

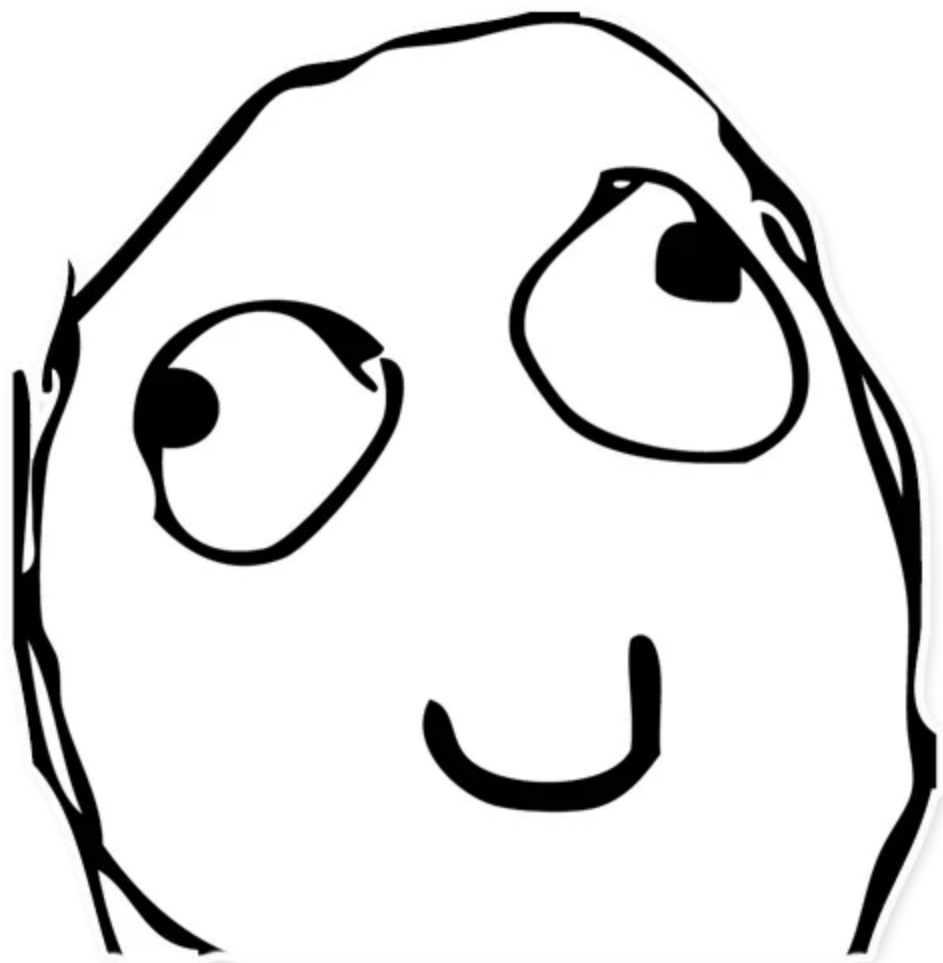
Software Engineering Conference Russia
October 2017, St. Petersburg

