Beginning Git

presentation_final_last_v1.pdf

Dhananjay Balan

Port Zero GmbH

DENOG12, November 8, 2020

Hi!

I’m Dhananjay

Software Team Lead at Port Zero GmbH

https://dbalan.in

https://port-zero.com

@notmycommit

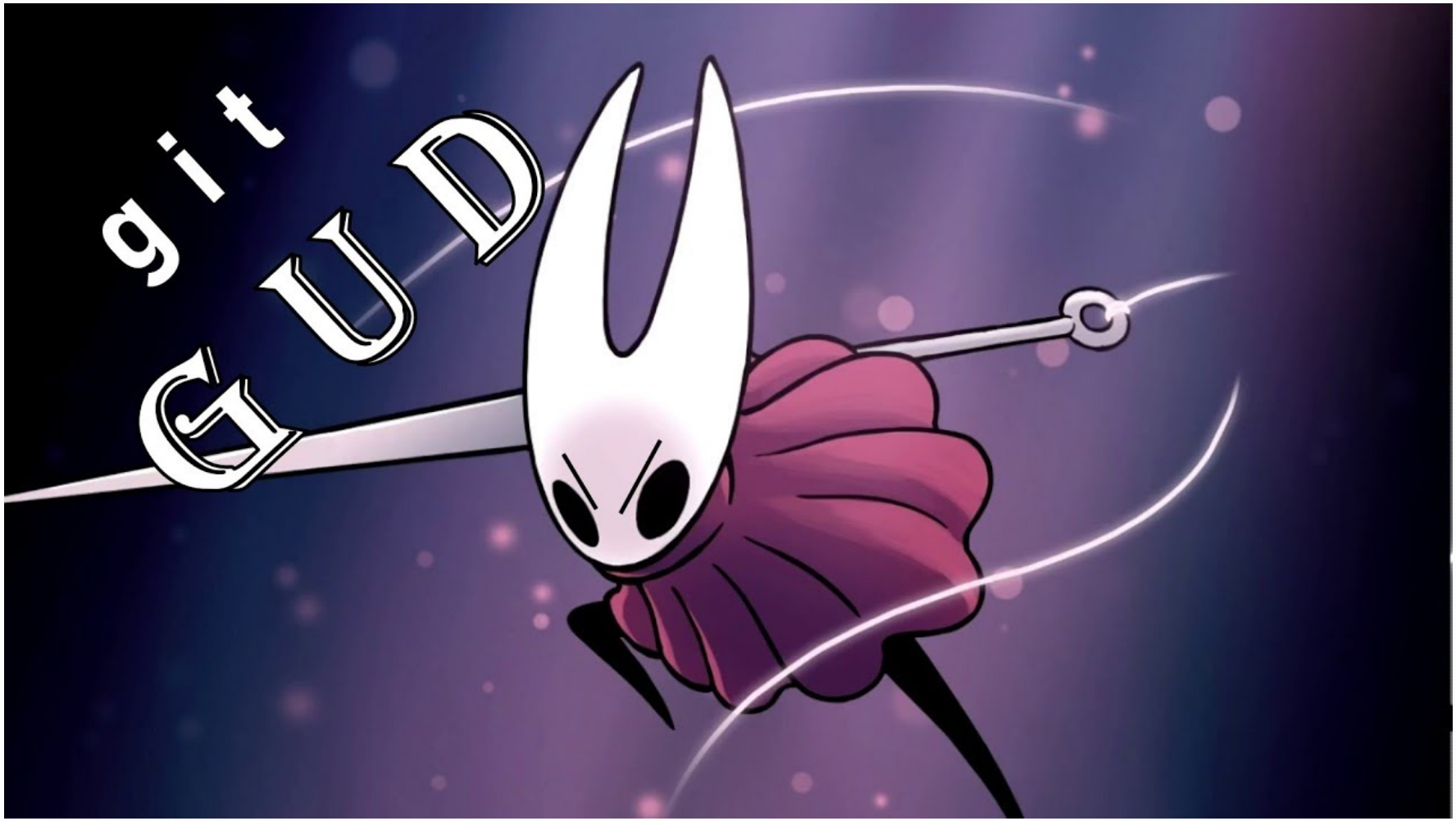
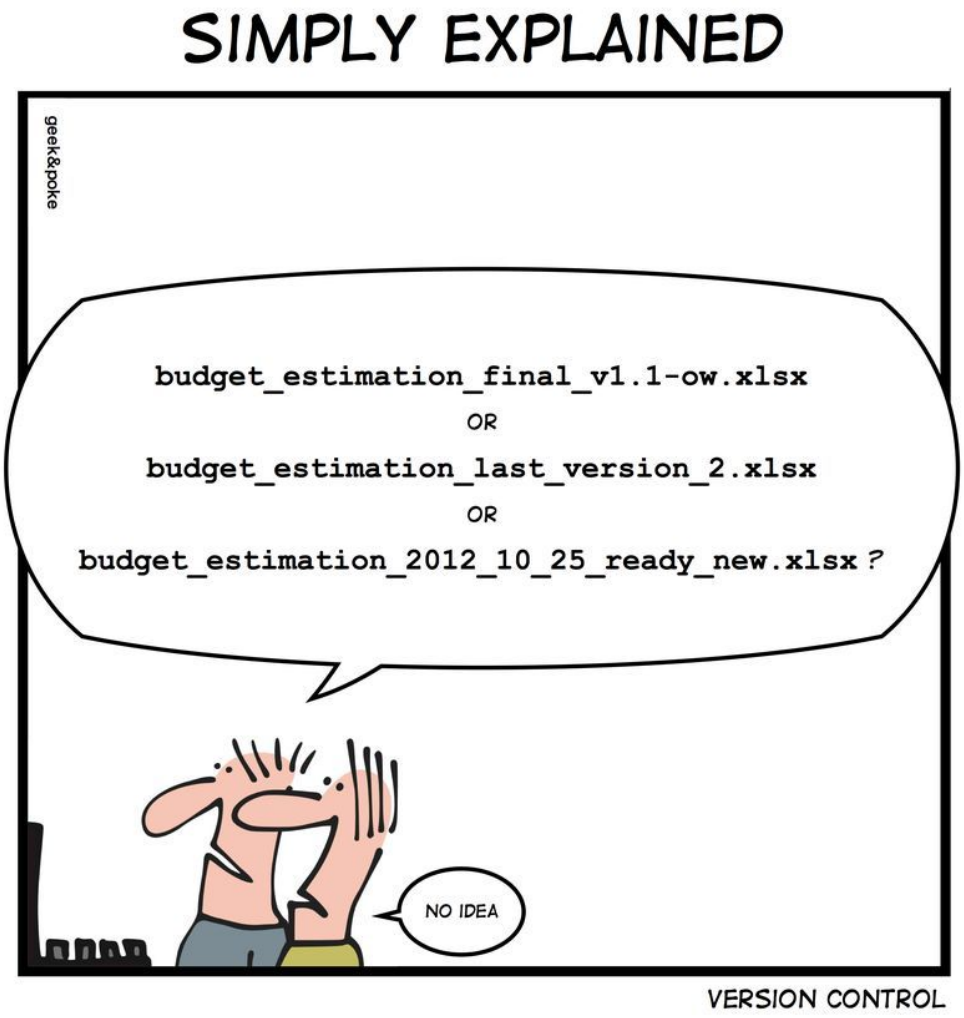


