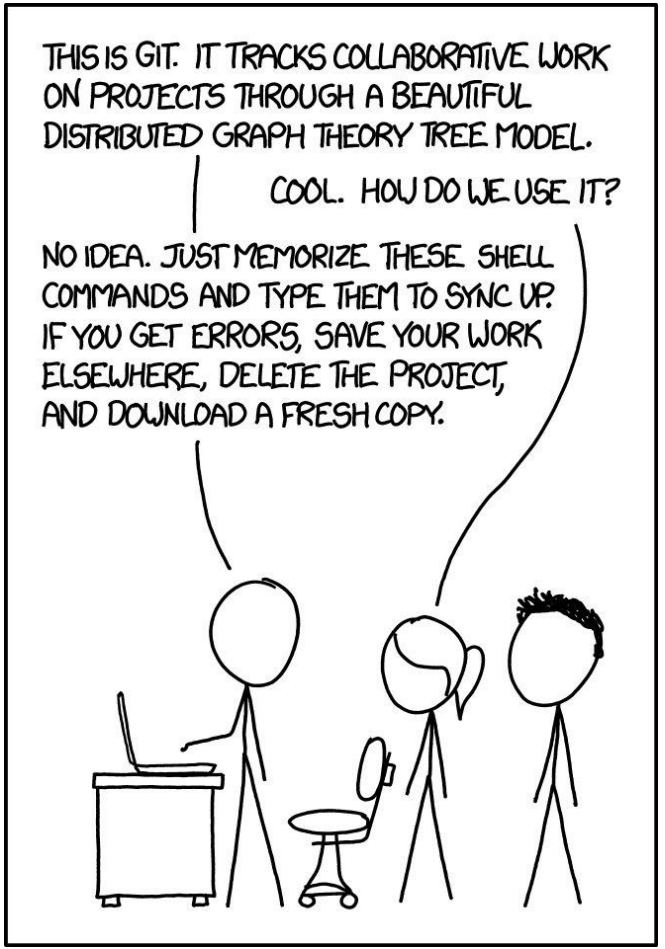
Advanced Git

“It's not that hard, you just need to git good”

Outline

- Quick flashback for those who missed basics
- Some theory about internals
- Demystifying rebase and merge
- Practical Session
  (some awesome stuff with git)
- Anecdotes



Flashback

Find the previous session's slides here.

- ➔ **git clone / git init**
- ➔ **git pull**
- ➔ **git add**
- ➔ **git status**
- ➔ **git commit**
- ➔ **git push**

What is git ?

What is github ?

Inside git

“May the forks be with you”

