Git Training

Ralph Ankele
Rachel Player
Ben Curtis

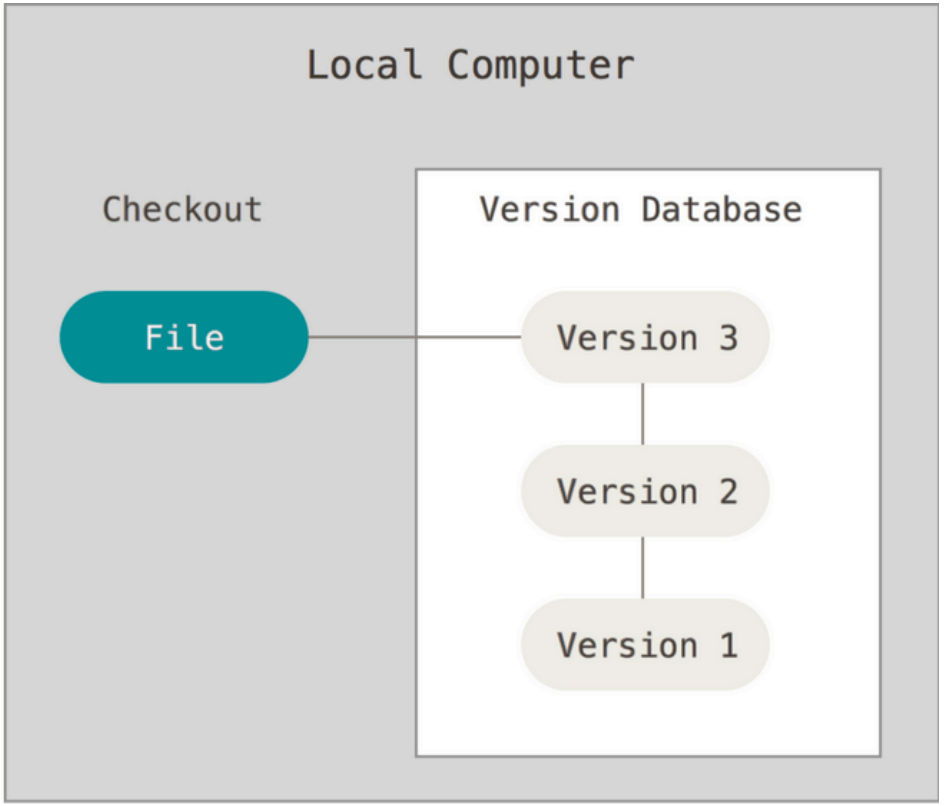
Royal Holloway University of London
27.April 2017

Motivation

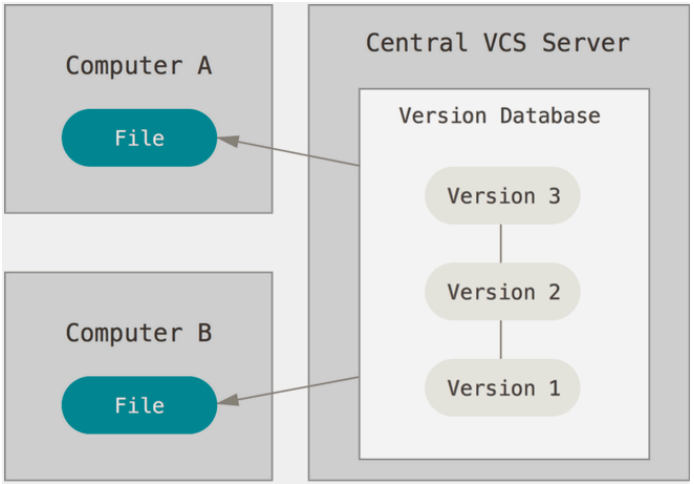
- Imagine a bunch of cryptographers want to write a paper together
  - They are sitting in different offices (all over the world)
  - They want to work together on one/more documents
  - They want to edit the documents at the same time
  - (One of them is a “stupid” PhD student, that makes a lot of mistakes)
- Solution: Exchange files per email?
  - S*** loads of emails
  - Who has the latest version?
  - Everyone has to wait until the other people finish

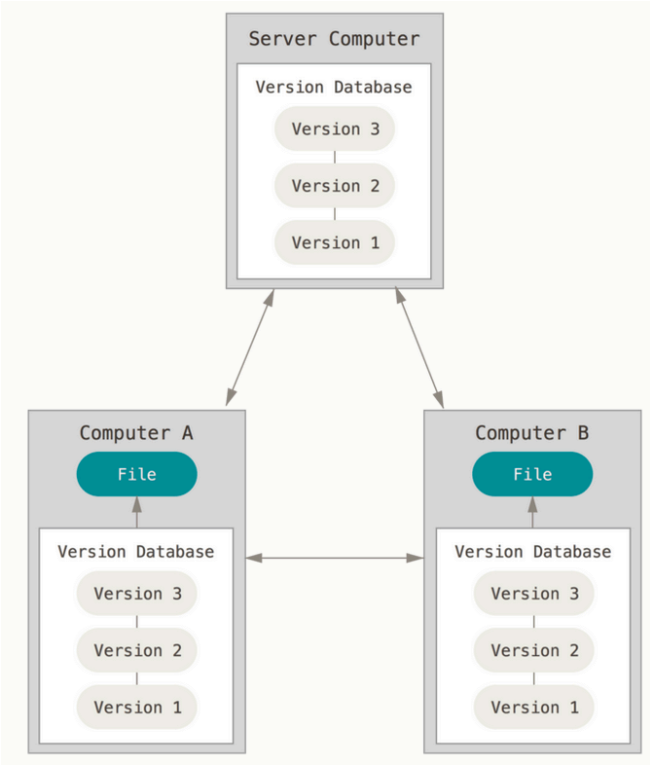
About Version Control

- Store “several copies” of a file
- History of the changes to a file



Version Control Systems





- Centralized Version Control Systems (e.g. SVN)
- Distributed Version Control System (e.g git

Installing git

- Linux:

```
$ sudo apt-get install git-all
```

- Mac OS:

Download from: https://git-scm.com/download/mac

- Microsoft Windows:

```
$ rm -rf C:/Windows/*
```

Installing git

- Windows
  - Download from: https://git-scm.com/download/win
  - Follow install instructions
  - Seriously, consider changing to Linux