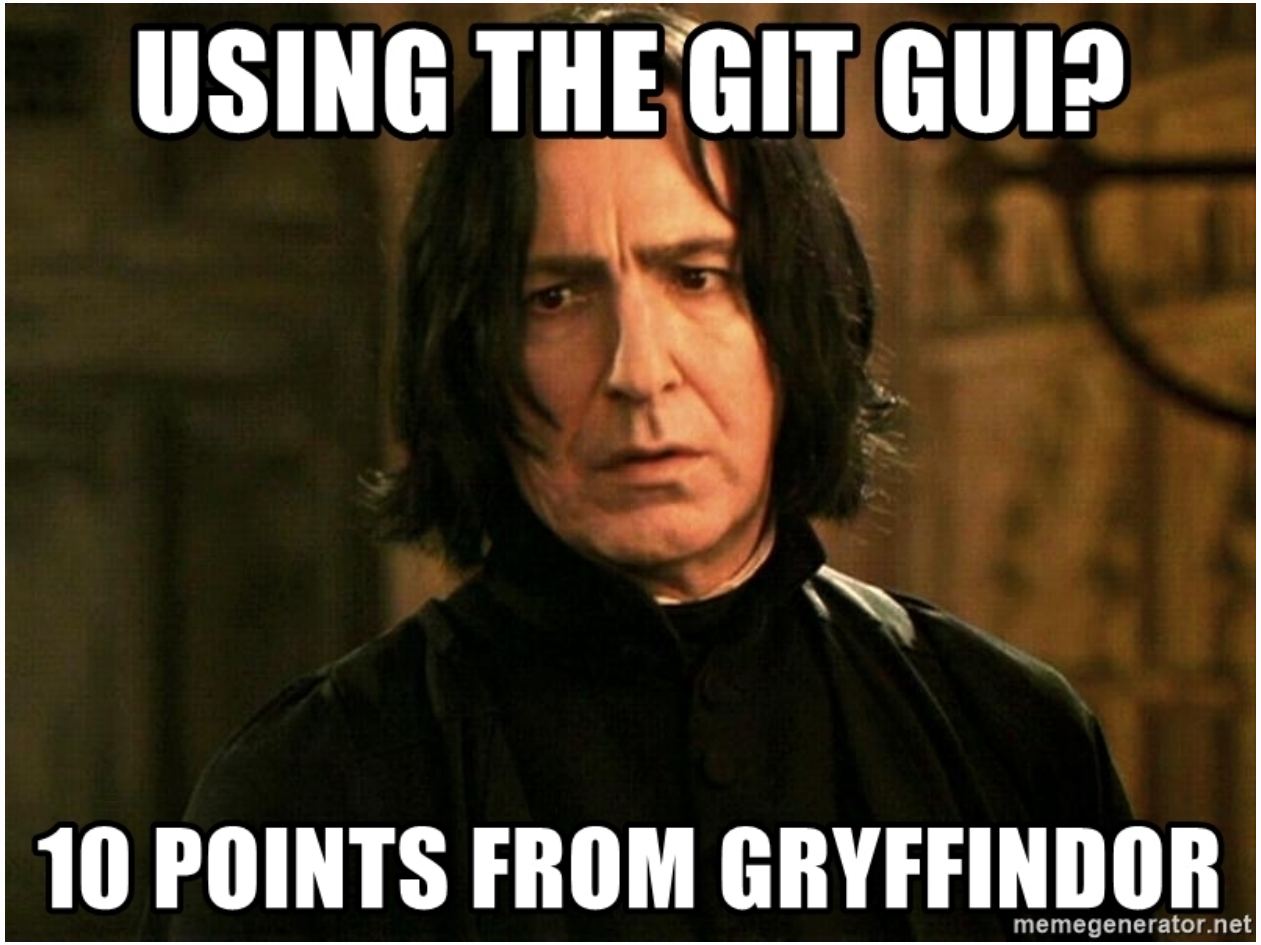
Figure 1: source: https://www.youtube.com/watch?v=ui9hFeUrRrI



