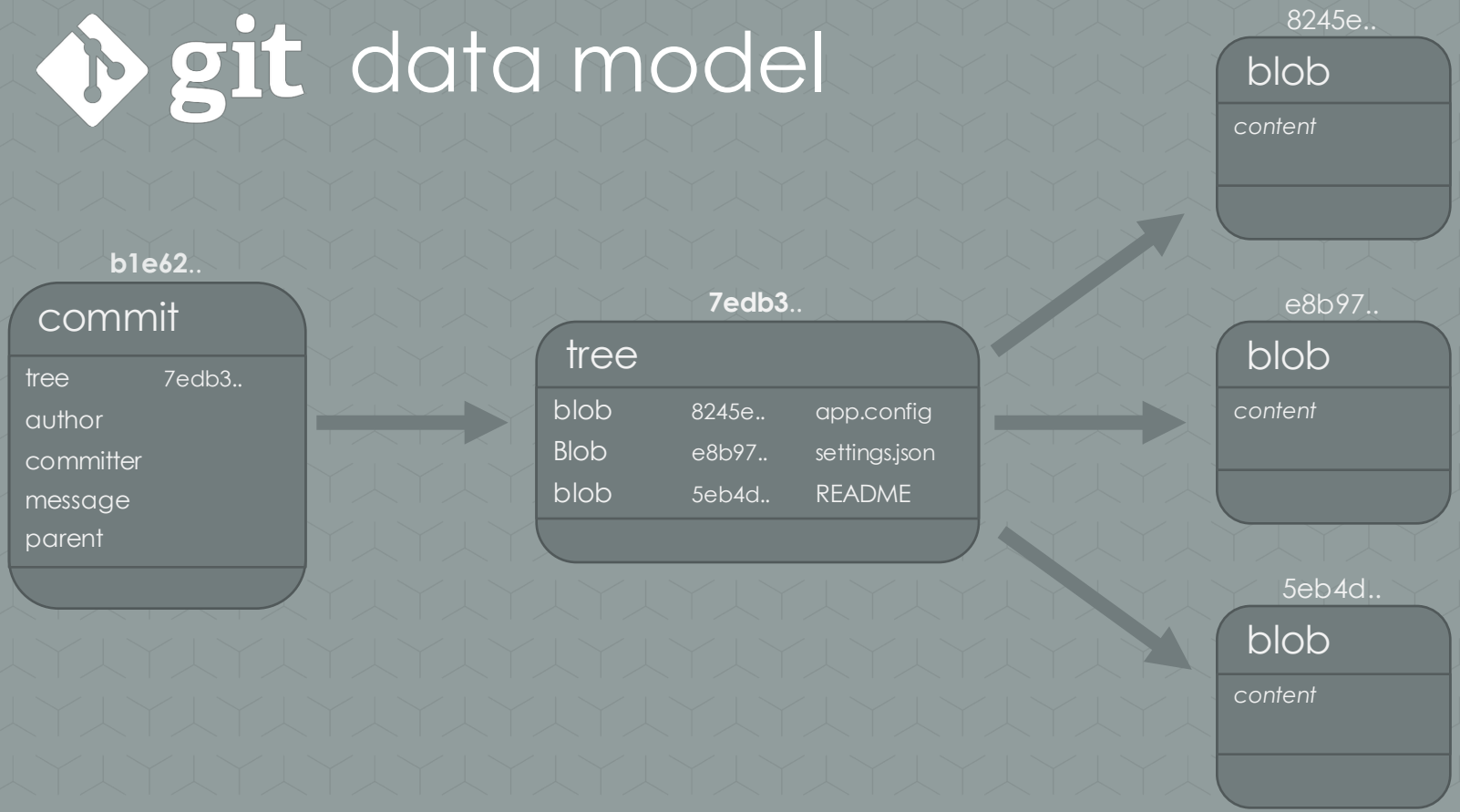
git, under the hood

Daniele Iasella

Thanks to Steve Smith

git data model



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

2 directories
```

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

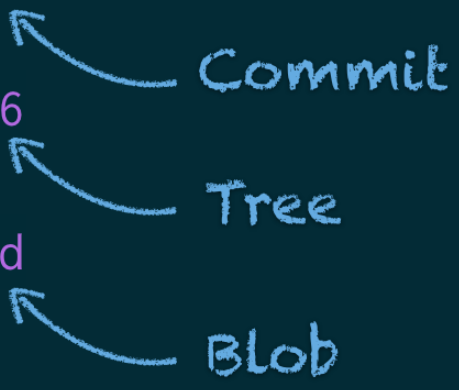
```
$> tree .git/objects
.git/objects
├── e6
│   └── 9de29bb2d1d6434b8b29ae775ad8c2e48c5391
├── info
└── pack