Plumbing

Vs

Porcelain

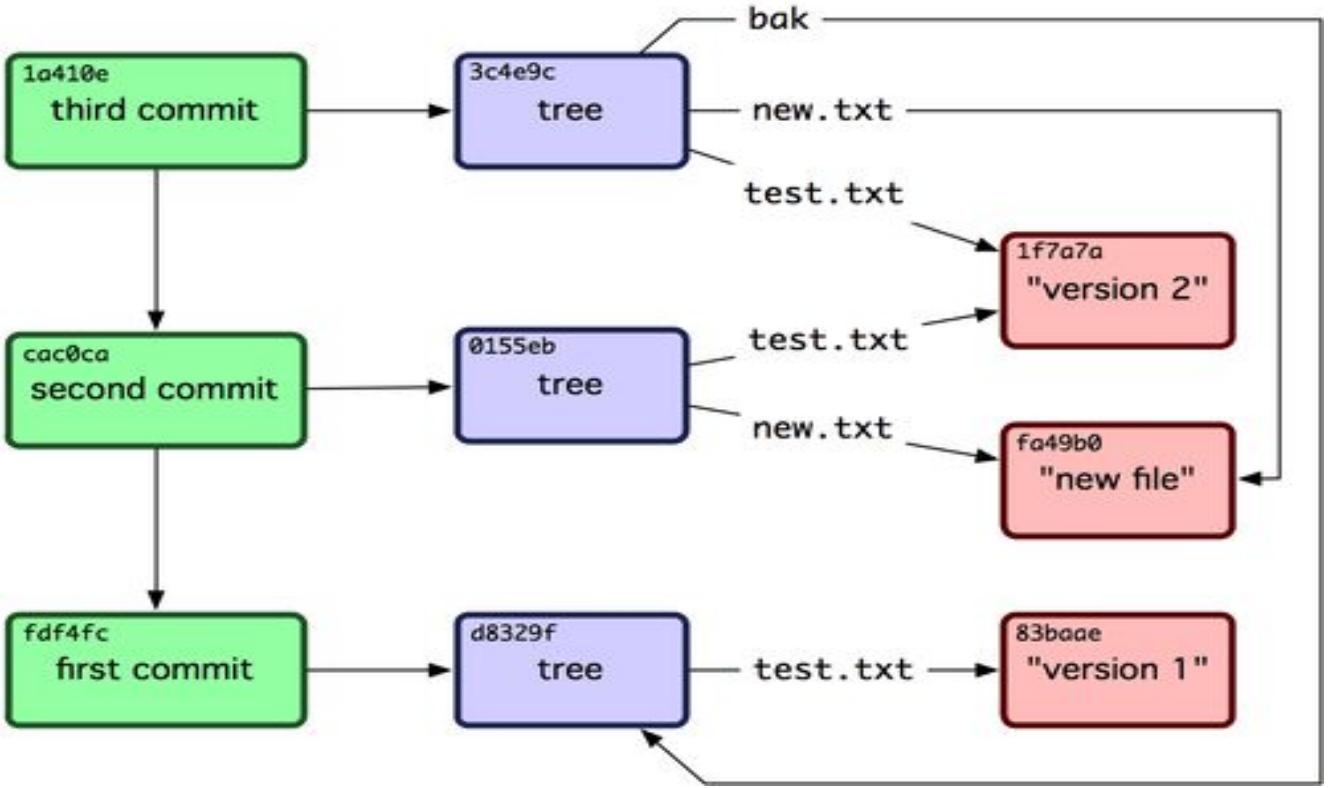
Inside git: The data structures used

- Index
  The staging area

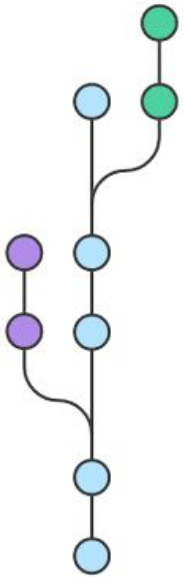
- Objects
  (Black magic)
  - Blobs: Content of a file
  - Tree: Refers to other objects
  - Commits: Links the trees
  - Tags: Refers to (objects + info)

```
git init

echo "Hello World" > firstfile.txt

git add firstfile.txt

git commit -m "Commit 1"

find .git/objects/ -type f

git cat-file <an sha of your choice>
```

