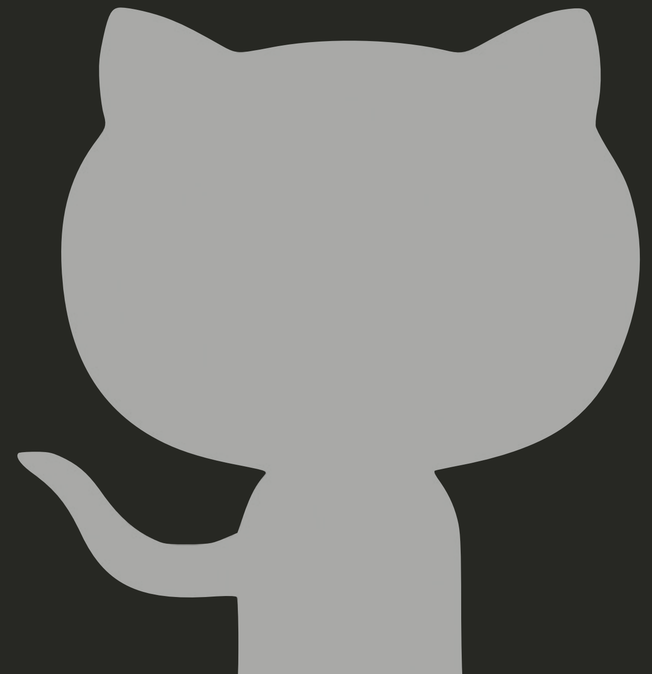
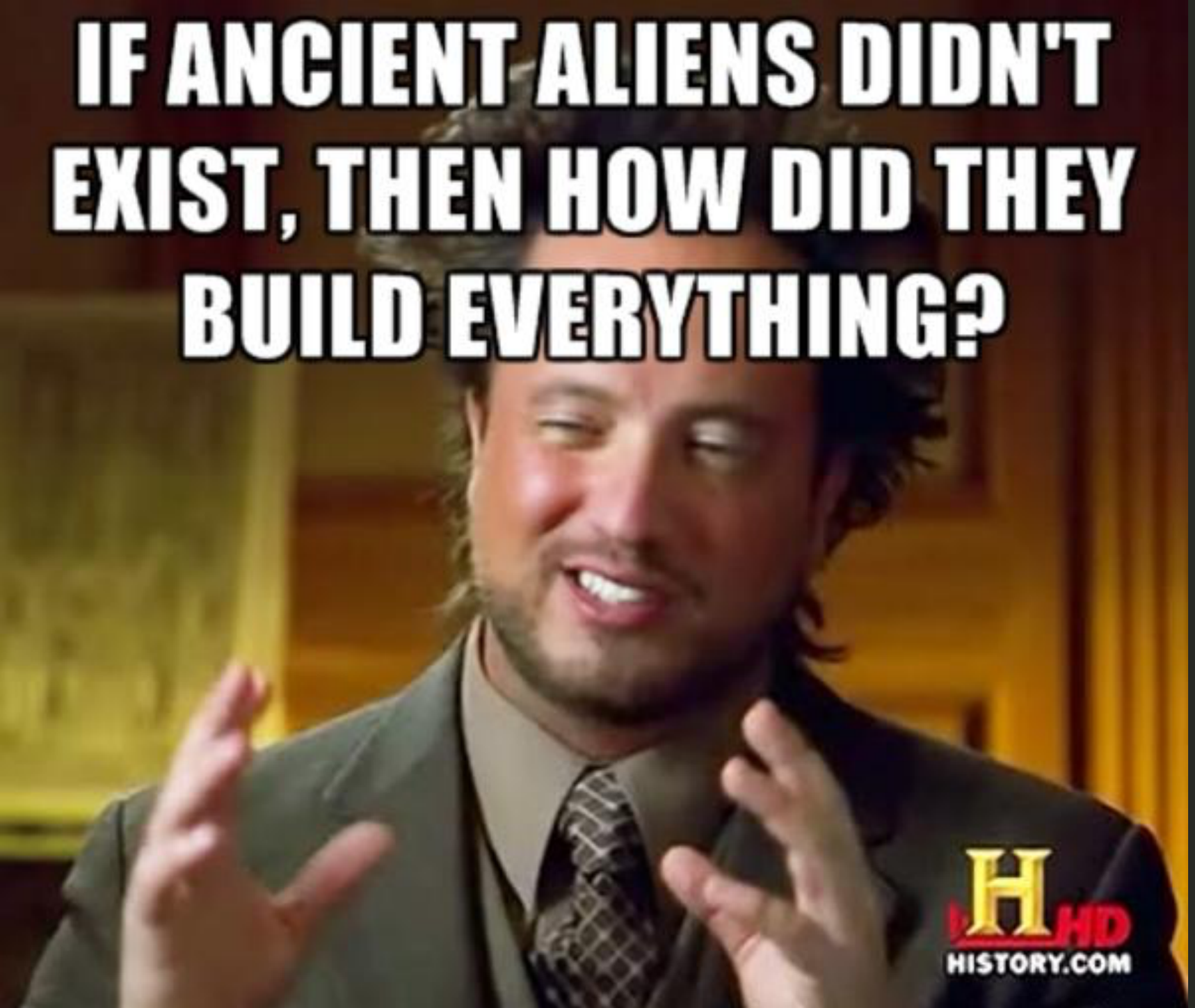
november 2017

