git

Git History

- Created by Linus Torvalds for work on the Linux kernel ~2005
- Used by:

  Nearly everybody at this stage…

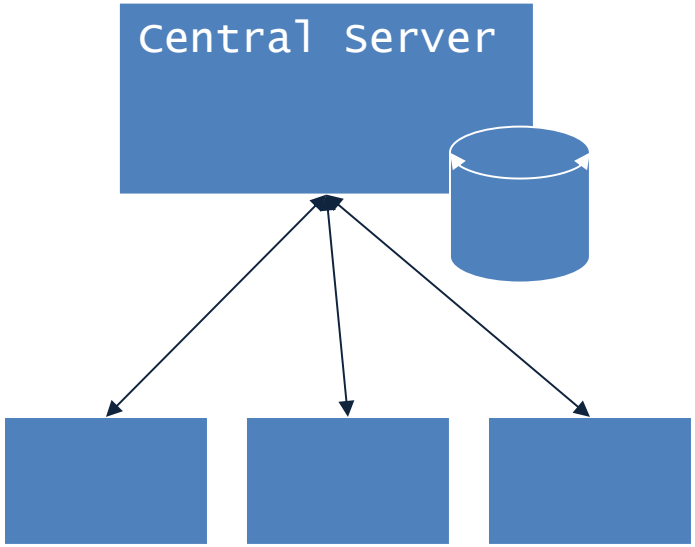
What’s Git

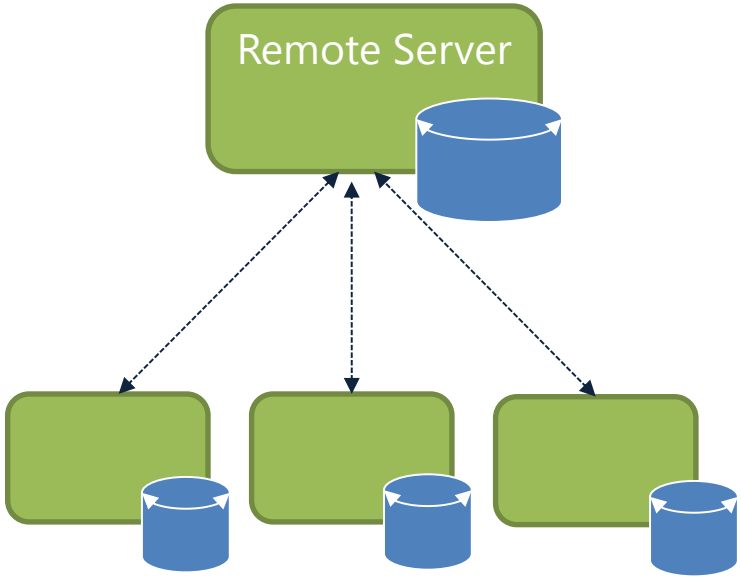
- Distributed Version Control
- Directory Content Management
- Tree Based History
- Everybody has complete history

Distributed Content

- Everyone has their own copy
- Work Offline
- No Central Authority
  - Except by mutual agreement
- Changes can be shared without a server…
  - Can be configured to work peer to peer
  - Can keep collaborating even if server is gone…

Centralised vs Distributed



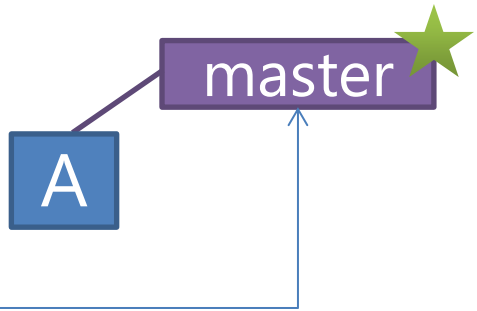


Branching

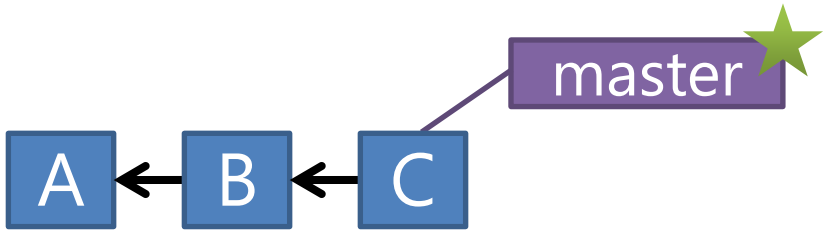
- Like a label on a graph node
- All branching takes place in the same folder/directory
  - Things might appear to disappear depending on what branch you work on…
- You can switch branches
  - Analogous to moving label from one node to another