3 directories, 1 file
```

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

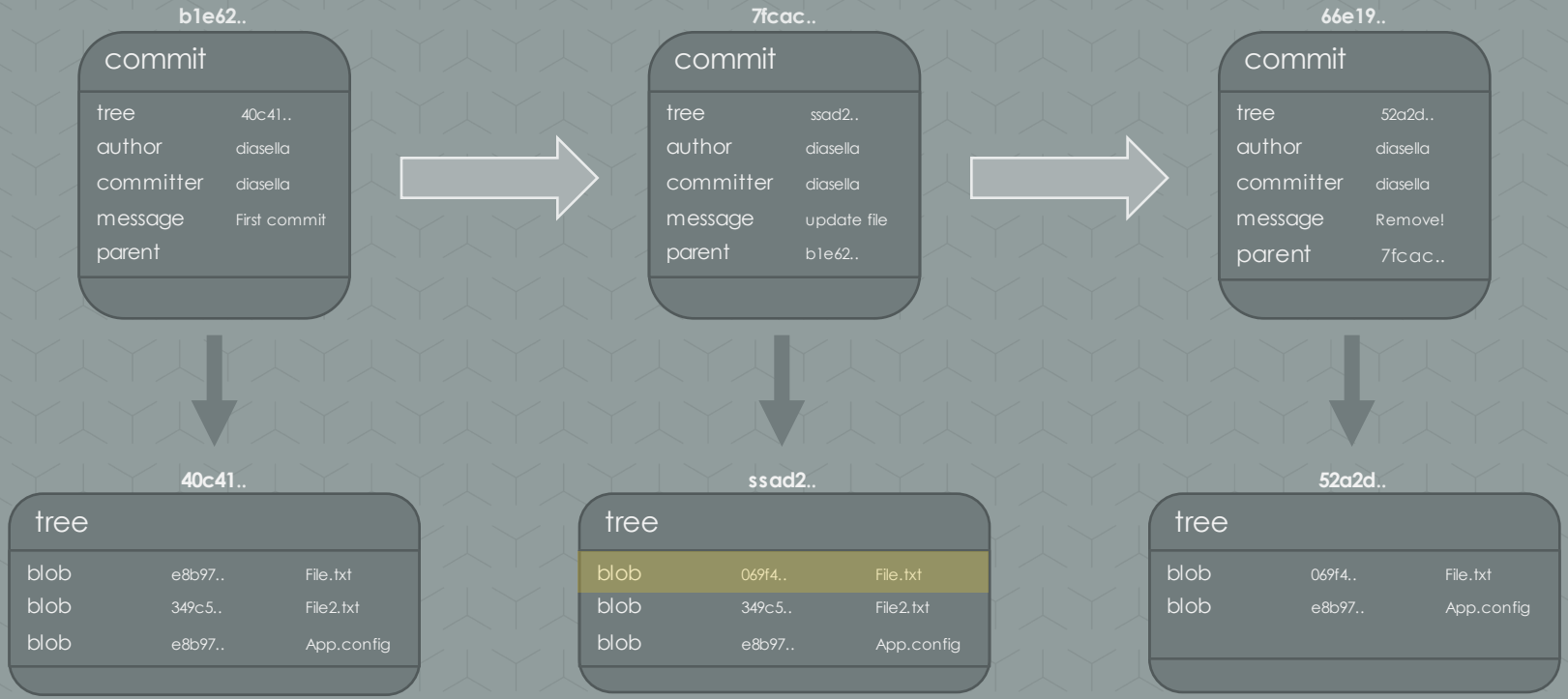
```
$> tree .git/objects
.git/objects
├── 13
|   └── 1e360ae1a0c08acd18182c6160af6a83e0d22f
├── 31
|   └── 995f2d03aa31ee97ee2e814c9f0b0ffd814316
├── e4
|   └── 3a6ac59164adadac854d591001bbb10086f37d
├── info
└── pack