version control

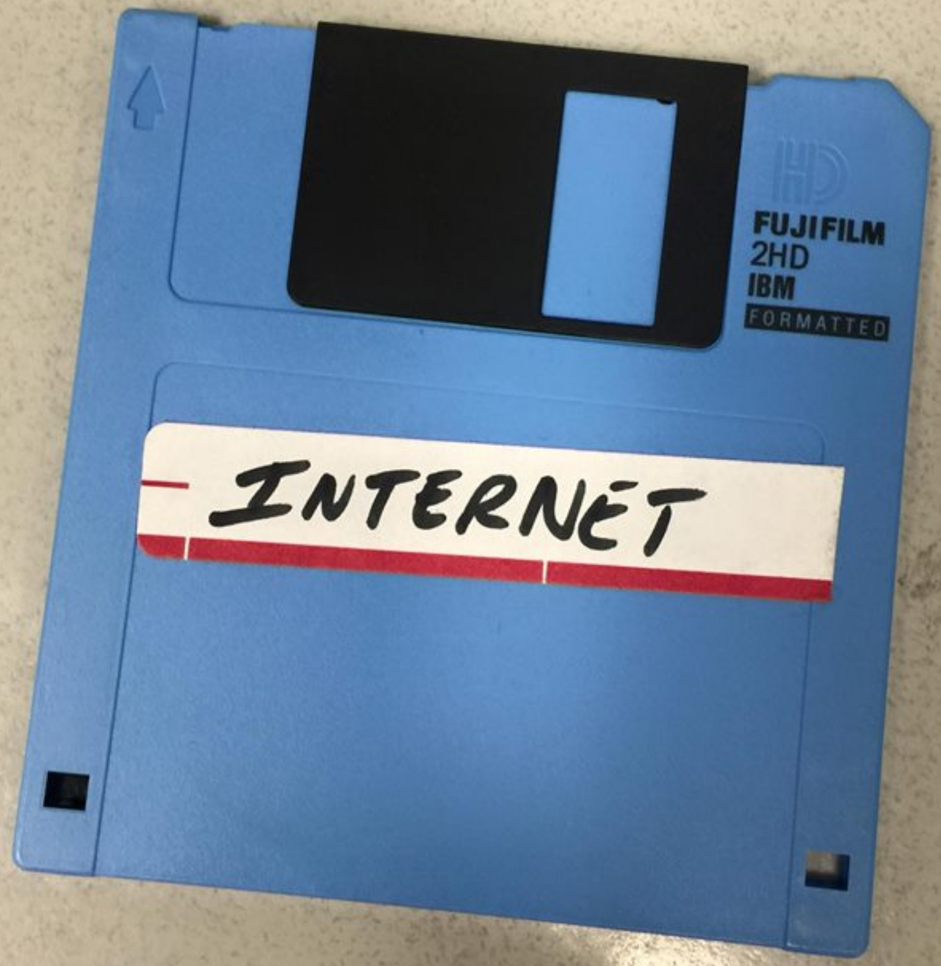
introduction to git

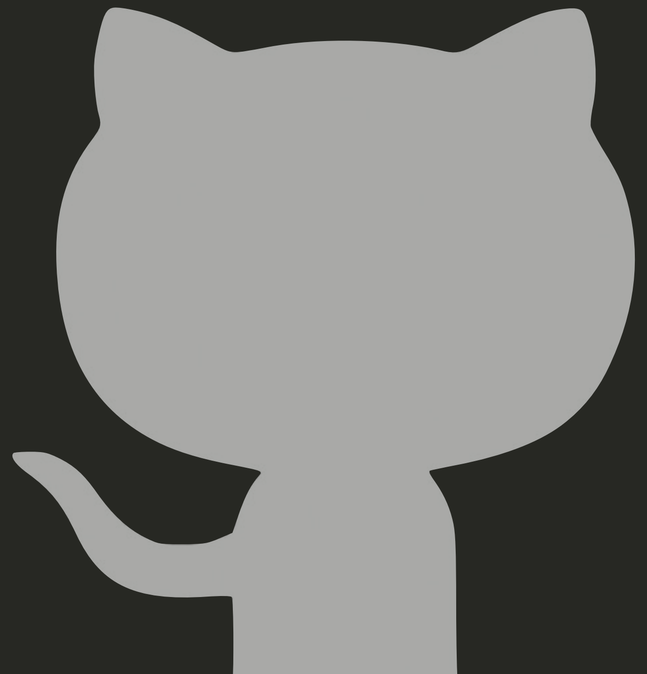


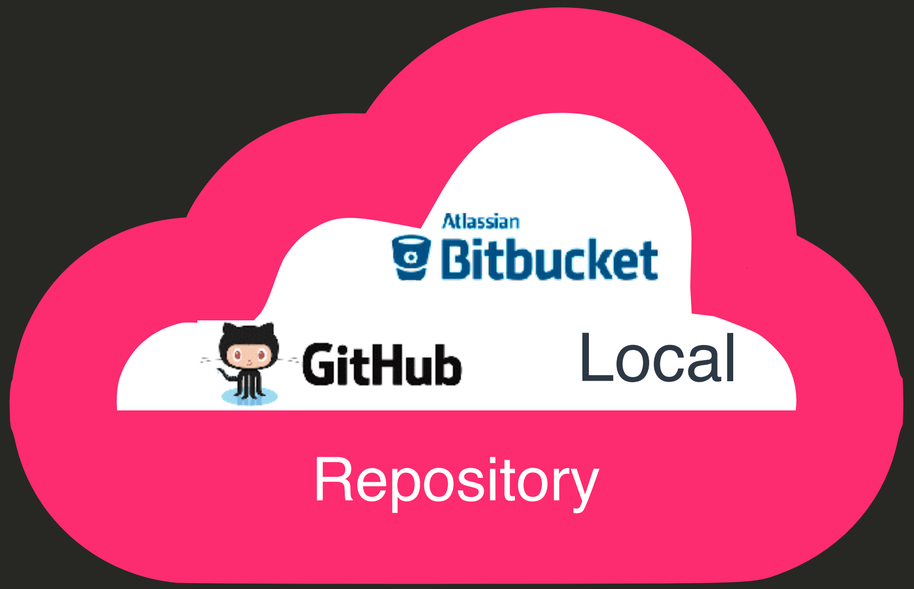
https://github.com/lfberge/versjonskontroll


IF ANCIENT ALIENS DIDN'T EXIST, THEN HOW DID THEY BUILD EVERYTHING?
HISTORY.COM


YAY! TEAMWORK!!!!


