Git History Manipulation

Viewing Logs

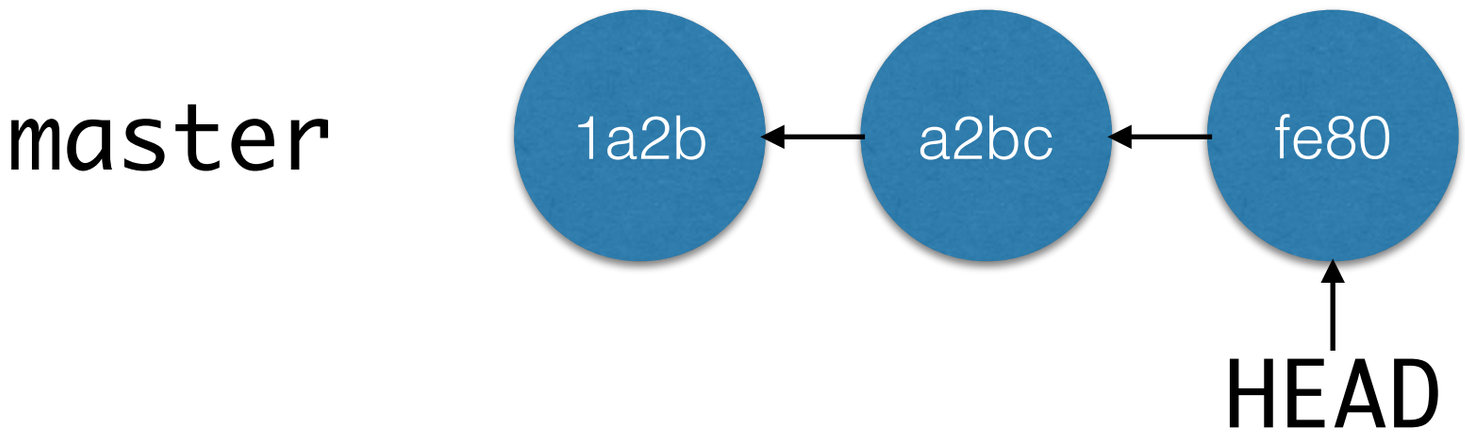
git log

```
commit 469a4f5ed5b3231397dad598db32867a19bae64d
Merge: 1b9fc1a b95ebb3
Author: Wilhansen Li <wil@byimplication.com>
Date:   Mon Nov 17 17:52:56 2014 +0800

    Merge remote-tracking branch 'refs/remotes/origin/centos7' into centos7

commit 1b9fc1a51b602a895b95a07cc492aea27a6091b3
Author: Wilhansen Li <wil@byimplication.com>
Date:   Mon Nov 17 17:49:47 2014 +0800

    Go gold.

commit b95ebb37a261305318be7f77adbfc77a3fe60db5
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Mon Nov 17 14:26:41 2014 +0800

    Fix /run directory

commit 38baa6430b429c95563083e483caad939e7507c4
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Tue Nov 11 17:07:01 2014 +0800

    Use common submodule
```