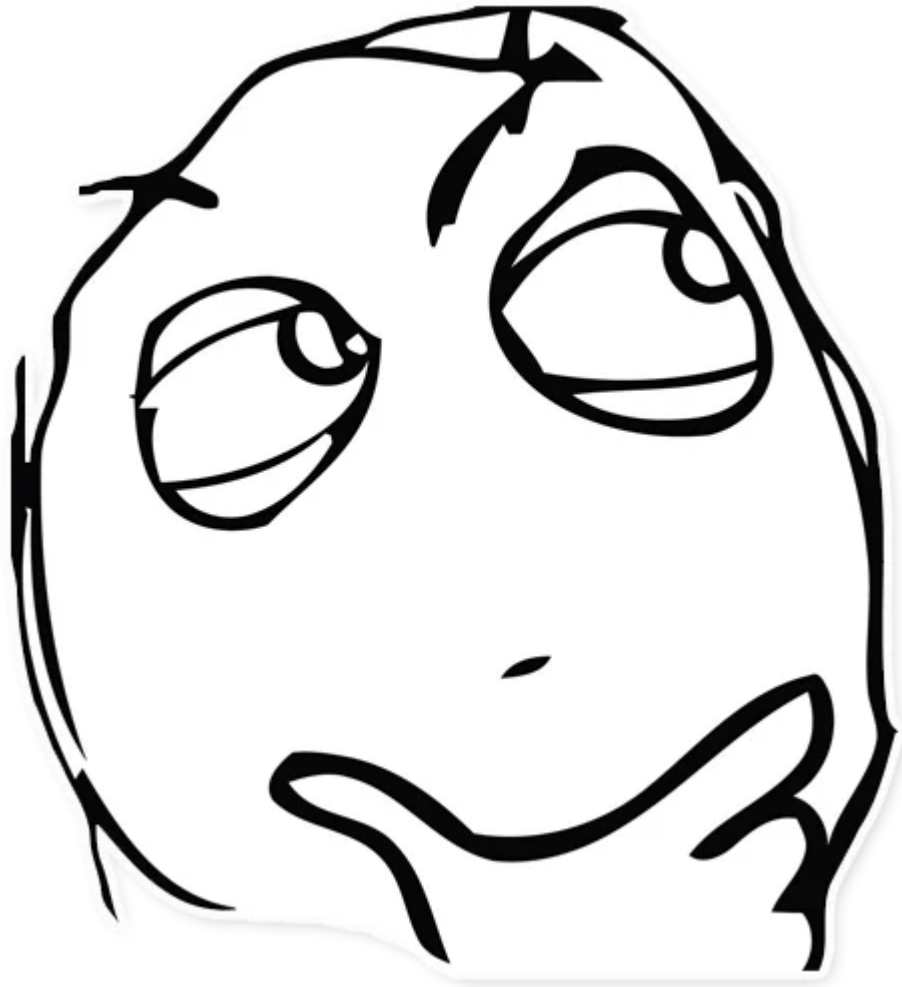


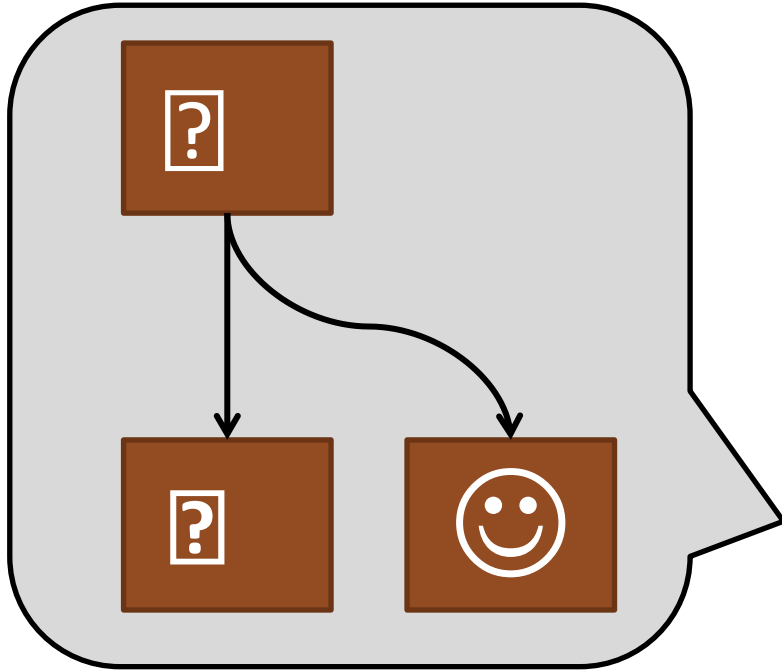
Reverting a merge

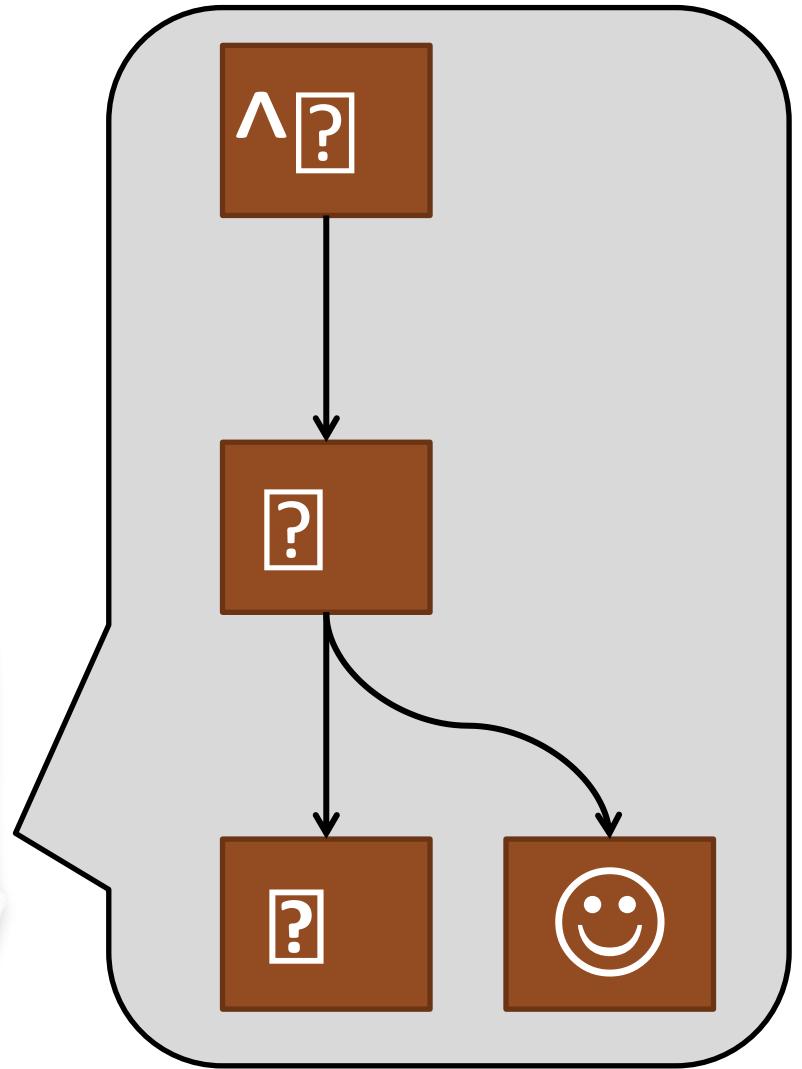
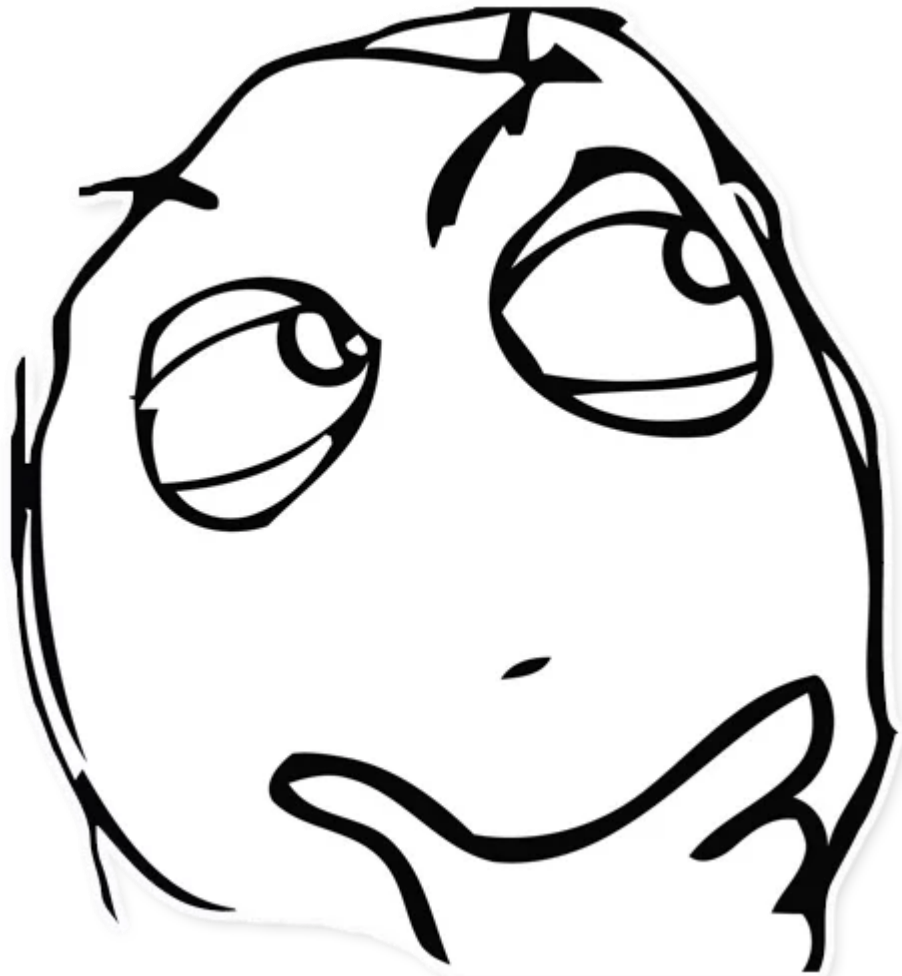
Mikhail Matrosov