FUJIFILM
2HD
IBM
FORMATTED
INTERNET





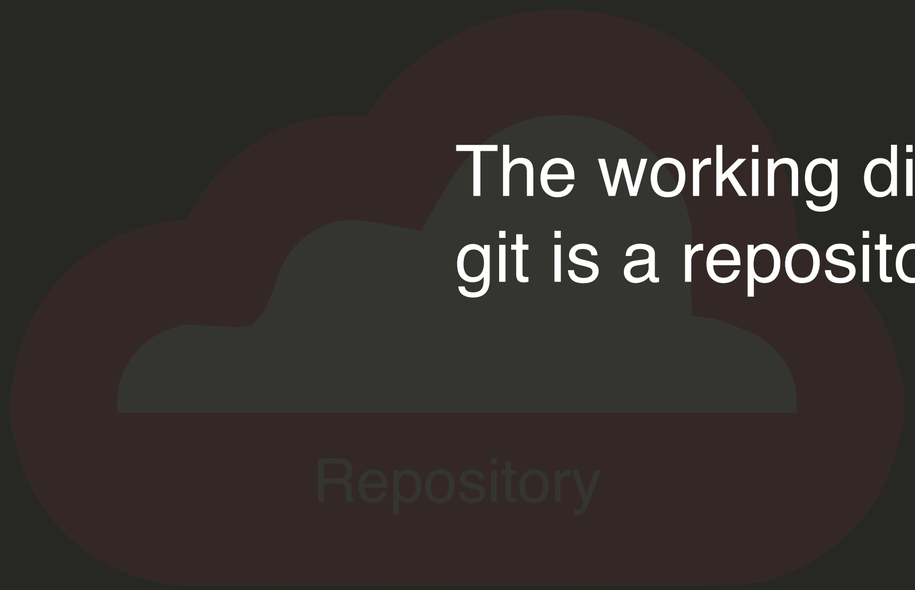
It’s just a folder.. sort of

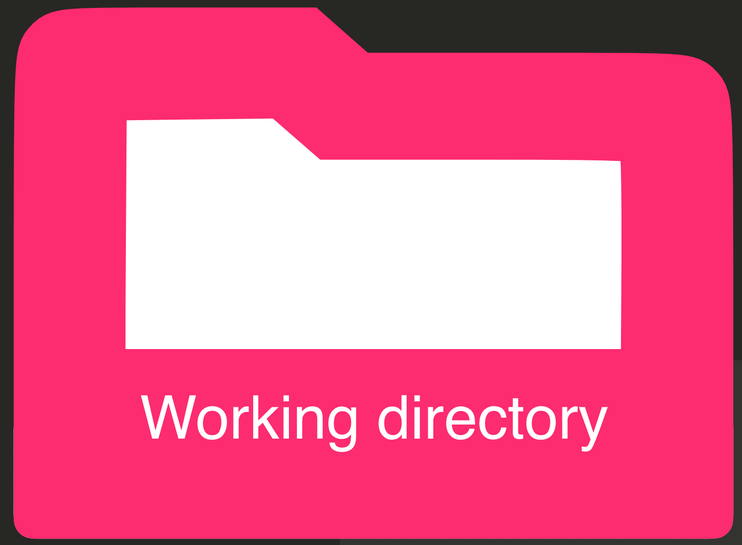
Record points in history
Branch out in parallel branches

Version control

Oh how we love recursiveness

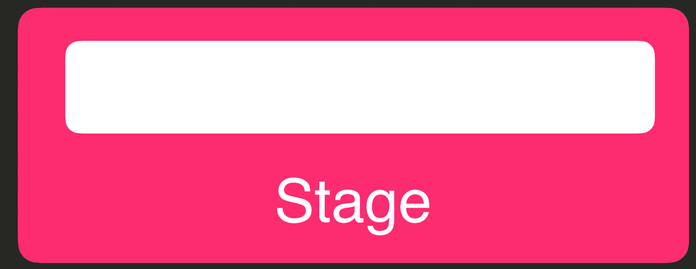
The working directory in git is a repository itself

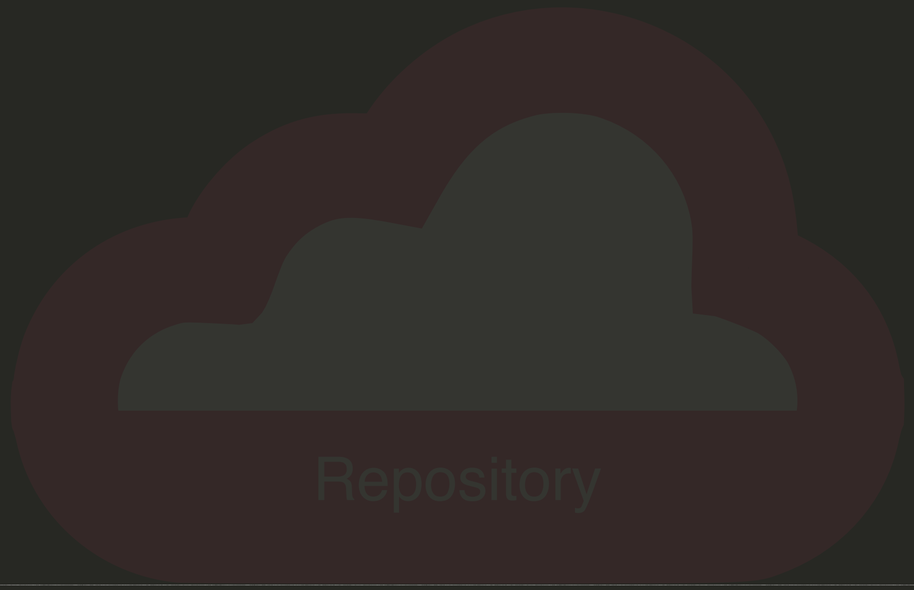


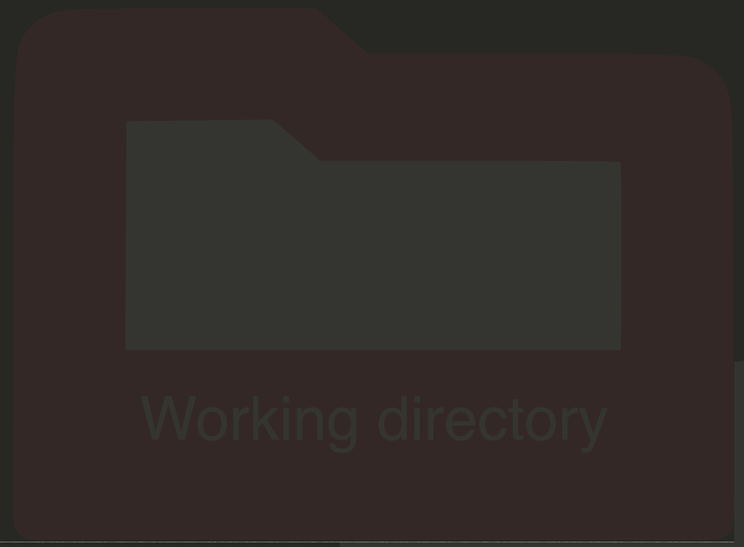


Version control

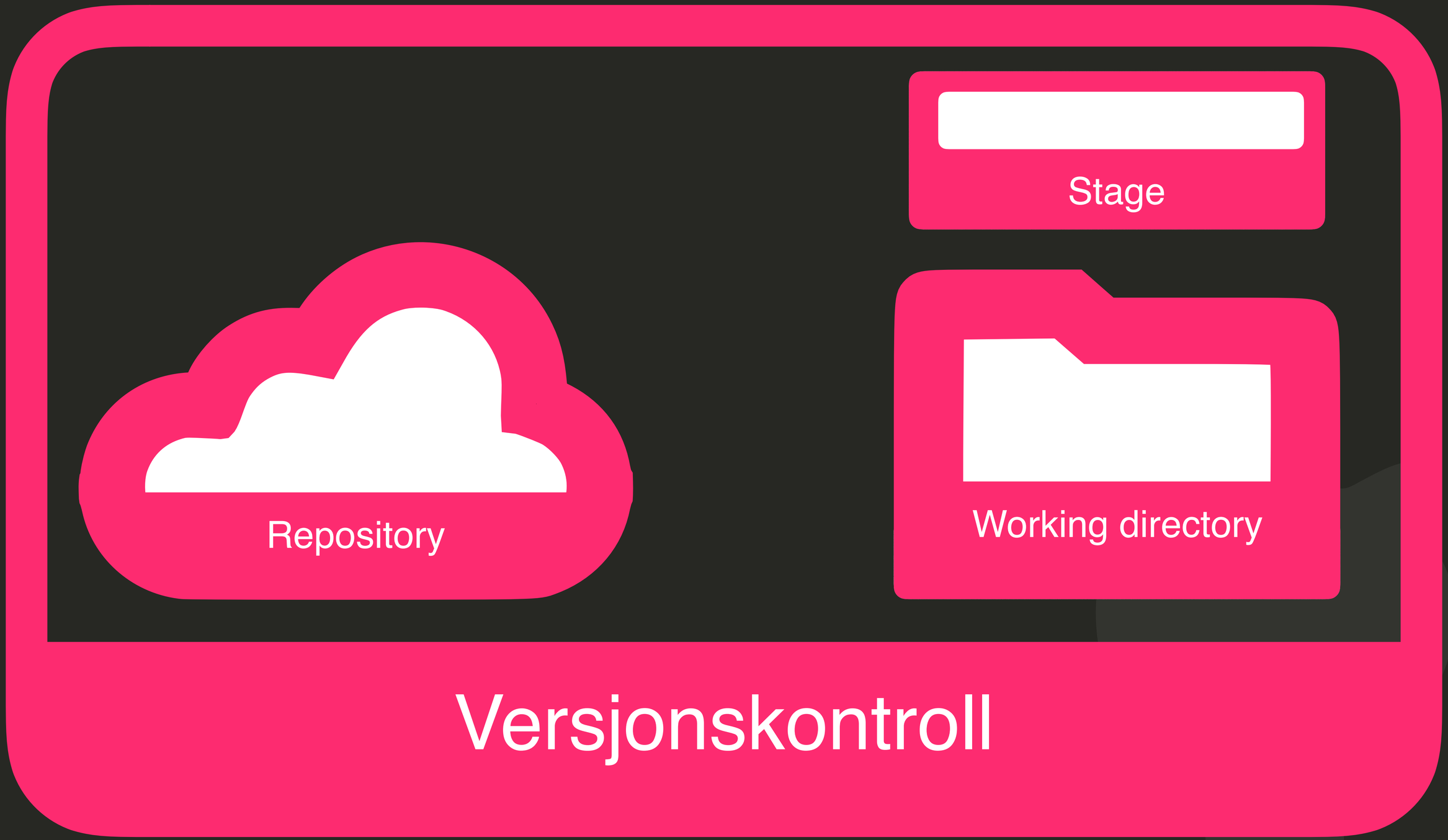
To bring files from working directory into your repo, it must go through state







