Git Training

Rachel Player
Jordy Gennissen
+ kind volunteer(s)

Royal Holloway University of London
15 February 2019

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
  - Under pressure, mistakes happen
    - Also when not under pressure

- 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 (1)

○ Store "several copies" of a file

○ History of the changes to a file
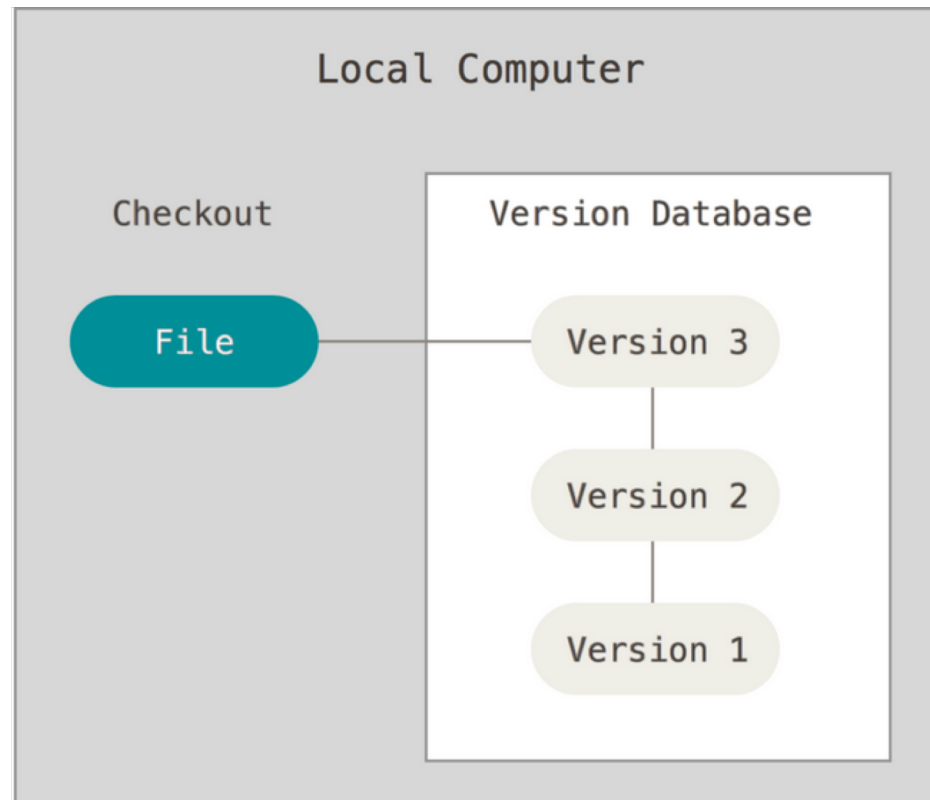
About Version Control (2)

Free backups for everyone!

Disclaimer

This workshop does not tell you:

- Good coding collaborative practice

- How to be a "git master"

But does tell you:

- How to use git well when collaborating on a paper

- How to use it **practically**

Installing git

- Linux:

  $ sudo apt-get install git-all

- Mac OS:

  Download from: https://git-scm.com/download/mac
  Or use brew: brew install git

- General link (including Windows):

  https://git-scm.com/downloads

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 nano
```

○ ## 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

o   Lists the files which have been modified since the last commit

o   Lists the untracked files in your local directory

git add

o Puts a file in the 'staging area' ready for a commit

o 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

o   Commits the files in the staging area (that have been added with the previous command)

o   Add a meaningful commit message so you/other people understand the change

o   Commits are labelled by a hash value (SHA-1)

$ git commit –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

o Upload the committed local changes to the remote repository

```
$ git push origin master
```

git pull

o Download the latest remote change to the local repository

```
$ git pull origin master
```

git log

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

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

Exercise ☺

Exercises

o **Setup**

 o Create Github account (FYI: you already should've done this)

 o Tell us your username (FYI: you already should've done this)

 o Start Exercises   (FYI: This is not something you
                 should've done already)

Exercise 1

o Create a new repository on Github

o Create a file named "test.txt"

o Write your name in the text document

o Upload the textfile to the repository

Exercise 2

o Checkout the following repository:

    o   https://github.com/rachelplayer/isg-playground.git

o Create a file "<your_firstname>.txt"

o Upload your file to the repository

o Download the files of the other people

Exercise 3

o   Use the repository from previous exercise

o   Write your name in the text document "names.txt"

o   Upload the changes in names.txt

o   Overall goal: Everyone's name should be in the file names.txt

(slightly)

Advanced Commands

Git IDs

- Every git commit has a unique ID

- If you want to go back to a commit, use the ID!

- To find the ID, use the website or

```
$ git log
```

- Git log example:

```
commit 0cb46a492bd91e0b4389dfeacd83ed2701701222
Author: Rachel Player <rachelplayer@gmail.com>
Date:   Fri Jan 18 15:06:41 2019 +0000

    added the file rachel.txt
```

git checkout

o **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 your version and the latest commit