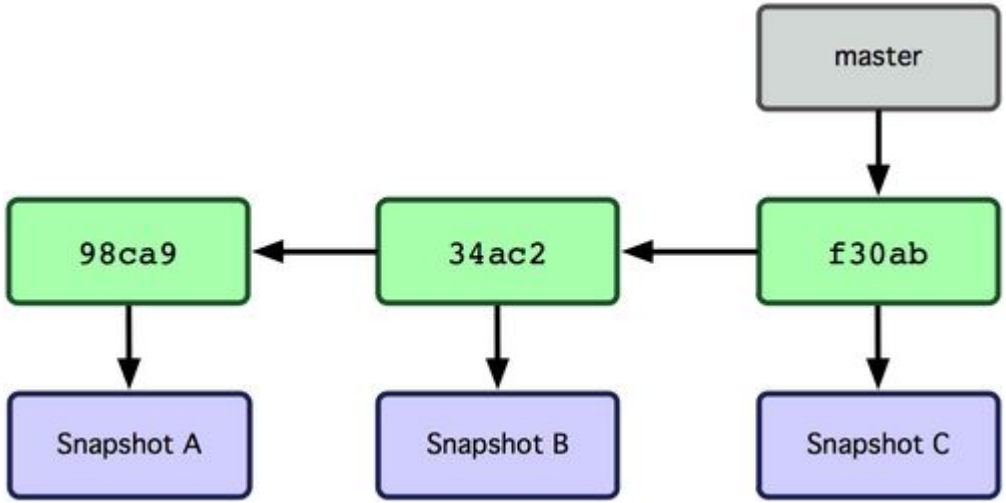
Inside git: The data structures used



Inside git: Branches



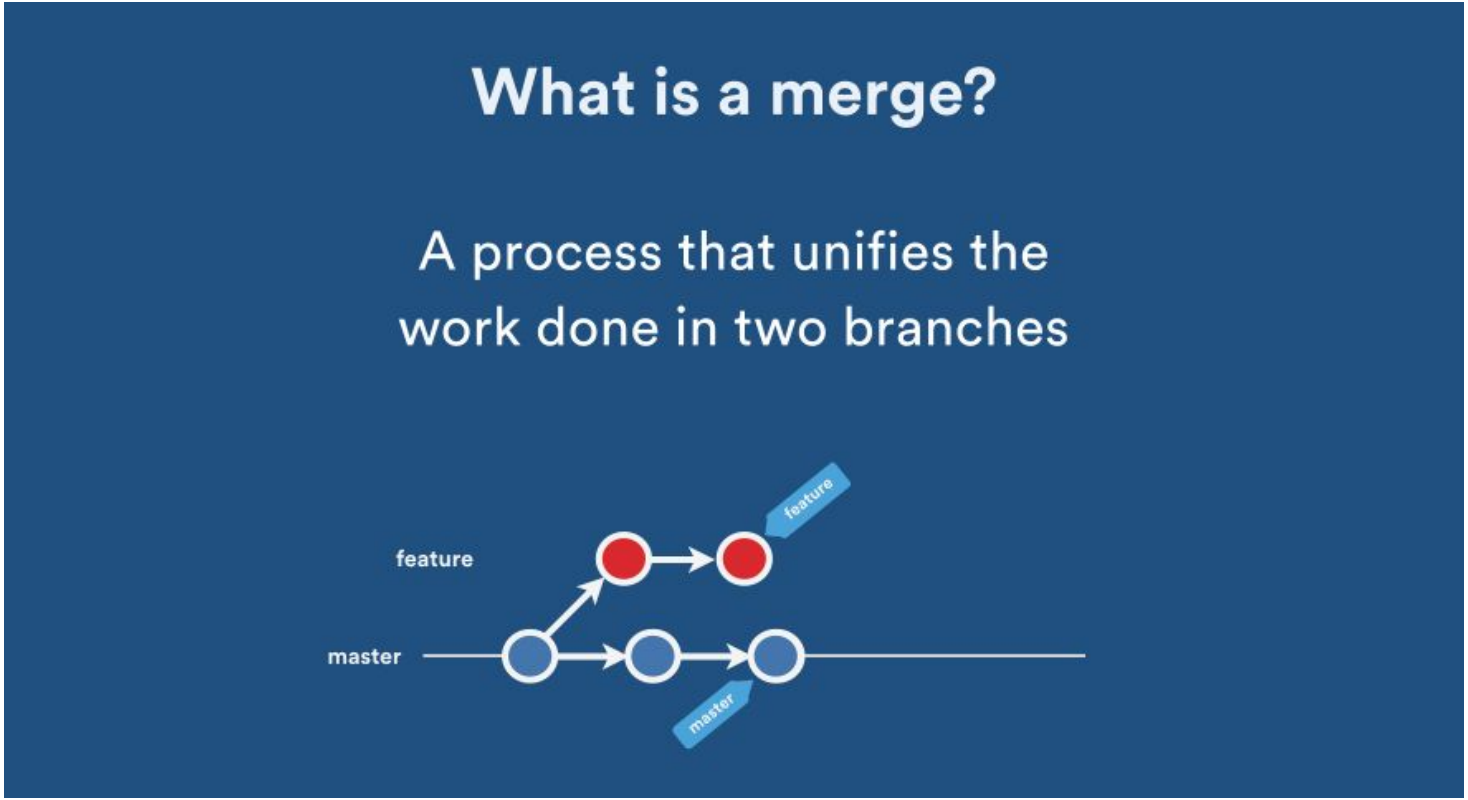
Branches are now simply movable pointers to the commit objects.