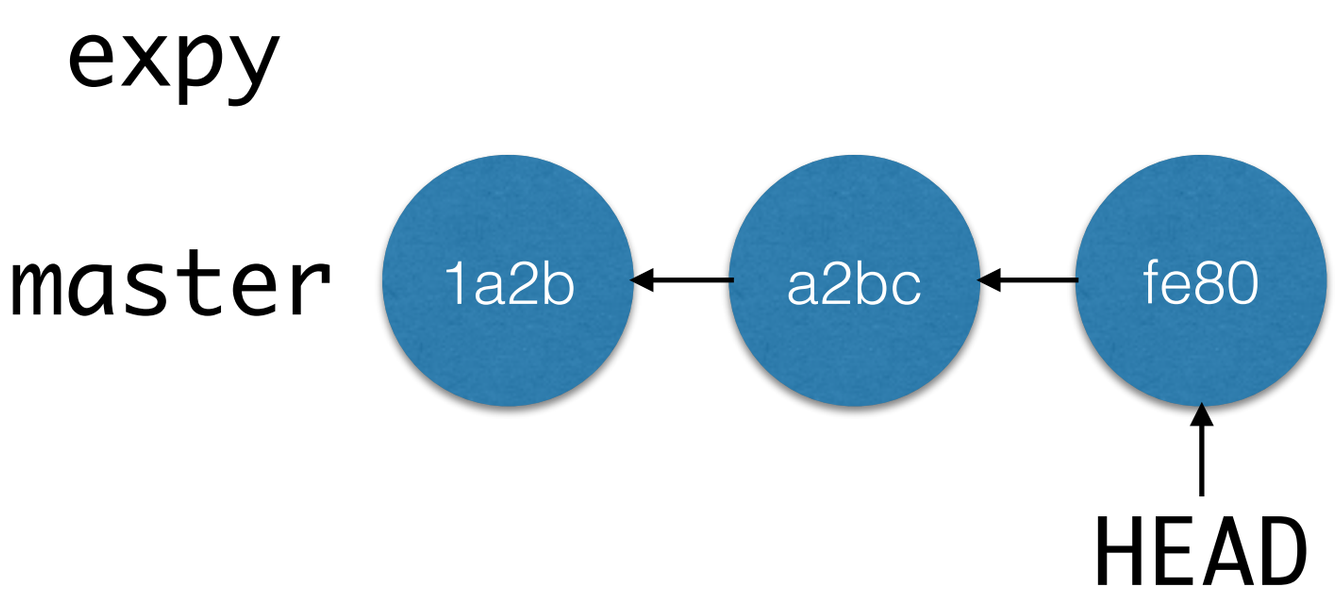
Branches

- “separate timelines”
- The “main timeline” is called `master`
- Used for simultaneous development without clashing with each other at every commit.
- View available branches with `git branch`
- Switch to a branch with `git checkout <branch>`
- Create branches with `git branch <name>`
    - Branches off from the current `HEAD`.