Idea from Jen Simmons and John Albin Wilkins during episode #40 of "Web Ahead" about Git:
http://5by5.tv/webahead/40

Figure 2: Why git?


GIT INIT
I GUESS THINGS ARE GETTING PRETTY SERIOUS
makeameme.org



UI’s add their own terminology and workflows (git has many), so to start we will not be using that

Getting help

- `git help commit`
- git glossary – https://git-scm.com/docs/gitglossary

```
# create a repository
$ git init

# check status of the repo
$ git status

# start tracking a file, add the current changes to
$ git add <file>

# commit changes
$ git commit

# query git history
$ git log
```

Repository

- Files are stored in a repository.
- Working directory and the `.git` directory that contains files.

Creating a repository

```
$ mkdir beginning_git
$ cd beginning_git
$ ls -al
$ git init
$ ls -al
```

Commits

```
# add a file and track with git
$ echo "hello denog12!" > readme.txt
$ git add readme.txt
$ git status

On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   readme.txt
```

Commits (Cont)

```
git diff --staged

git commit
```

Time travel

```
$ git log --graph

* commit c304b64a276824a40e1089b00dac716df9be4974 (HEAD -> maste
| Author: Dhananjay Balan <mail@dbalan.in>
| Date:   Sat Nov 7 11:43:24 2020 +0100
|
|     Third commit
|
* commit bc4a9cdbf398924abdf6c9df7e8fab867b2f918b
| Author: Dhananjay Balan <mail@dbalan.in>
| Date:   Sat Nov 7 11:32:57 2020 +0100
|
|     My Second commit
|
* commit 7400ea427e13bf627b19044bfbca32a0063732cd
  Author: Dhananjay Balan <mail@dbalan.in>
  Date:   Sat Nov 7 11:32:16 2020 +0100

      My first commit
```