```
$ git diff
```

.gitignore

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

○ For example all intermediate files from LaTeX:

$$*.bbl$$
$$*.blg$$
$$*.aux$$
$$*.out$$
$$*.log$$

git mv

o Move/Rename a file

`$ git mv test.txt introduction.txt`

This renames the file test.txt to introduction.txt

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 stash trick

o How do we go back to the latest commit?

   o We changed a lot of files, but it was a bad decision.

   o git checkout a; git checkout b; git checkout c; … ?
     OR:

   `$ git stash; git stash drop`

Exercise ☺

Exercise 4

o Go back to your own repo

o Add 2 files: "test2.txt, oops.txt".

o Commit

o Revert the changes using *only git*

o Commit

o Recover these files using *only git*

o Remove oops.txt again

Demo ☺

Exercise ☺

Exercise 5

o In pairs, invite someone to join your repo

o Add a file "review.txt" and push

o Both collaboratively write a review about this workshop. Push regularly, and resolve conflicts.
Hint:

　　o Git diff

　　o Git log (also verify their useful commit messages!)

Exercise 6

o Copy the review into the shared repo

    o   isg-playground

o add it, commit and push!

    o   Hint: git mv

Useful Stuff for Paper Writing

Github

o Web-based git/version control repository

o Distributed version control

o Source code management

o 20 million users (57 million repositories) – largest host of source code in the world

o Offers public and private repositories

o Free private repositories with an academic email address

CryptoBib

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

https://cryptobib.di.ens.fr

Advanced Commands

Git Branches
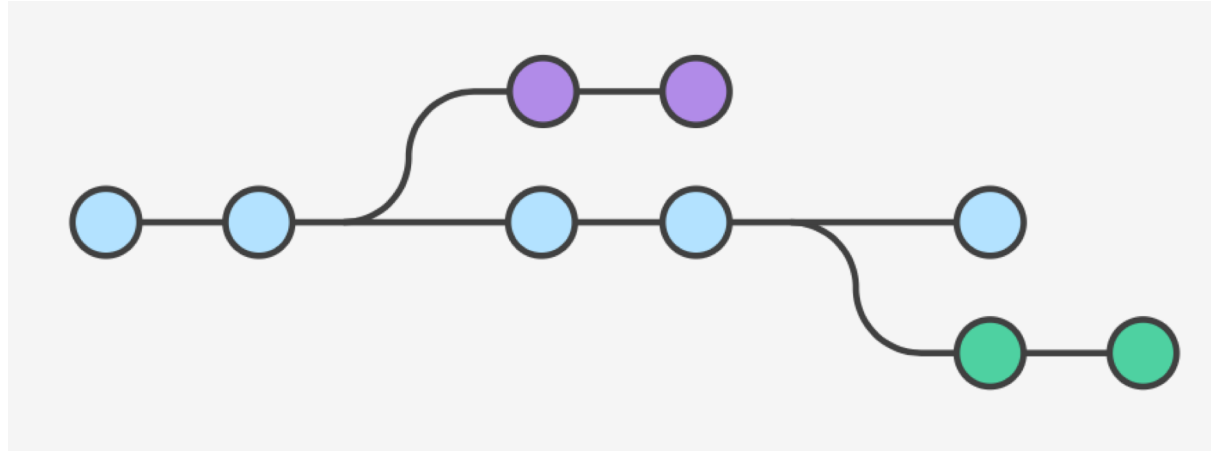
- A branch represents a independent line of development

- There are local and remote branches

Git Branches

o List all branches in your repository:

```
$ git branch
```

o Create a new branch:

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

o Delete a branch:

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

o Switch to /checkout a branch:

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

Git Branches - Example
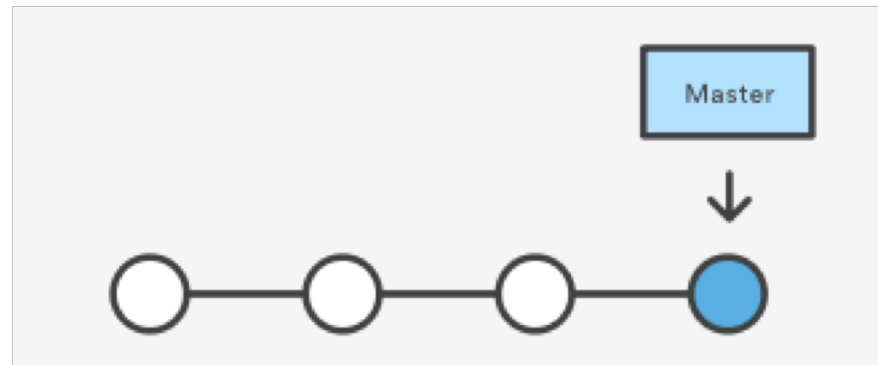
$ git branch <some feature>
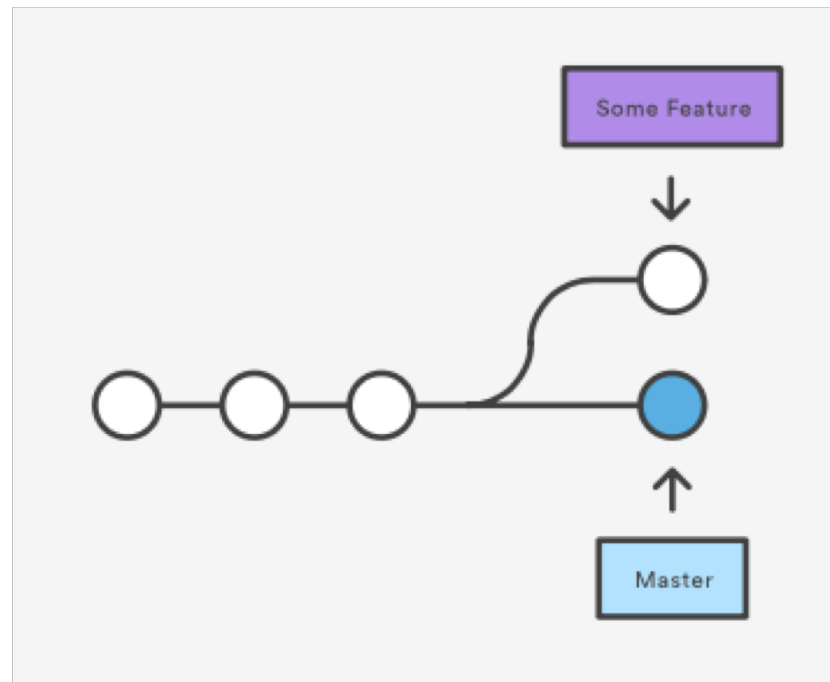
Git Branches - Example

$ git checkout <some feature>
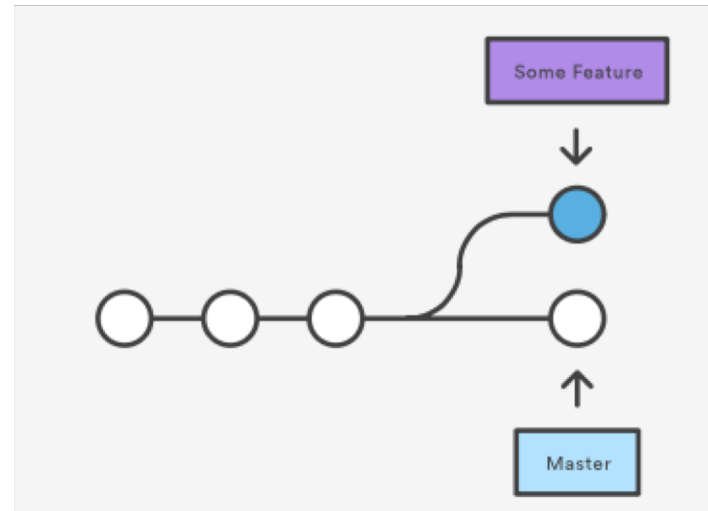
$ touch test.txt
$ git add test.txt
$ git commit test.txt –m "add test.txt"
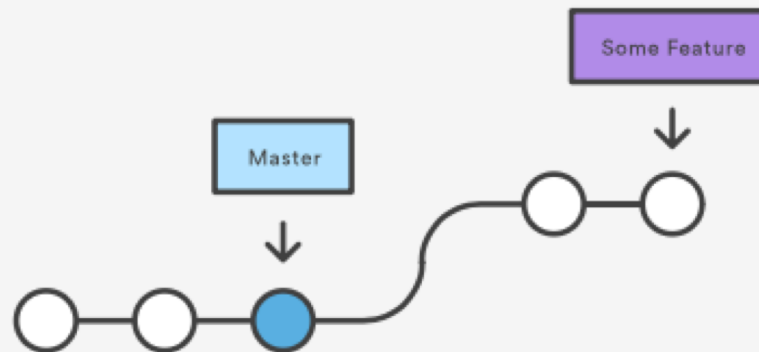
Git Branches - Merge

o Merge branch back to current branch:

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

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

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

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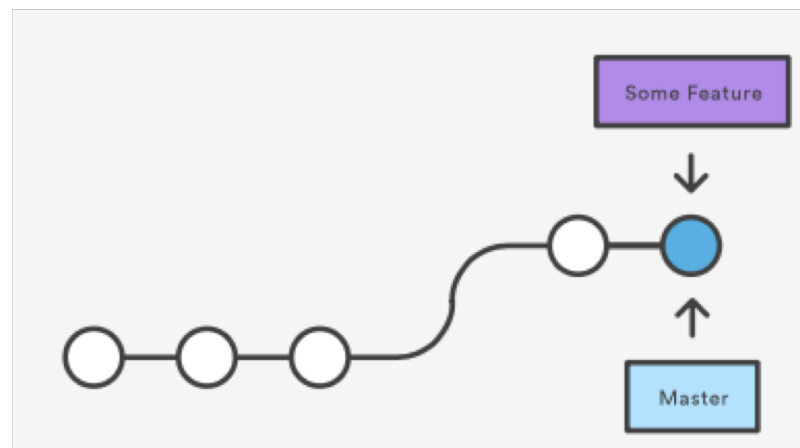 o Fast-forward merge
  o 3-way merge

Git Branches – Fast-Forward Merge