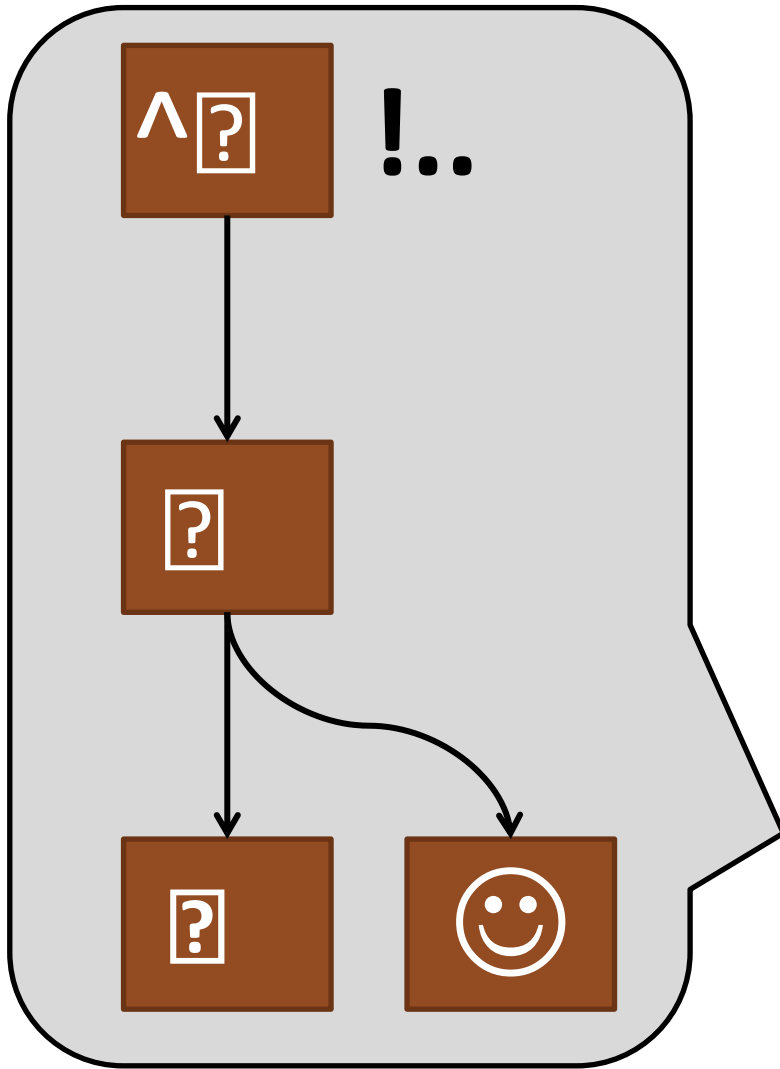
Technical Manager at Align Technology

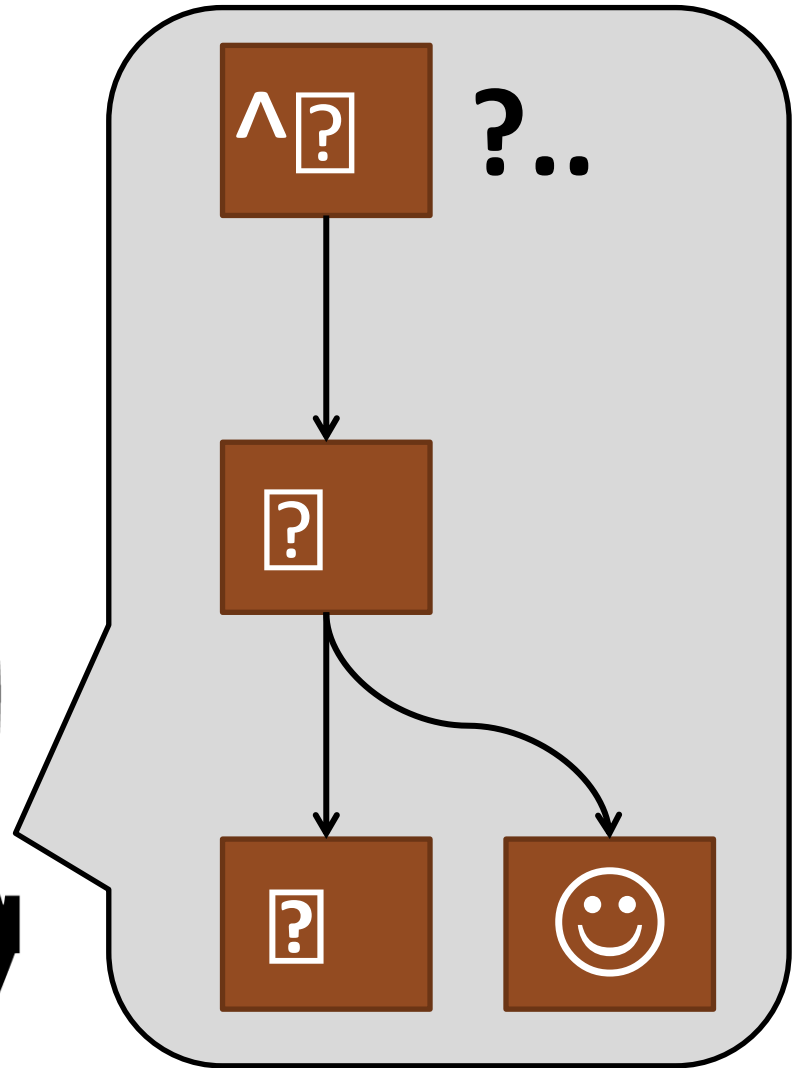
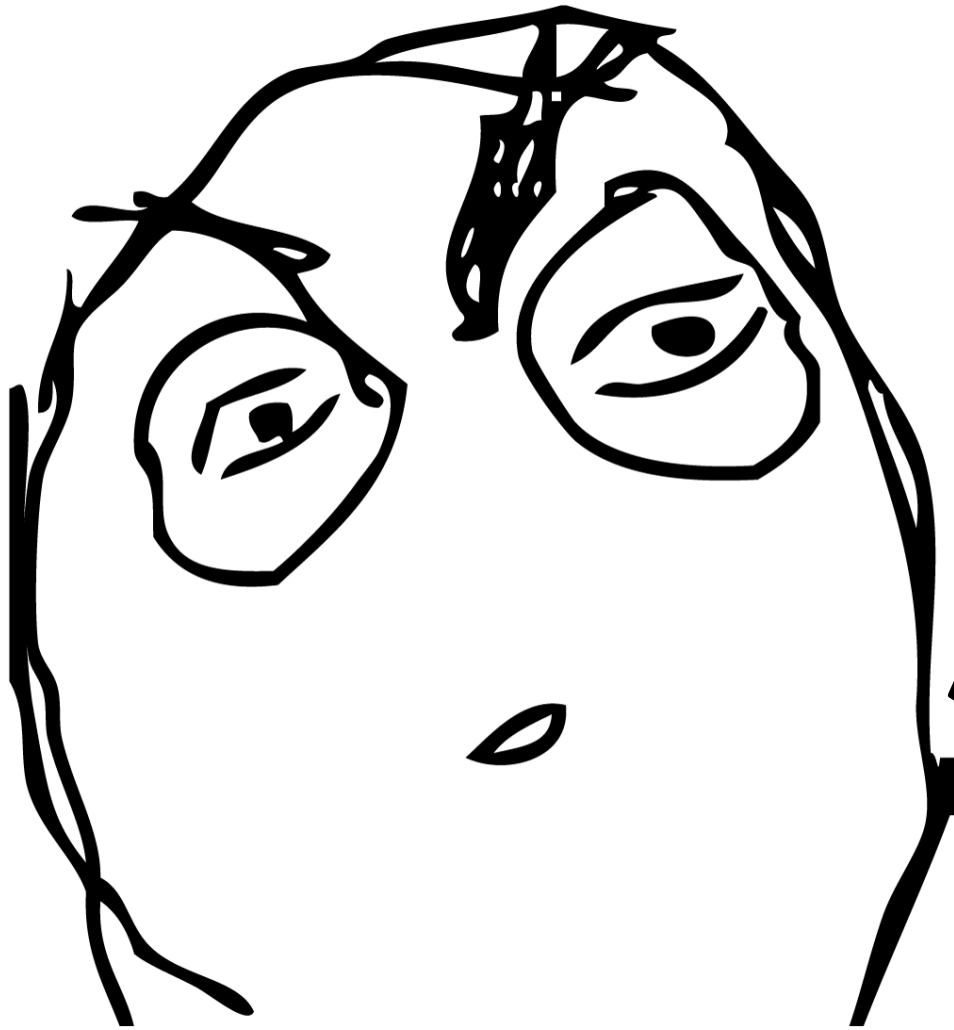


