LONDON
INTERNATIONAL
SOFTWARE DEVELOPMENT
CONFERENCE 2015

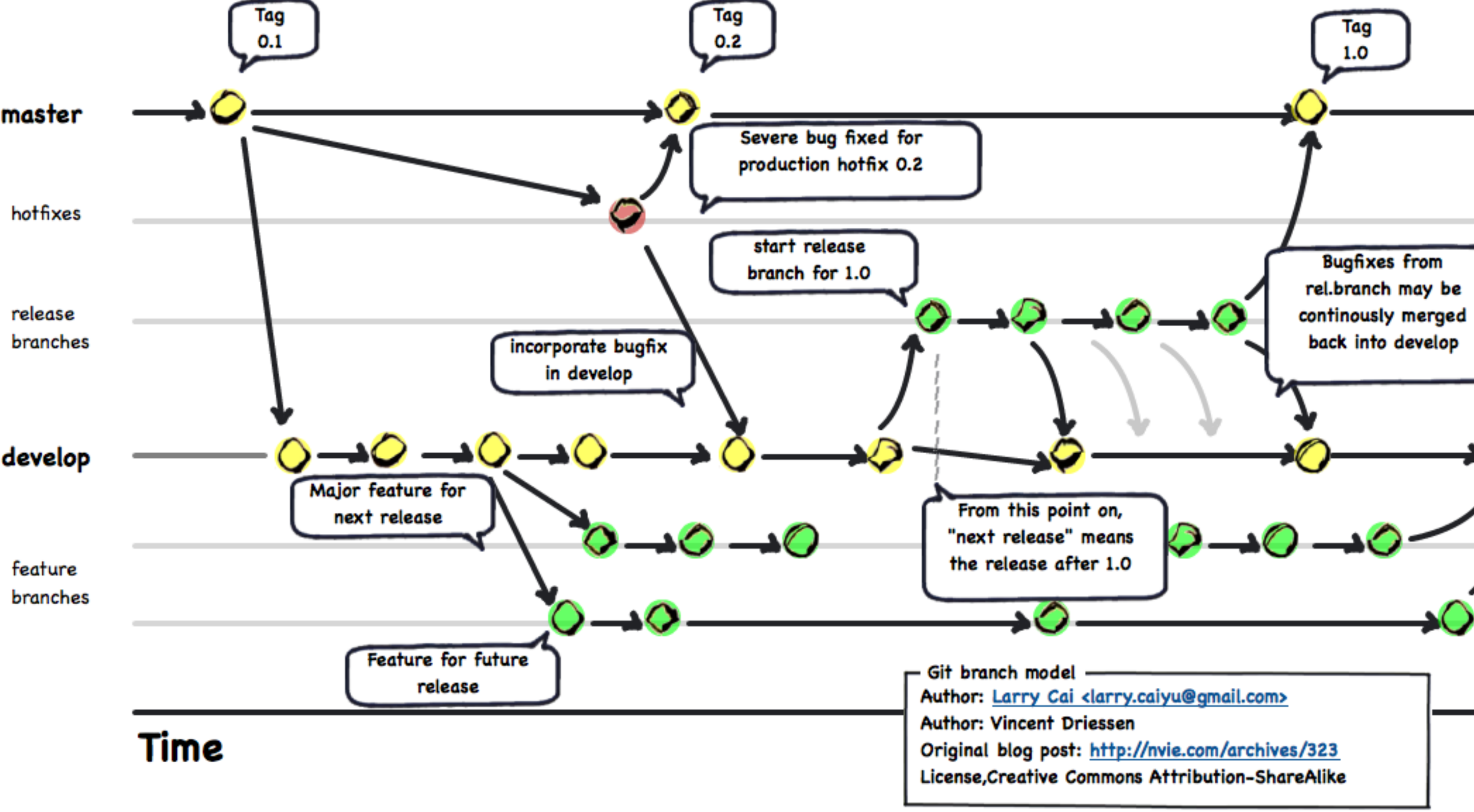
goto; conference

Understanding git

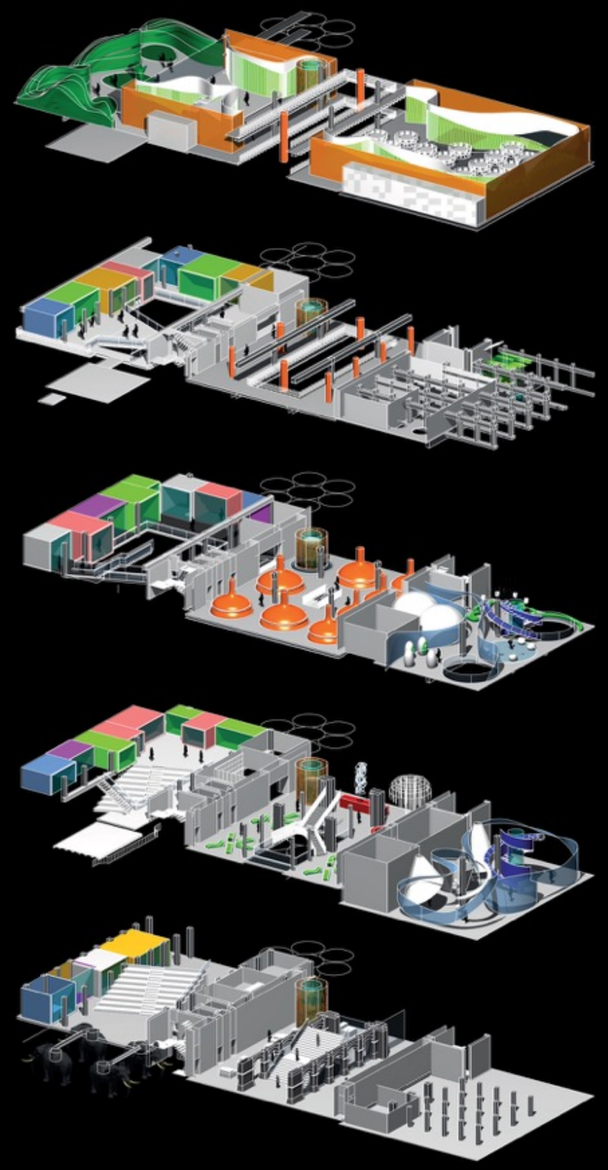
Steve Smith
@tarkasteve

“It's easy, it's just a directed acyclic graph!”
If I hear that one more time I may have to punch something!