```
$ git checkout master
$ git merge <some feature>
```
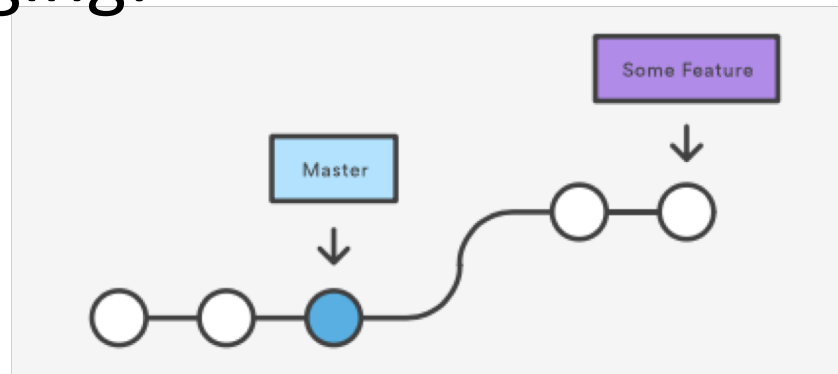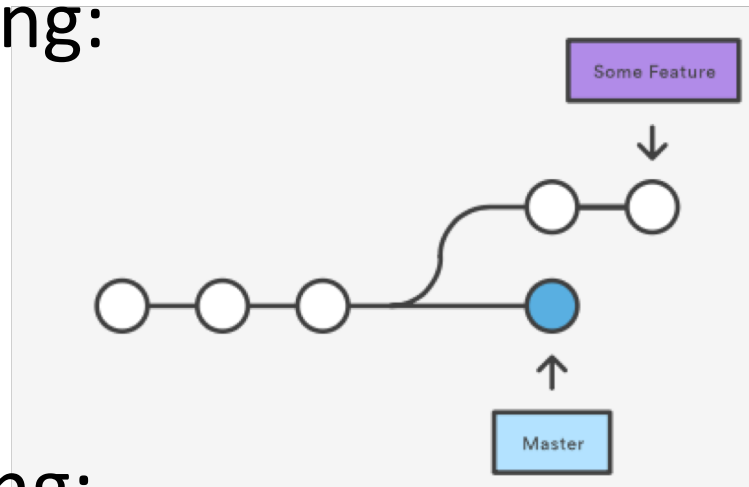
o Before merging:

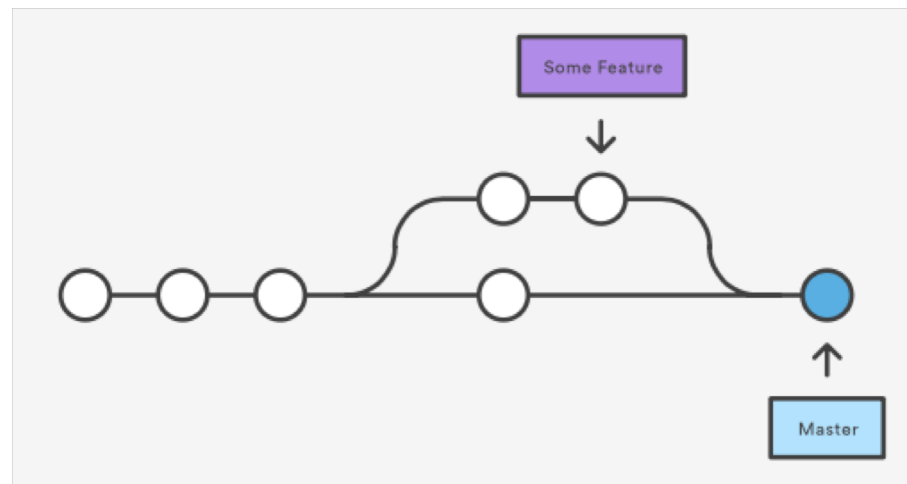o After merging:

Git Branches – 3-way Merge

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

o Before merging:

o After merging:

Git Branches – Merge conflicts

o 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
#
```

o Resolve conflict manually
o Commit resolved conflict

Git Branches – Remote branches

o Publish/Push a local branch:

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

o Pull a remote branch:

```
$ git checkout –b <localbranch> origin/<remotebranch>
```
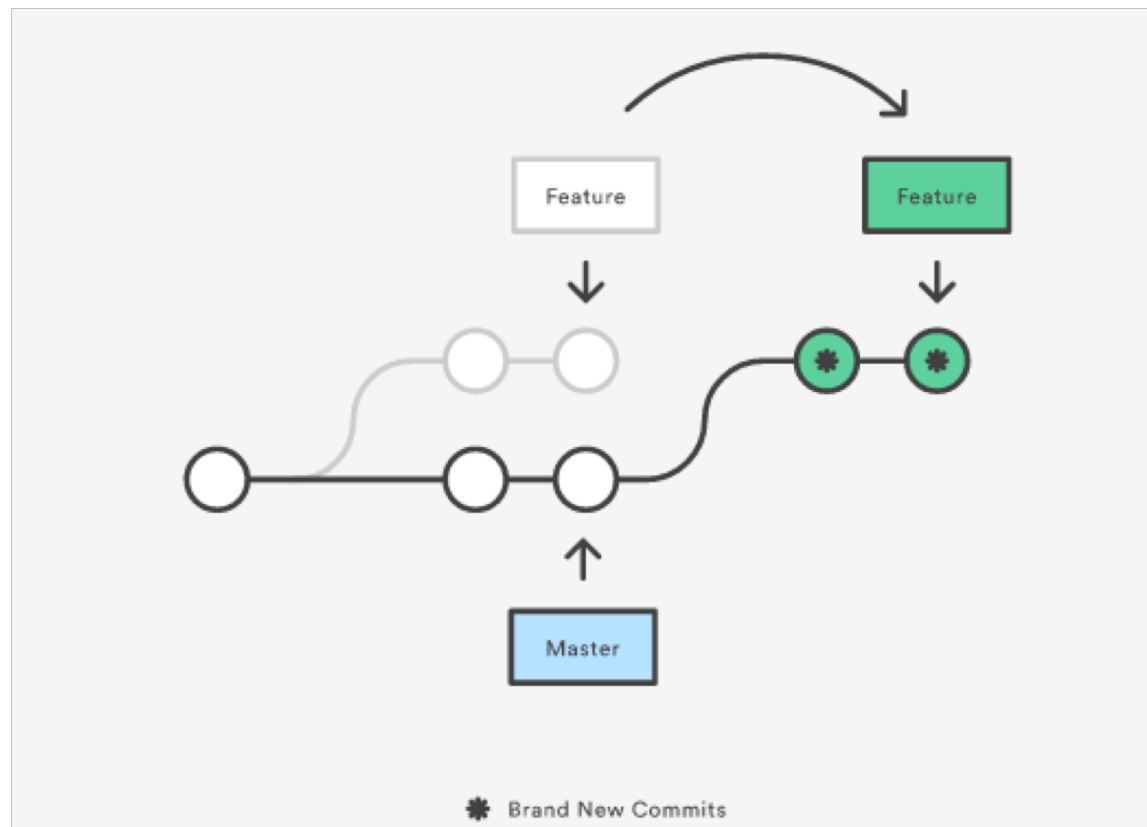
o List all branches (local and remote):

```
$ git branch -a
```

o Delete remote branch
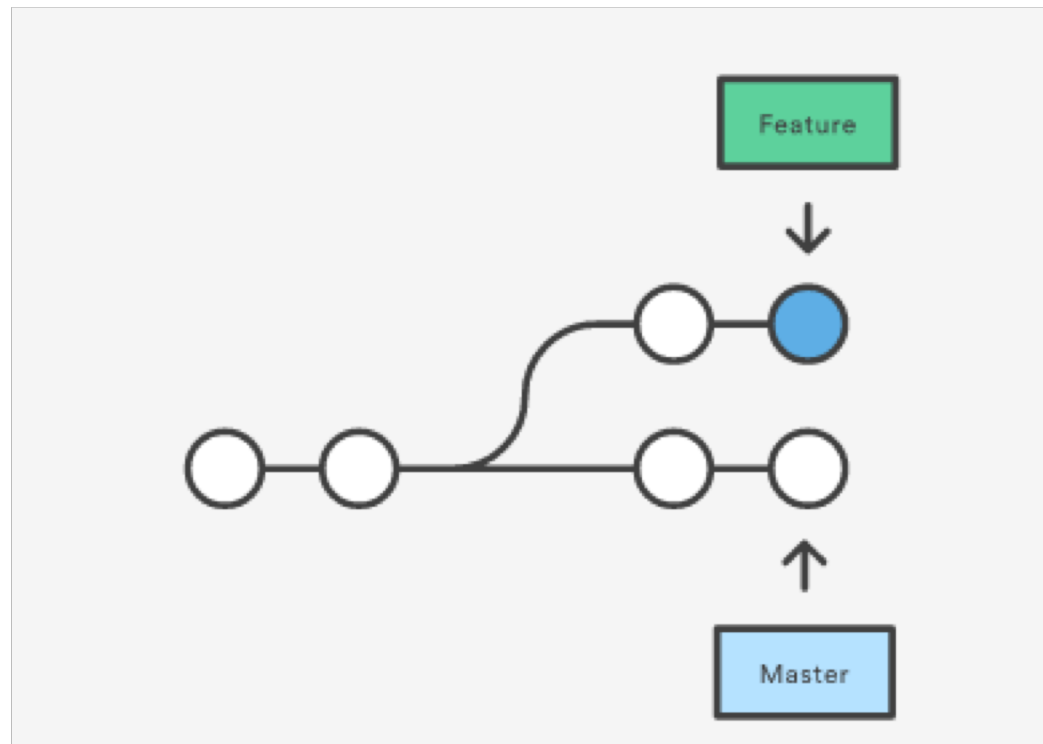
```
$ git push origin --delete <remotebranch>
```