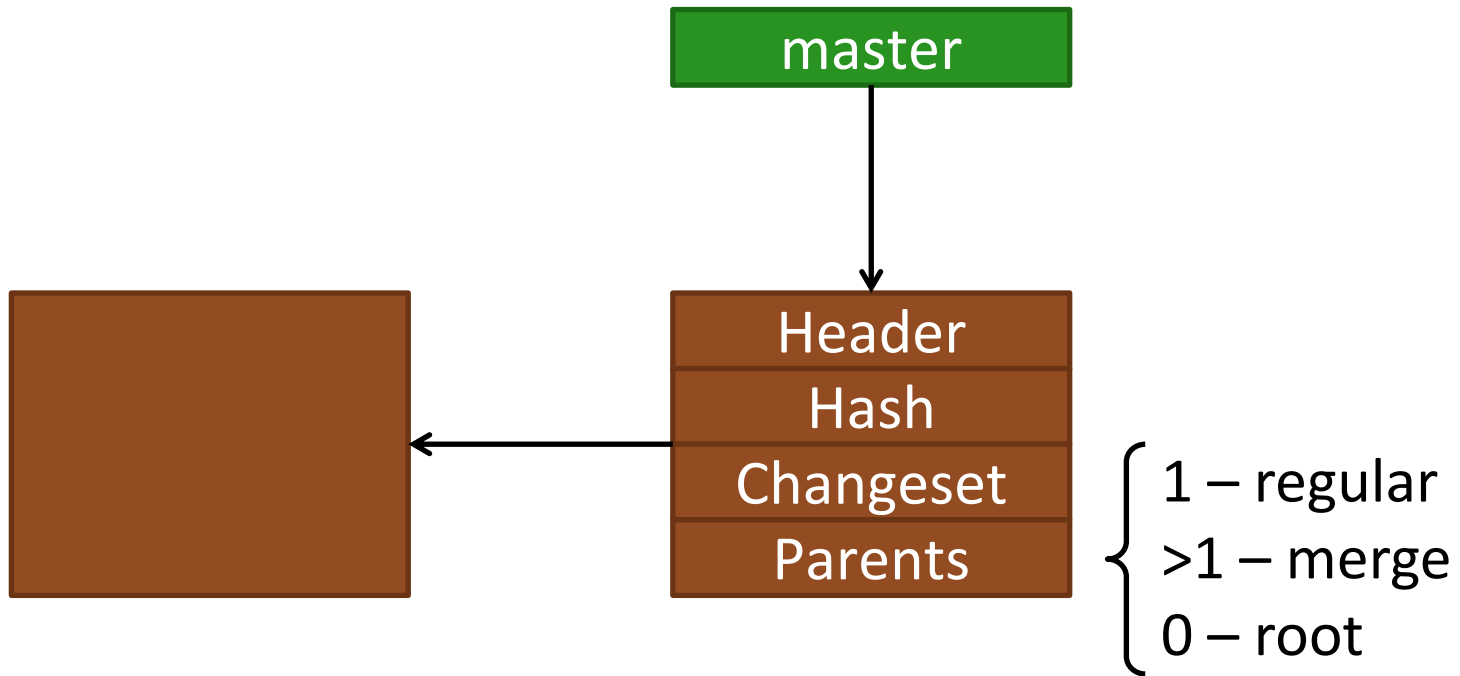


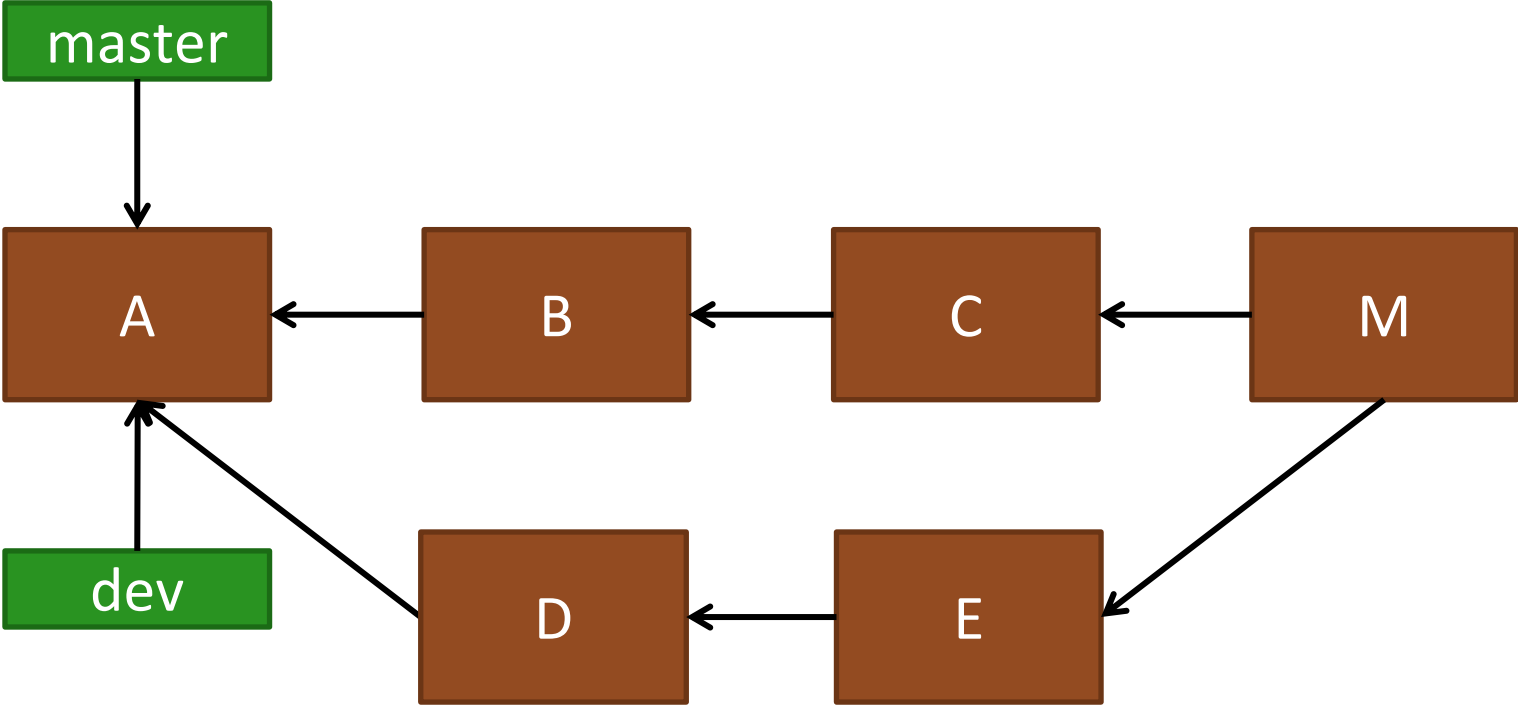




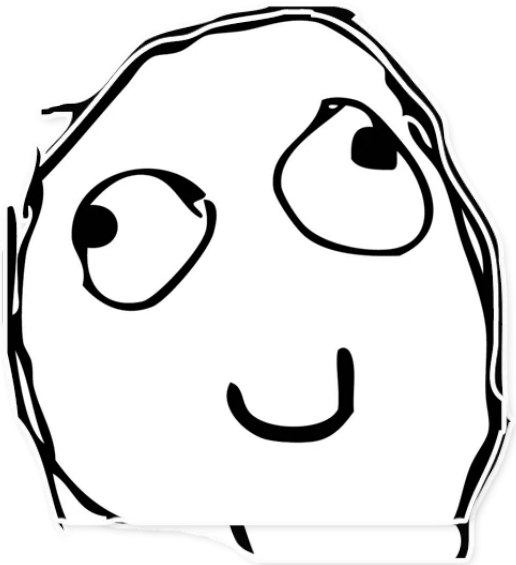
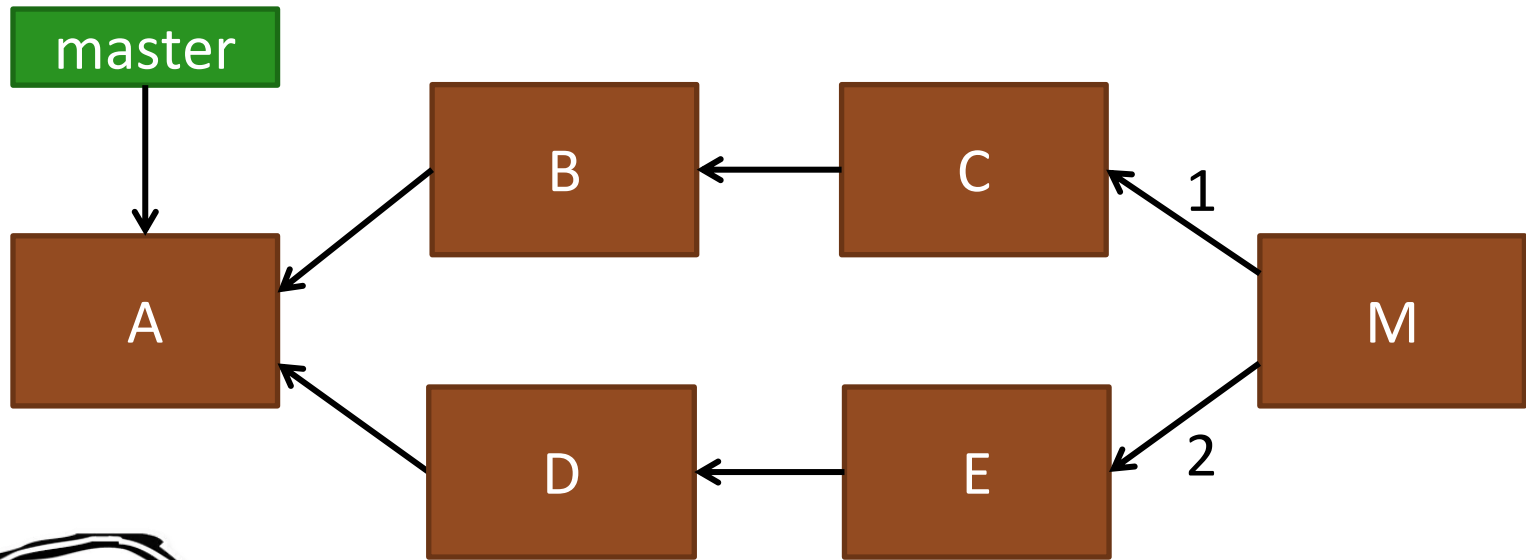