Initialising a repo…

```
[ec2-user@ip-10-34-209-81 ~]$ mkdir myproject
[ec2-user@ip-10-34-209-81 ~]$ cd myproject
[ec2-user@ip-10-34-209-81 myproject]$ git init
Initialized empty Git repository in /home/ec2-user/myproject/.git/
 git config --global user.name “fxwalsh“
git config --global user.email fxwalsh@wit.com
[ec2-user@ip-10-34-209-81 myproject]$ vi README.txt
[ec2-user@ip-10-34-209-81 myproject]$ git add .
[ec2-user@ip-10-34-209-81 myproject]$ git commit -m 'initial commit'
[master (root-commit) 7d738f4] initial commit
1 file changed, 1 insertion(+)
 create mode 100644 README.txt
[ec2-user@ip-10-34-209-81 myproject]$
```



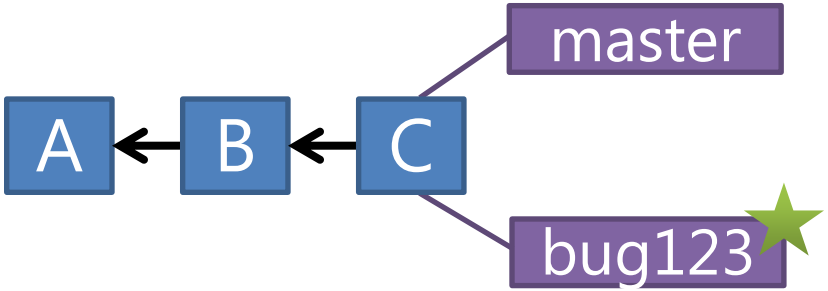
Multiple Commits



git commit –m “updated text file”

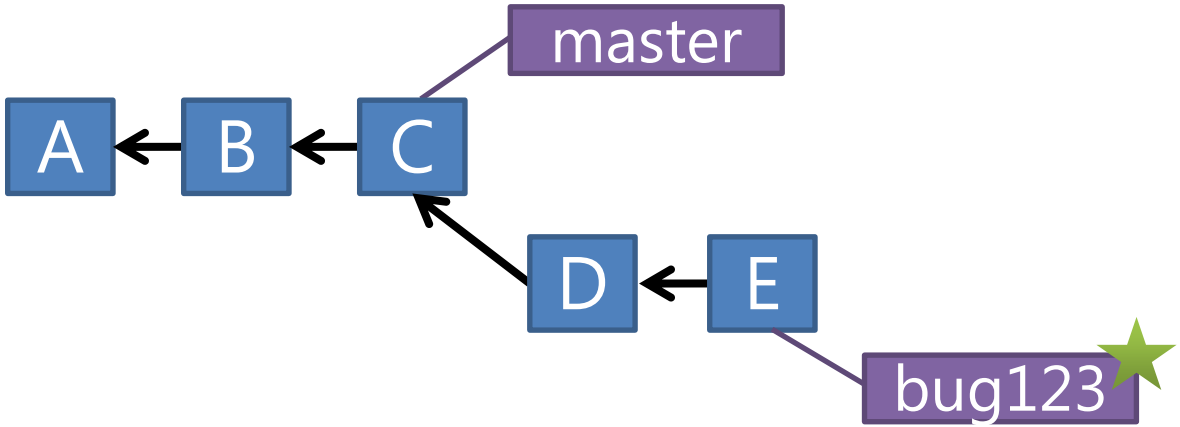
git commit –m “updated text file again”

Branching




```
git checkout -b bug123
Switched to a new branch 'bug123'
```

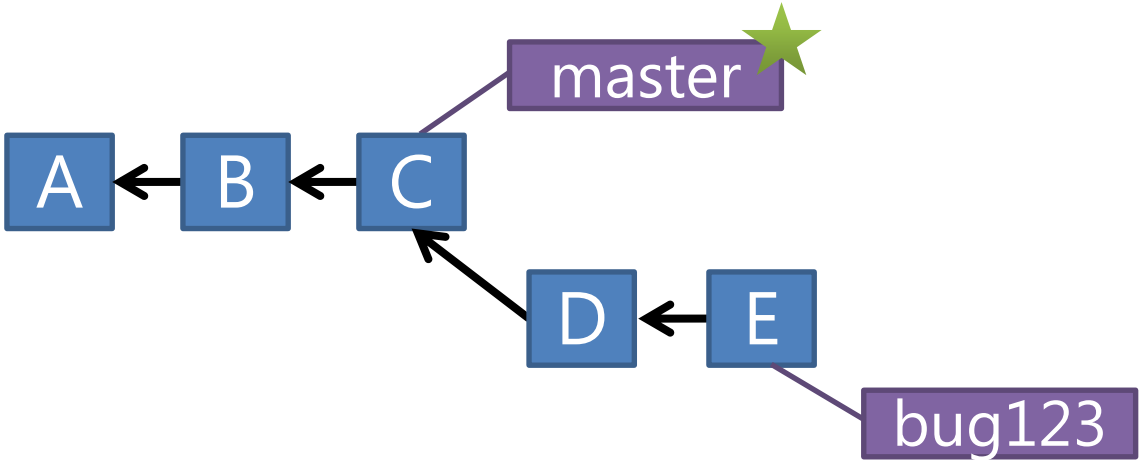
Branching



git commit –m “bug fix”

git commit –m “another code fix”