Branching

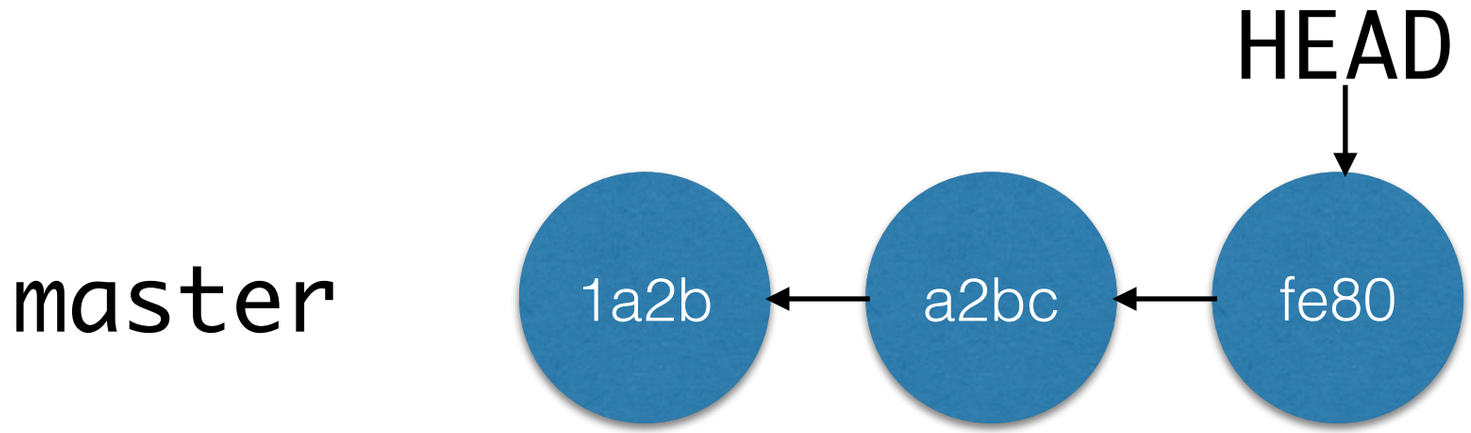



```
git branch expy
```

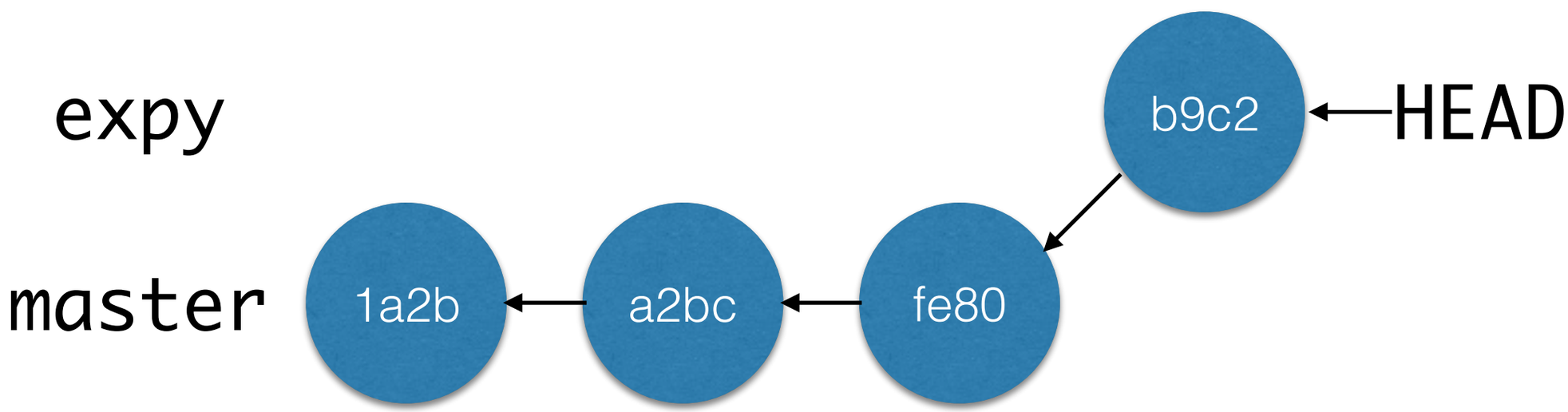


Branching

```
git checkout expy
```




```
git add (…)
git commit (…)
```



Merging Branches

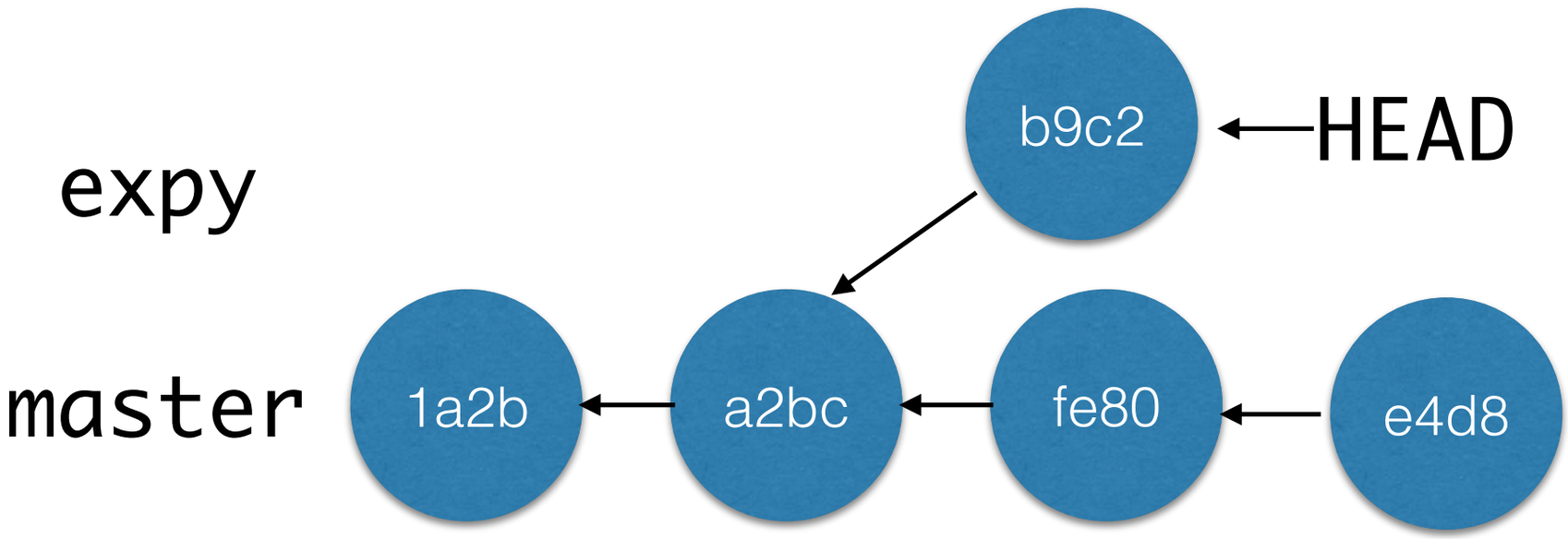
- To merge a branch **to** the current branch

```
git merge target
```