- Each commit is identified by a hash.
- `git show`
- `git diff`

- git stores full snapshots of every file (git repos can get really big)
- Fails miserably on binary files.

```
# create a new branch
$ git branch

# switch branches
$ git switch <branchname>

# merge branches
$ git merge
```

Branches

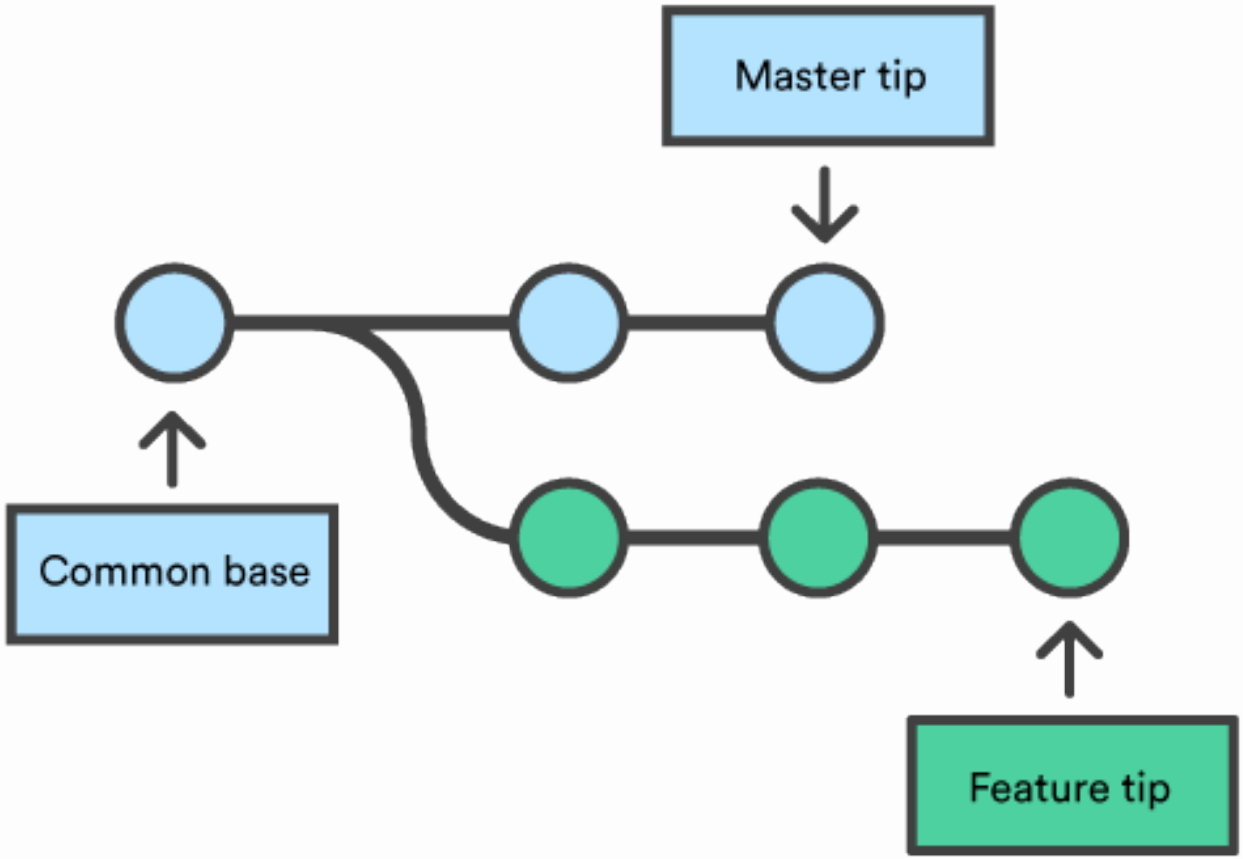


Figure 3:
https://www.atlassian.com/cs/git/tutorials/using-branches/git-merge

```
# look at branches
$ git branch

# create a new branch
$ git branch feature

# query branch again
$ git branch

# switch
$ git switch feature

# what does branch information say?
```

Merging changes back.