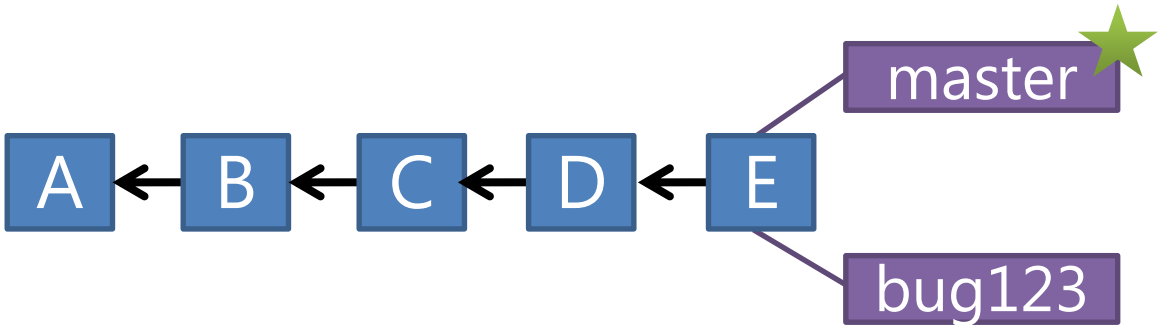
Branching




```
git checkout master
vi README.txt
```

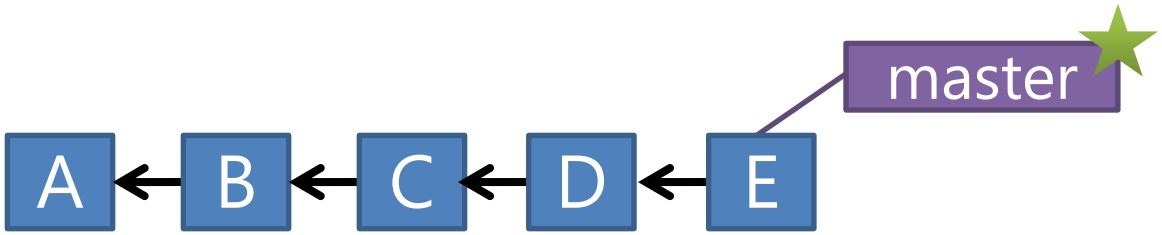
Changes wont be visible…

Branching



git merge bug123

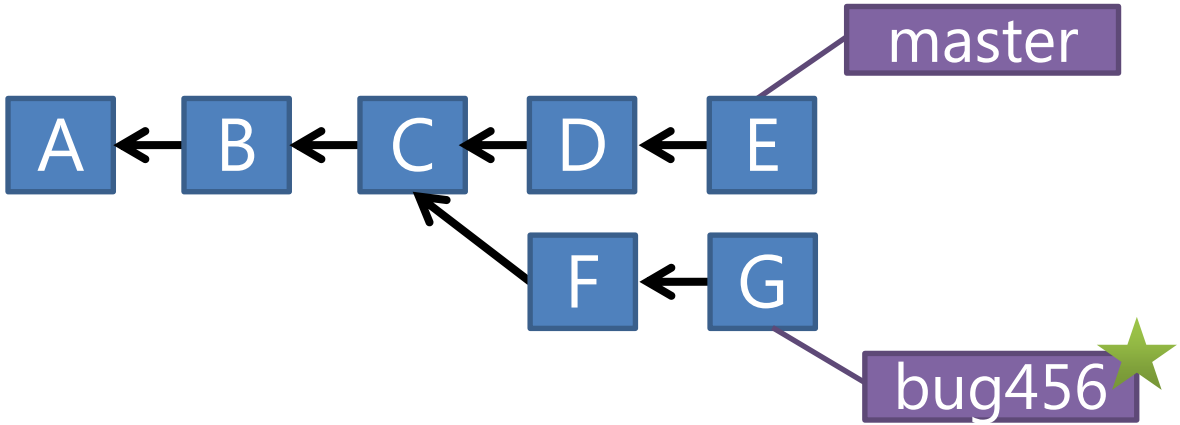
Delete Branch




```
git branch -d bug123
```