Versjonskontroll


Stage
Repository
Working directory
Versjonskontroll

Tools

VCS

GUI tools

Repo hosting

Tools


git
SUBVERSION
mercurial
CVS
VCS
Atlassian
SourceTree
BASH
THE BOURNE-AGAIN SHELL
GUI tools
GitLab
GitHub
Atlassian
Bitbucket
Repo hosting

> pull

Get the latest
from the server

Slightly different
than clone or
fetch

```
$git pull
```

concepts

> pull

> push

Push your local commits to the remote server

At this point your commits are visible for everyone, so don’t be stupid

```
$ git push <remotename> <branchname>
```

concepts

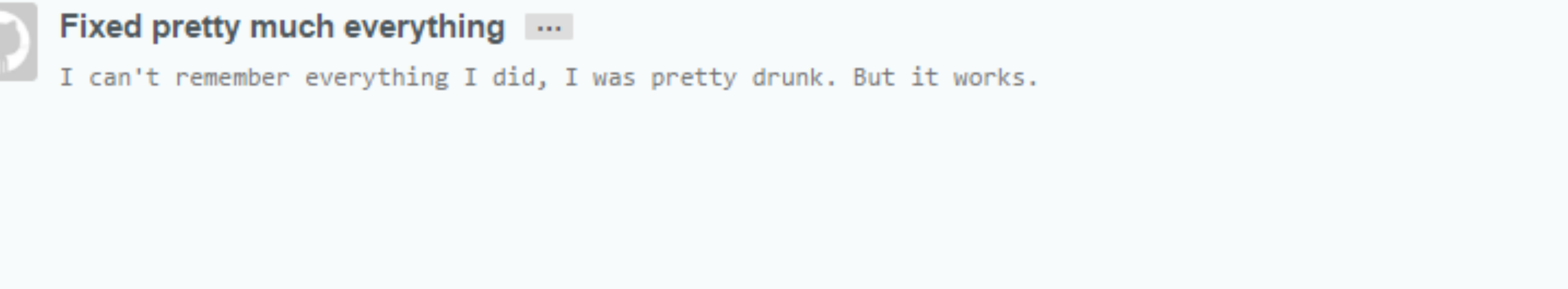
> pull

> push

> commit

Adds a points to the history, it will only exist locally before you push to the server

```
$git commit -m ”beskrivelse”
```



> pull
> push
> commit

> status

It just gives you a status of your repo. Obviously.

Use this all the time, it will save you so much headache

```
$git status
```

concepts

> pull

> push

> commit

> status

> add

Adds the file to stage, ready to be committed. Use glob patterns to save some time, or . to add all

```
$git add [filnavn]
```

concepts

> pull

> push

> commit

> status

> add

> branch

It makes a branch… Or delete it if you add -D

```
$git branch [ny-branch]
```

concepts

> pull
> push
> commit
> status
> add
> branch

> checkout

Changes active branch in your working directory. Add param -b to create a branch and check it out in one step

```
$git checkout [branchnavn]
```

concepts

> pull
> push
> commit
> status
> add
> branch
> checkout

> merge

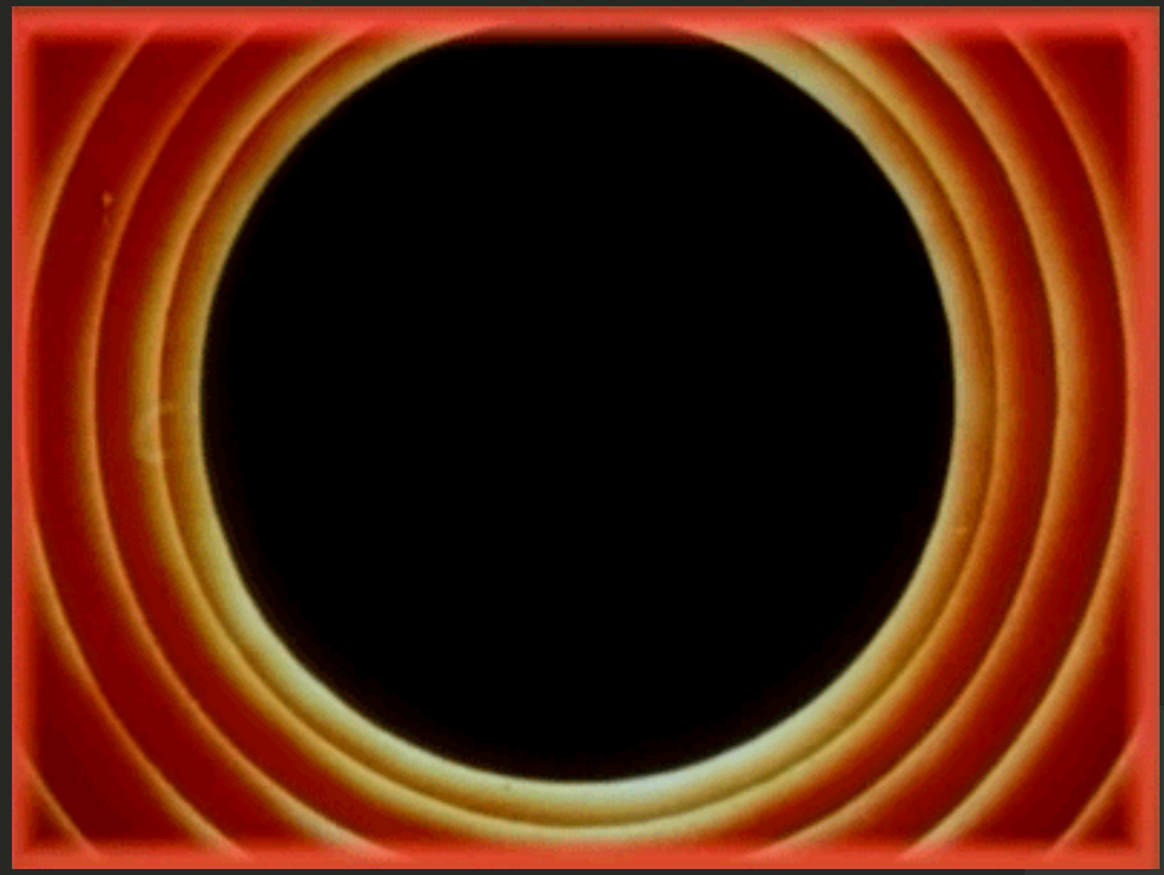
Take content from target branch and merge it into the current branch of the repo

```
$git merge [branchnavn]
```