- Create few commits in feature branch.
- Observer the repository structure with `git log --graph --all` (all queries all branches)

Merging changes back

```
git diff <other branch>

git switch master
git merge feature
```

Detour, getting a remote repository

```
git clone - Clone a repository into a new directory
git clone https://github.com/dbalan/merge-conflict-w
```

Merging changes back, hard way

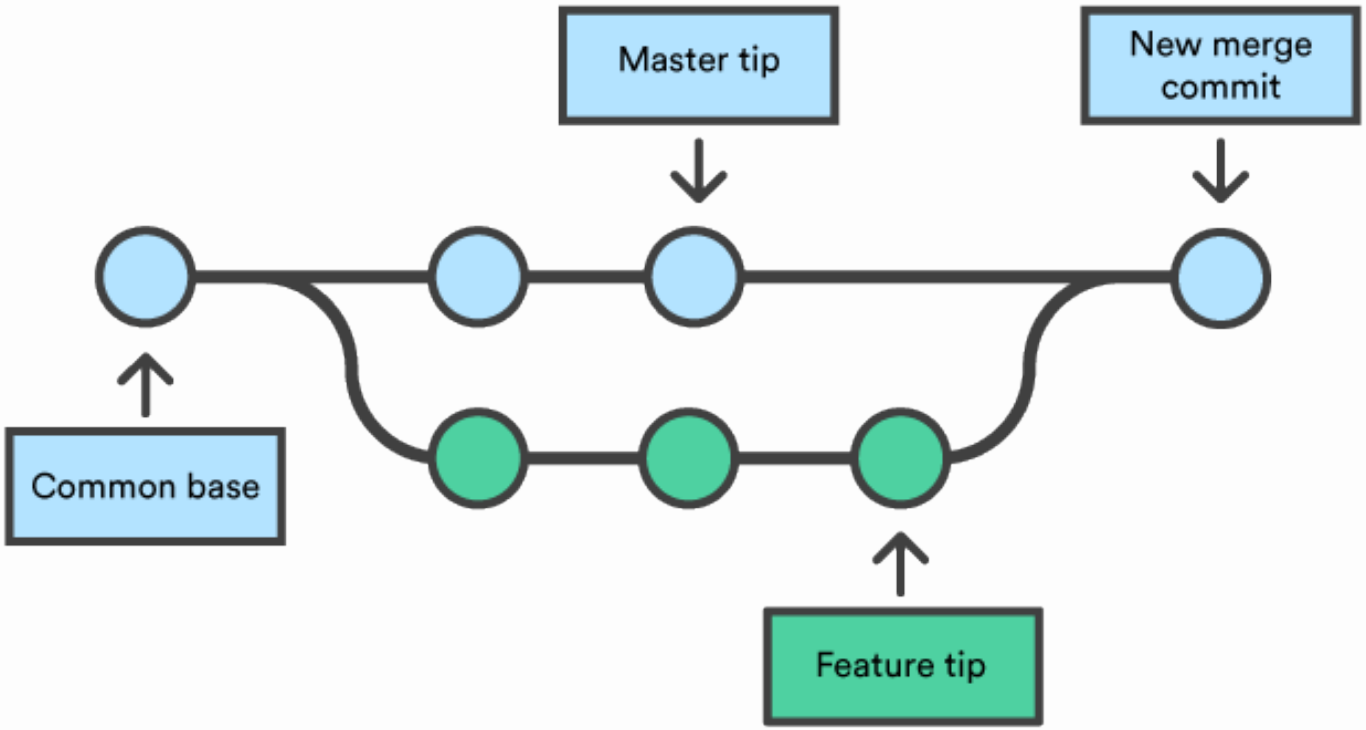


Figure 4:
https://www.atlassian.com/cs/git/tutorials/using-branches/git-merge

How do we merge feature branch into master?

Conflicts!

```
CONFLICT (add/add): Merge conflict in conflict.txt
```

```
<<<<<<< HEAD
# Nothing to see here.

The best branch ofcourse is *master*.

=======
# Which one is the best branch.

The answer of course is the *feature* branch
>>>>>>> feature
```

Many Ways.