Git Branches – Rebasing

o Move a branch to a new base commit
o Maintain linear project history
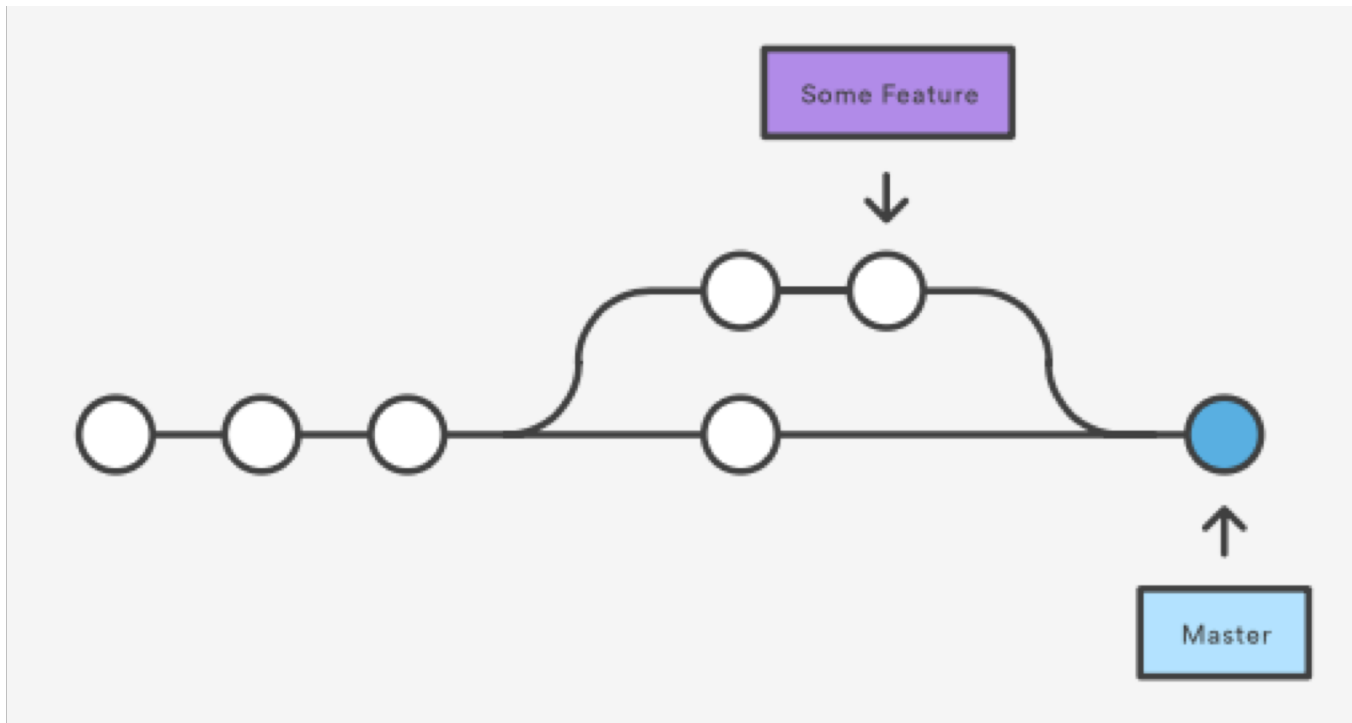o Don't loose history from a branch