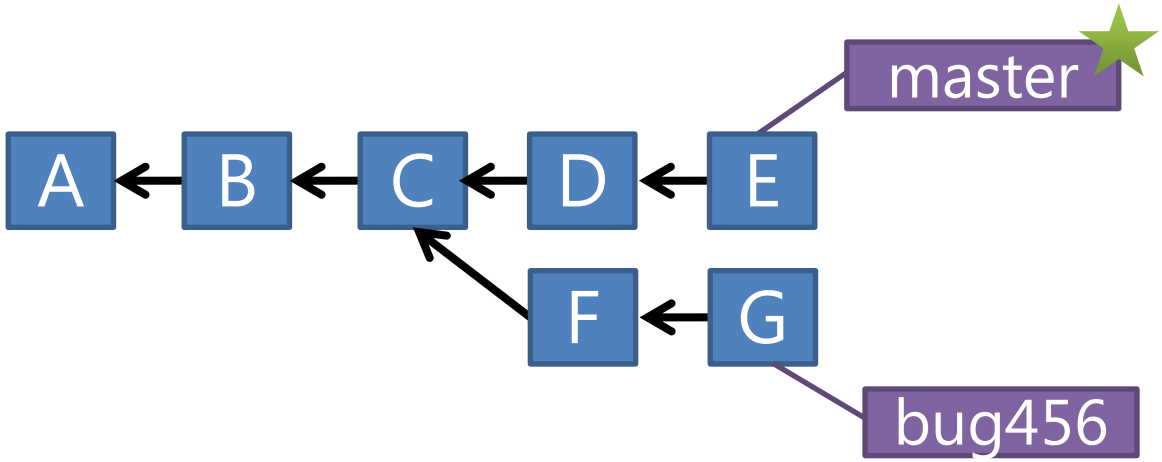
Deleted branch bug123 (was 0e85eb8).

Branching



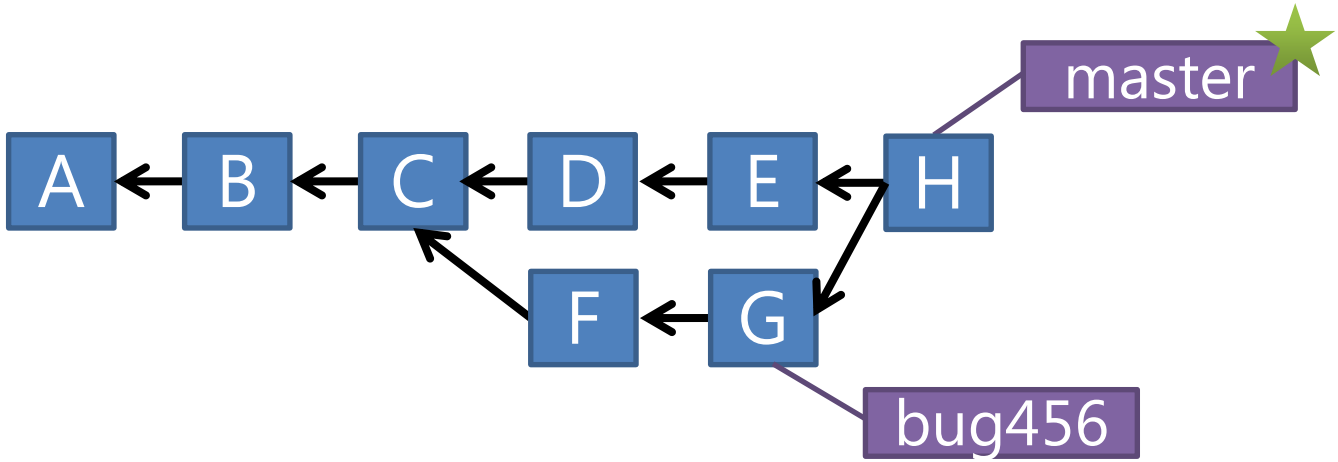
- Suppose another bug branch off of (C).
- Also, changes have happened in master (bug 123 which we just merged) since then.
- Also, two commits in bug456.