- `git send-email`
- `git push/pull`
- `git format-patch`

GitHub/GitLab

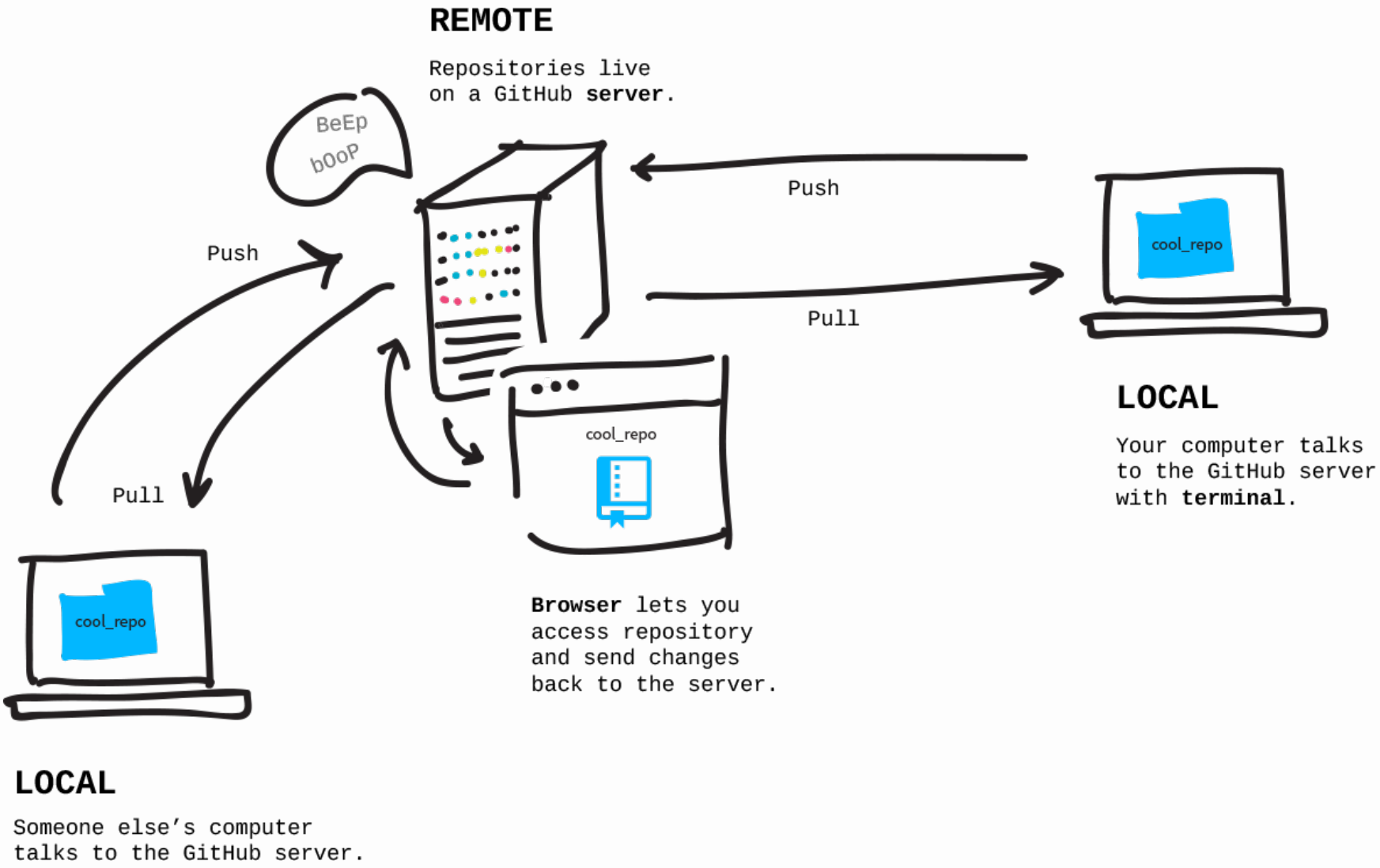


Figure 5: http://ilord.us/git_it/challenges/remote_control.html

- User creates github.com/username/the-awesome-code
- Clone github.com/username/the-awesome-code
- use git push, pull to sync code.

Pull Requests

- Contribute to repositories that you don't have access to.
- ..or for review processes.
- Lets look at an example; Denog Chatterlist https://github.com/denog/chatterliste

Explain the workflow

- Make a pull request to github repo
- Get those changes back.

Part IV: Intermediate git foo

- go back on changes
- git blame
- Git for large files https://git-lfs.github.com/

Part V: Advanced git

- git cherry-pick
- git rebase
- git bisect
- git reflog

Fin

- https://rogerdudler.github.io/git-guide/
- https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud
- http://jlord.us/git-it/