Emma Jane Hogbin Westby, Git-Merge 2015


Tag 0.1
Tag 0.2
Tag 1.0
master
hotfixes
release branches
develop
feature branches
Severe bug fixed for production hotfix 0.2
start release branch for 1.0
Bugfixes from rel.branch may be continously merged back into develop
incorporate bugfix in develop
Major feature for next release
From this point on, "next release" means the release after 1.0
Feature for future release
Time
Git branch model
Author: Larry Cai <larry.caiyu@gmail.com>
Author: Vincent Driessen
Original blog post: http://nvie.com/archives/323
License,Creative Commons Attribution-ShareAlike

git internals

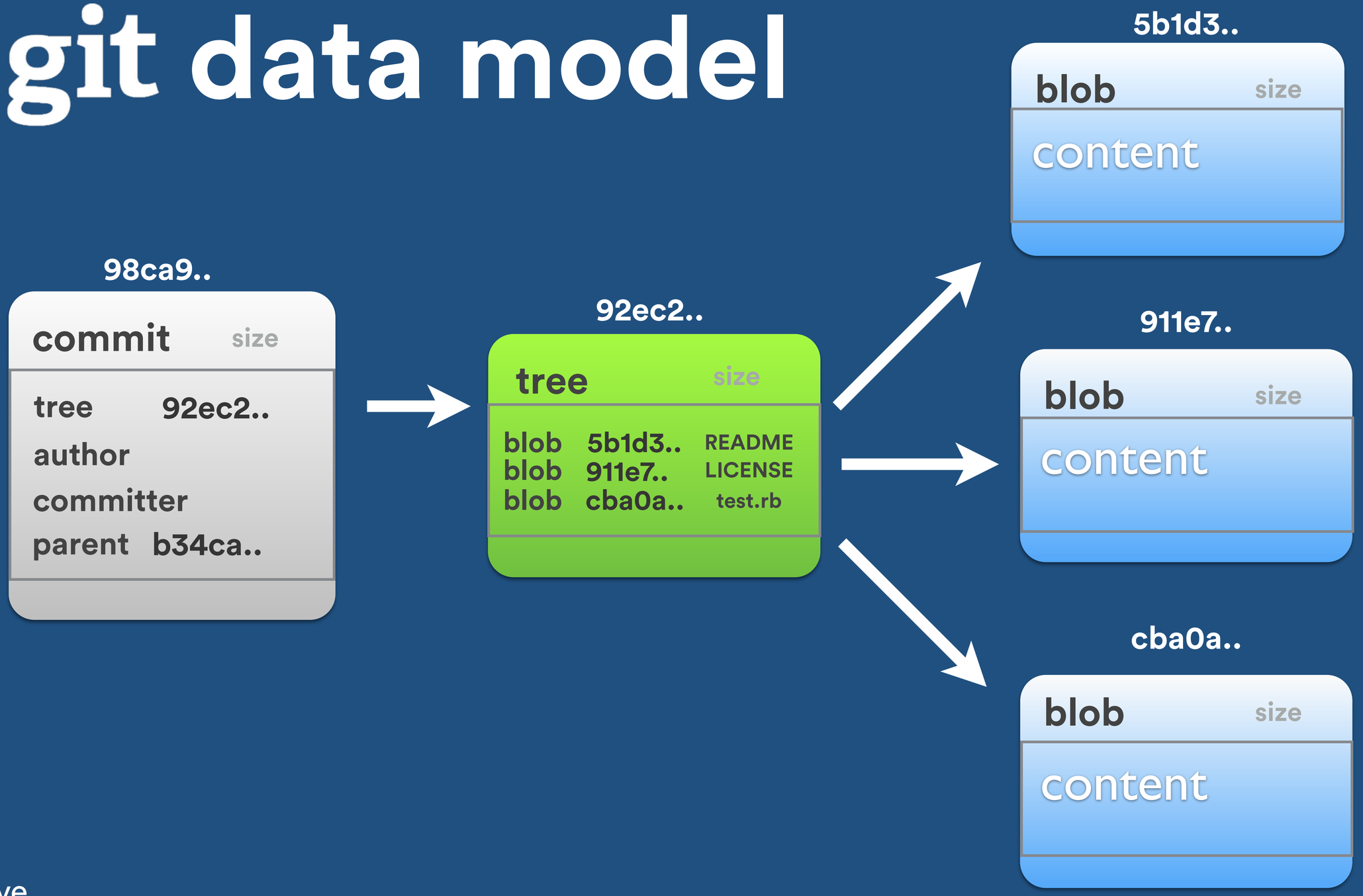


@tarkasteve

Carlsberg


COLONIA
AIR CONDITIONED
HORROR

git data model



@tarkasteve

```
$> git init
$> tree .git/objects
.git/objects
├── info
└── pack

2 directories
```

```
$> touch some-file.txt
$> git add some-file.txt
```

```
$> tree .git/objects
.git/objects
├── e6
│   └── 9de29bb2d1d6434b8b29ae775ad8c2e48c5391
├── info
└── pack

3 directories, 1 file
```

zlib compressed
SHA1

```
$> git commit -m "First commit"
```

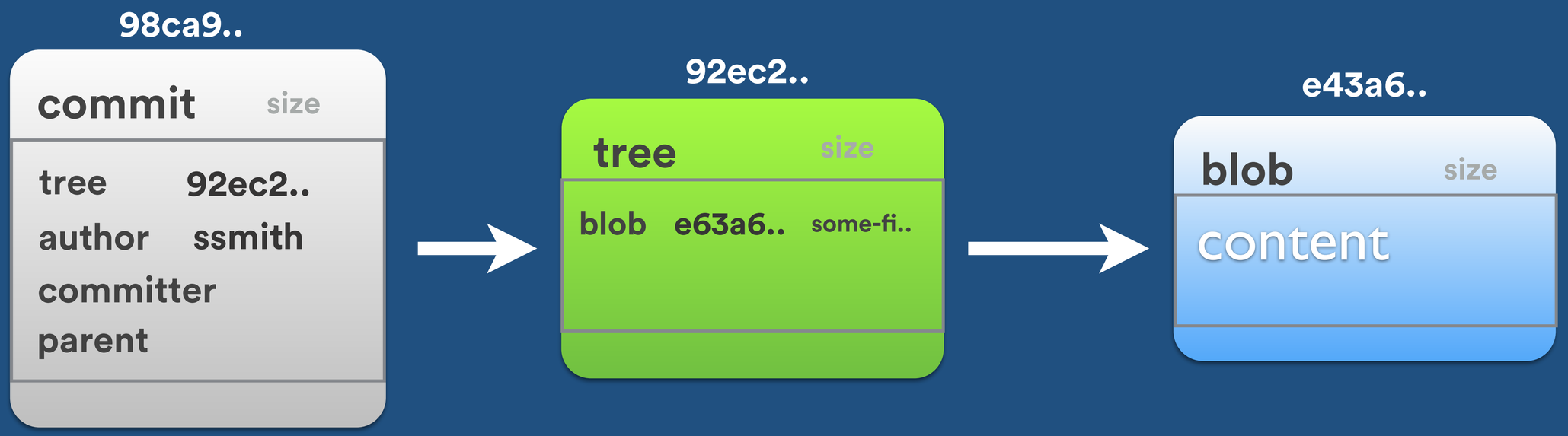
```
$> tree .git/objects
.git/objects
├── 13
│   └── 1e360ae1a0c08acd18182c6160af6a83e0d22f
├── 31
│   └── 995f2d03aa31ee97ee2e814c9f0b0ffd814316
├── e4
│   └── 3a6ac59164adadac854d591001bbb10086f37d
├── info
└── pack

5 directories, 3 files
```