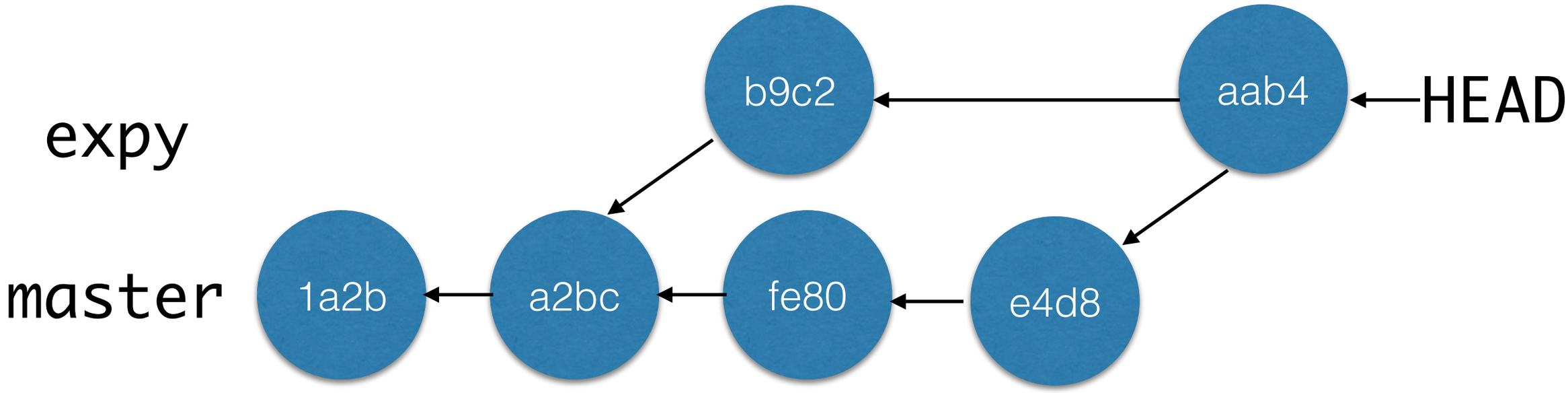
- Leaves the target branch untouched.

- May cause “merge conflicts” which must be resolved manually.

Merging



`git merge master`



Viewing Logs

`git log --since last.month --until 5.days.ago`

```
commit 38baa6430b429c95563083e483caad939e7507c4
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Tue Nov 11 17:07:01 2014 +0800

    Use common submodule

commit 93b6804acc0e5c9b5d5a165974ca79a4cbe71995
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Tue Nov 11 16:52:20 2014 +0800

    Fix permissions for server certs

commit 5017ae46a1e54d52984c53b099fe552ec56c4f4c
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Tue Nov 11 16:16:31 2014 +0800

    Move tester sudo config to sudoers.d

commit d532a1ce0c0c1863b2de643f26eb40e32df1f44c
Author: Thomas Dy <thatsmydoing@gmail.com>
Date:   Tue Nov 11 15:36:01 2014 +0800

    Add sysad aliases for Start/Stop etc
```