Merging



git checkout master

Merging

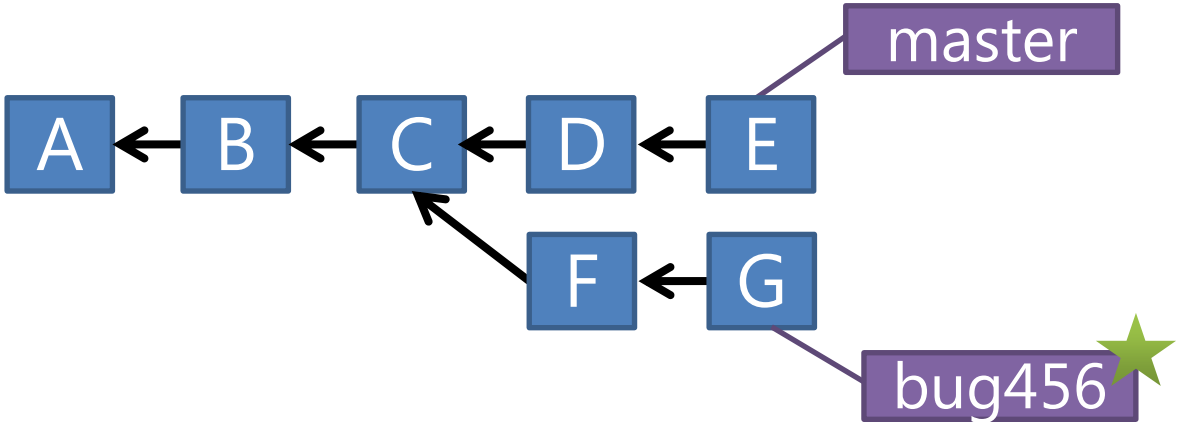


git merge bug456

if there are conflicts, they need to be resolved manually

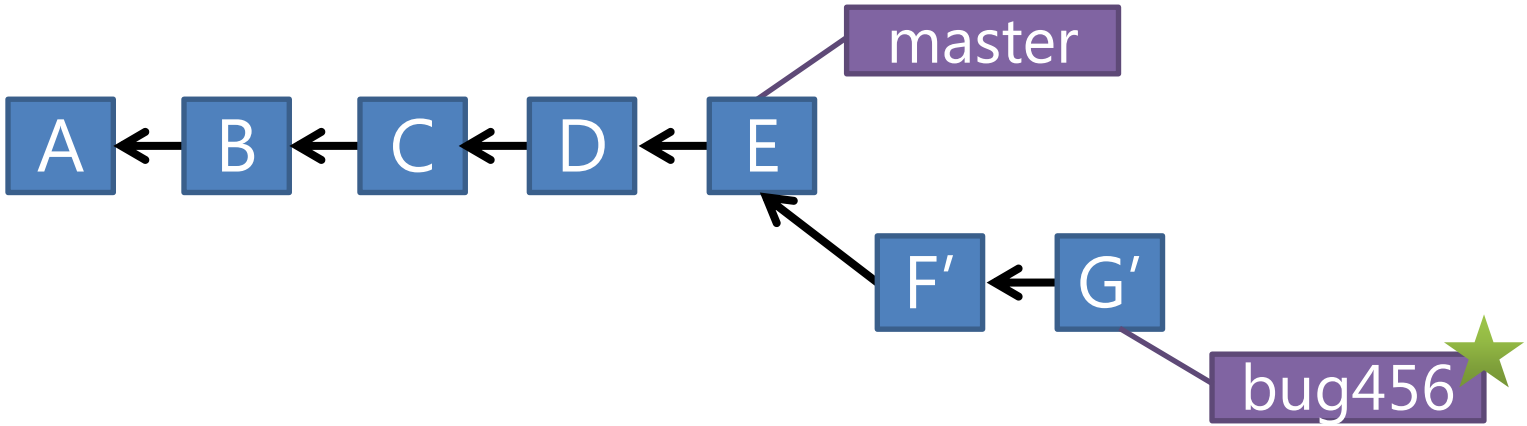
Also deleting the bug456 branch can leave a non-linear, messy structure.

Merging - Rebase



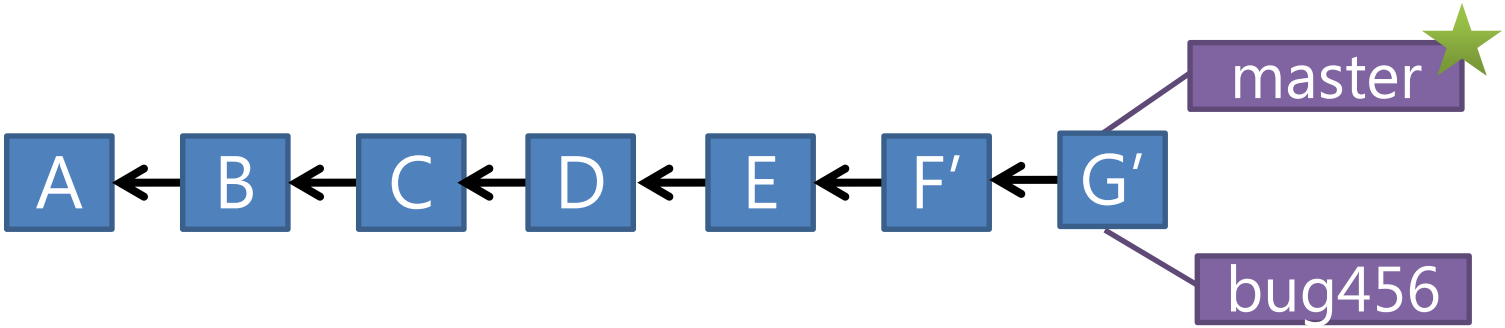
As before, but this time we rebase first….

Merging: Rebase



- Changes on (C) are undone and applied to (E) instead.