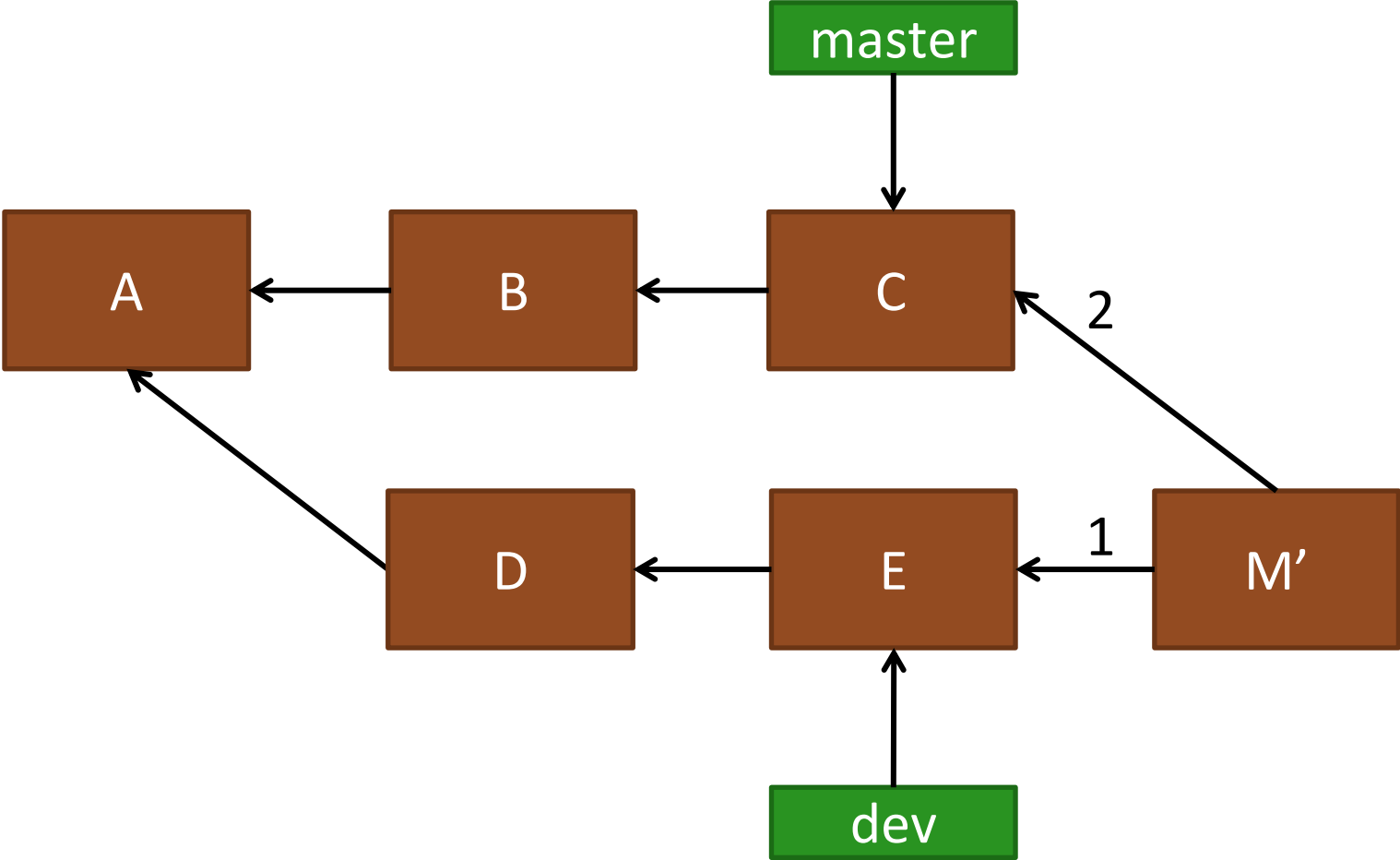


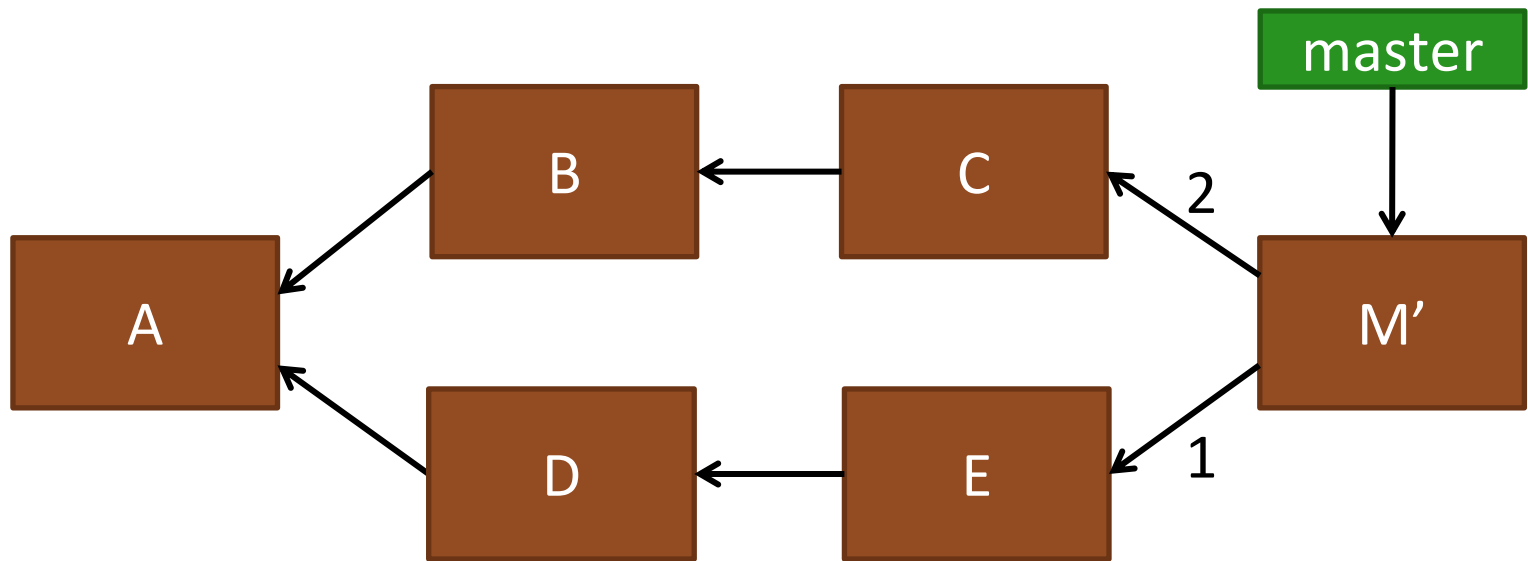
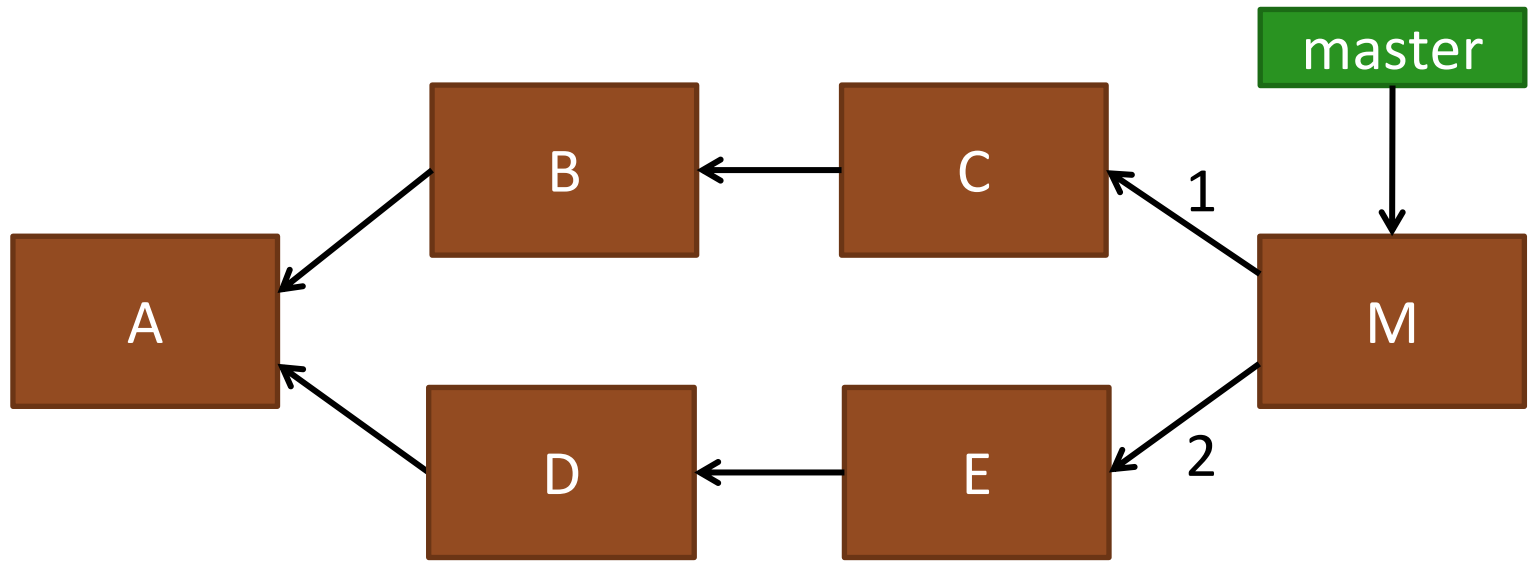