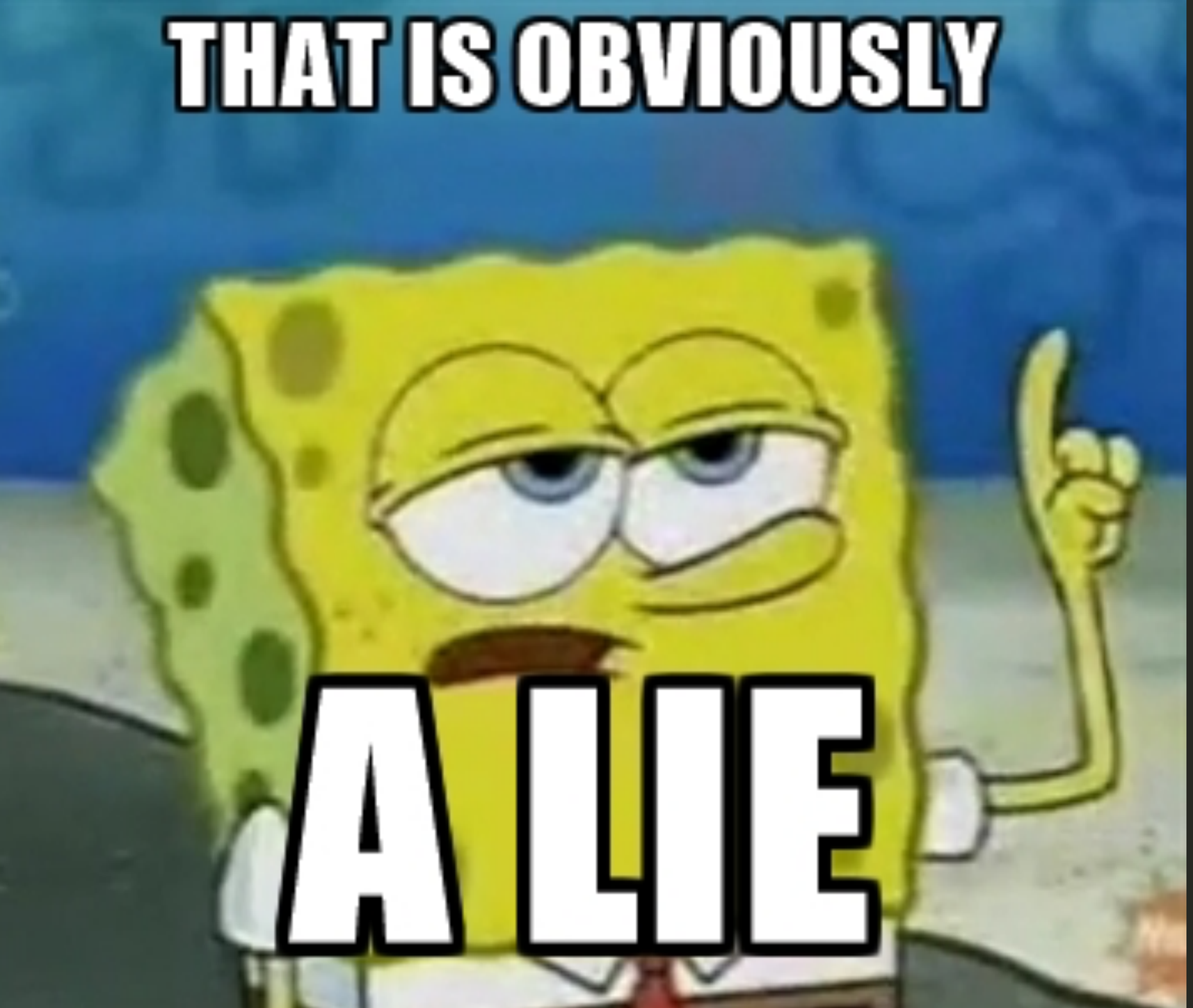
Yes, we can discuss rebase.. spoiler: It’s awesome

concepts

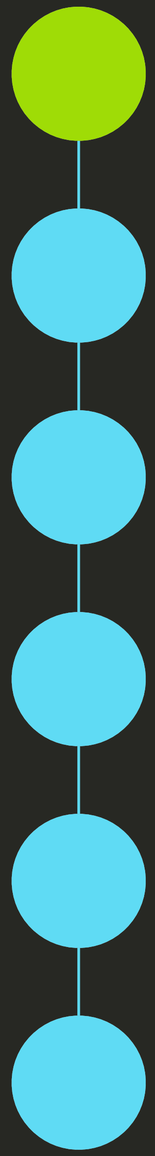
> pull
> push
> commit
> status
> add
> branch
> checkout
> merge