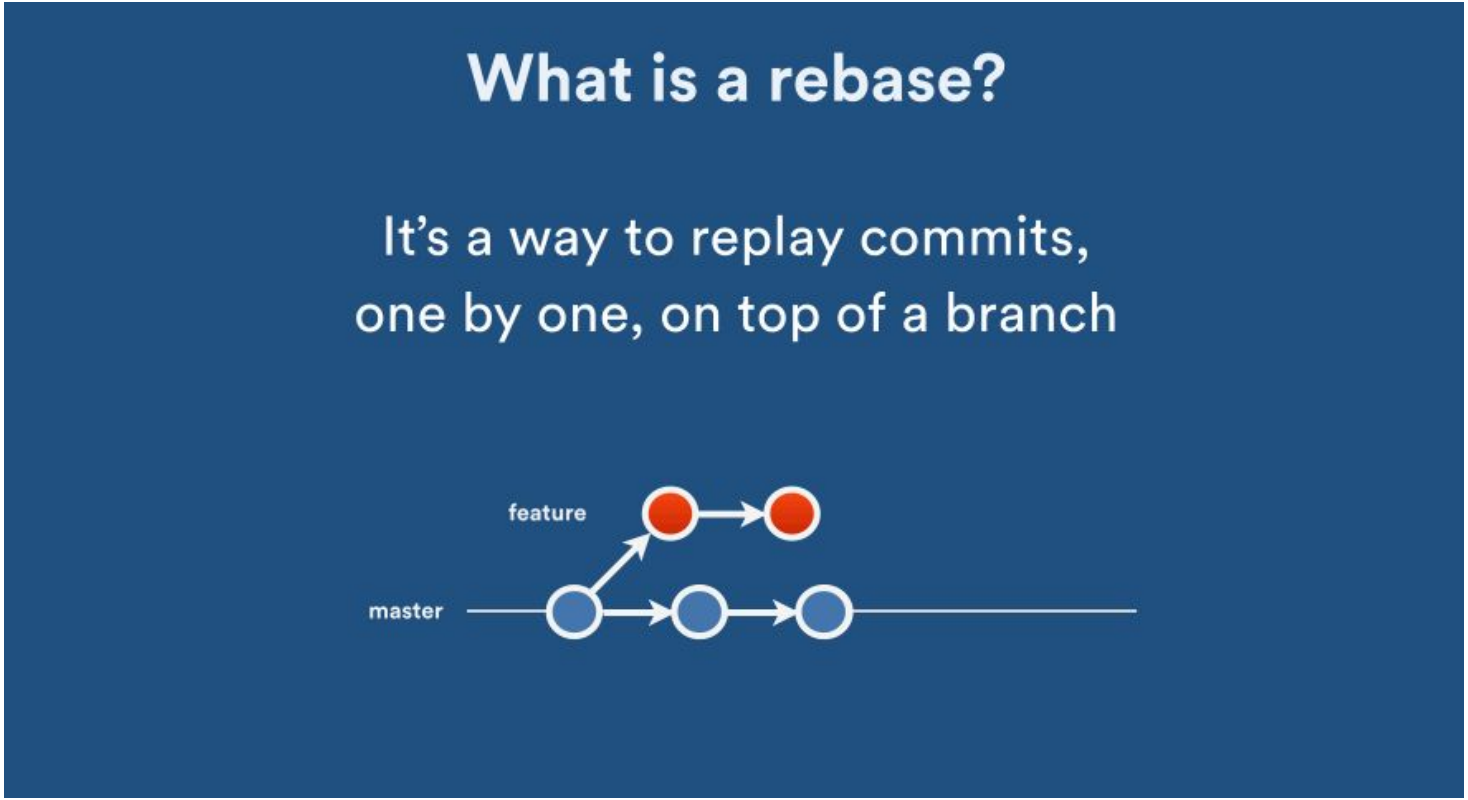
Combining Code

Rebasing and Merging

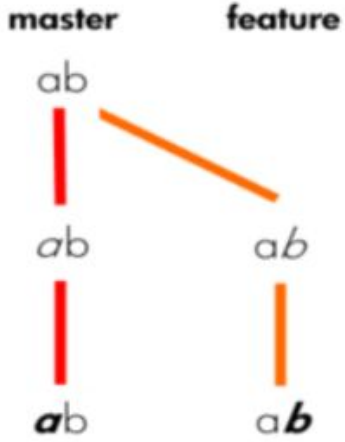
Merging


What is a merge?
A process that unifies the work done in two branches
feature
master
feature
master

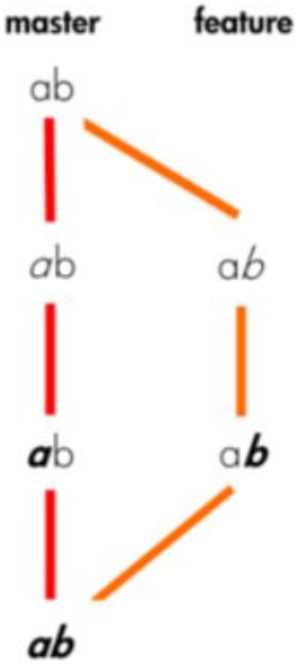
Rebasing


What is a rebase?
It's a way to replay commits,
one by one, on top of a branch
feature
master