Commit

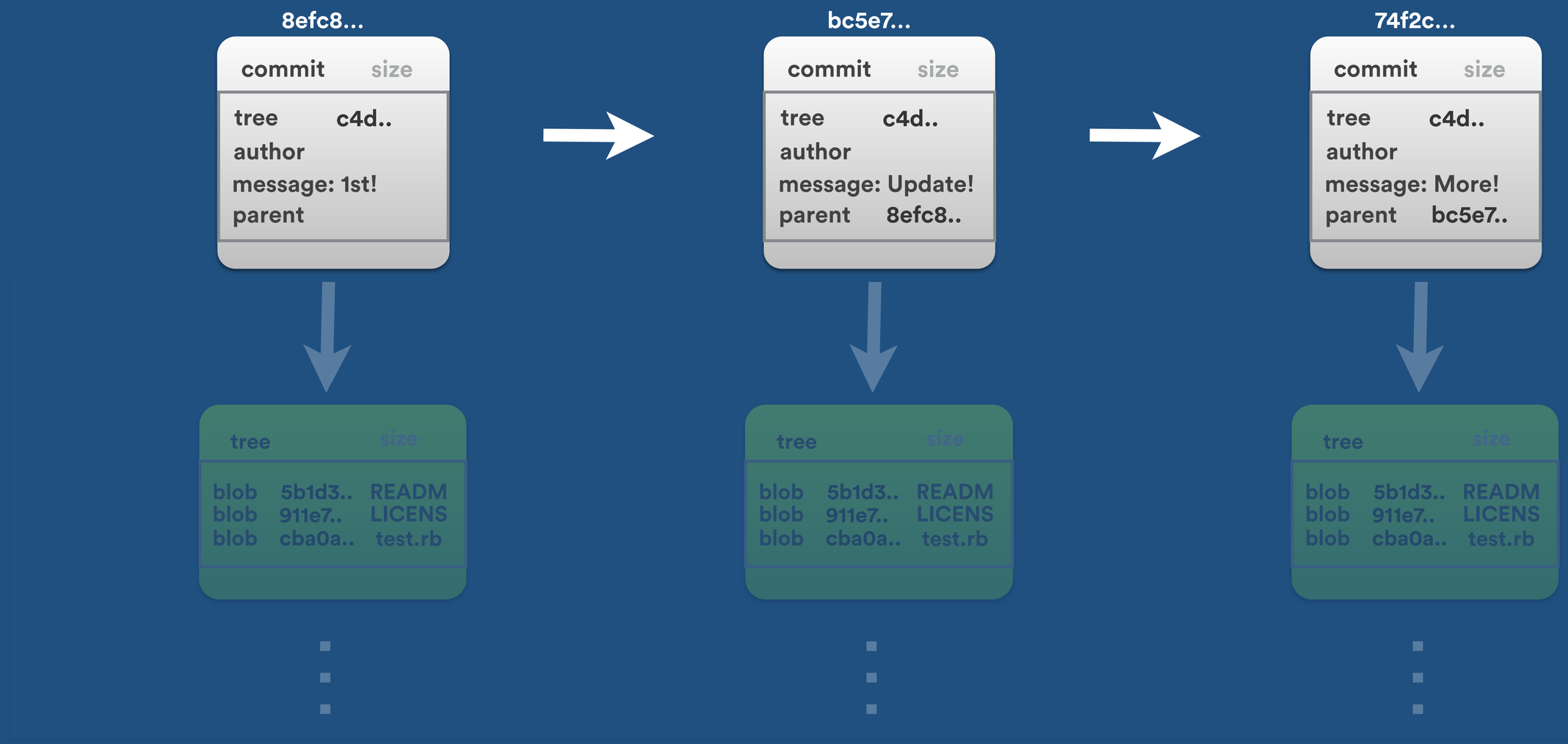
Tree

Blob

git data model



git data model



@tarkasteve

```
$> echo "// Comment" >> some-file.txt
$> git add some-file.txt
```

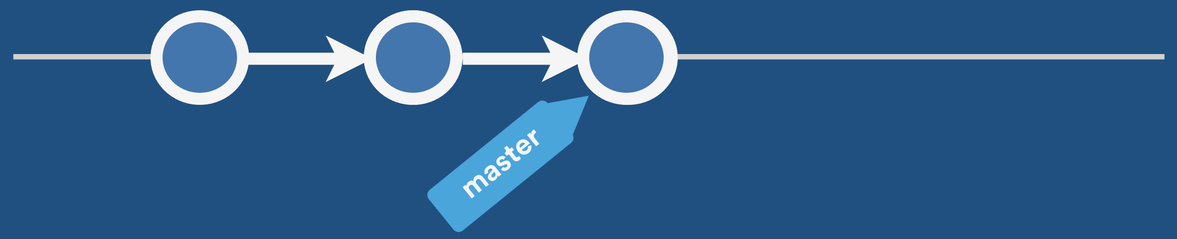
```
$> tree .git/objects
.git/objects
├── 13
│   └── 1e360ae1a0c08acd18182c6160af6a83e0d22f
├── 31
│   └── 995f2d03aa31ee97ee2e814c9f0b0ffd814316
├── c1
│   └── 9e6823e34980033917b6427f3e245ce2102e6e
├── e4
│   └── 3a6ac59164adadac854d591001bbb10086f37d

6 directories, 4 files
```

Entirely new BLOB

Refs and Branches And Tags

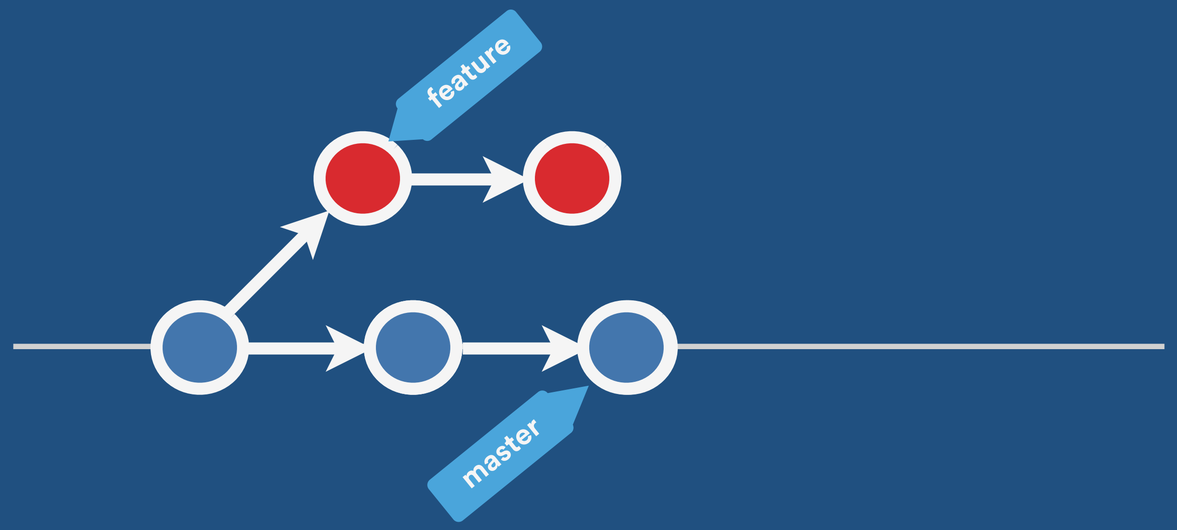
What is a ‘ref’?