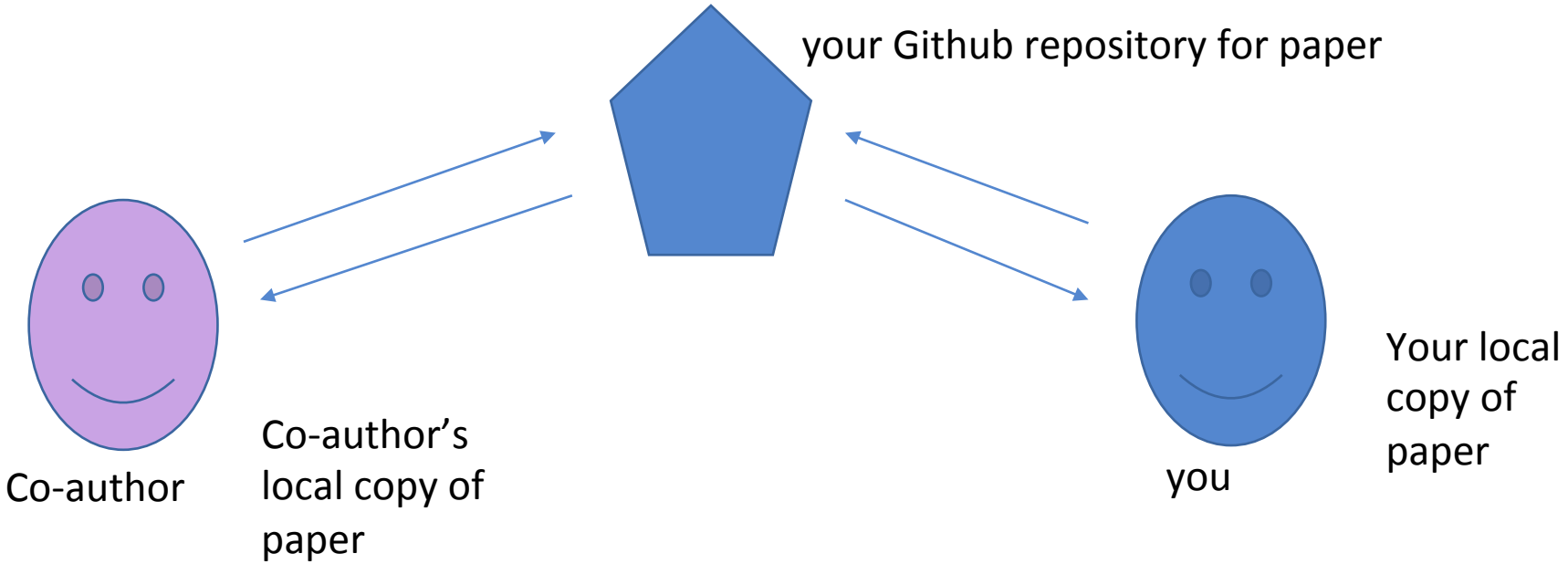
Github repositories

Our workflow:

-One central repository

-All parties commit to

-Fix conflict where needed



First-Time Git Setup

- Configure name

```
$ git config -- global user.name “John Doe”
```

- Configure email address

```
$ git config -- global user.email “johndoe@example.com”
```

- Configure editor for commit messages

```
$ git config -- global core.editor vim
```

- Check settings

```
$ git config – list
user.name=John Doe
user.email=johndoe@example.com
...
```

Basic Commands

git clone

- Get a copy of an existing remote repository on your local machine
- The typical way to start any project

git status

- Lists the files which have been modified since the last commit
- Lists the untracked files in your local directory

git add

- Puts a file in the ‘staging area’ ready for a commit
- You can add several files ready for one commit

```
$ git add test.txt
```

Adds a new file called test.txt to the staging area (which can then be uploaded to the server)

git commit

- Commits the files in the staging area (that have been added with the previous command)
- Add a meaningful commit message so you/ other people understand the change
- Commits are labelled by a hash value (SHA-1)

```
$ git commit introduction.tex –m “refer to [XYZ17] in introduction”
```

This means ‘Commit the file[s] that have been added to the local repository, with the message given after the symbol -m’.

git push

- Upload the committed local changes to the remote repository

```
$ git push origin master
```

git pull

- Download the latest remote change to the local repository

```
$ git pull origin master
```