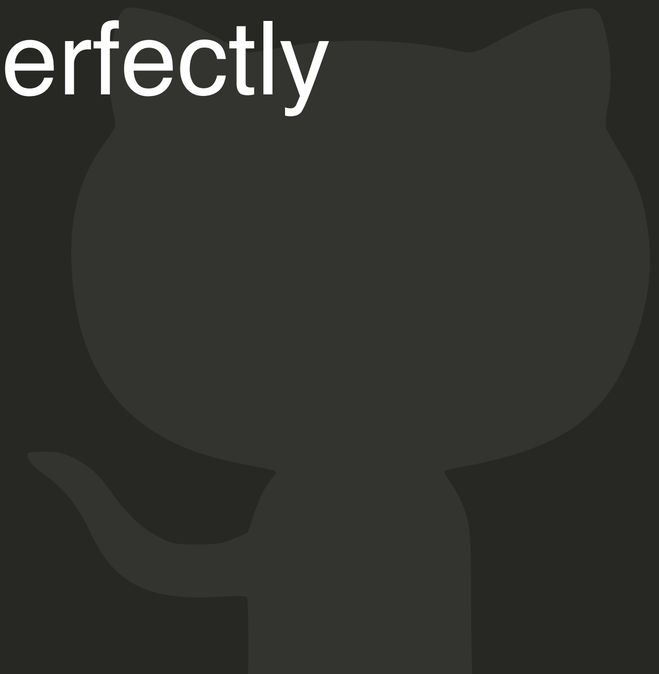
concepts

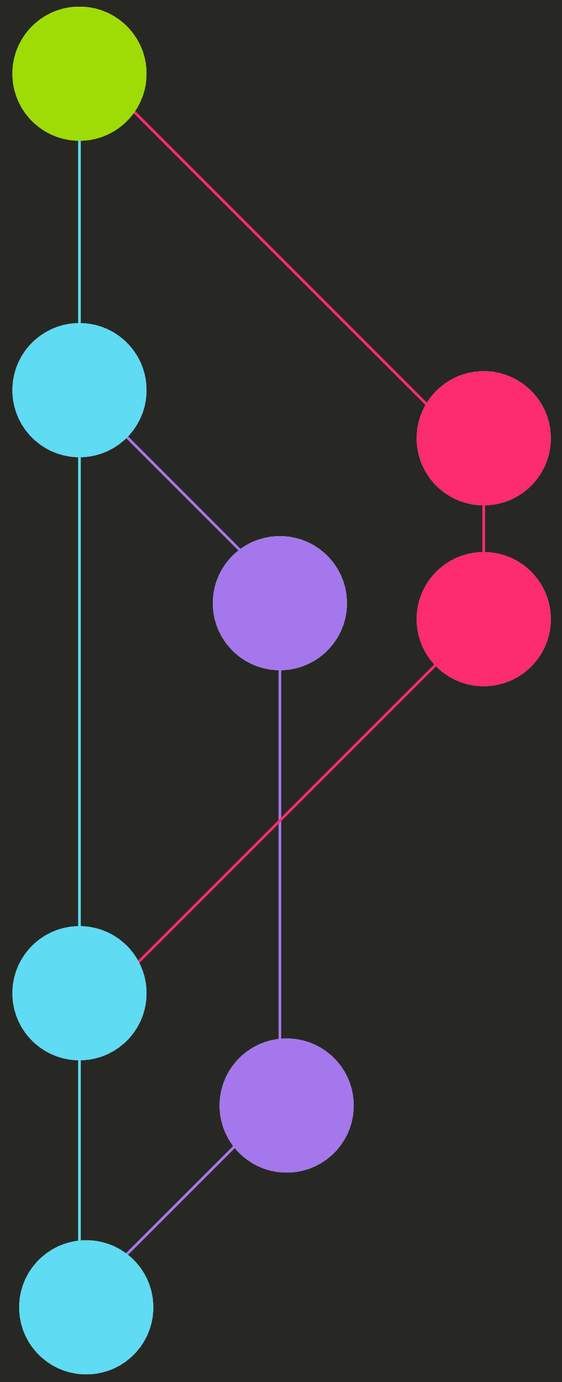

THAT IS OBVIOUSLY
A LIE

workflow



In your private project, on a single computer, a single line is perfectly adequate

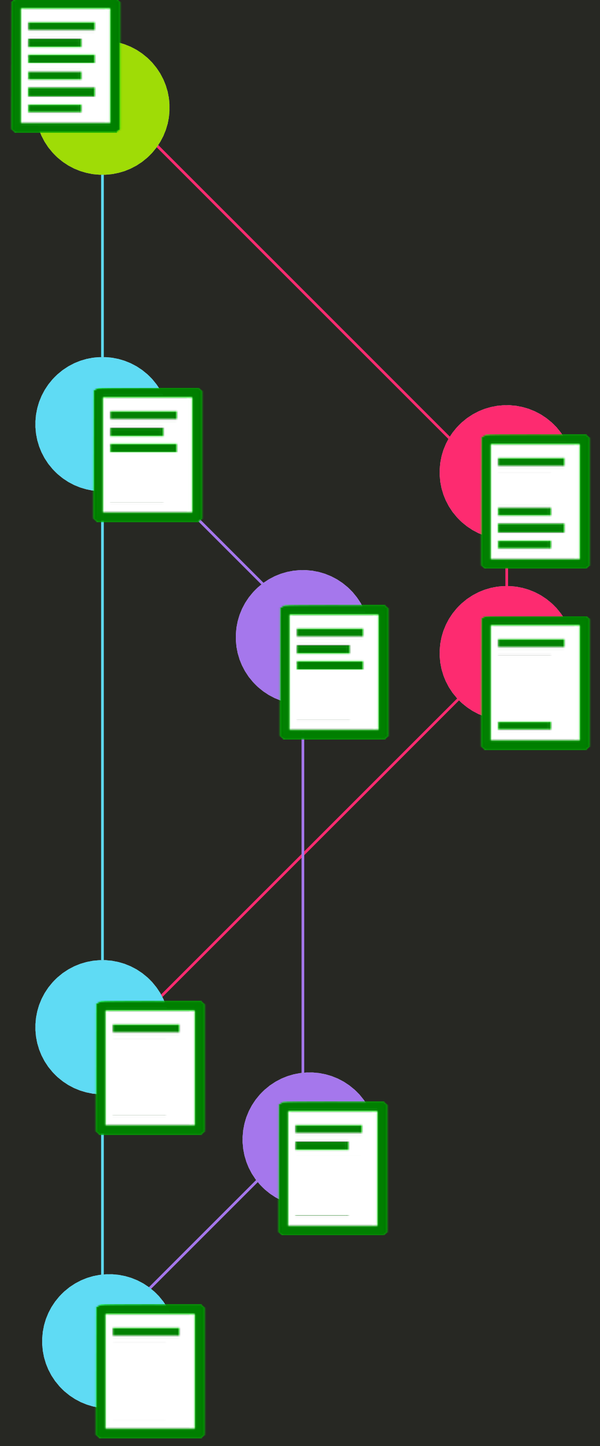




However in real life, you will work with others. Then you will encounter a somewhat more feature based flow, with a ton of branches…