A ref is just a pointer to an object



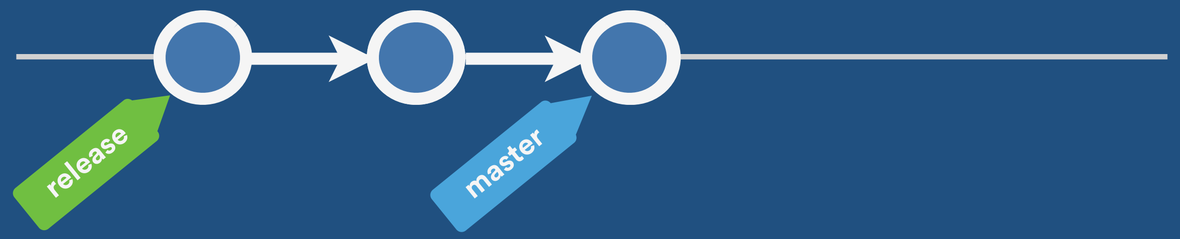
What is a ‘branch’?

A branch is divergence from a common point, and ref to a commit, the “HEAD”



What is a ‘tag’?

A tag is just special ref used a mark
a commit in the history

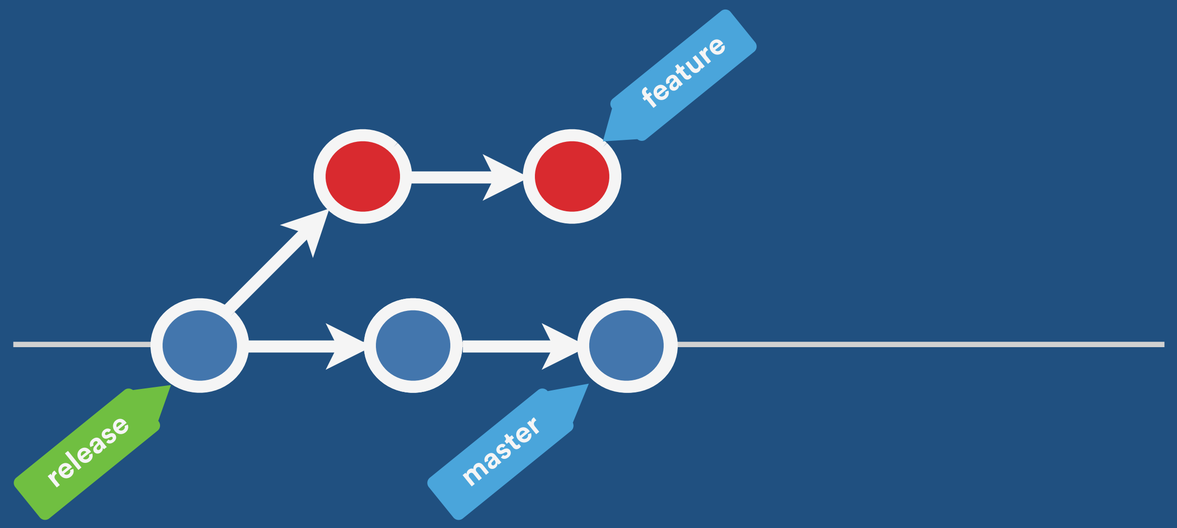


```
$> git tag a-tag -m”A tag”
$> git branch a-branch

$> tree .git/refs/
.git/refs/
├── heads
|   ├── a-branch
|   └── master
└── tags
    └── a-tag

$> cat .git/refs/heads/a-branch
c13e27cdfd15c5acdcd8b510eefed7be68c41c8e
```

What is a ‘reset’?

Manipulates the branch ‘head’

```
$> git reset --hard feature^          # ‘^’ means ‘parent’
```



Reflog keeps a history

```
$> git reflog
0c35628 HEAD@{1}: reset: moving to HEAD^
6cc6637 HEAD@{2}: commit: Add B
0c35628 HEAD@{3}: merge: Merge made by the 'recursive' strategy.
e0c0d65 HEAD@{4}: cherry-pick: A
80bb854 HEAD@{5}: checkout: moving from alpha to master
5044136 HEAD@{6}: commit: A
80bb854 HEAD@{7}: checkout: moving from master to alpha
80bb854 HEAD@{8}: commit (initial): 1
```

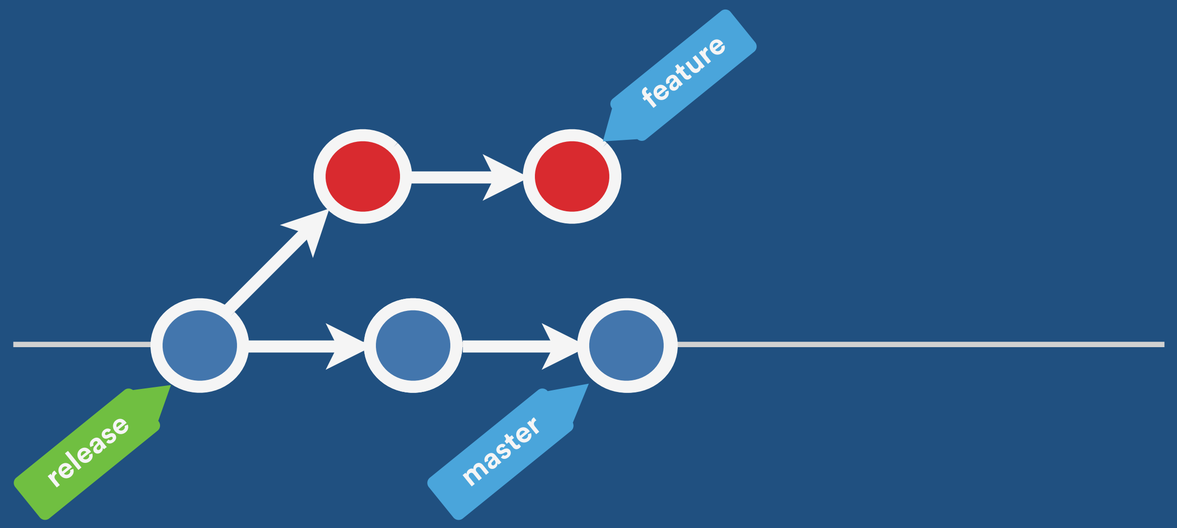
(Only 90 days by default though!)

Keeping things *clean*

What is a ‘gc’?