Viewing Logs

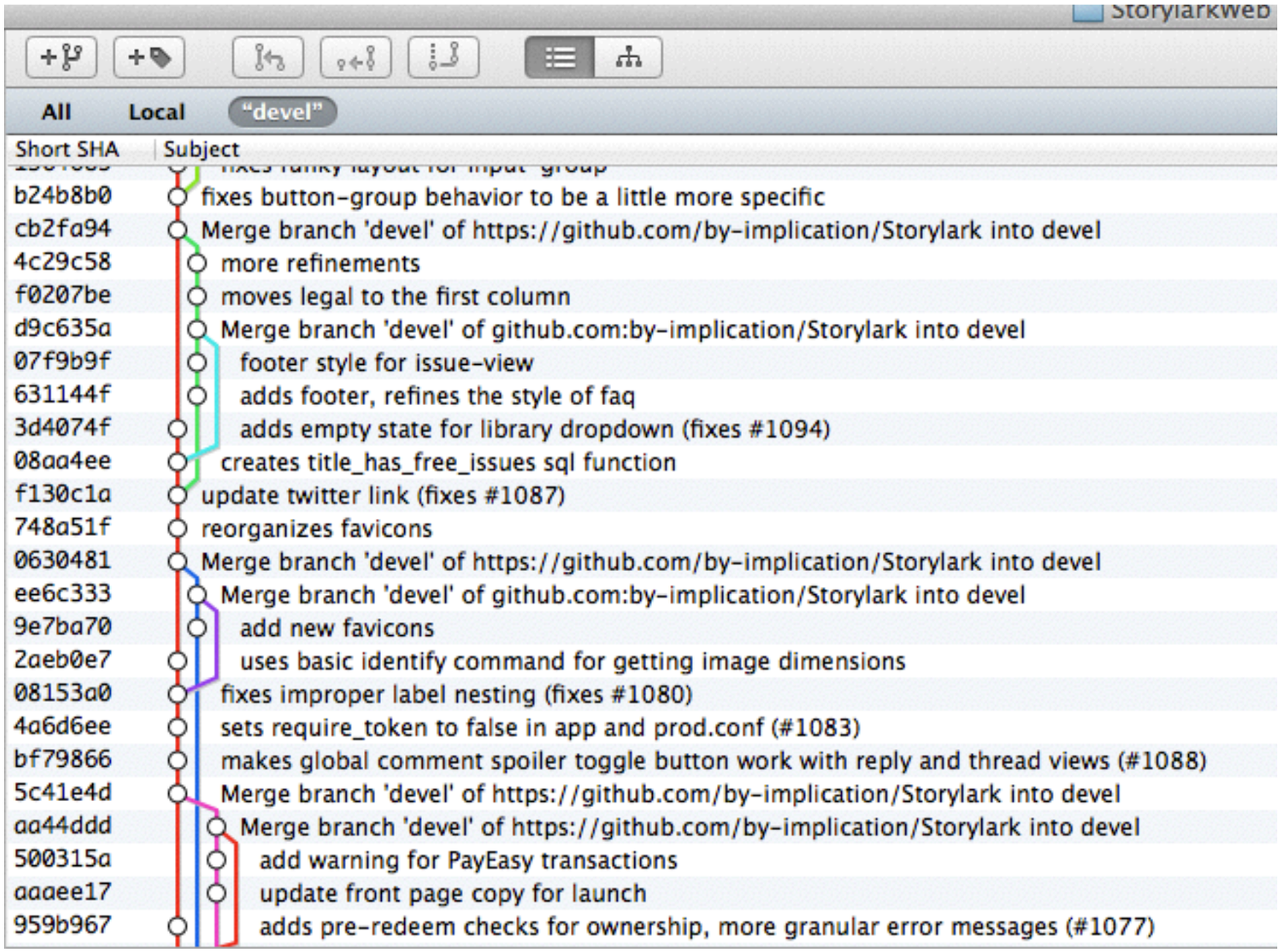
`git log --graph`

(also shows the merge tree)

```
*   commit 469a4f5ed5b3231397dad598db32867a19bae64d
|\  Merge: 1b9fc1a b95ebb3
| | Author: Wilhansen Li <wil@byimplication.com>
| | Date:   Mon Nov 17 17:52:56 2014 +0800
| |
| |     Merge remote-tracking branch 'refs/remotes/origin/centos7' into centos7
| |
| * commit b95ebb37a261305318be7f77adbfc77a3fe60db5
| | Author: Thomas Dy <thatsmydoing@gmail.com>
| | Date:   Mon Nov 17 14:26:41 2014 +0800
| |
| |     Fix /run directory
| |
| * commit 38baa6430b429c95563083e483caad939e7507c4
| | Author: Thomas Dy <thatsmydoing@gmail.com>
| | Date:   Tue Nov 11 17:07:01 2014 +0800
| |
| |     Use common submodule
```