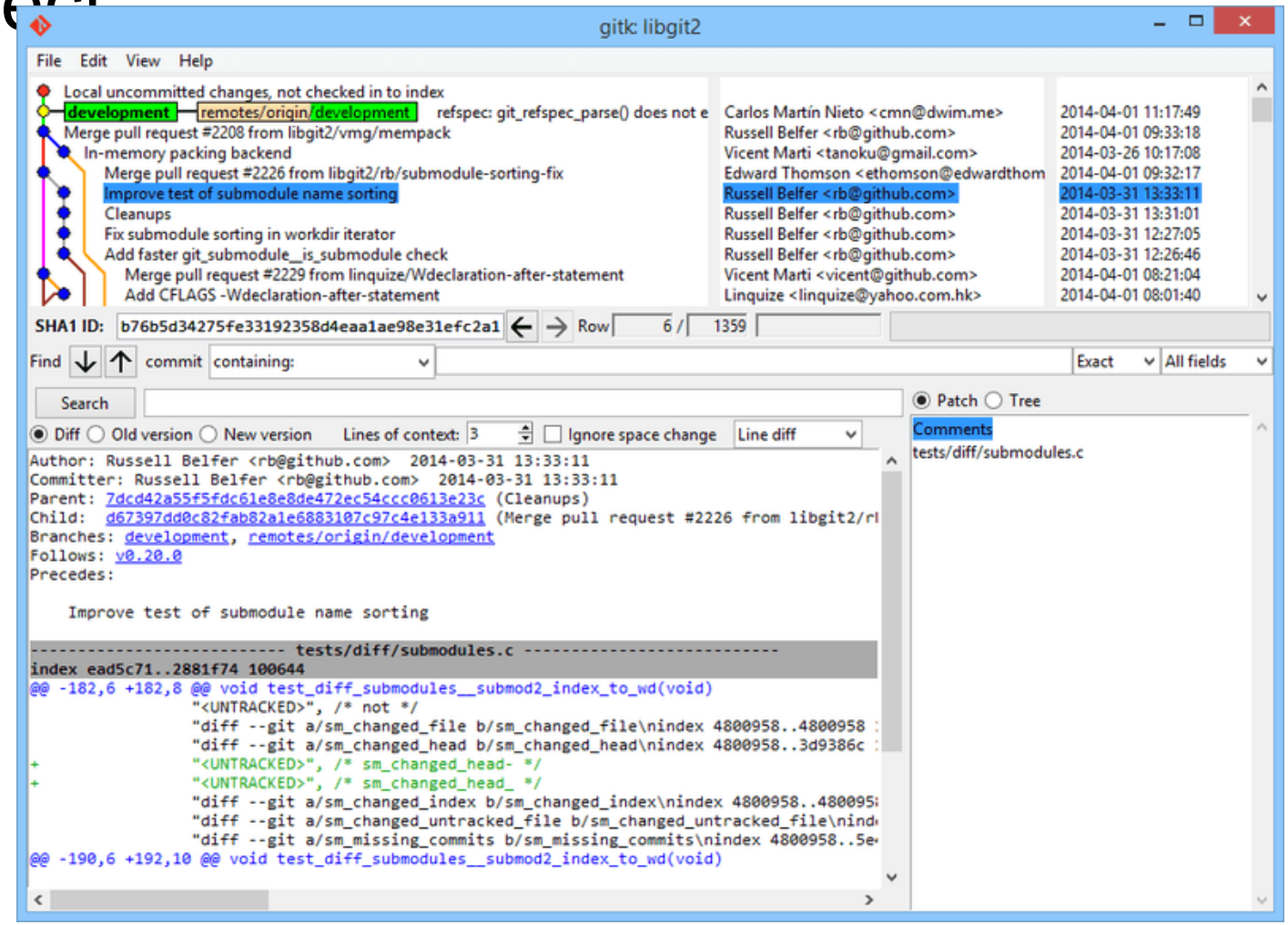
git log

- Shows the history of commits (author/date/ commit message)

```
$ git log --graph
```

gitk

- Another way to visualise the history of the project



git rm

- Deletes a file from the git repository
  - If you delete the local file, but don’t commit the deletion, it can be recovered

```
$ git rm test.txt
```

git mv

- Move/Rename a file

```
$ git mv test.txt introduction.txt
```

This renames the file test.txt to introduction.txt

git checkout

- Revert a file to a version of the file from a previous commit

```
$ git checkout test.txt
```

This restores the file test.txt to the last uploaded version

```
$ git checkout 397344c2 test.txt
```

This restores the file test.txt to the version with commit id 397344c2

git diff

- Shows the differences between two files/ commits

```
$ git diff
```

git tag

- Tag “important” points in your history (e.g. paper versions)
- Create tags

```
$ git tag -a <tagname> -m “<tag message>”
```

- Share tags

```
$ git push origin <tagname>
```

- Checkout tags

```
$ git checkout <tagname>
```

- List all tags

```
$ git tag
```

.gitignore

- One can create a file and list all files that should be ignored by git

Exercise ☺

Exercises

- Setup
  - Create Github account (if you haven’t already)
  - Start Exercises

Exercise 1

- Create a new repository on Github
- Create a file named “test.txt”
- Write your name in the text document
- Upload the textfile to the repository

Exercise 2

- Checkout the following repository:
  - https://github.com/TheBananaMan/exercise2.git
- Create a file “<your_firstname>.txt”
- Upload your file to the repository
- Download the files of the other people

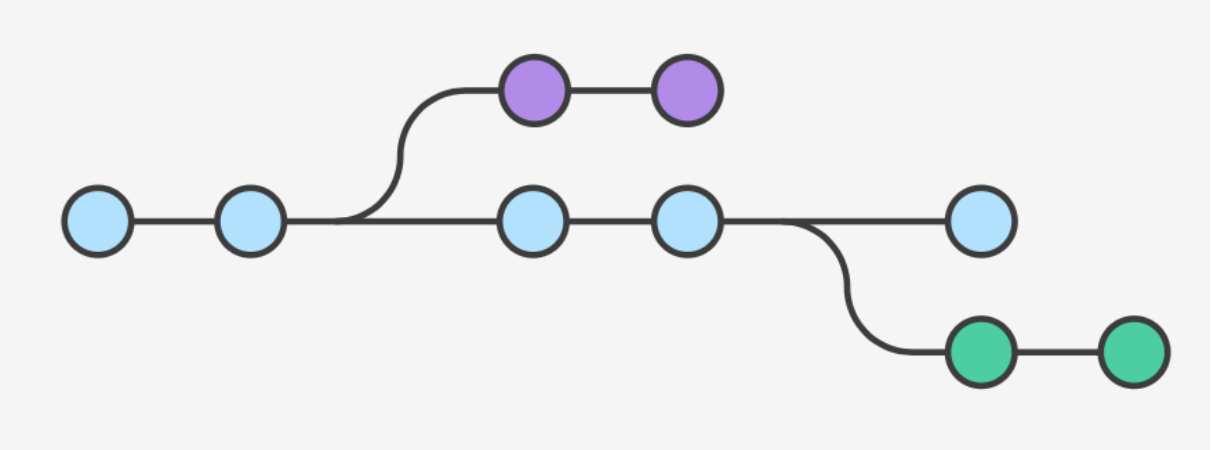
Exercise 3

- Use the repository from previous exercise
- Write your name in the text document “test.txt”
- Upload the changes in test.txt

- Overall goal: Everyone’s name should be in the file test.txt

Advanced Commands

Git Branches



- A branch represents a independent line of development
- There are local and remote branches

Git Branches

- List all branches in your repository:

```
$ git branch
```

- Create a new branch:

```
$ git branch <branch>
```

- Delete a branch:

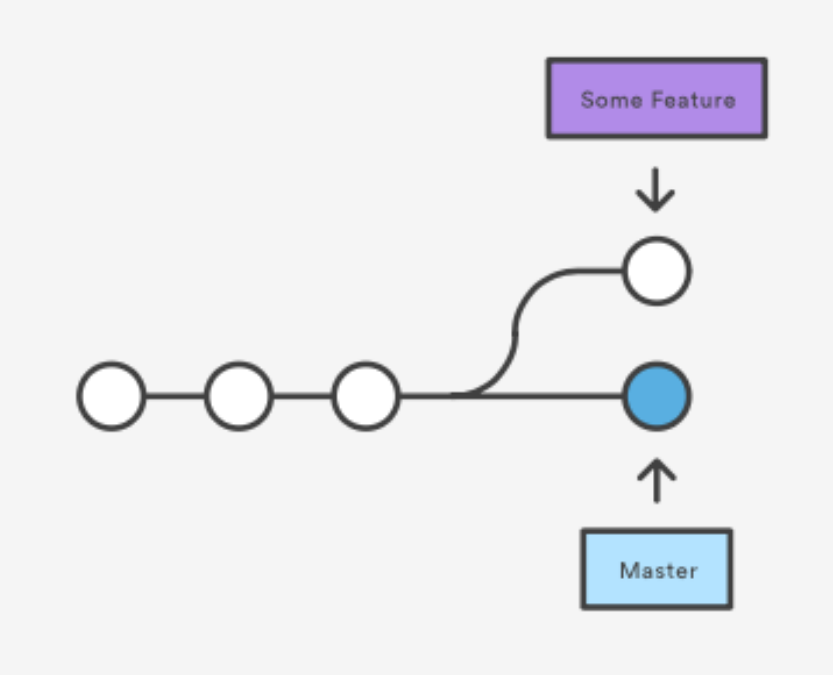
```
$ git branch -d <branch>
```

- Switch to /checkout a branch:

```
$ git checkout <branch>
```

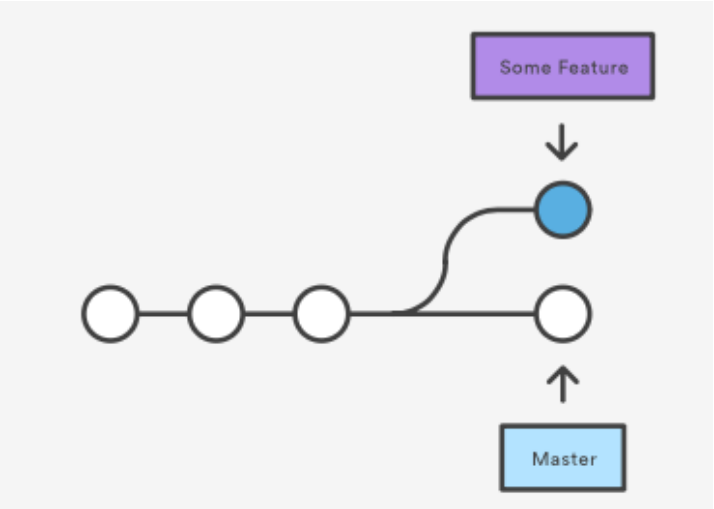
Git Branches - Example


```
$ git branch <some feature>
```

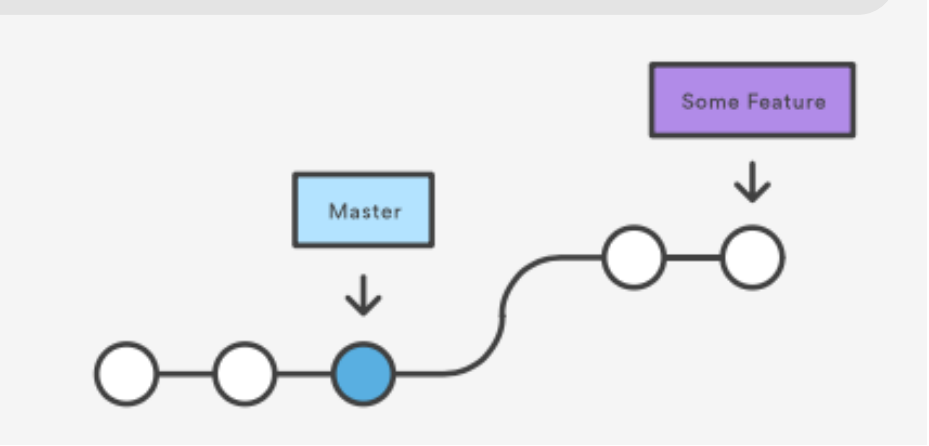


Git Branches - Example

```
$ git checkout <some feature>
```




```
$ touch test.txt
$ git add test.txt
$ git commit test.txt –m “add test.txt”
```



Git Branches - Merge

- Merge branch back to current branch:

```
$ git merge <branch>
```

- Merge branch (but always create a merge commit):

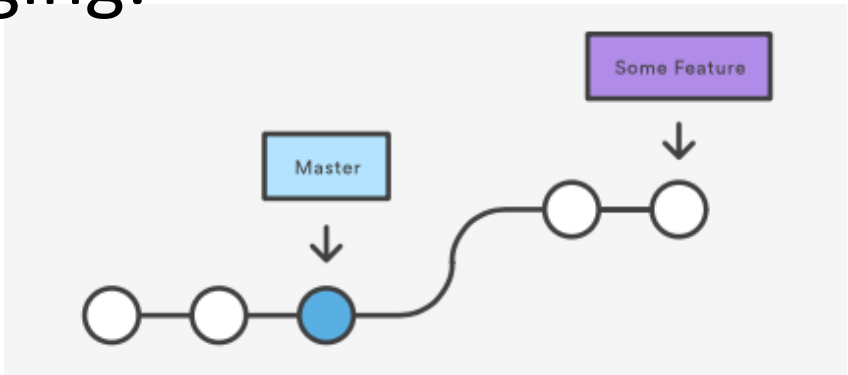
```
$ git merge --no-ff <branch>
```

- Several types of possible merges
  - Fast-forward merge
  - 3-way merge

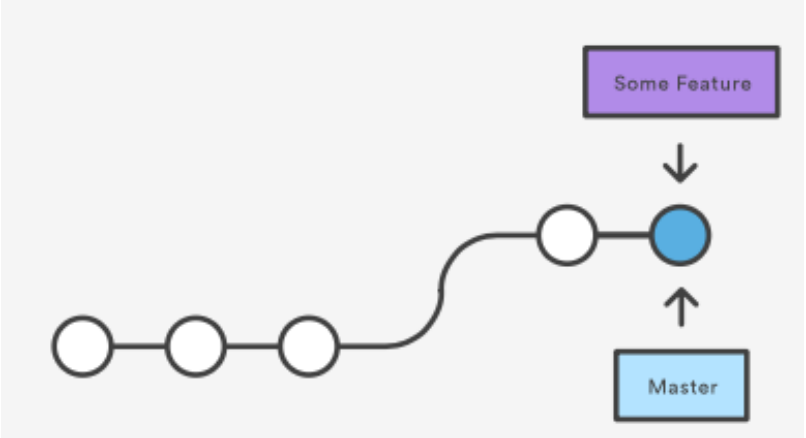
Git Branches – Fast-Forward Merge

```
$ git checkout master
$ git merge <some feature>
```

- Before merging:



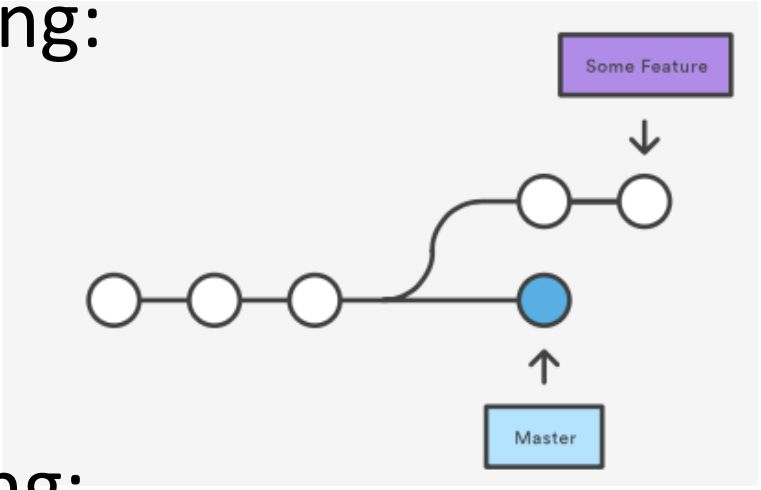
- After merging:



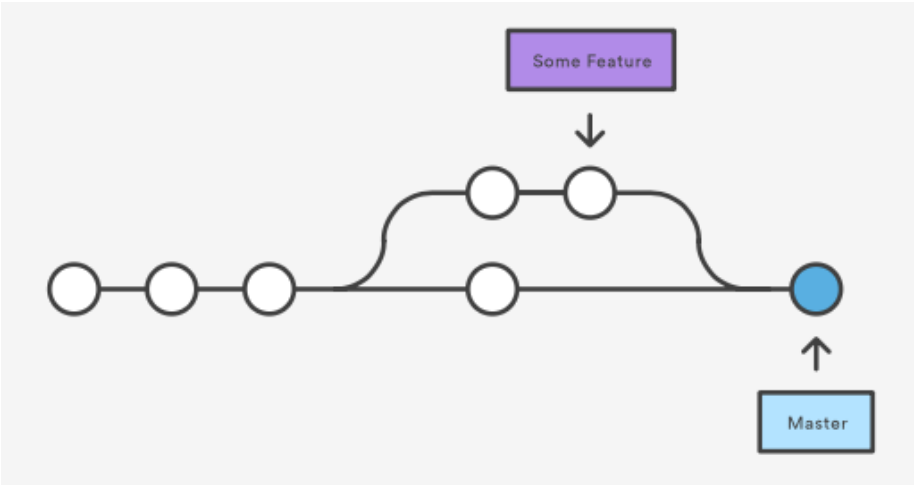
Git Branches – 3-way Merge

```
$ git checkout master
$ git merge <some feature>
```

- Before merging:



- After merging:



Git Branches – Merge conflicts

- If two branches change the same part of the same file, git can't handle the conflict

```
# On branch master
# Unmerged paths:
# (use "git add/rm ..." as appropriate to mark resolution)
#
# both modified: hello.py
#
```

- Resolve conflict manually
- Commit resolved conflict

Git Branches – Remote branches

- Publish/Push a local branch:

```
$ git push origin <branch>
```

- Pull a remote branch:

```
$ git checkout –b <localbranch> origin/<remotebranch>
```

- List all branches (local and remote):