Git Branches – Rebasing

o Master branch has progressed since the start of a feature

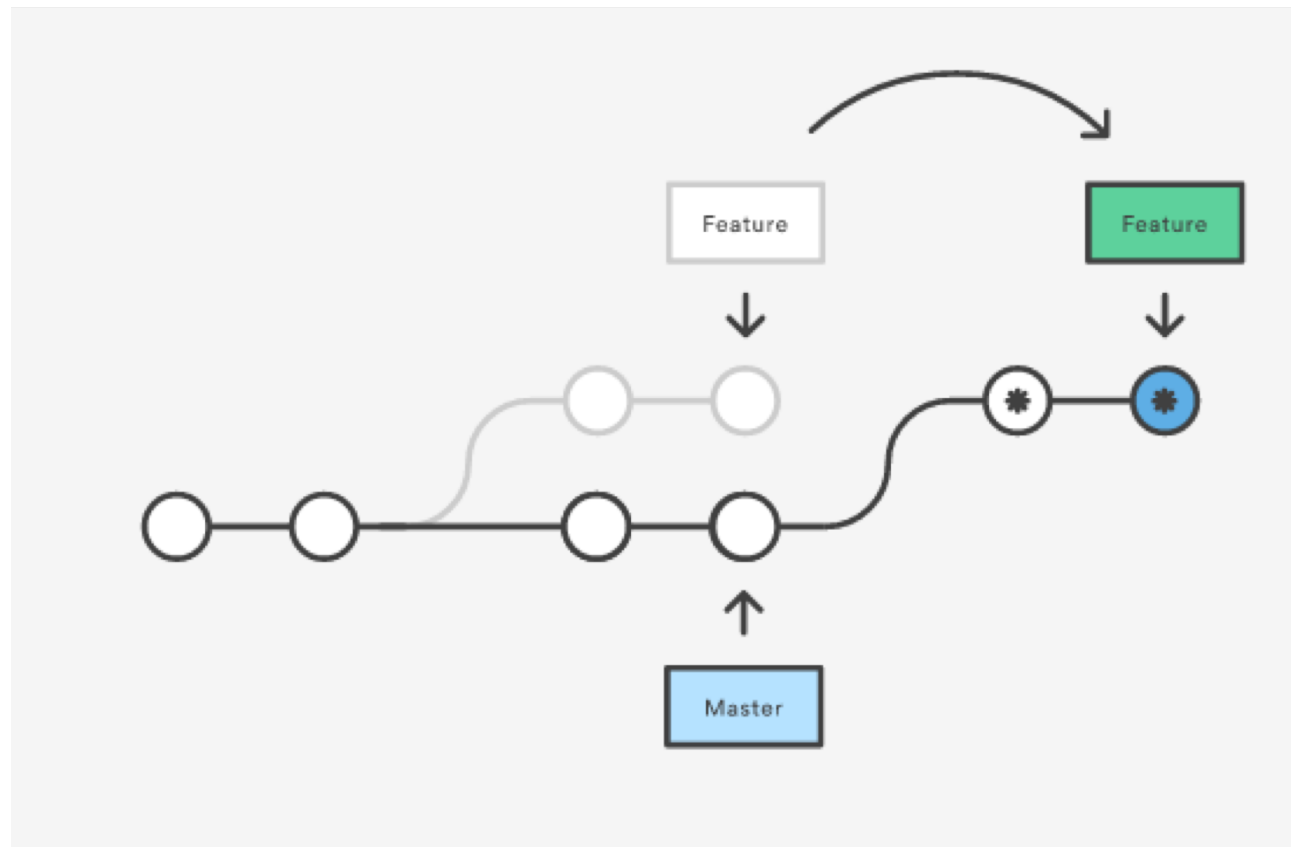o The feature depends on some commits of the master branch