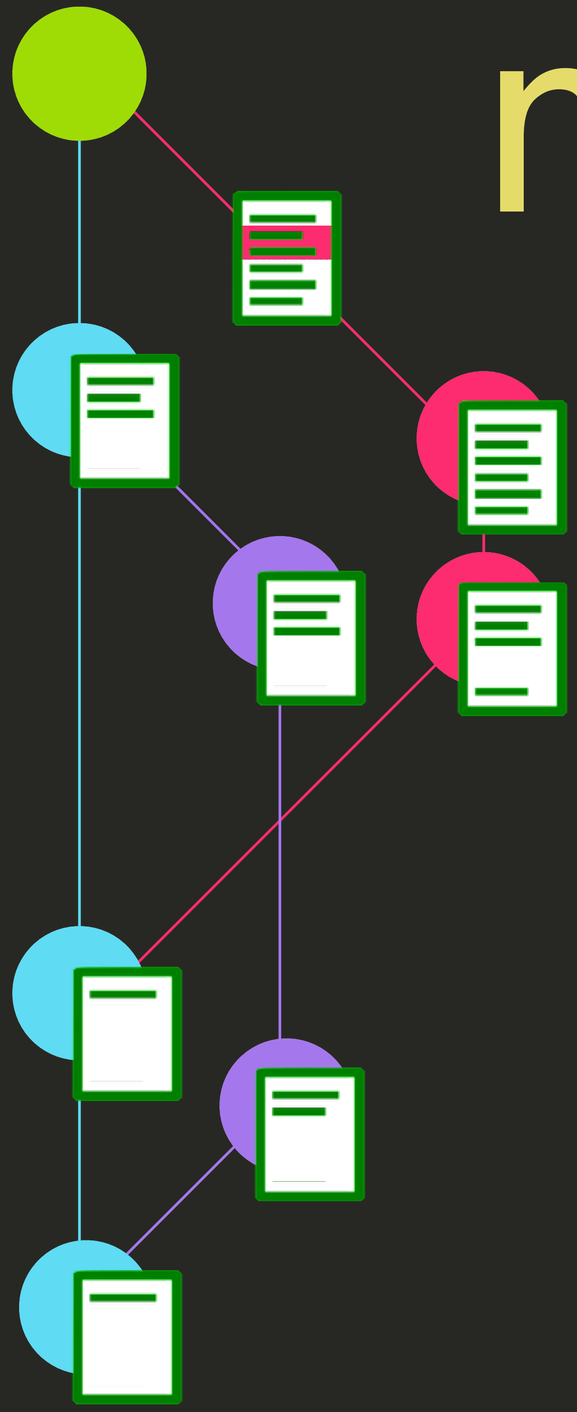
Workflow

merge



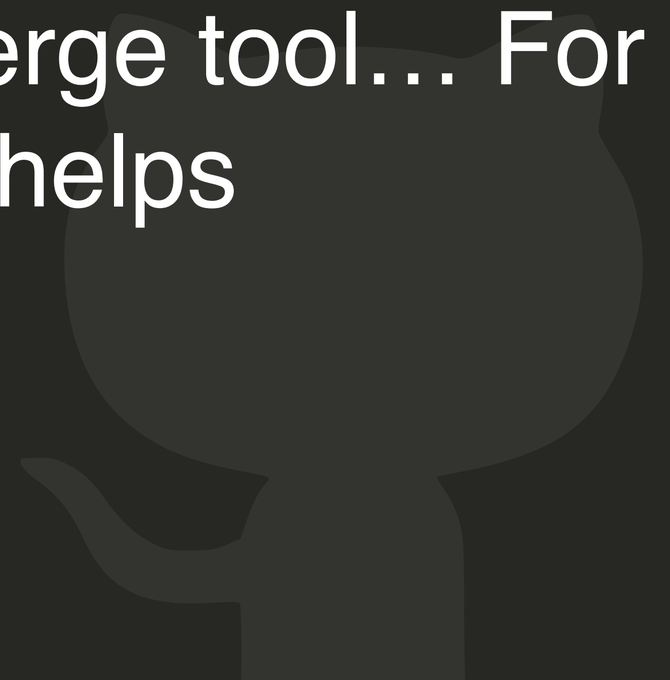
Lets take a look at this awesome animation to se how a merge works

merge conflicts



This happens… a lot…

As much as I don’t like to admit it, Intellij has a pretty awesome merge tool… For VS Code, `better merge` helps



> linux

sudo yum install git-all

sudo apt-get install git-all

> macOS

git

> Windows

git-scm.com/download/win

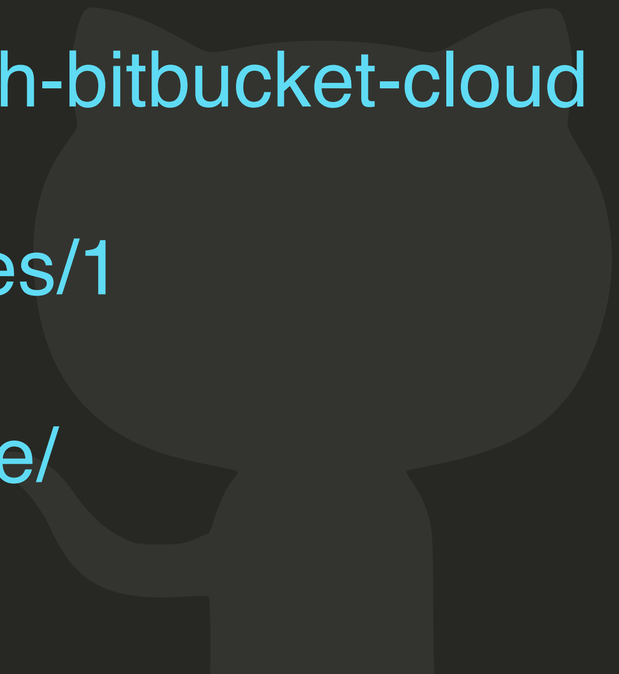
```
$ git clone https://github.com/lfberge/versjonskontroll
```