Merging: Rebase



git checkout master
git merge bug456

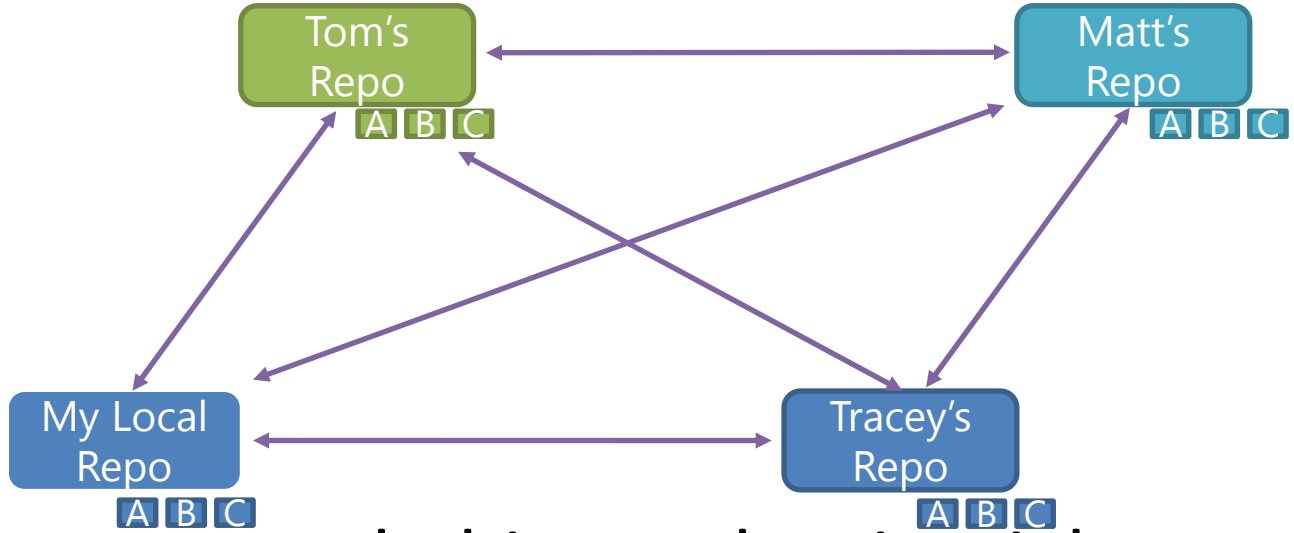
- Linear, causal flow of changes.
- Less snapshots in repository

Branching and Merging: Key points

- Quick and Easy to create ‘Feature’ Branches
- Very capable tool to manage changes
- Rebasing helps keep things clearer.