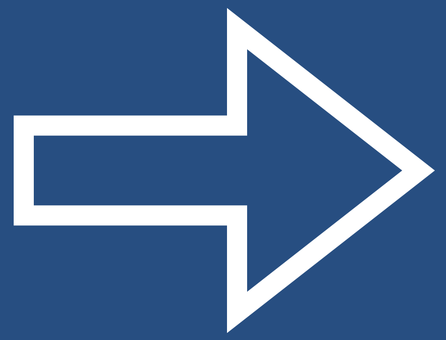
Orphaned objects are eligible for removal

```
$> git reset feature^                    # ‘^’ means ‘parent’
$> git gc --prune=all
```



GC also packs objects

Loose Objects



Packfile

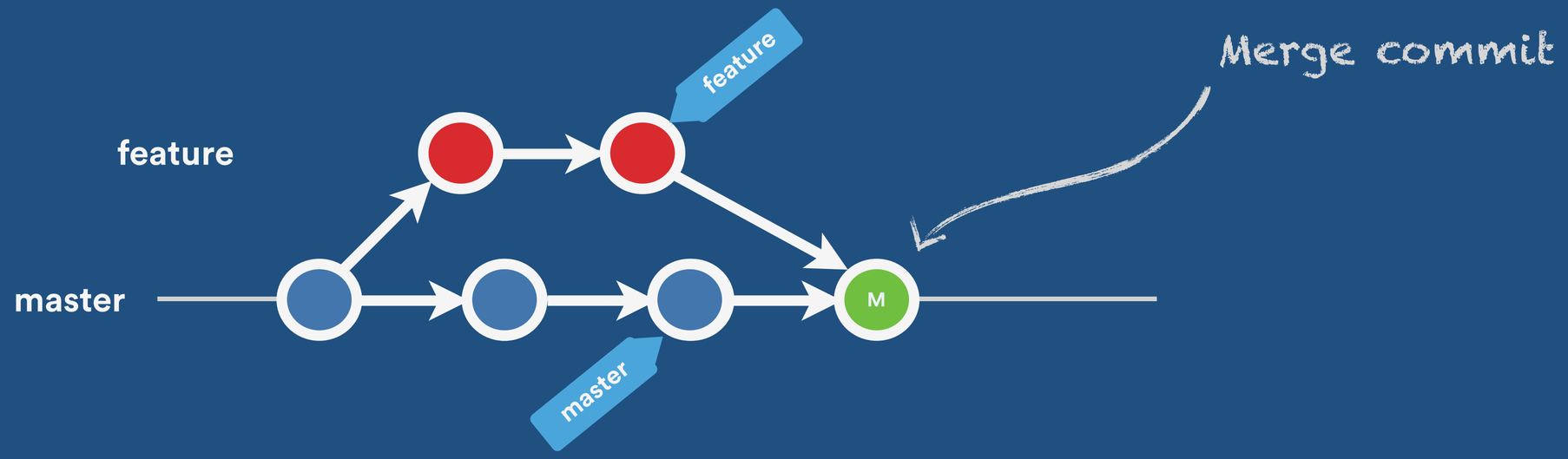
1. zlib compressed
2. Delta encoded

```
$> tree .git/objects
.git/objects
├── info
|   └── packs
└── pack
    ├── pack-7475314b451a882d77b1535d215def8bad0f4306.idx
    └── pack-7475314b451a882d77b1535d215def8bad0f4306.pack

2 directories, 3 files
```

git merge

What is a merge?

merges keep the context of
the feature's commits

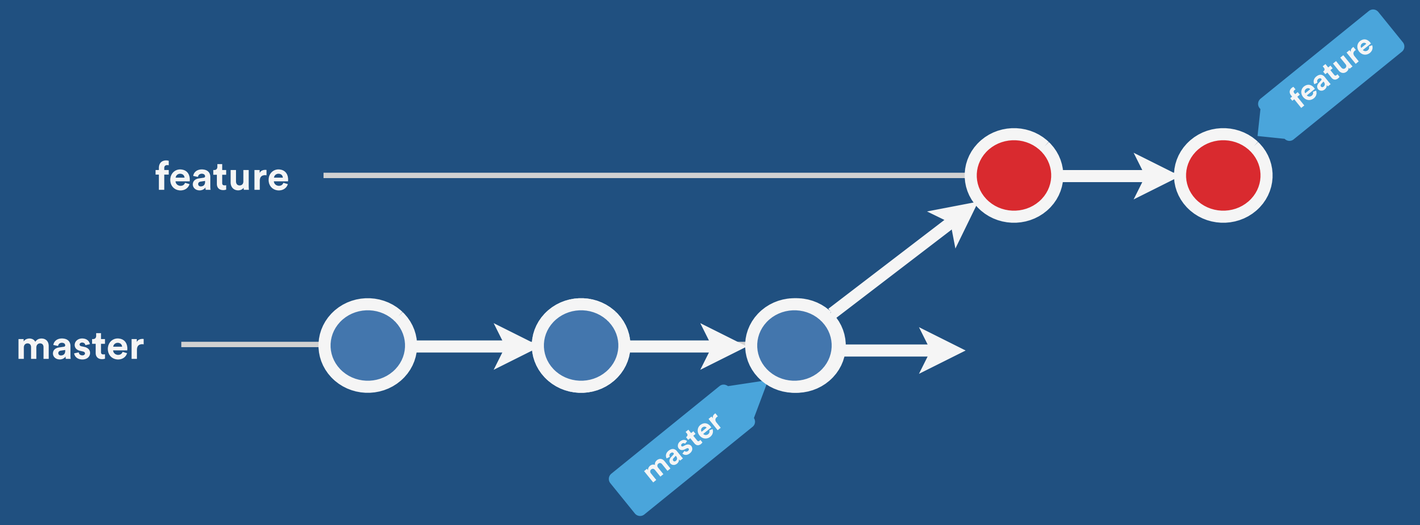


Anatomy of a merge

```
$> git cat-file 3680d8c8fd182f97cb0e75045e2fed5c7b7613ed
tree f362c42032aff677c1a09c3f070454df5b411239
parent 49a906f5722ad446a131778cea52e3fda331b706
parent bd1174cd0f30fe9be9efdd41dcd56256340f230e
author Marcus Bertrand <mbertrand@atlassian.com> 1409002123 -0700
committer Marcus Bertrand <mbertrand@atlassian.com> 1409002123 -0700

Merge branch 'foo/mybranch'
```

What is a fast-forward merge?

It will just shift the HEAD tag



What are ‘merge strategies’?