5 directories, 3 files
```



git data model

git data model



Ok, I get it now but…

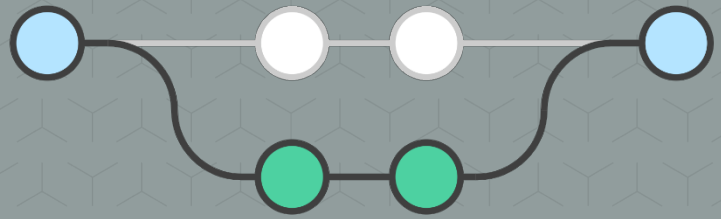
what about **ref**, **branches** and **tags**?

ref who???

Ref is just a pointer to an object.

Branches what???

A branch is a fork from a common point and ref to a commit, the "*HEAD*"



Tags ???

A tag is special ref to 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
```

git 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'
```

Merge strategies

Why is that plural?

Well, there is more than just default…

git merge strategies

resolve

recursive

subtree

2 HEADs

octopus

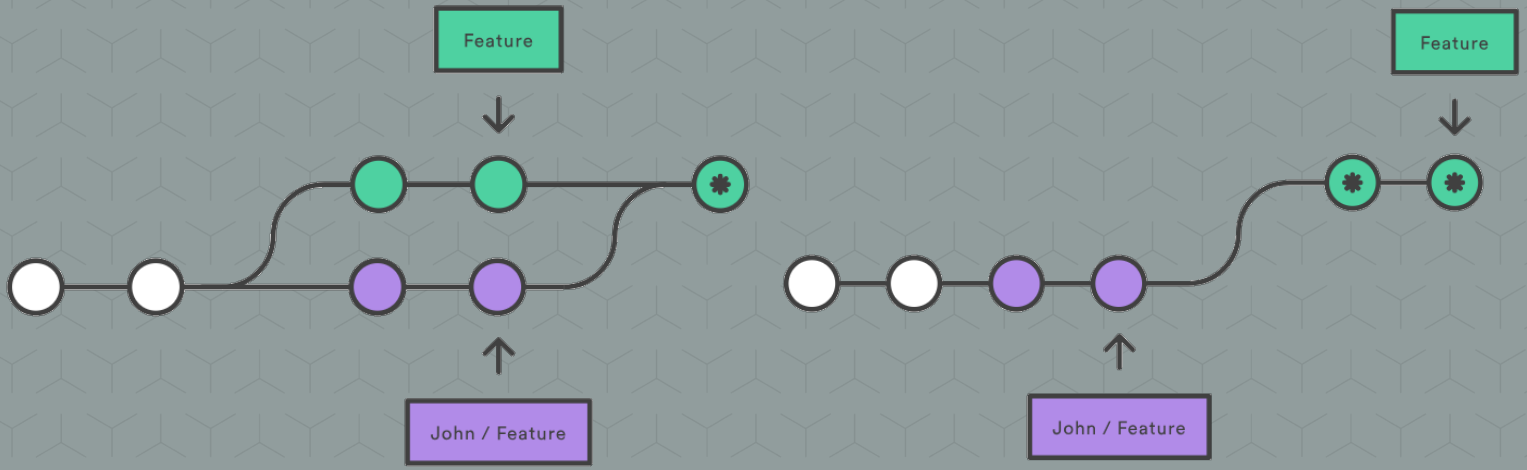
3+ HEADs

ours

any number HEADs

git rebase (vs merge)

Instead of using a merge commit, rebasing *re-writes* the project history by creating brand new commits for each commit in the original branch



git reflog :

here, there is my 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
```

(n.b. up to 90 days by default)

git reset --hard

removes all staged and working changes

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

S**t happens soooo.....

reflog + reset  redo! ☺

```
$> git reflog
0c35628 HEAD@{1}: reset: moving to HEAD^
6cc6637 HEAD@{2}: commit: Add B
$> git reset --hard 6cc6637
```

git bisect

New bug introduced?

Binary search at your service!

git GUIs

SourceTree (free)

gMaster (free non-commercial)

GitKraken (free community version)

gitExtensions (free)

Thanks people!