Git Branches – Rebasing

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

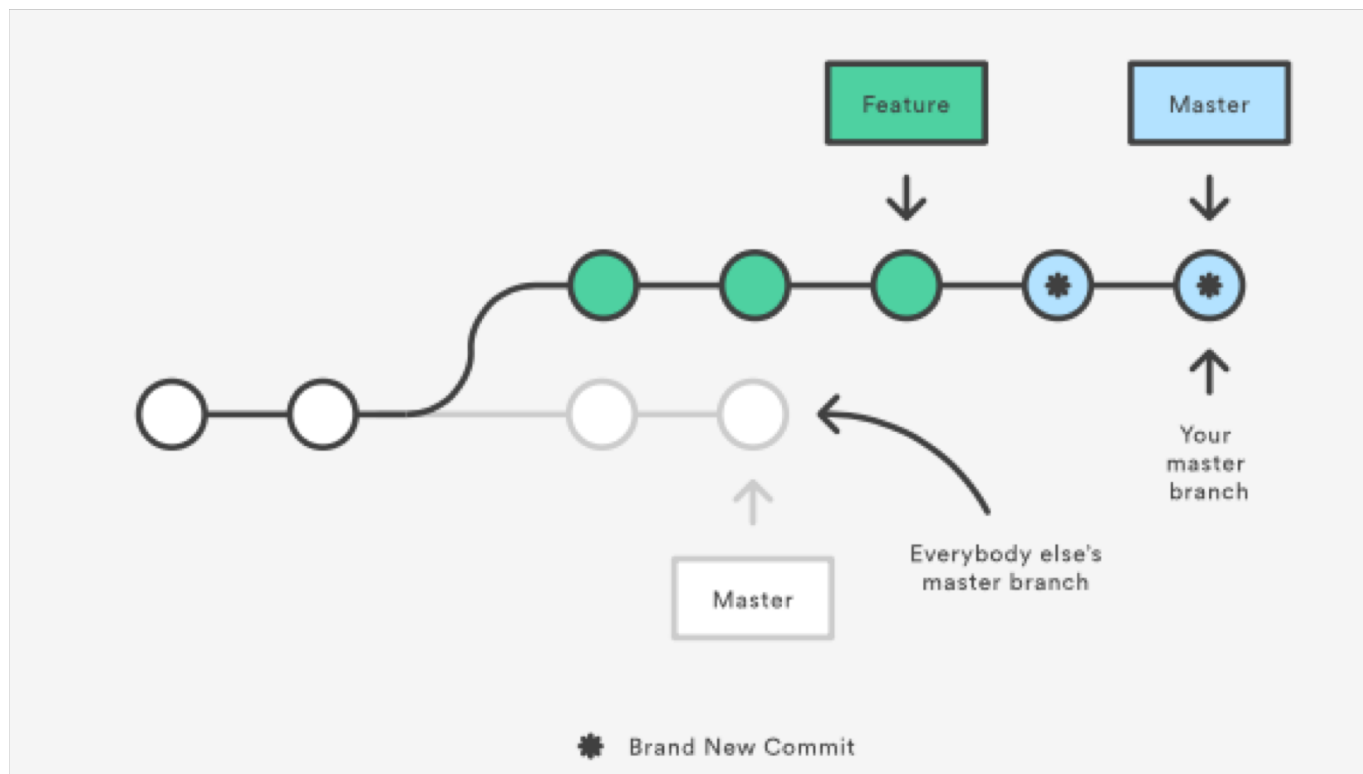# Git Branches – Rebasing

o Solution 2: Rebase

Git Branches – Rebasing
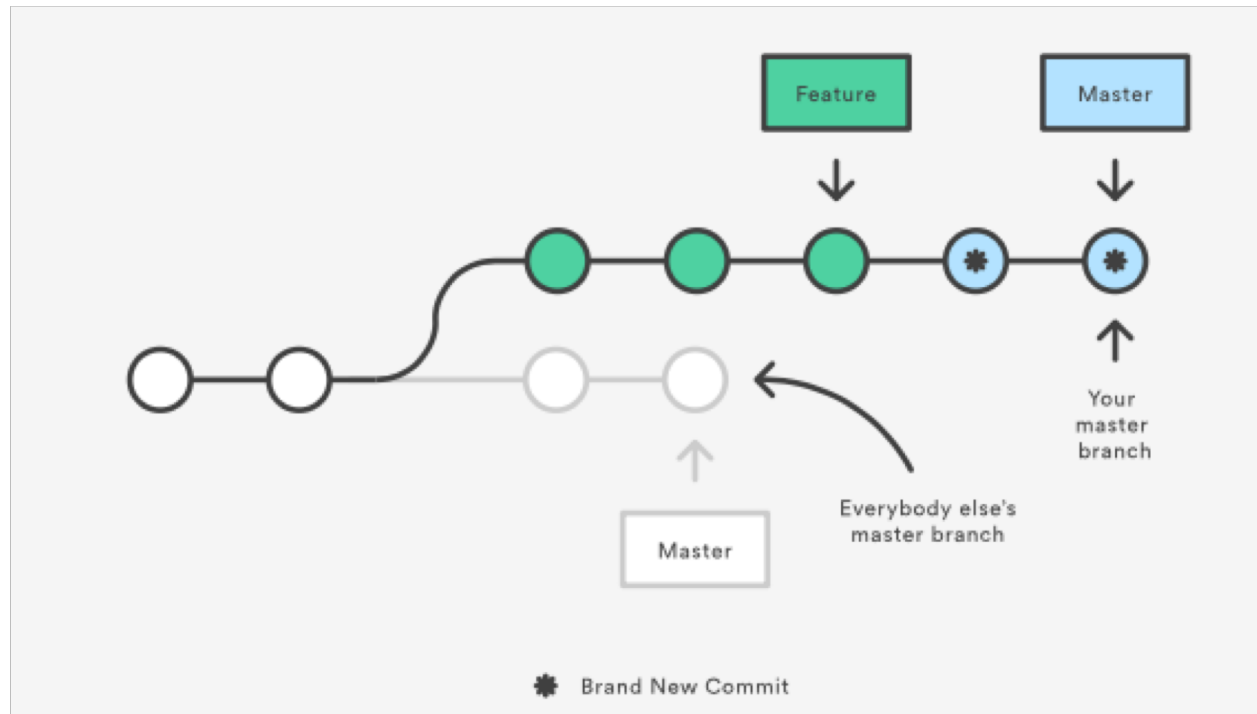
o Solution 2: fast-forward merge

Git Branches – Rebasing

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

Git Branches – Rebasing

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

Git Submodules

o Use other git repository in your git repository

o Use external libraries managed in a git repository

o Create a new submodule:

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

o Clone a git repository with submodules:

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

o Update a submodule:

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

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/