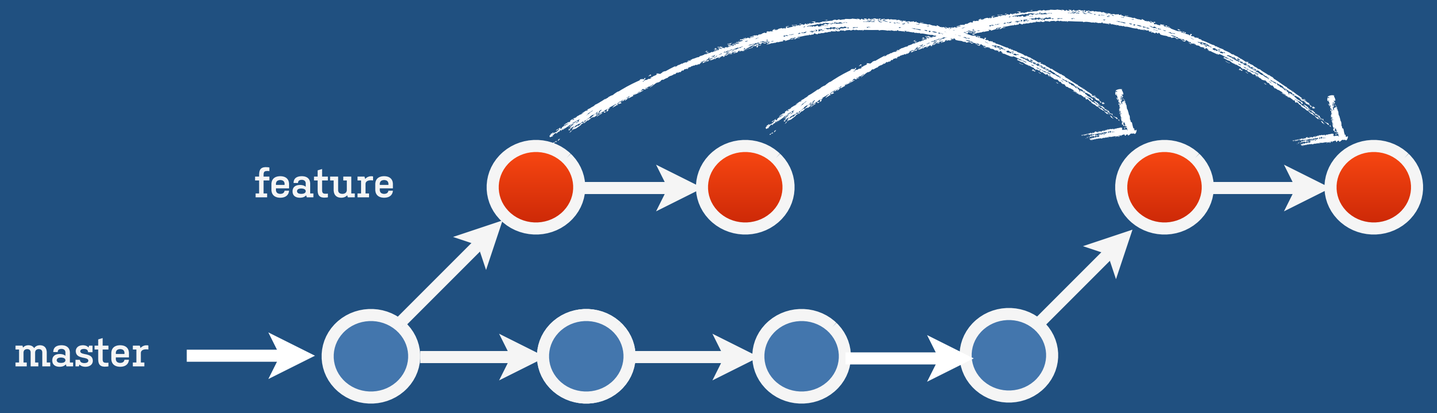
git has breadth of choice on
how to merge changes!

resolve **recursive** **octopus**

ours **subtree** **yours?**

Rebase

It's a way to replay commits,
one by one, on top of a branch

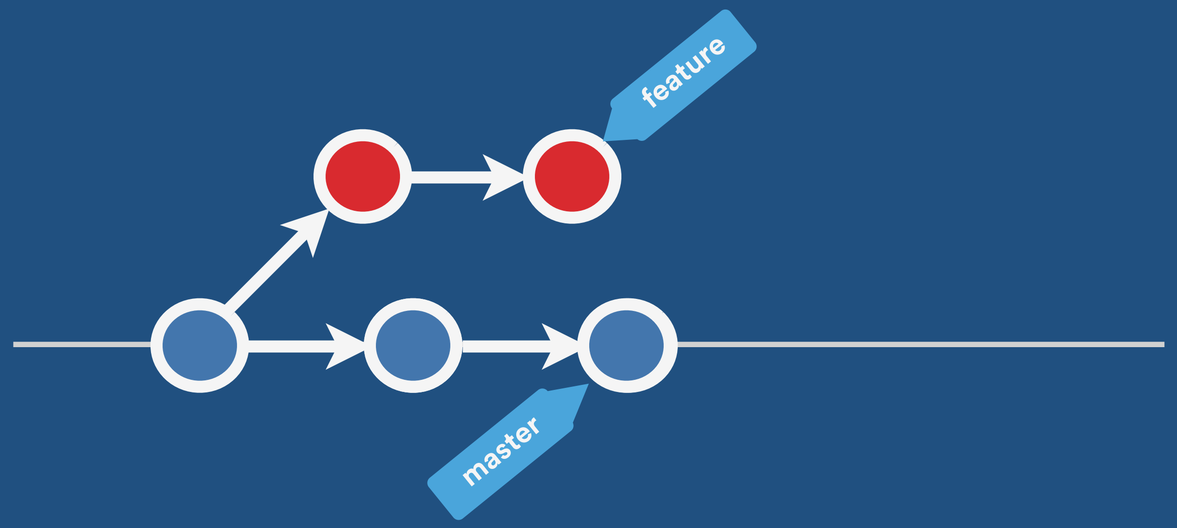


Getting out of trouble

reset —hard ohshit

—hard removes all staged and working changes!

```
$> git reset --hard feature^          # '^' means 'parent'
```



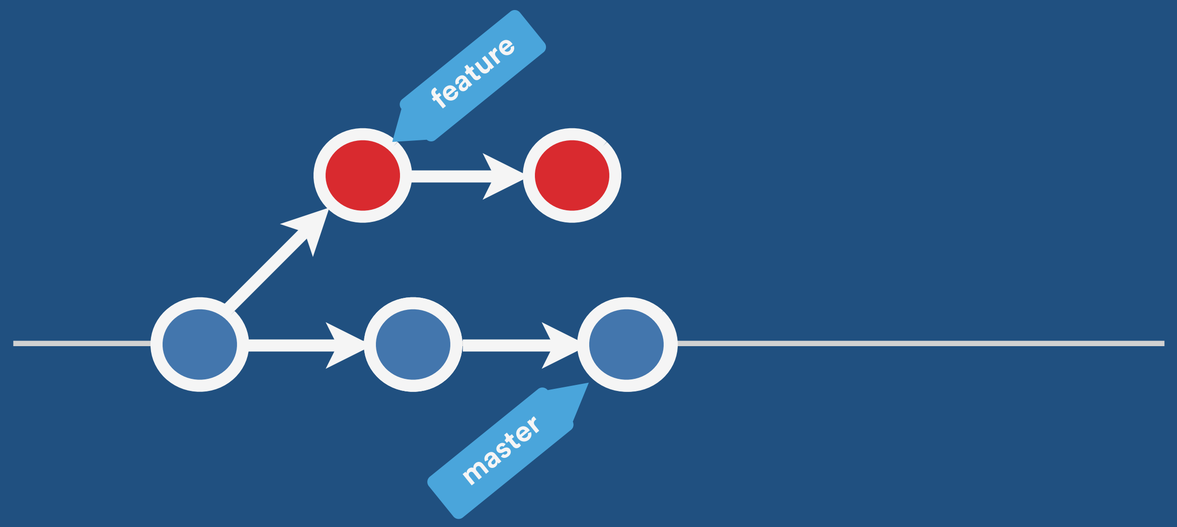
Reflog!

```
$> git reflog
0c35628 HEAD@{1}: reset: moving to HEAD^
6cc6637 HEAD@{2}: commit: Add B
0c35628 HEAD@{3}: merge: Merge made by the 'recursive' strategy.
e0c0d65 HEAD@{4}: cherry-pick: A
80bb854 HEAD@{5}: checkout: moving from alpha to master
5044136 HEAD@{6}: commit: A
80bb854 HEAD@{7}: checkout: moving from master to alpha
80bb854 HEAD@{8}: commit (initial): 1
```