https://services.github.com/kit/downloads/github-git-cheat-sheet.pdf

https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud

https://try.github.io/levels/1/challenges/1

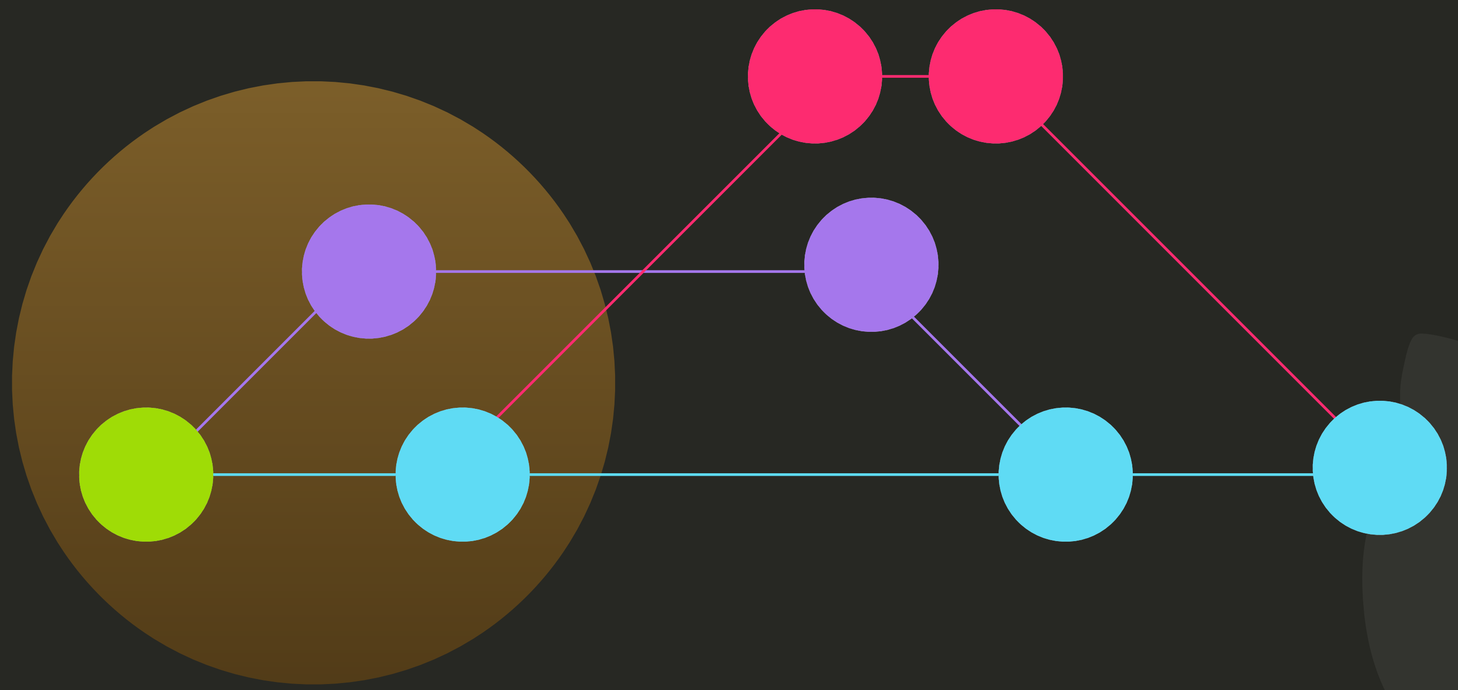
http://rogerdudler.github.io/git-guide/



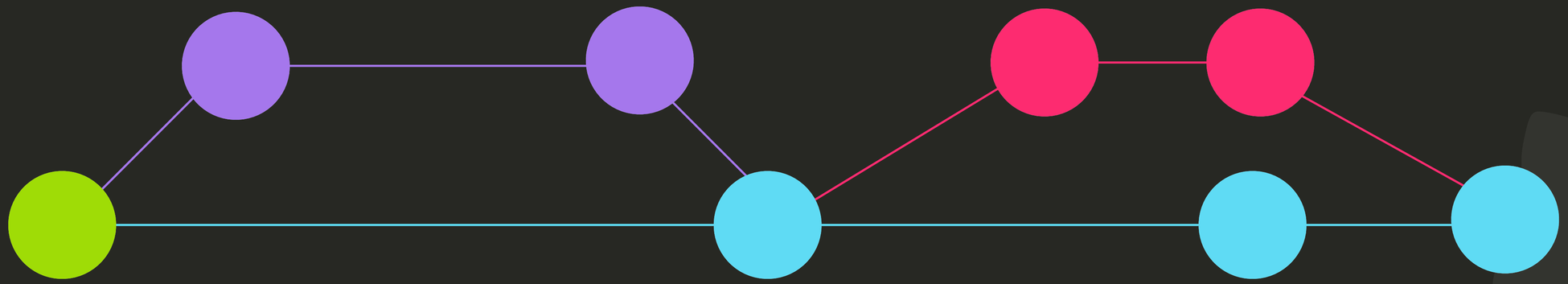
Advanced

So yeah… What was wrong with merge again?

rebase

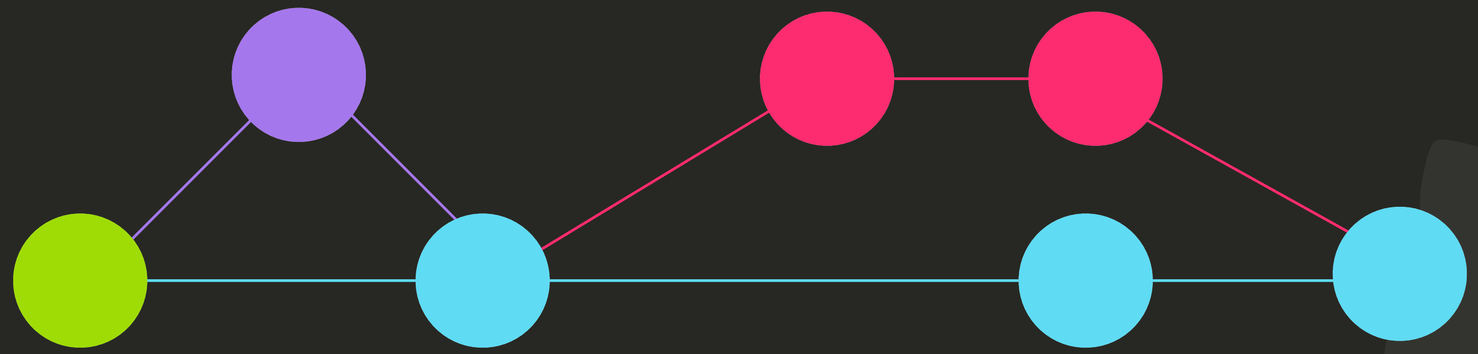


rebase



rebase

Lets squash the dumb stuff, nobody needs to see that



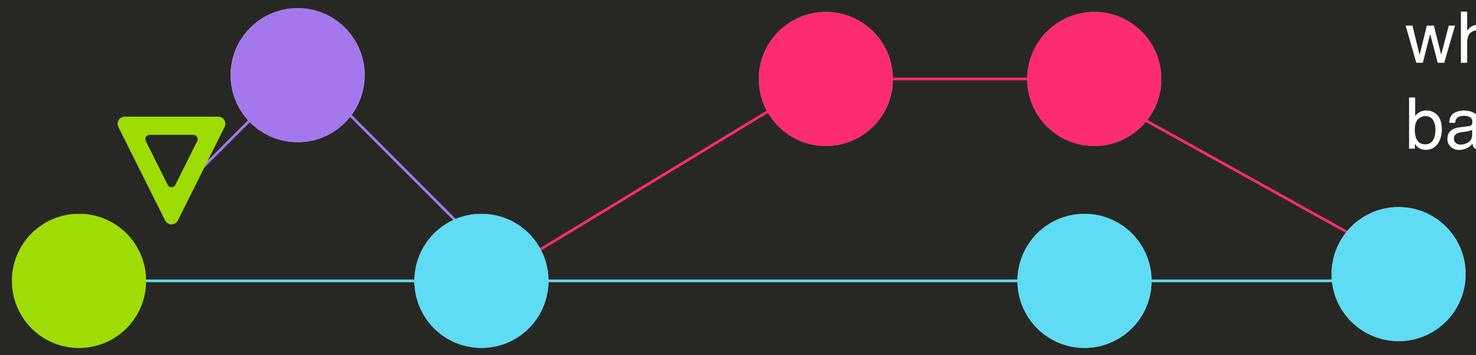
Pull request

Or merge request, which is a superior name, that only gitlab people use… So lets be cool and use the stupid name Pull request or PR

Pull request

Should you do this for your thesis?

Yes.
Yes you should.
Really.



1. Do some awesome coding
2. Create a Pull request
3. Someone on your team reviews your awesome code
4. They do not understand and reject
5. You document your code and reopen the PR
6. And this goes on for a while, making the code base much much better

A nice side effect is that you become a better coder