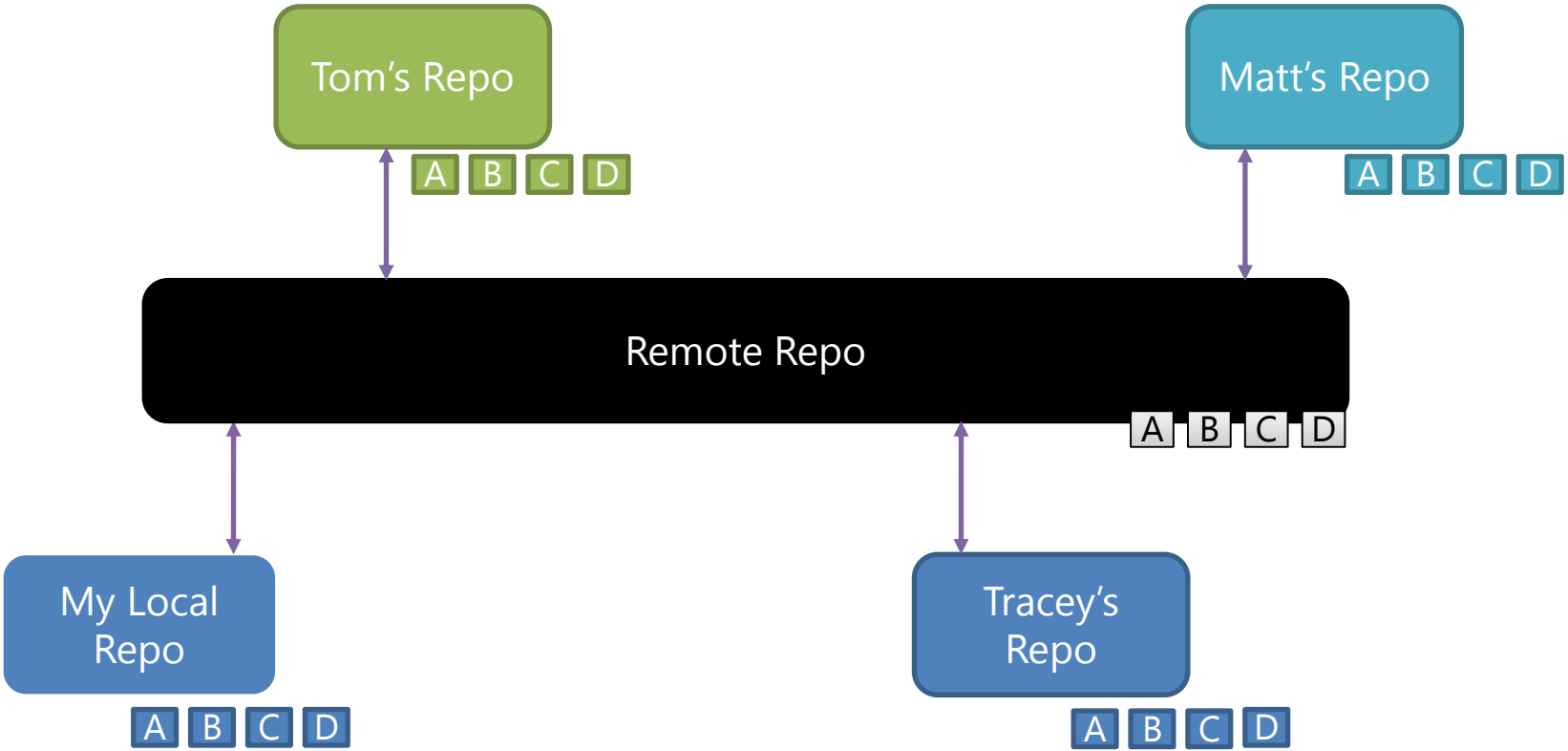
COLLABORATING WITH GIT

Peer – to Peer



- You can work this way but it might get complicated

Central Repository (e.g. GitHub)



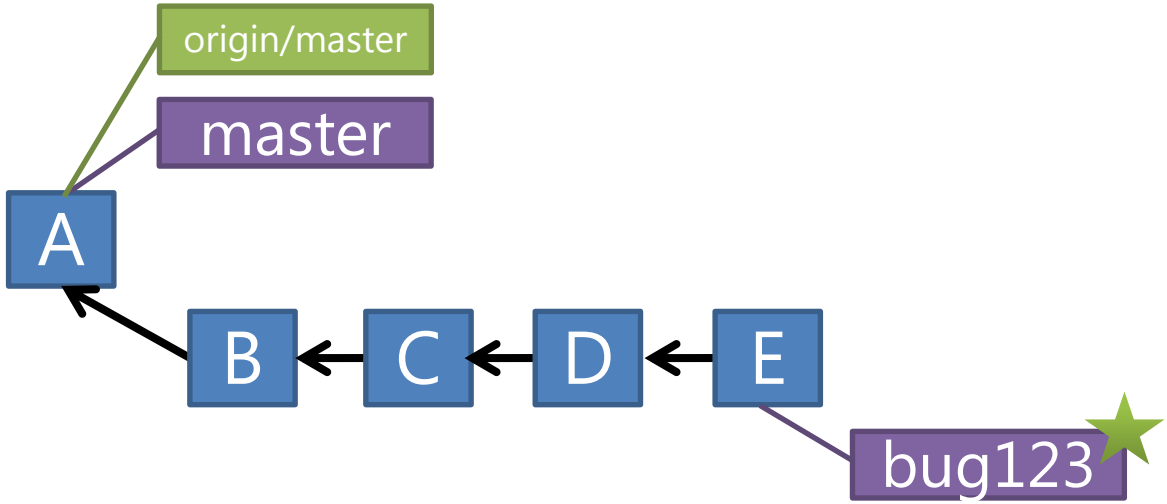
Adding a Remote Repo to Existing Project

```
 git remote add origin https://github.com/fxwalsh/BSc4Repo.git
 git remote -v
origin  https://github.com/fxwalsh/BSc4Repo.git (fetch)
origin  https://github.com/fxwalsh/BSc4Repo.git (push)
```

Setting up Remote via Cloning

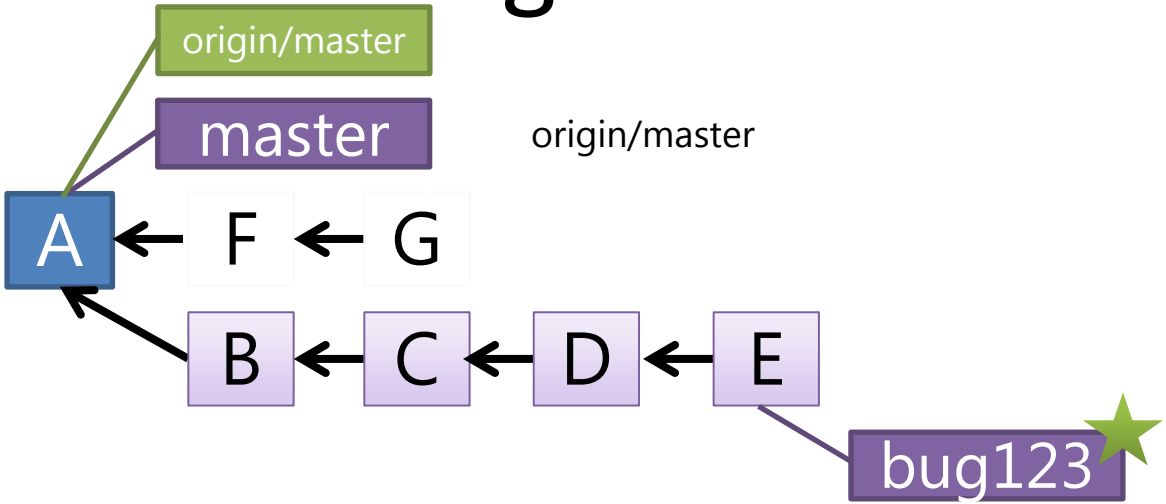
```
 git clone https://github.com/fxwalsh/BSc4Repo.git
……
 git remote -v
origin  https://github.com/fxwalsh/BSc4Repo.git (fetch)
origin  https://github.com/fxwalsh/BSc4Repo.git (push)
```

Branching with Remote



Changes on Bug123 branch are only local.