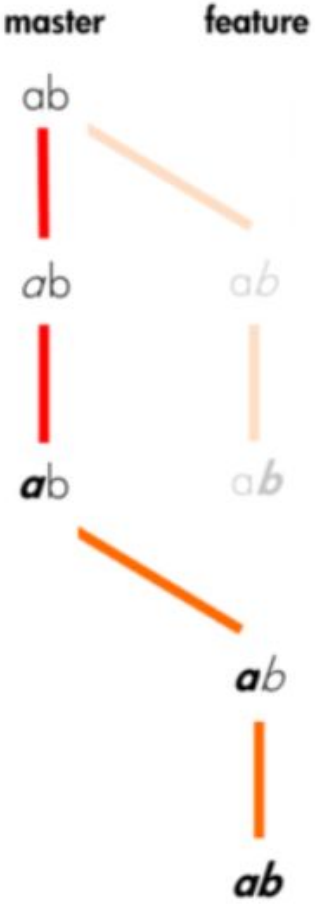
Commits



Merge



Rebase



Merging vs Rebasing

Merging:

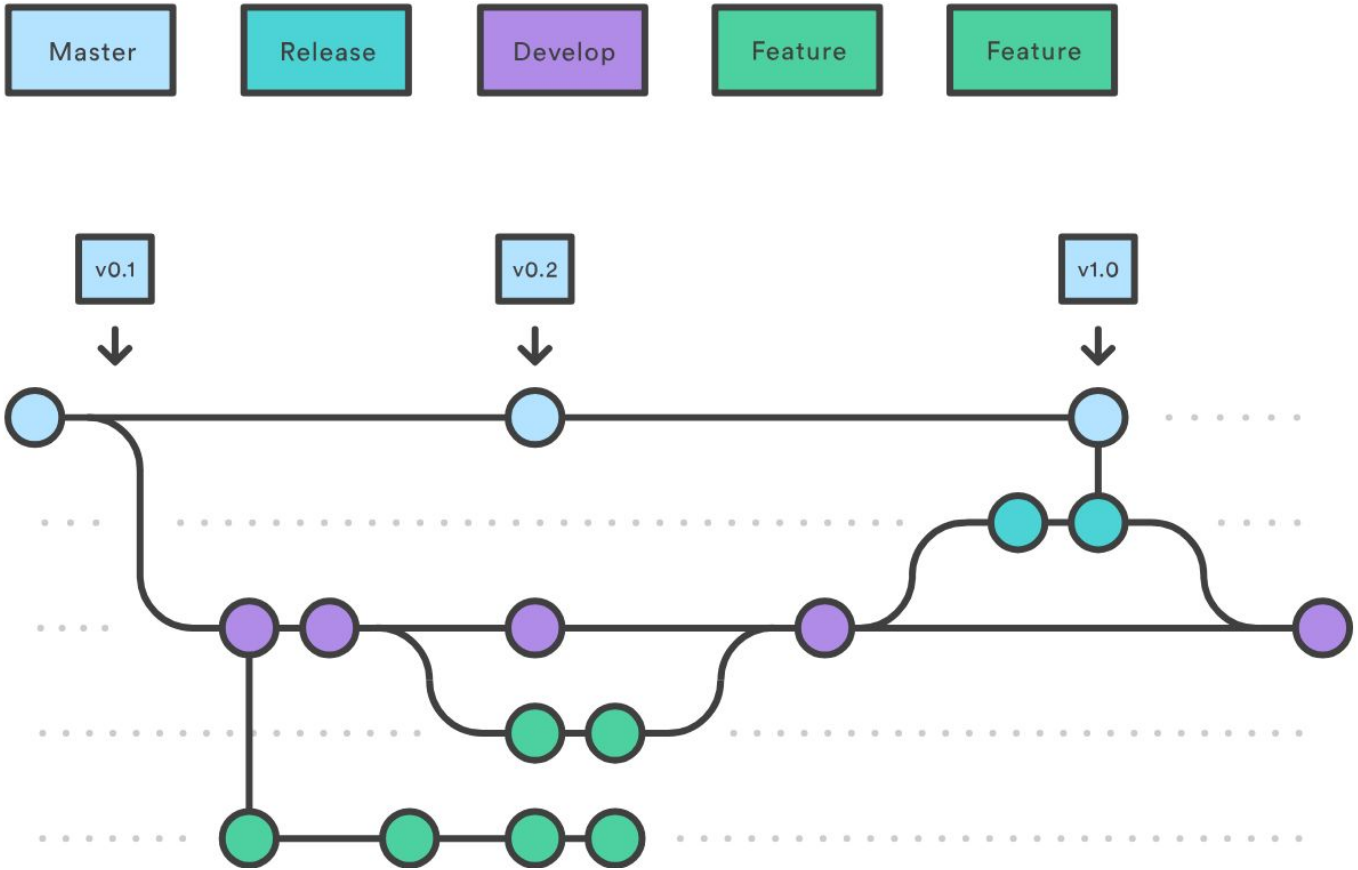
- Doesn’t change any hashes
- Creates a messy git history
- Leaves an extra merge commit
- Can have more than one parent commits
- Completely non-destructive and thus it’s the default