1 = merged to
2 = merged from





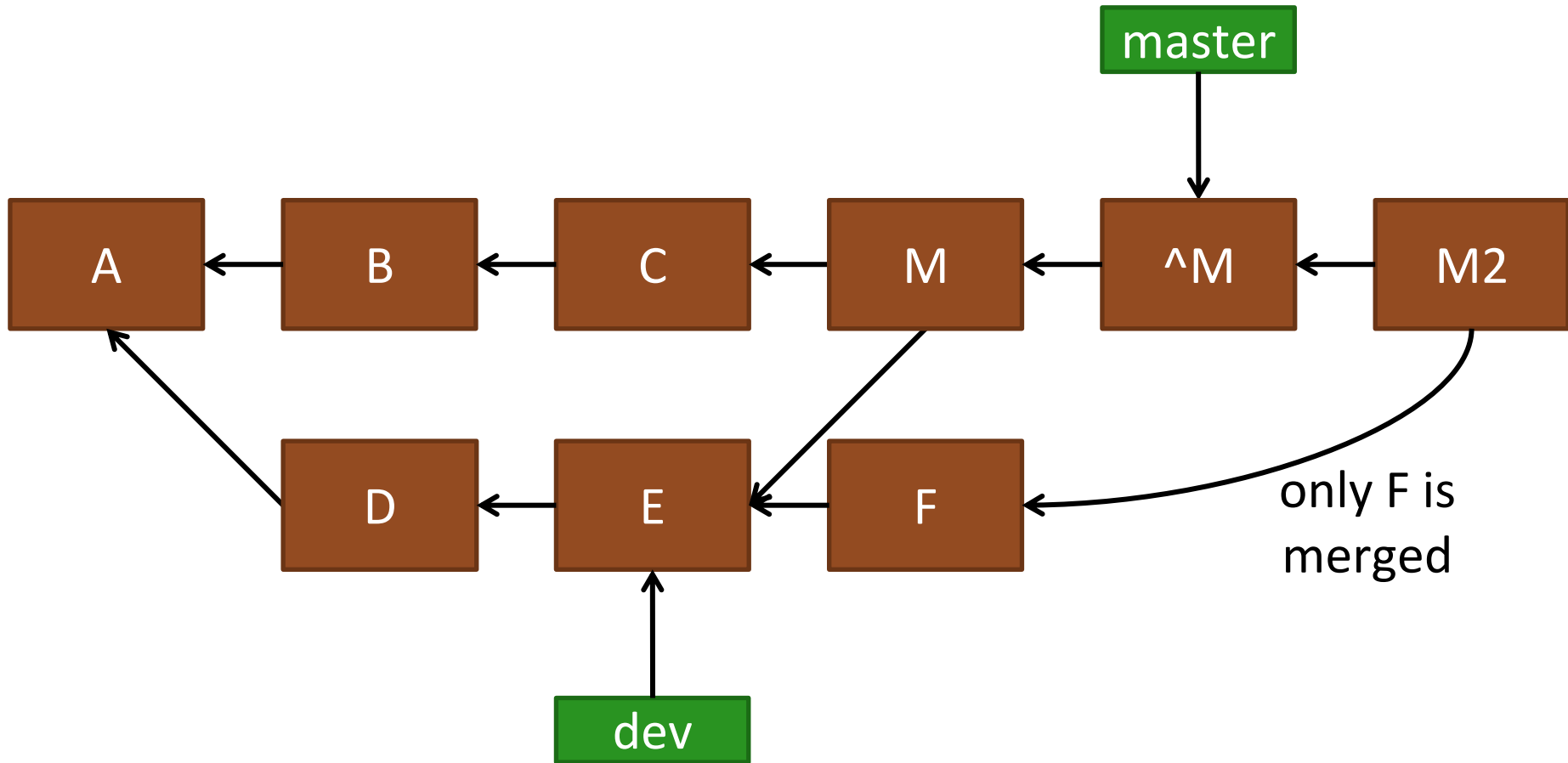




TO REVERT A MERGE

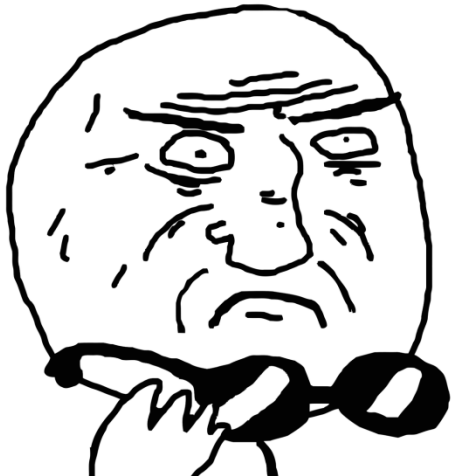
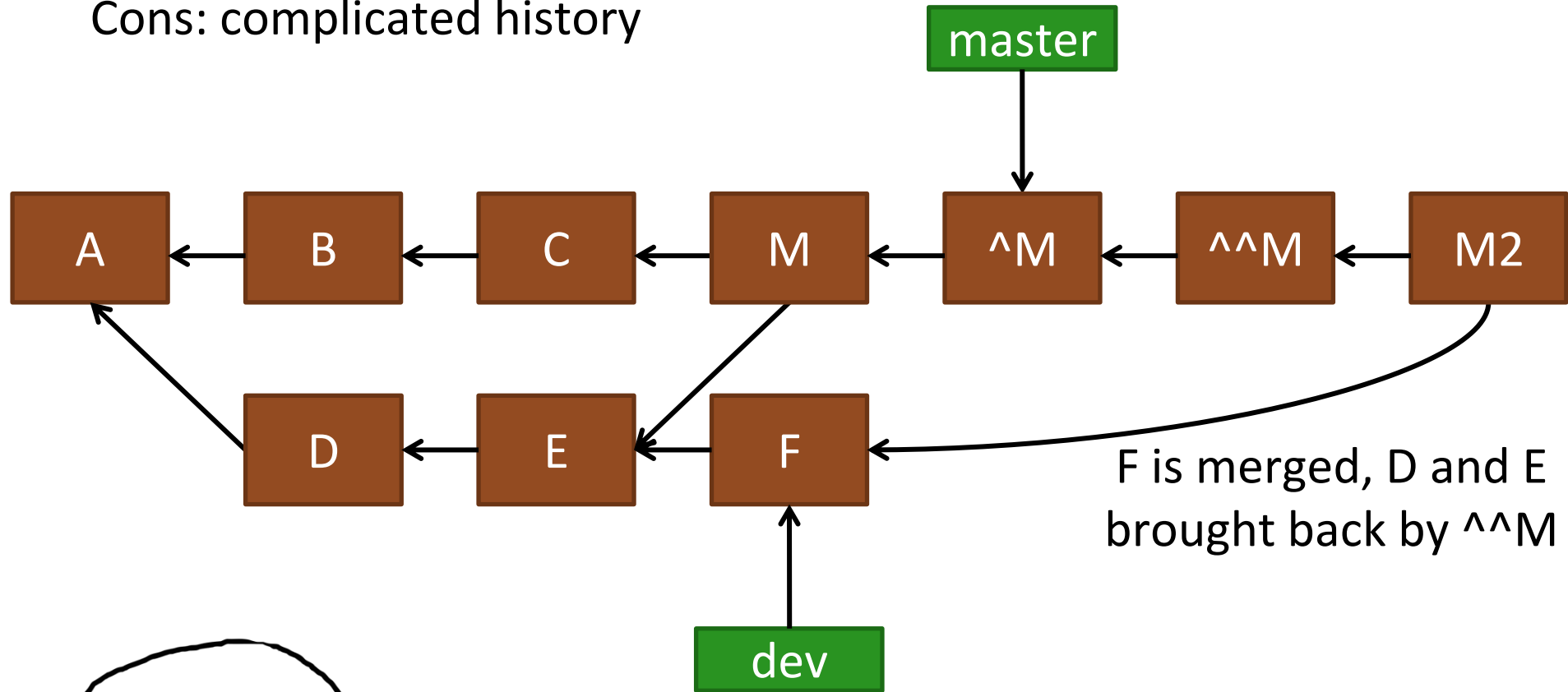