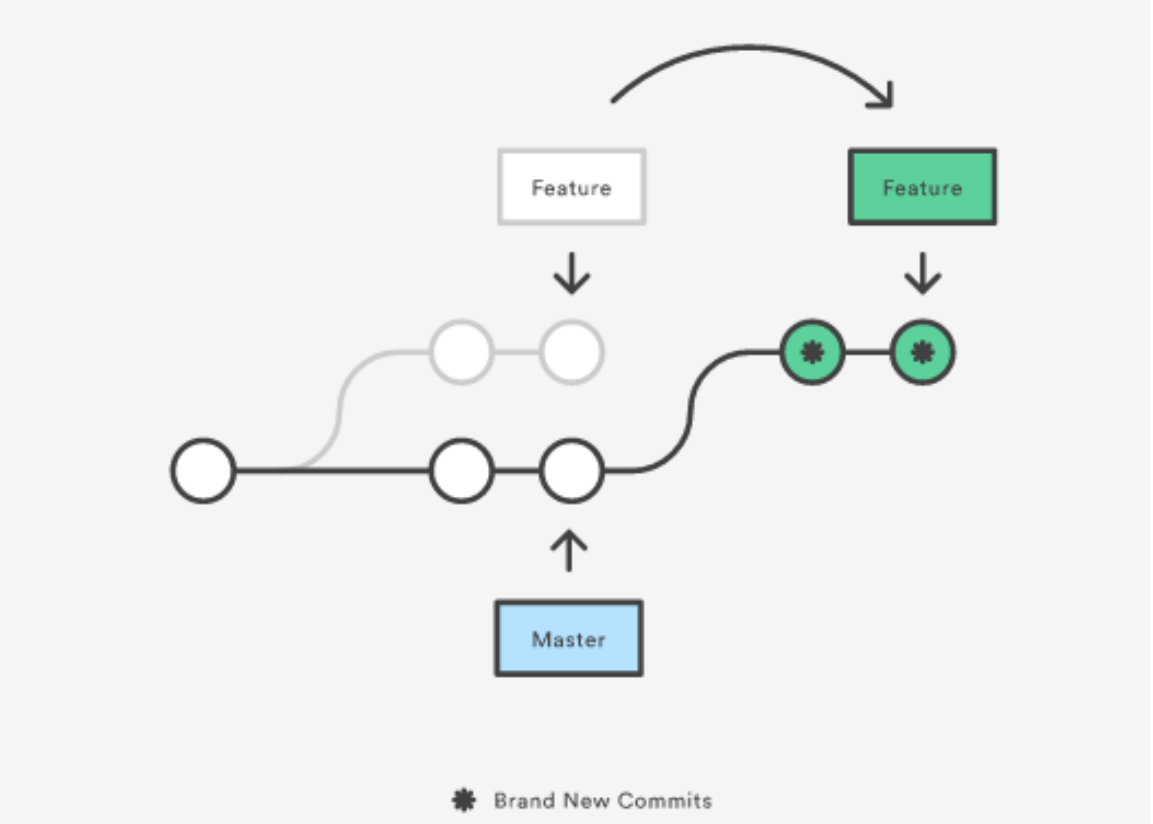
```
$ git branch -a
```

- Delete remote branch

```
$ git push origin --delete <remotebranch>
```

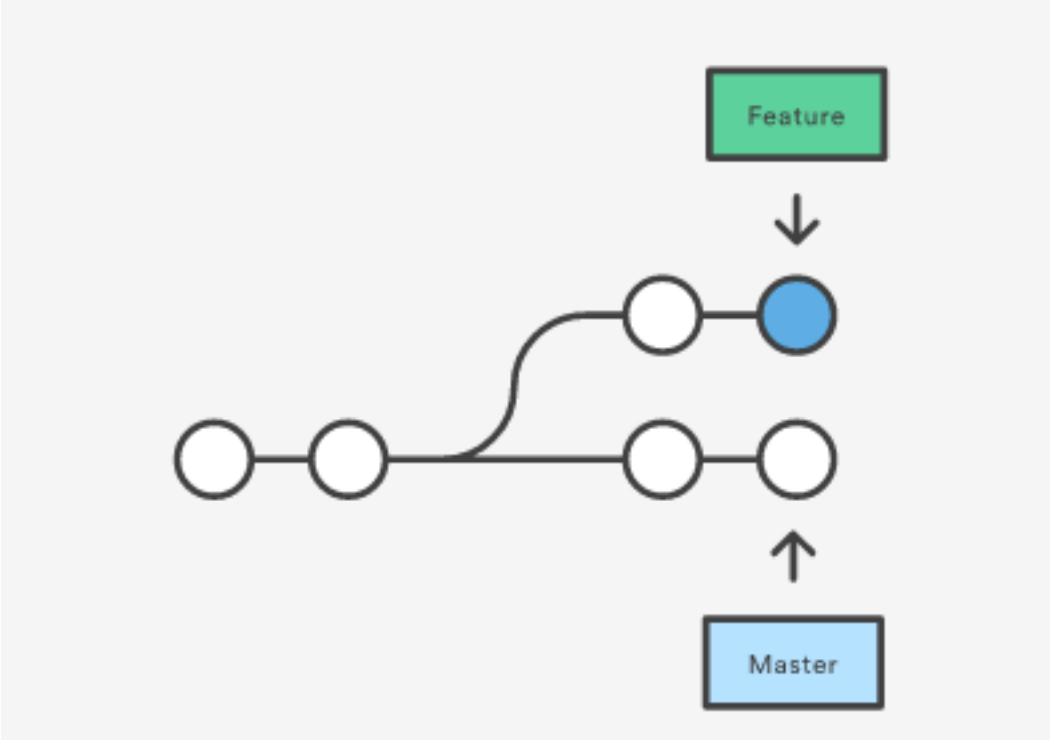
Git Branches – Rebasing

- Move a branch to a new base commit
- Maintain linear project history
- Don’t loose history from a branch



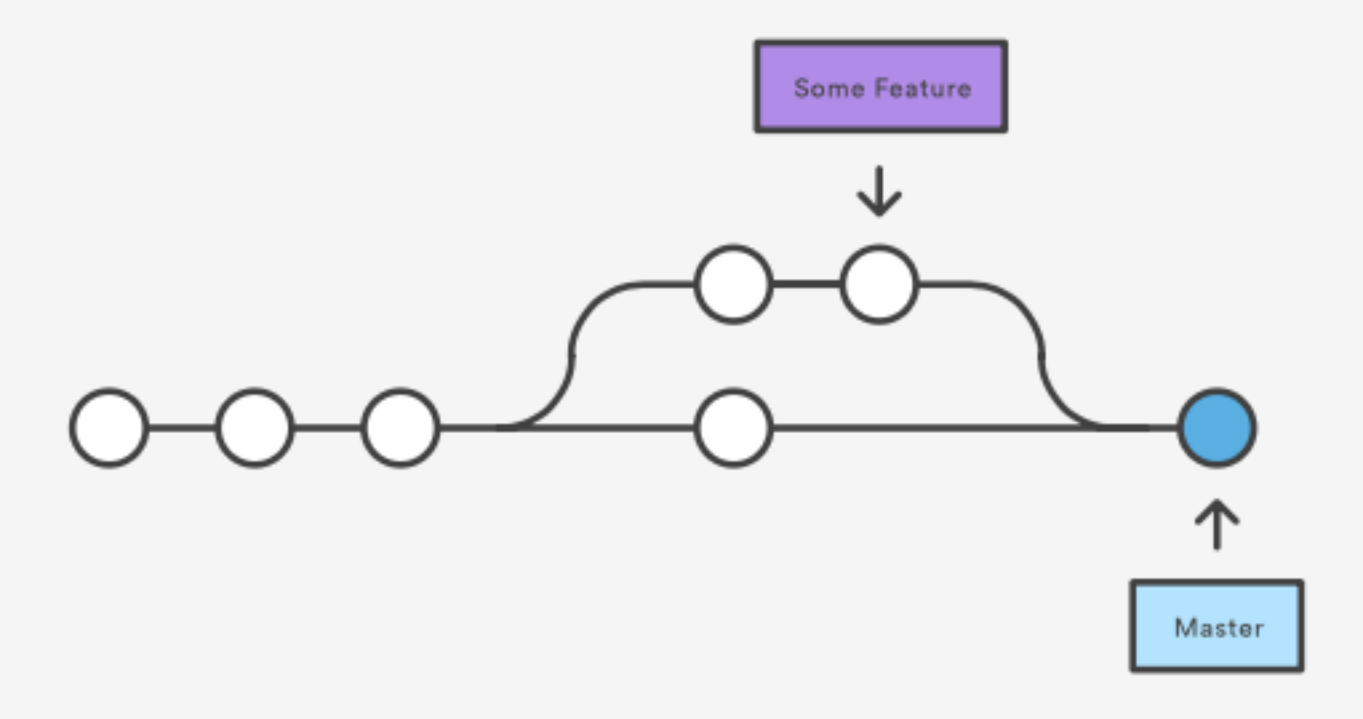
Git Branches – Rebasing

- Master branch has progressed since the start of a feature
- The feature depends on some commits of the master branch