GitX

(or any other Git GUI client)



Referring Commits

- Full commit id: `469a4f5ed5b3231397dad598db32867a19bae64d`
- Partial commit id: `469a4f5ed`
- Symbolic reference (**`HEAD`**, tags, etc).
- Traversals: **`HEAD^`**, **`HEAD@{1}`**,**`HEAD@{yesterday}`**
- See **`git rev-parse`** for more info

```
commit 469a4f5ed5b3231397dad598db32867a19bae64d
Merge: 1b9fc1a b95ebb3
Author: Wilhansen Li <wil@byimplication.com>
Date:   Mon Nov 17 17:52:56 2014 +0800

    Merge remote-tracking branch 'refs/remotes/origin/centos7' into centos7
```

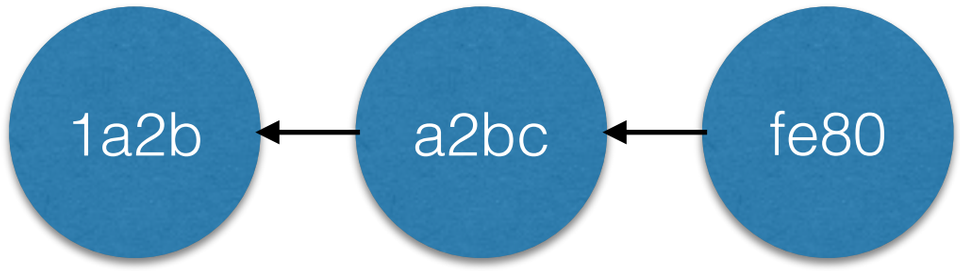
Rewinding Commits

1. Public Apology
2. The Politician
3. Separate Timeline

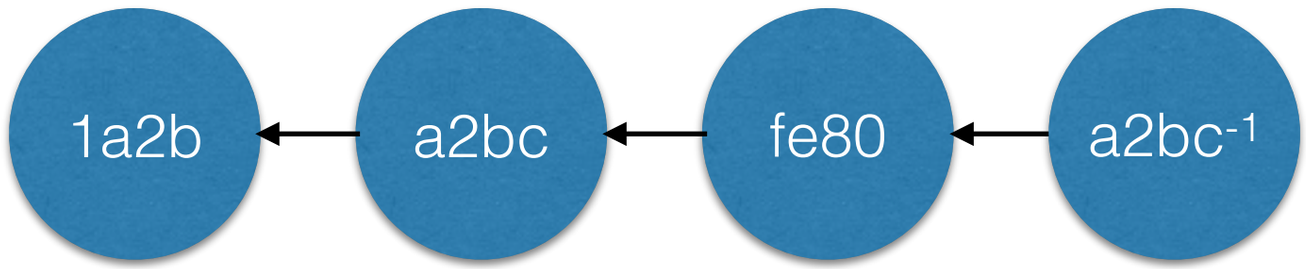
Public Apology

- Issue another commit which reverts a given commit.
- Undos one commit at a time (unless a range is specified).
- Use for commits that went public (“it got out”)
- `git revert <commit id>`
- If the commit is a merge, you need to specify which merge branch you are reverting to using `-m`
  - `1` for merge destination