**CHOOSE YOUR PATH
WISELY YOU MUST**



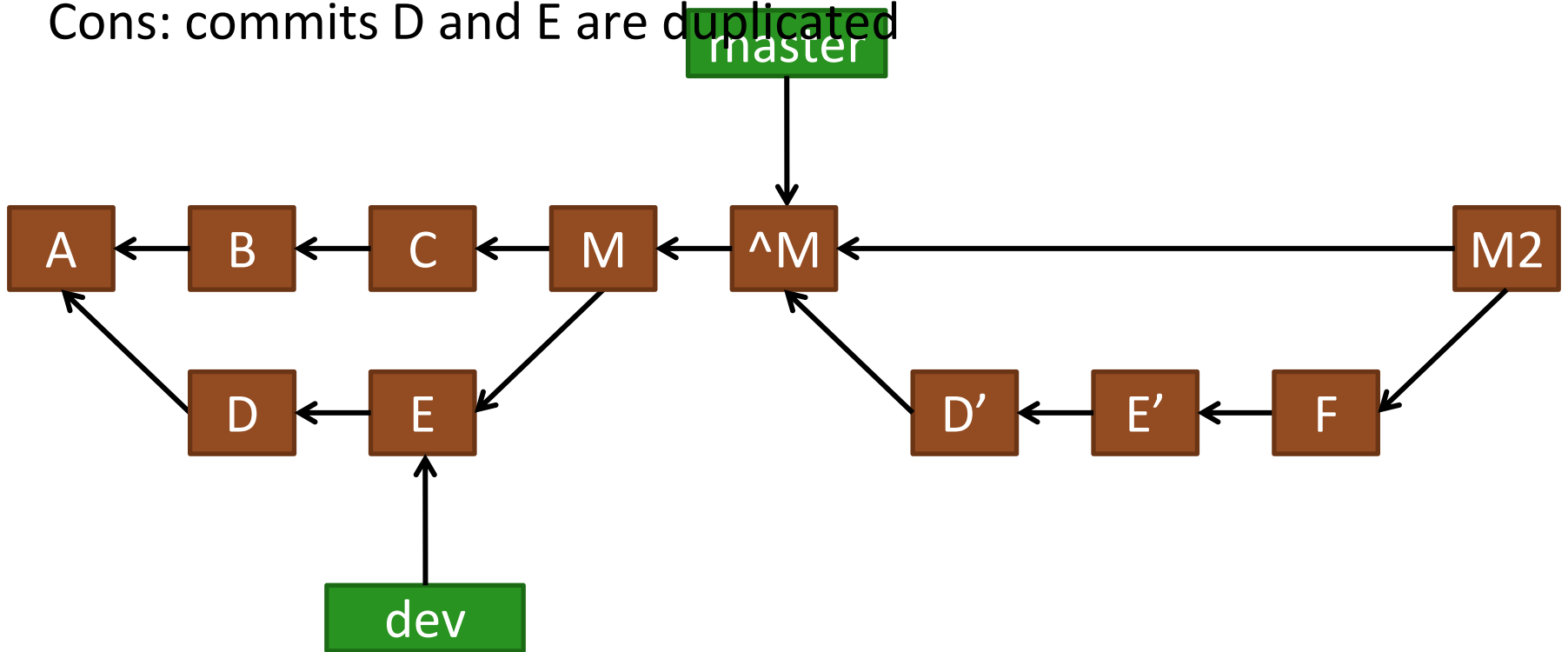
Pros: commits D and E are preserved

Cons: complicated history



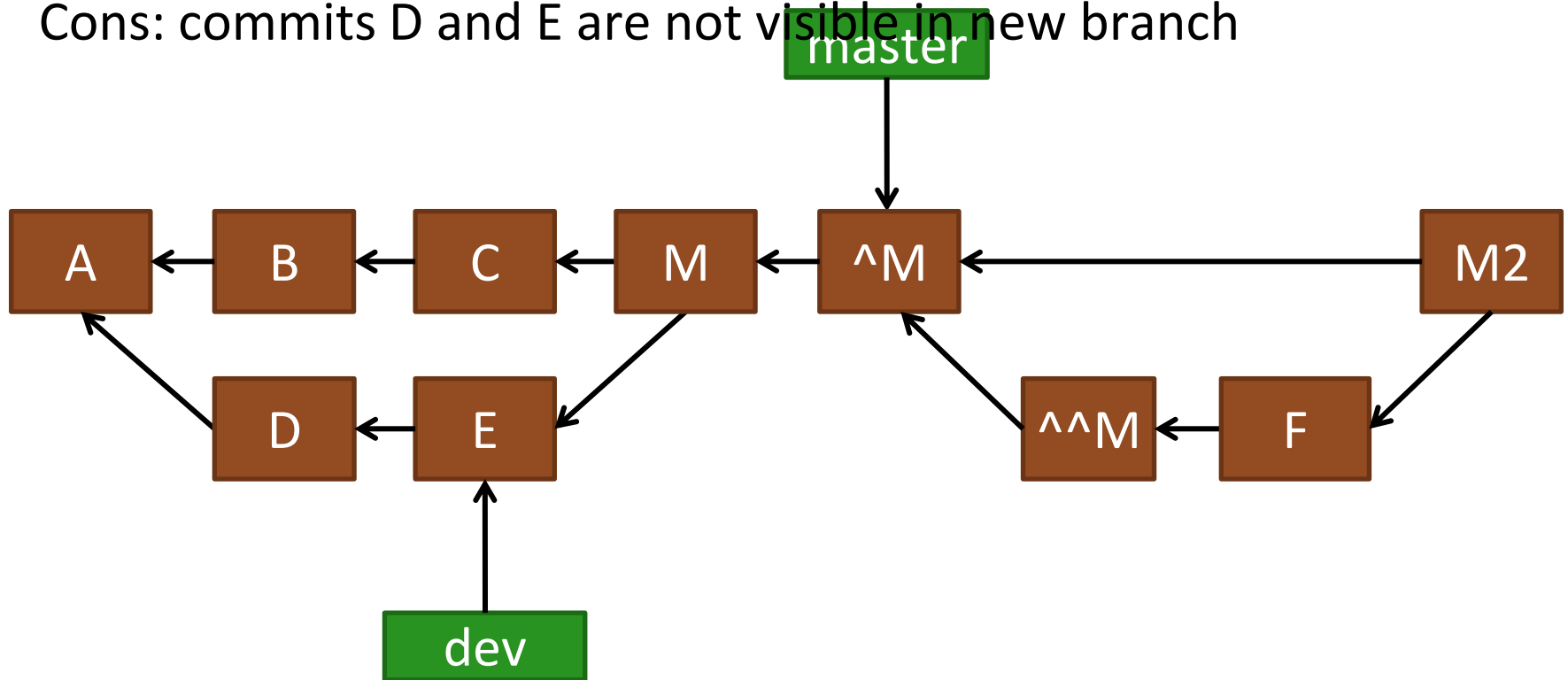
Pros: new branch has regular workflow

Cons: commits D and E are duplicated



Pros: preserves conflicts resolution happened in M

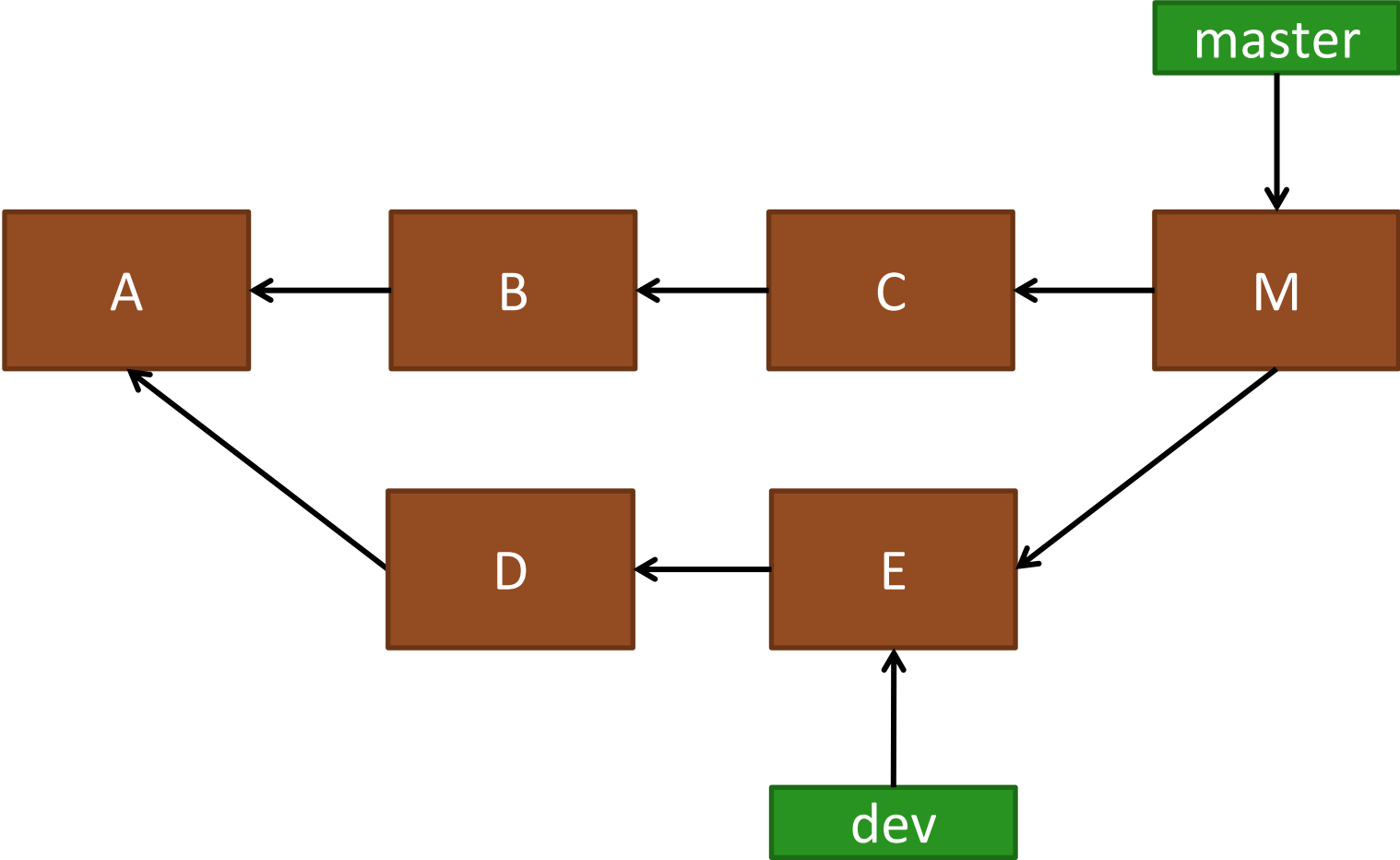
Cons: commits D and E are not visible in new branch





**TO MERGE AGAIN
ONCE READY**

**YOU MUST
UNLEARN WHAT
YOU HAVE LEARNT**



Resetting the merge

- When you immediately realize the problem
- `--force-with-lease`

TO RESET A MERGE



USE FORCE...

TO RESET A MERGE



**USE FORCE...
WITH LEASE**

References

- Basic info: <https://git-scm.com/blog/2010/03/02/undoing-merges.html>
- More details: <https://opensource.apple.com/source/Git/Git-26/src/git-htmldocs/howto/revert-a-faulty-merge.txt>
- On order of commits: <http://devblog.nestoria.com/post/98892582763/maintaining-a-consistent-linear-history-for-git>