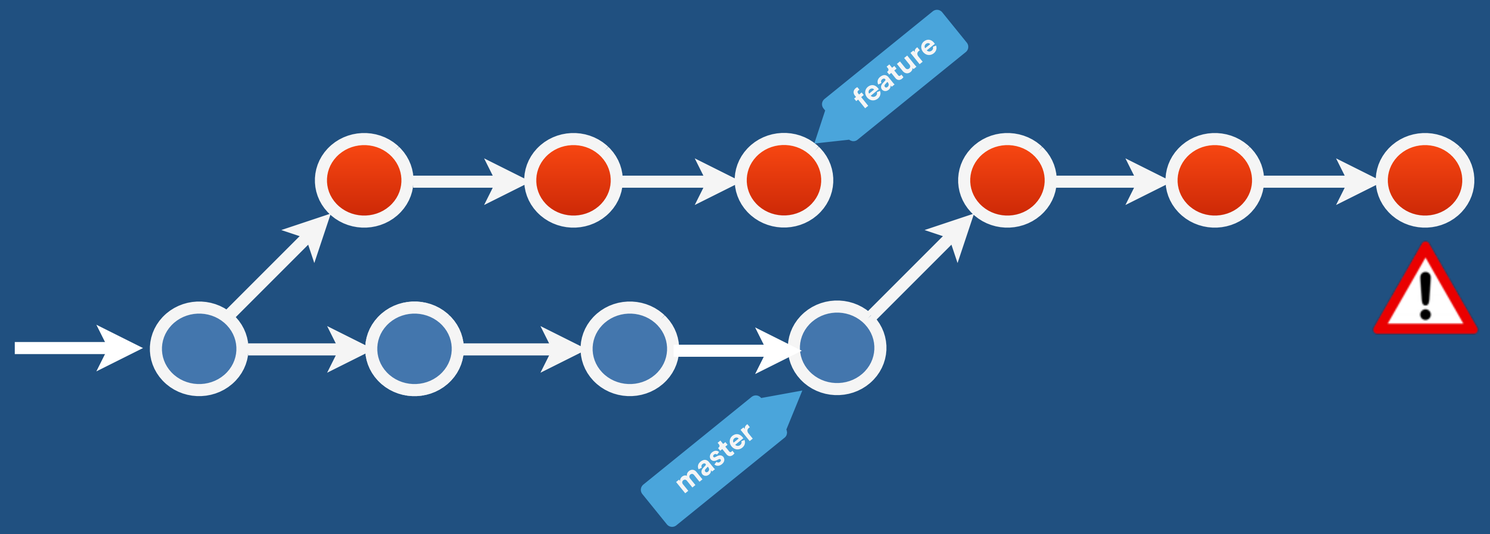
Reflog+Reset = Redo

Reset back to our commit!

```
$> git reset --hard 6cc6637
```



Rebase Broke The Build!

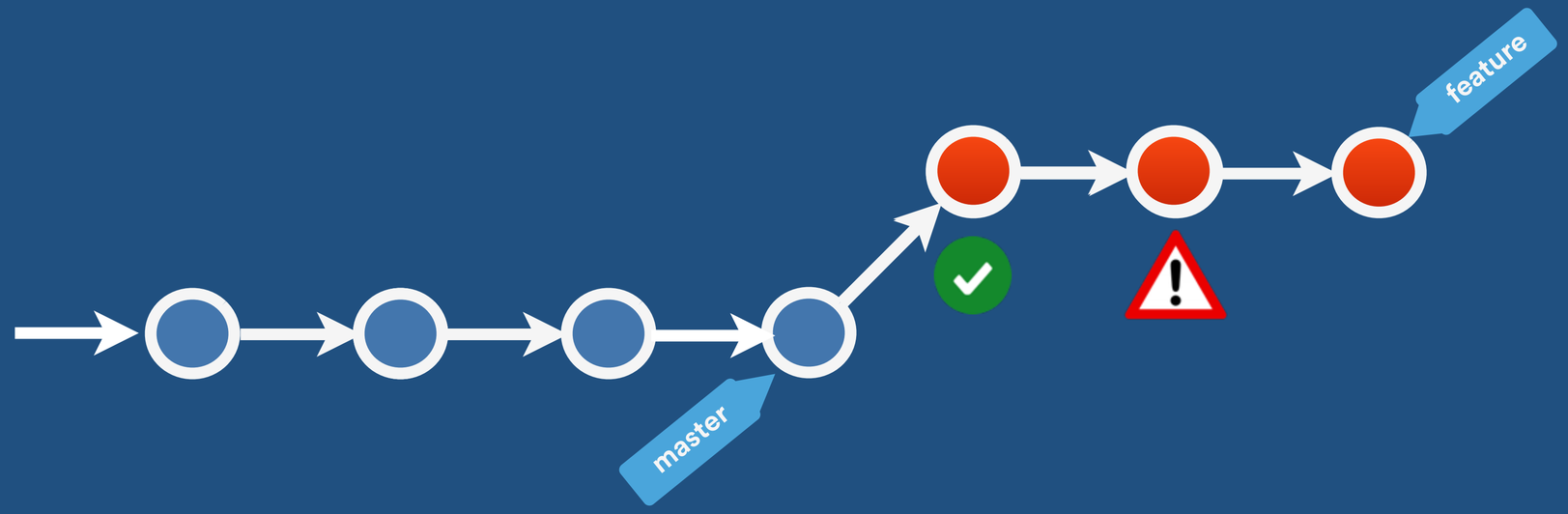


Fix with rewind/replay

Reflog to reset.

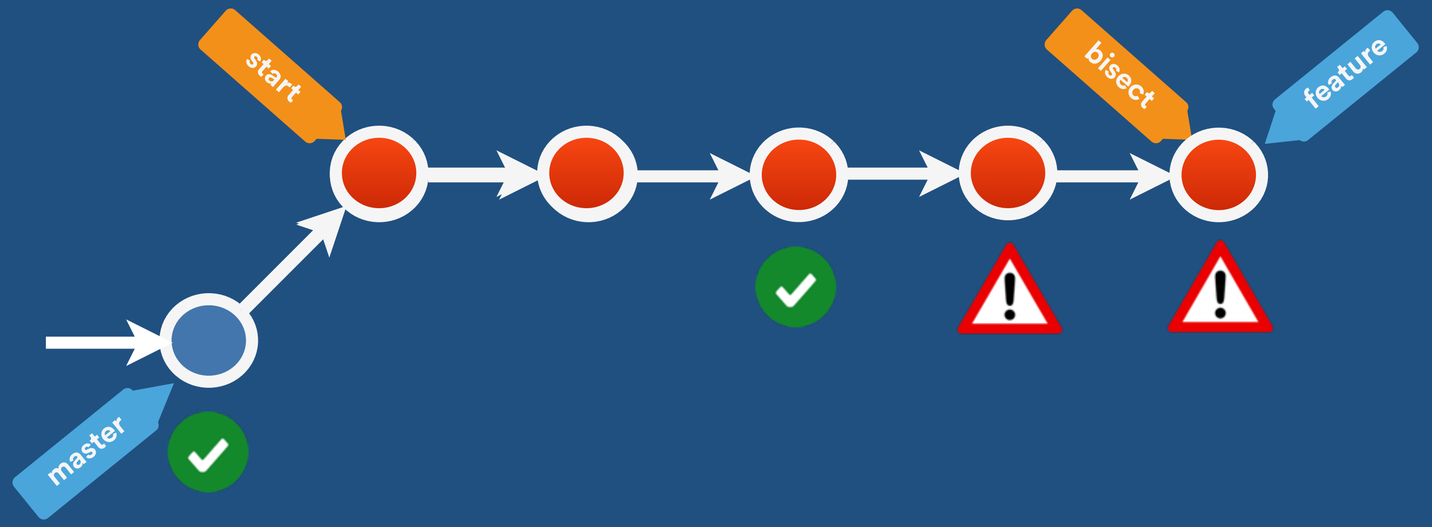
rebase --exec to test each step.

```
$> git rebase master  --exec “make test”
```