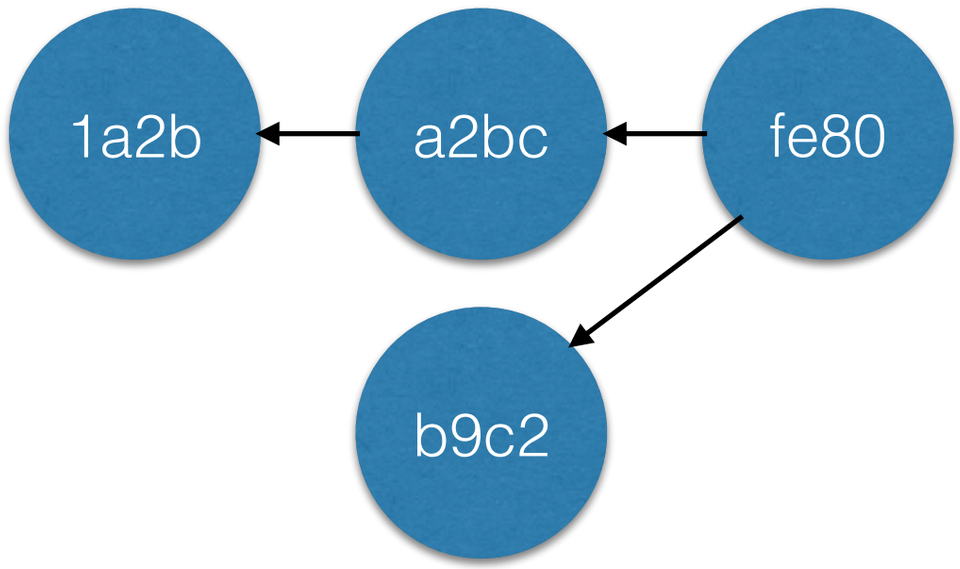
Revert



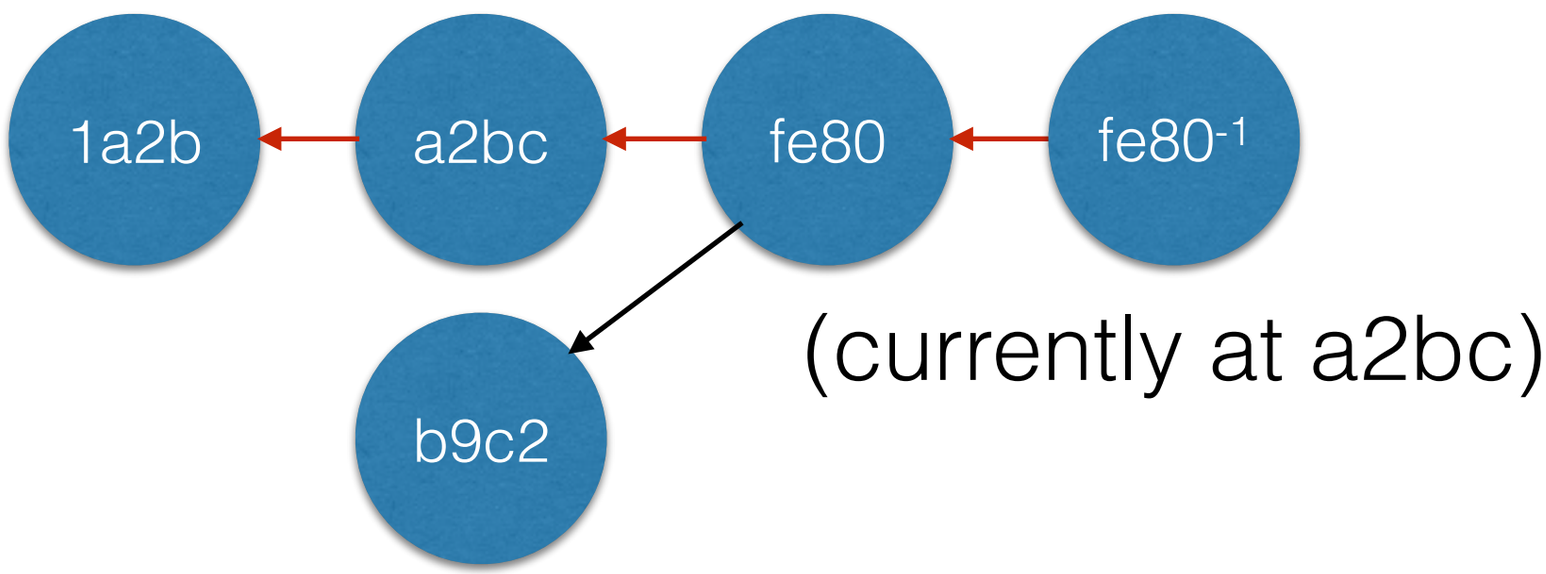

```
git revert a2bc
```



Revert




```
git revert -m 1 fe80
```



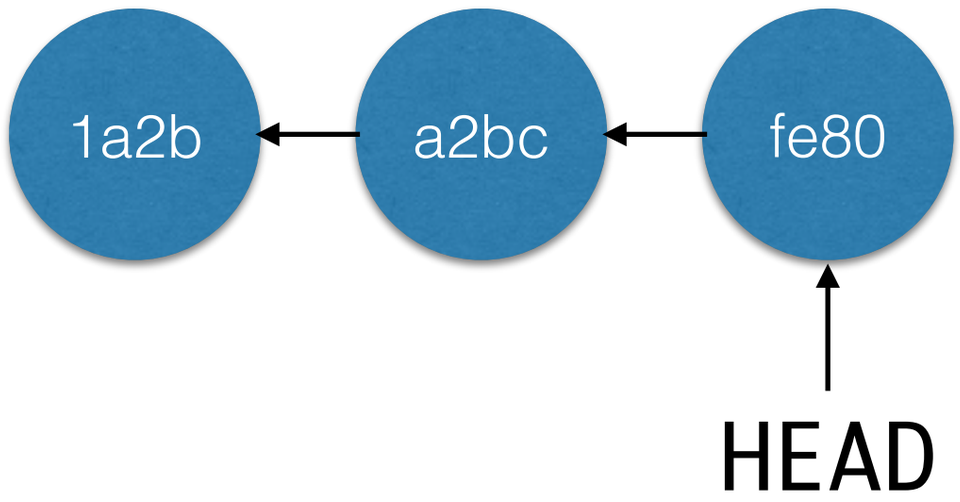
The Politician

- Erase the existence of the commit.
- Not safe if the commit went out to public.
  - Real life tip: Erase your history if you wish but if there was a witness to that… nice try.
  - Usually helpful if you messed something up locally.

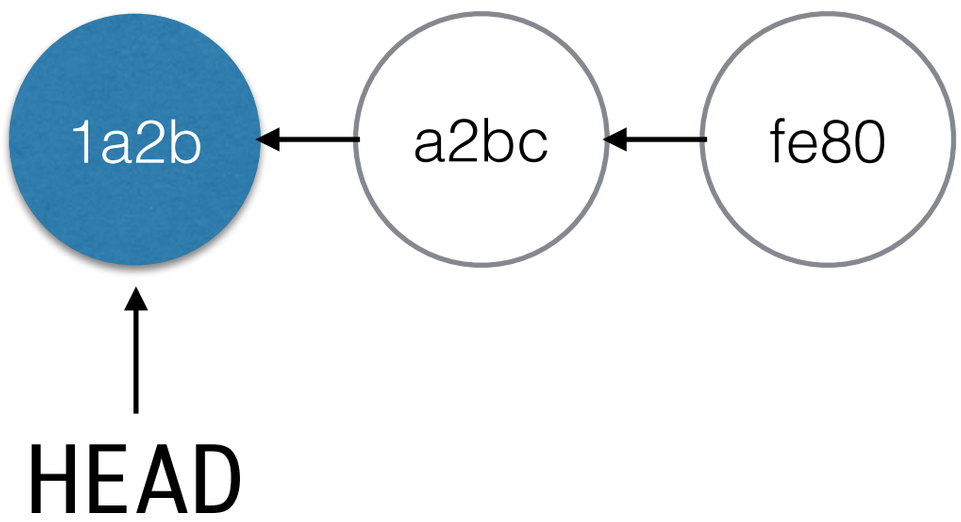
```
git reset <commit>
```

- Reverts from the specified commit all the way to the `HEAD`.

Revert



`git reset 1a2b`



3 modes of reset

- `git reset --hard`
  - Resets everything, including your current data. (dangerous!)
- `git reset --mixed` (default)
  - Leaves your current files untouched, resets the staging area.
- `git reset --soft`
  - Just resets the current `HEAD`, does not touch anything else.

Resetting a Reset

- To view commits that got lost in history (because of resets, deleted branches, etc). Check the ref log (not re-flog).

`git reflog`

- Can be used to reset a reset. (note: reset means re-set)

- In accessible commits are deleted after a certain amount of time (when `git prune` runs).

Separate Timeline

- Create a separate branch off a previous commit.
- Usually useful in order to preview a previous state.
- Doesn’t really undo a commit.

`git checkout <commit>`

- By default, it creates an “anonymous branch”.
  - Any commit done to it disappears when you switch out.
  - Use the `-b` flag (or do a `git branch`) to name it.