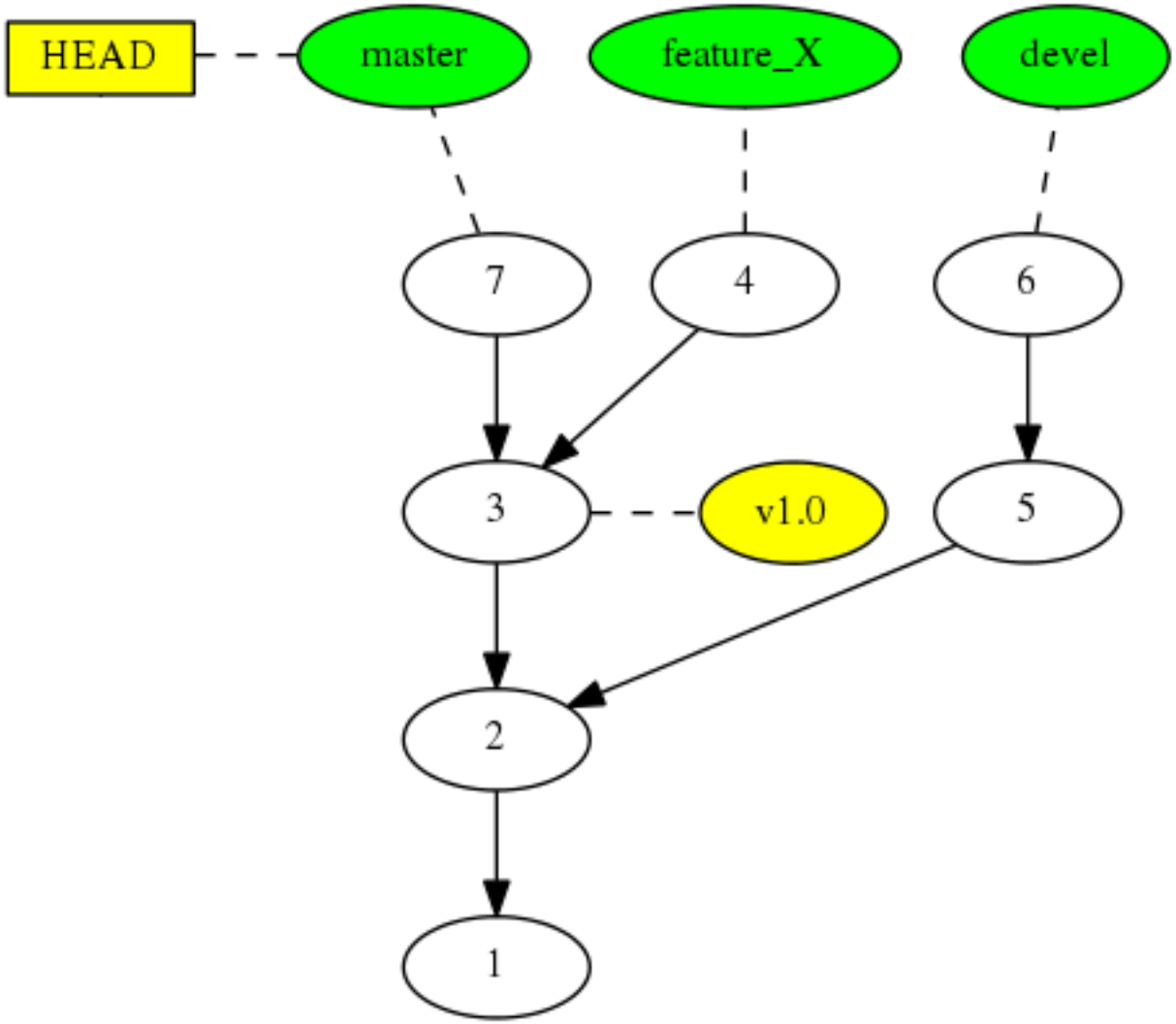


Objects - tag

- Depends on commits, holds additional info

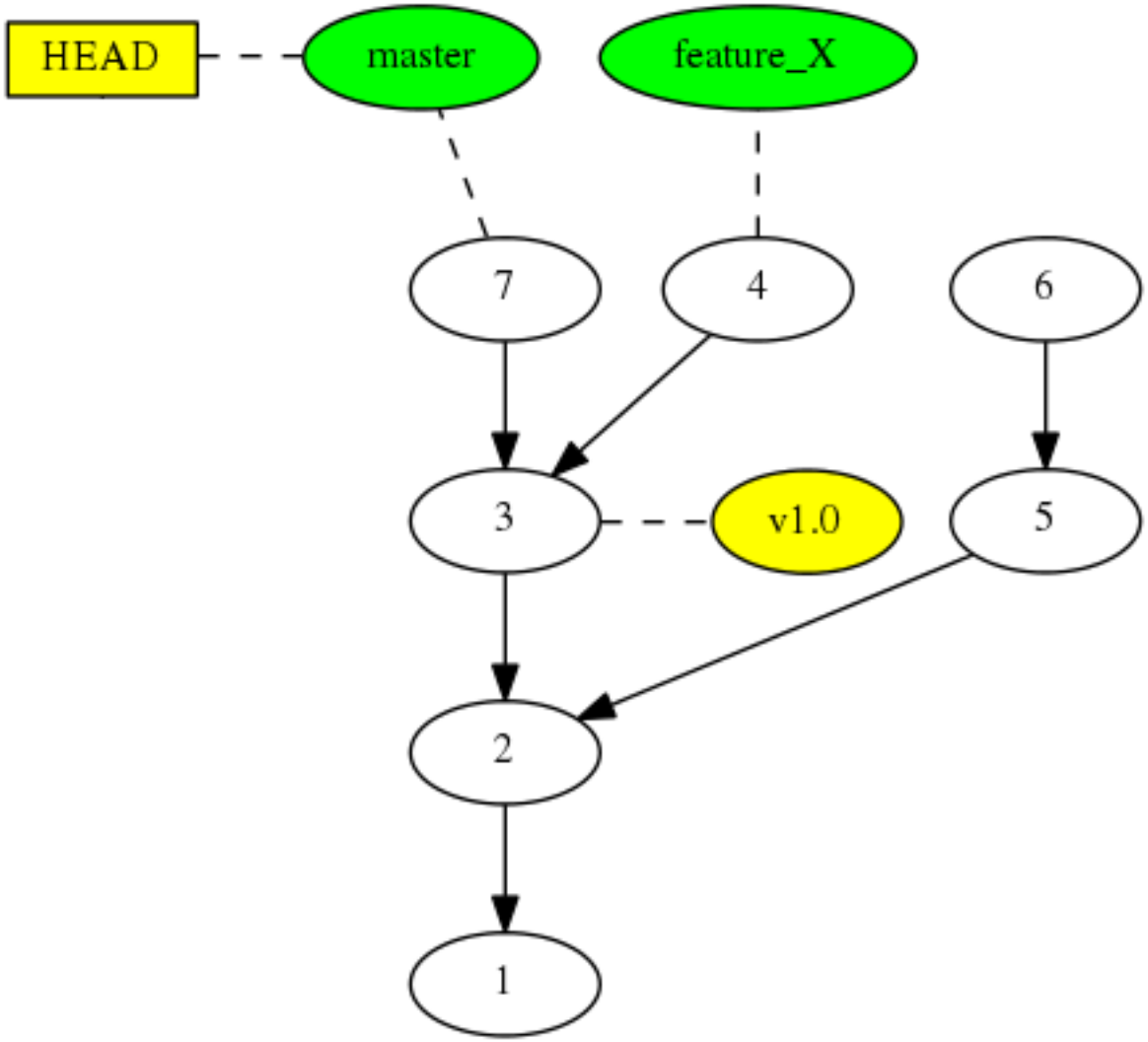
References - symbolic refs

- Misc references used for special purposes



Dangling objects

- Objects without ref / dependency upon



The .git/ directory structure (subset only)

The .git/ directory structure

```
COMMIT_EDITMSG    hooks/
FETCH_HEAD        index
HEAD              info/
ORIG_HEAD         logs/
branches/         objects/
config            packed-refs
description       refs/
```

```
refs/
refs/heads
refs/heads/branch1
refs/heads/branch2
refs/remotes
refs/remotes/origin
refs/remotes/origin/hisbranch1
refs/tags
refs/tags/sometag
```

```
objects/
objects/info
objects/info/packs
objects/eb
objects/eb/56dc85d2336d927252909f21949d2734041889
objects/94
objects/94/a14c0f90ffcaed7a583aef409ea8a33093011d
objects/94/c2d5c38364c817d79759c2a35ff1fb1693440f
objects/f9
objects/f9/350dab48335cbcc81d9a3ea46c3d121dcf1808
objects/pack
objects/pack/pack-e7e74fef42f3583516bf887e0ddd559224fbb665.pack
objects/pack/pack-e7e74fef42f3583516bf887e0ddd559224fbb665.idx
```

You now understand Git

Presentation: Questions?

Workshop >>

Workshop

goal:

```
file1
dir1/file2
dir1/file3
```

file1 = file2 = same blob

setup:

```
mkdir testme; cd testme
git init
echo cont1 > /tmp/gitshop1
echo cont2 > /tmp/gitshop2
```

- git hash-object -w <file>
- git mktree
  - format: man git-mktree says man git-ls-tree:
    - <mode> SP <type> SP <object> TAB <file>
      - where "SP" = separator = space
      - <mode> = UNIX mode, see "man 2 stat"
        - first two digits: 10 = regular file, 04 = dir
        - last four digits: well-known filemode, 0000 for default
      - <file> = desired filename in the tree object
- git commit-tree <root_tree_hash>
  - enter commit msg (multiline), then Control-D
- git log --stat --format=raw <commit_hash>
- git checkout -f <commit_hash>

The End

Thanks for listening

Links

- Git-scm.com - docs & reference manual

```
http://git-scm.com/documentation
```

- Old "git book", now only @ github

```
https://github.com/schacon/gitbook
```

- "Git for computer scientists" + rework

```
http://eagain.net/articles/git-for-computer-scientists/
http://sitaramc.github.com/gcs/
```