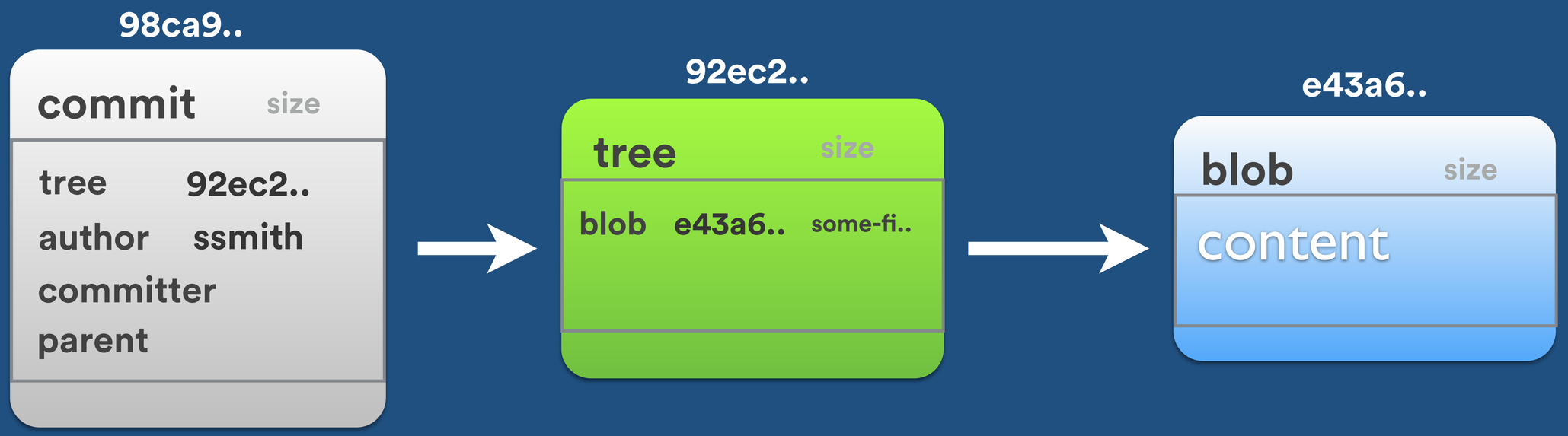
Fix in place: bisect

```
$> git bisect start
$> git bisect good master
$> git bisect run make test
```



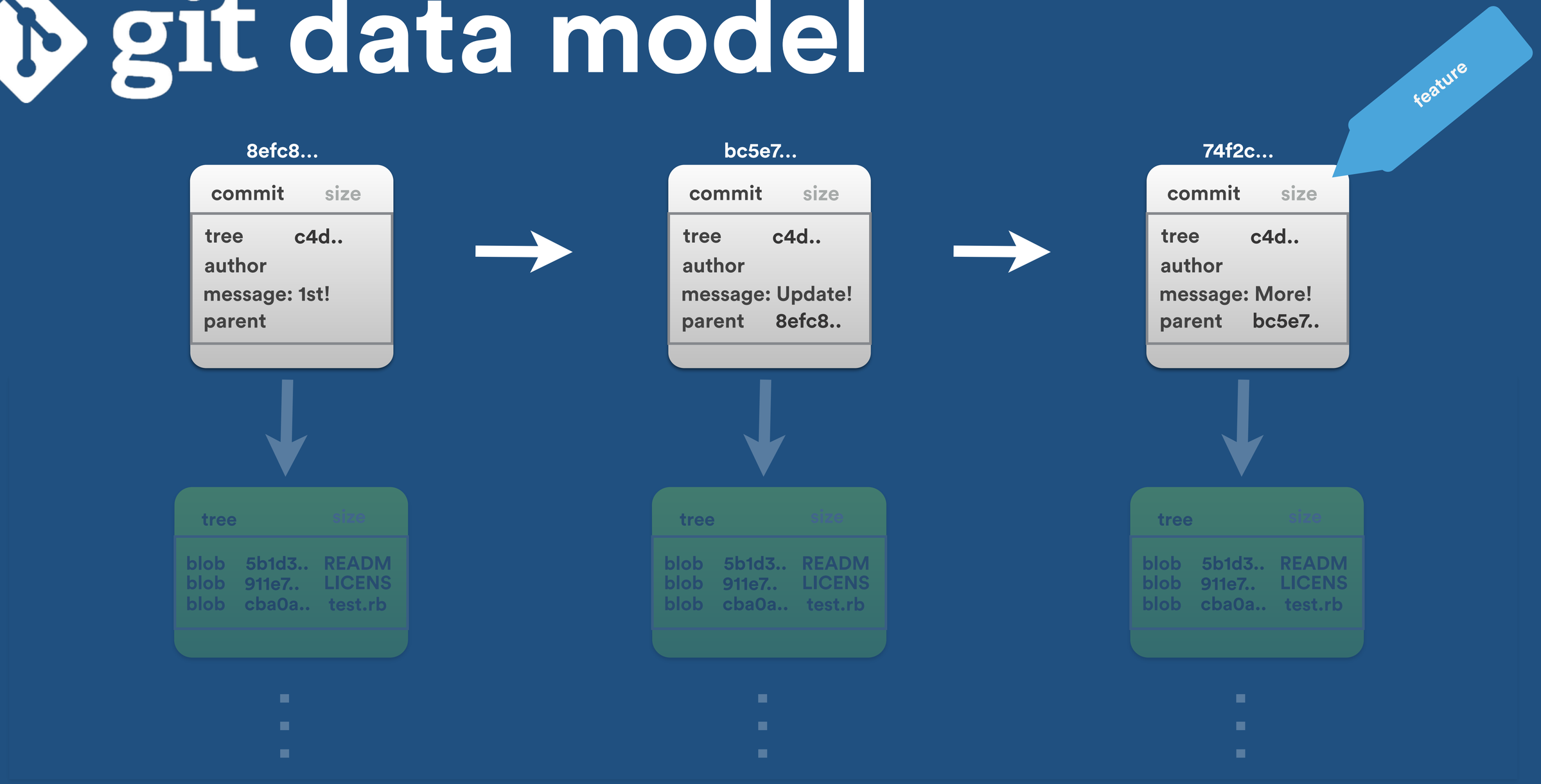
Out of Time

git data model



@tarkasteve

git data model



@tarkasteve


So now we're samrt!

goto; conference

Click ‘engage’ to rate session.

Rate **12** sessions to get the supercool GOTO reward

Join the conversation #gotoldn