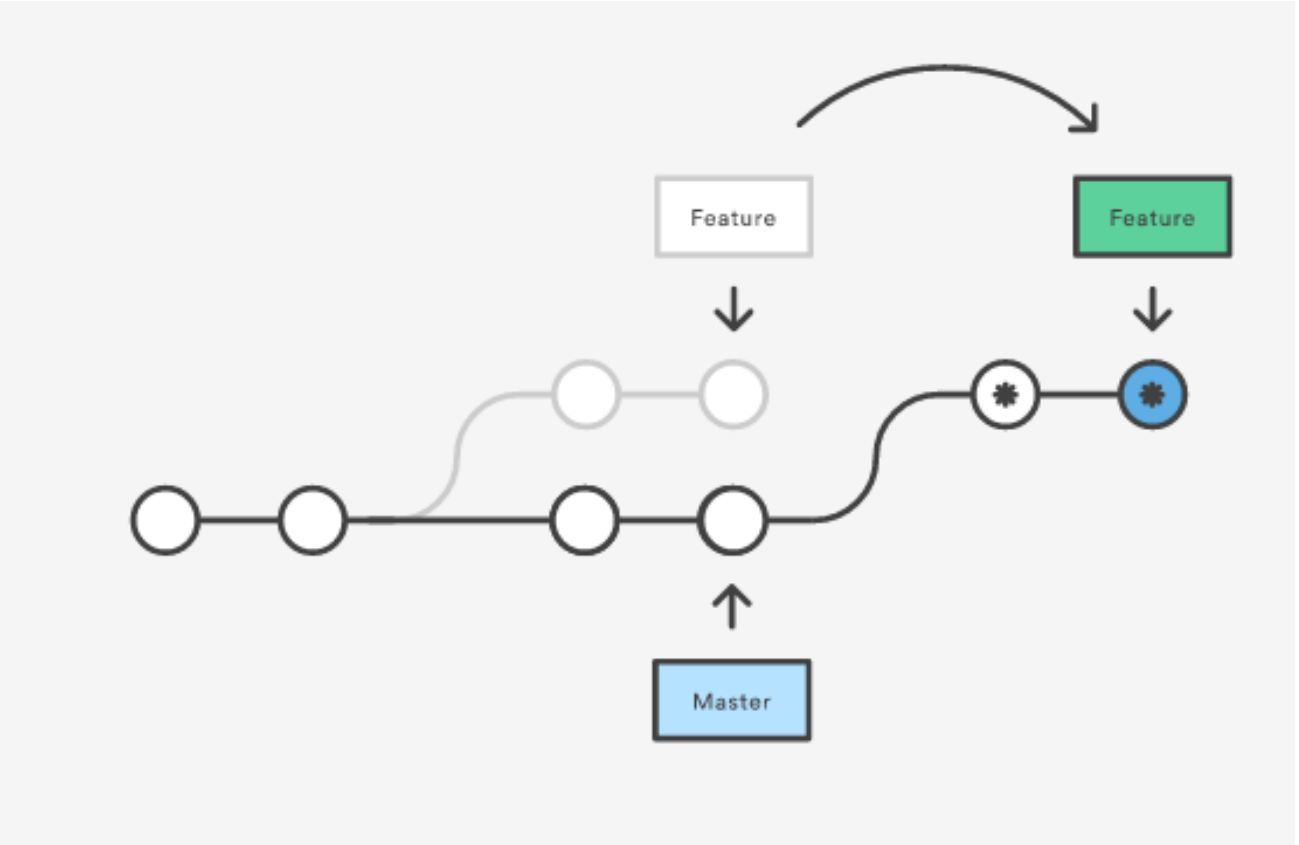
Git Branches – Rebasing

- Solution 1: Merge directly with a 3-way merge and a merge commit



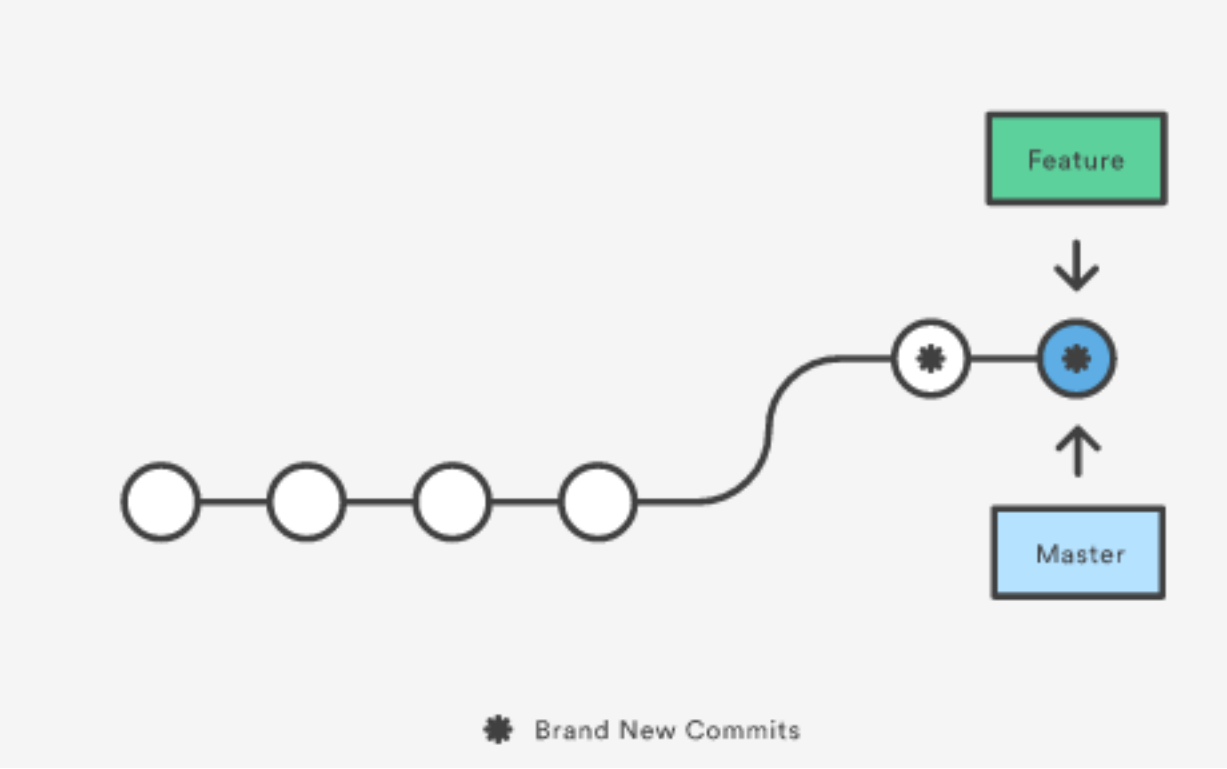
Git Branches – Rebasing

- Solution 2: Rebase



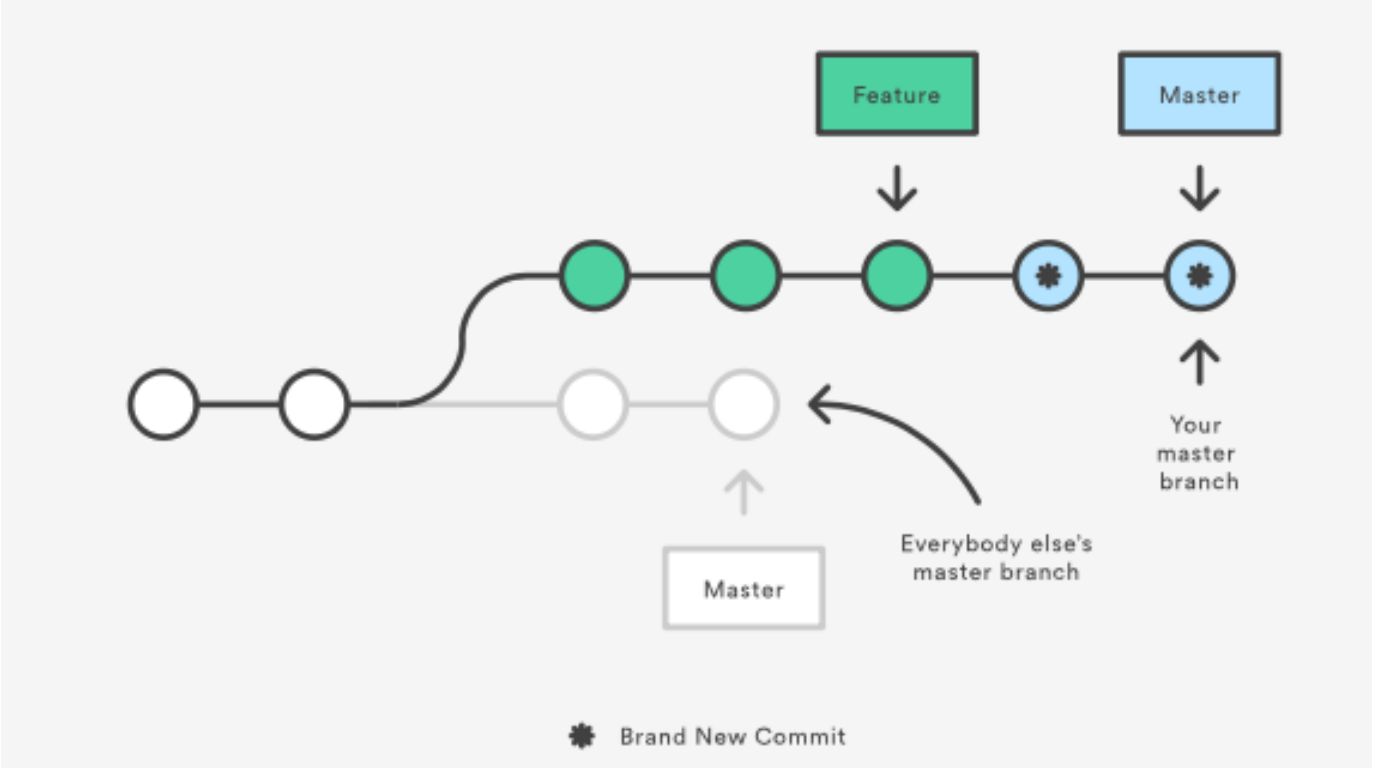
Git Branches – Rebasing

- Solution 2: fast-forward merge

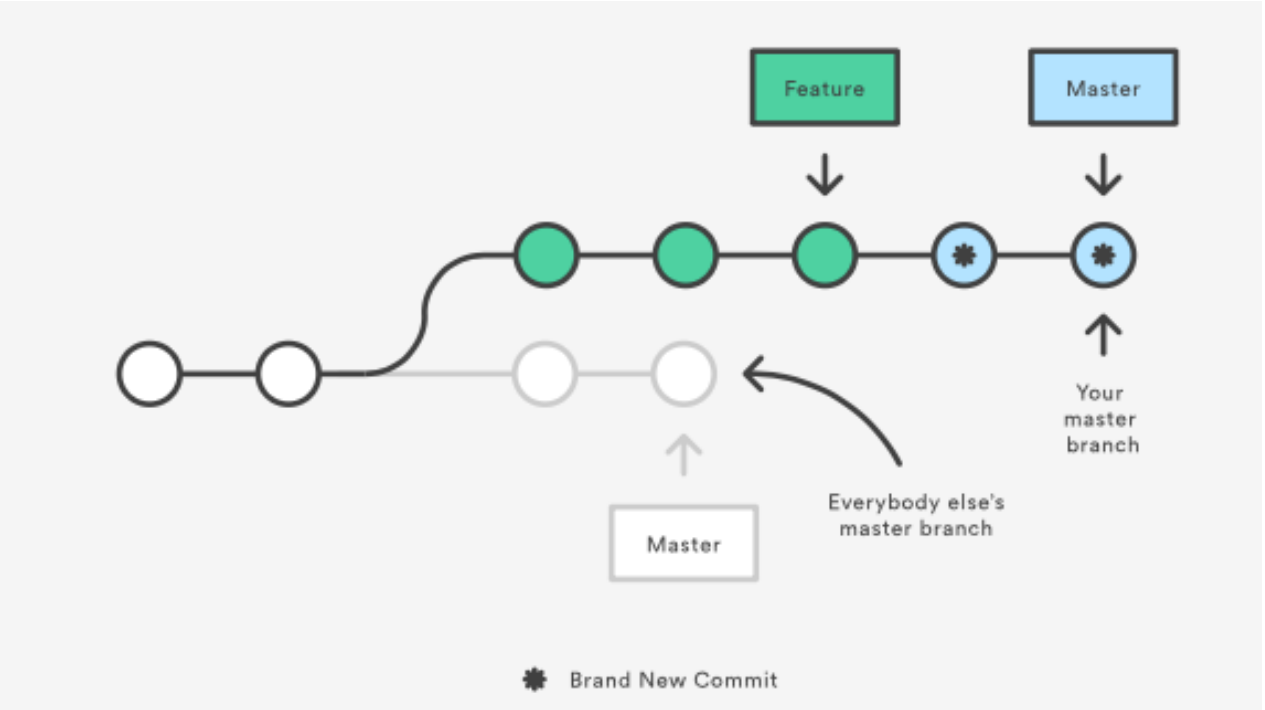


Git Branches – Rebasing

- Golden Rule of Rebasing: Don’t rebase public branches
  - Example: Rebase the master branch onto your feature branch



Git Branches – Rebasing



- This **only** happens in **your** repository
- Everyone else will work on the old master
- Rebase creates new commits – git thinks that your master branches diverge from the other master
- Merging them together will results in a merge commit with two different histories

Git Submodules

- Use other git repository in your git repository
- Use external libraries managed in a git repository
- Create a new submodule:

```
$ git submodule add <link to repository> <directory>
```

- Clone a git repository with submodules:

```
$ git clone -- recursive <link to repository>
```

- Update a submodule:

```
$ git submodule update --init
```

Useful Stuff for Paper Writing

CryptoBib

CryptoBib is a BibTeX database containing papers related to Cryptography, with manually checked entries and uniform BibTeX data.

https://cryptobib.di.ens.fr

Github

- Web-based git/version control repository
- Distributed version control
- Source code management
- 20 million users (57 million repositories) – largest host of source code in the world
- Offers public and private repositories
- Free private repositories with an academic email address

Exercise ☺

Exercise 4

Clone the following repository:

- https://github.com/TheBananaMan/exercise4.git

Create a branch with your name

Edit the file ”test.txt” in your branch and write your name in it

Upload your branch to the repository

Exercise 5

Merge your branch back to the master branch

Overall goal: All your names should be in test.txt

Exercise 6

Create a repository with CryptoBib as a submodule

Exercise 7

Checkout the following repository (which contains CryptoBib as a submodule)

https://github.com/TheBananaMan/testwithcryptobib.git

Further Tutorials

- https://git-scm.com/book/en/v2
- https://www.atlassian.com/git/tutorials
- https://www.git-tower.com/blog/git-cheat-sheet/