Rebasing:

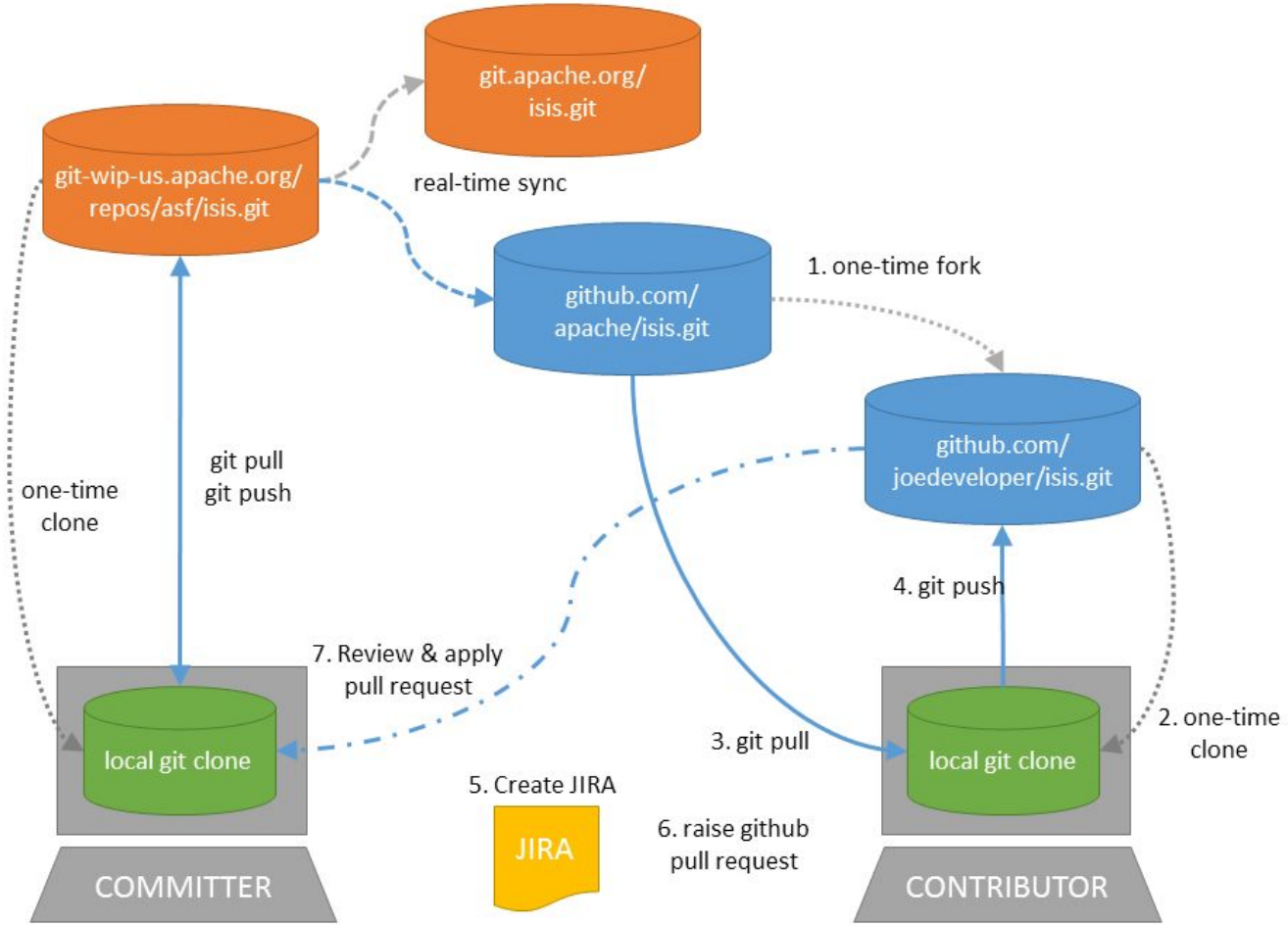
- Changes hashes of all new commits
- Creates a beautiful linear git history
- No extra merge commit
- Always a single parent
- Destructive operation, thus needs people who know more about git