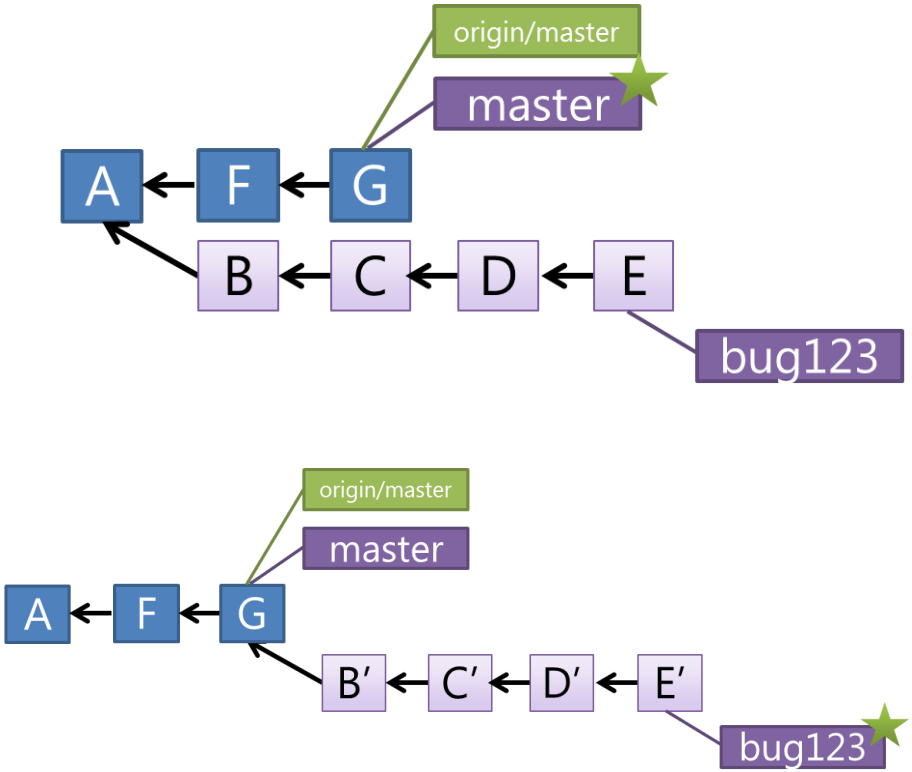
Branching with Remote



- Can have situation where there's two versions of the origin/master
  1. what was last known about the upstream master
  2. what is actually up there (which we don't know about).

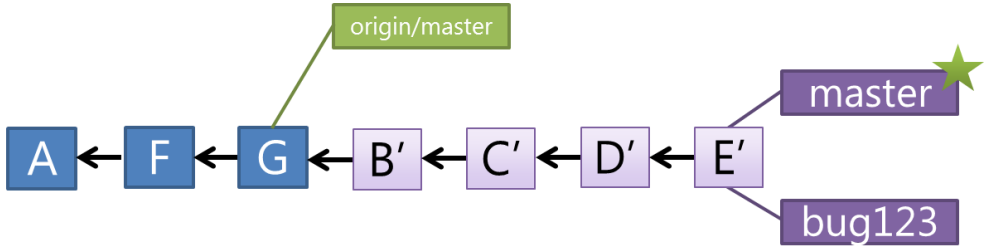
Branching with Remote

- Update Master to what's on remote
  - git checkout master
  - git pull origin
- Rebase the bug123 branch
  - git checkout bug123
  - git rebase

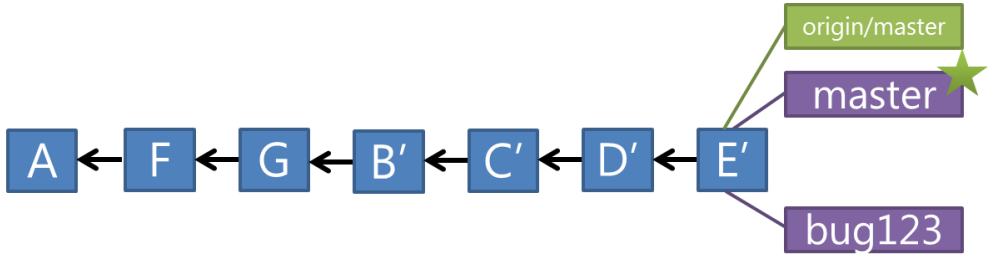


Branching with Remote

```
git merge bug123
```




```
git push origin
```



Push

- Pushes your changes to remote
- Changes will be rejected if newer changes exist on remote
- Good to pull then push
  - merge locally, then push the results.