Git Workflows

Gitflow



Fork flow


git.apache.org/
isis.git
git-wip-us.apache.org/
repos/asf/isis.git
real-time sync
github.com/
apache/isis.git
1. one-time fork
github.com/
joedeveloper/isis.git
one-time
clone
git pull
git push
4. git push
7. Review & apply
pull request
local git clone
3. git pull
local git clone
2. one-time
clone
5. Create JIRA
JIRA
6. raise github
pull request
COMMITTER
CONTRIBUTOR

Let's git our hands dirty

What we will do

- ➔ **Work using the Fork flow**
- ➔ **Connect local repo to our remote repos**
- ➔ **Fix an obvious bug**
- ➔ **Rebase and deal with merge conflicts**
- ➔ **Make a PR from a branch in our fork to master branch**

Connecting git to a service

Collaborating code with other people !
Done using remotes.
We will use https://github.com/virresh/advgit2019w.git (or bit.ly/advgit2019w)
And connect to that from our local repository.

Story:
BatMan has made a feature for that site (in a branch my_feature_1) .

We need to make a PR to include that

```
git init

git remote add upstream
https://github.com/virresh/advgit2019w.git

git remote add origin <your fork link>

git fetch upstream

git merge upstream/master --ff-only

git checkout my_feature_1
```

The button doesn’t work

So we, as the maintainer of our fork, have this responsibility pressed upon our shoulders to fix that and create a pull request to our master branch.

The bug seems easy to fix, Look at the closing head tags, a wrong jquery include and we’re good to go.

But, as you shall see, rebasing or merging will land you with merge conflicts. Let’s deal with them !

```
git fetch upstream
# merging gives conflicts
git merge --ff-only master
git merge --abort

# rebasing gives conflicts too
git rebase upstream/master
```

What to do ?

Resolve conflicts manually !

And then push your tree ! (often forcefully)

Let’s visualise our repository

There are lots of tools out there such as gitkraken and github desktop, but we will stick to the minimalistic tool gitk and git gui for this purpose.

```
#this suffices to observe the current branch
gitk

#This can do a big subset of what command line can
git gui
```

Is git all there is ?

No !

There are alternatives like SVN, mercurial, Bazaar etc out there too !

Lots of service providers are out there too such as GitHub, BitBucket, Gitlab, Launchpad (And many of them also run on many types other than git such as hg or svn !)

Good Reads

These slides: http://bit.ly/advgit2019w
More about workflows: https://gist.github.com/blackfalcon/8428401
Origin of the name of git: https://github.com/git/git/blob/master/README.md
Emoji Commit Sample guide: https://github.com/slashsBin/styleguide-git-commit-message
Gnome's Commit guidelines: https://wiki.gnome.org/Git/CommitMessages (There are different guidelines for different organisations though)

Acknowledgements and references

https://medium.com/datadriveninvestor/git-rebase-vs-merge-cc5199edd77c
https://medium.com/@tedwu_55861/how-to-git-rebase-like-a-boss-c90abe110e95
https://www.atlassian.com/git/tutorials/using-branches/merge-strategy
https://git-scm.com/
https://svn.apache.org/repos/infra/websites/production/isis/content/contributing.html

Upcoming Events

- First Hackathon - 12th Jan
- HackTogether - 15th Jan
- PyGame by Chirag and Aman - 22nd Jan
- Darwin Dev - 29th Jan
- Intro to ML by Kanika - 5th Feb

Feedback

http://bit.ly/advgit2019