Uploading Large Files to GitHub

3 ways to avoid getting error messages during upload

Eden Au May 26, 2019 · 3 min read ★

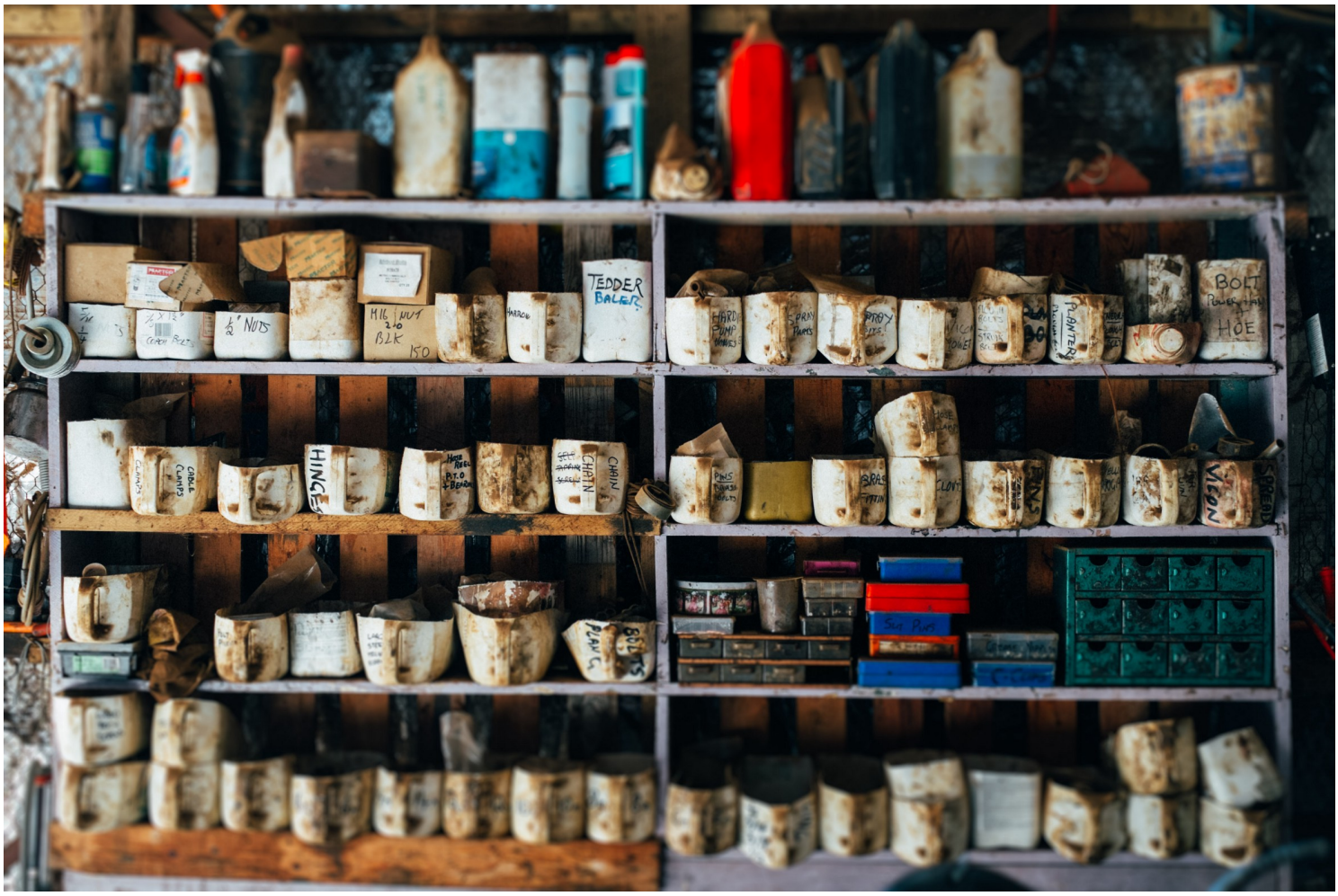

Photo by Jay Wennington on Unsplash

GitHub has a strict file limit of 100MB. If you are just uploading lines of codes, this is not

ways to overcome the **100MB limit**.

Originally published on my blog edenau.github.io.

1. .gitignore

Create a file ***.gitignore*** in the parent directory of the repository, and store all the file directories that you want Git to ignore. Use `*` for wildcard so that you do not need to add file directories manually each time you create a new large file. Here is an example:

```
*.nc
*.DS_store
```

These ignored files will be automatically ignored by Git and will not be uploaded to GitHub. No more error messages.

2. Repository Cleaner

If you have accidentally committed files locally that exceeds 100MB, you would have a hard time trying to push it to GitHub. It cannot be solved by removing the large files and committing again. This is because GitHub keeps track of every commit, not just the latest one. You are technically pushing files in your entire commit record.

Photo by Samuel Zeller on Unsplash

While you could technically resolve it by branching, it is by no means straightforward. Fortunately, you can run a ***repository cleaner*** and it automatically cleans all the large file commits.

Download BFG Repo-Cleaner *bfg.jar* and run the following command:

```
java -jar bfg.jar --strip-blobs-bigger-than 100M <your_repo>
```

It automatically cleans your commits and produces a new commit with the comment 'remove large files'. Push it and you are good to go.

3. Git LFS

You might have noticed that the abovementioned methods both avoid uploading the large files. What if you really want to upload them so that you could gain access to them on another device?

Git Large File Storage lets you store them on a remote server such as GitHub. Download and install ***git-lfs*** by placing it into your *$PATH*. You will then need to run the following command **once per local repository**:

Large files are selected by:

```
git lfs track '*.nc'
git lfs track '*.csv'
```

This will create a file named ***.gitattributes***, and voilà! You can perform add and commit operations as normal. Then, you will first need to a) push the files to the LFS, then b) push the **pointers** to GitHub. Here are the commands:

```
git lfs push --all origin master
git push -u origin master
```

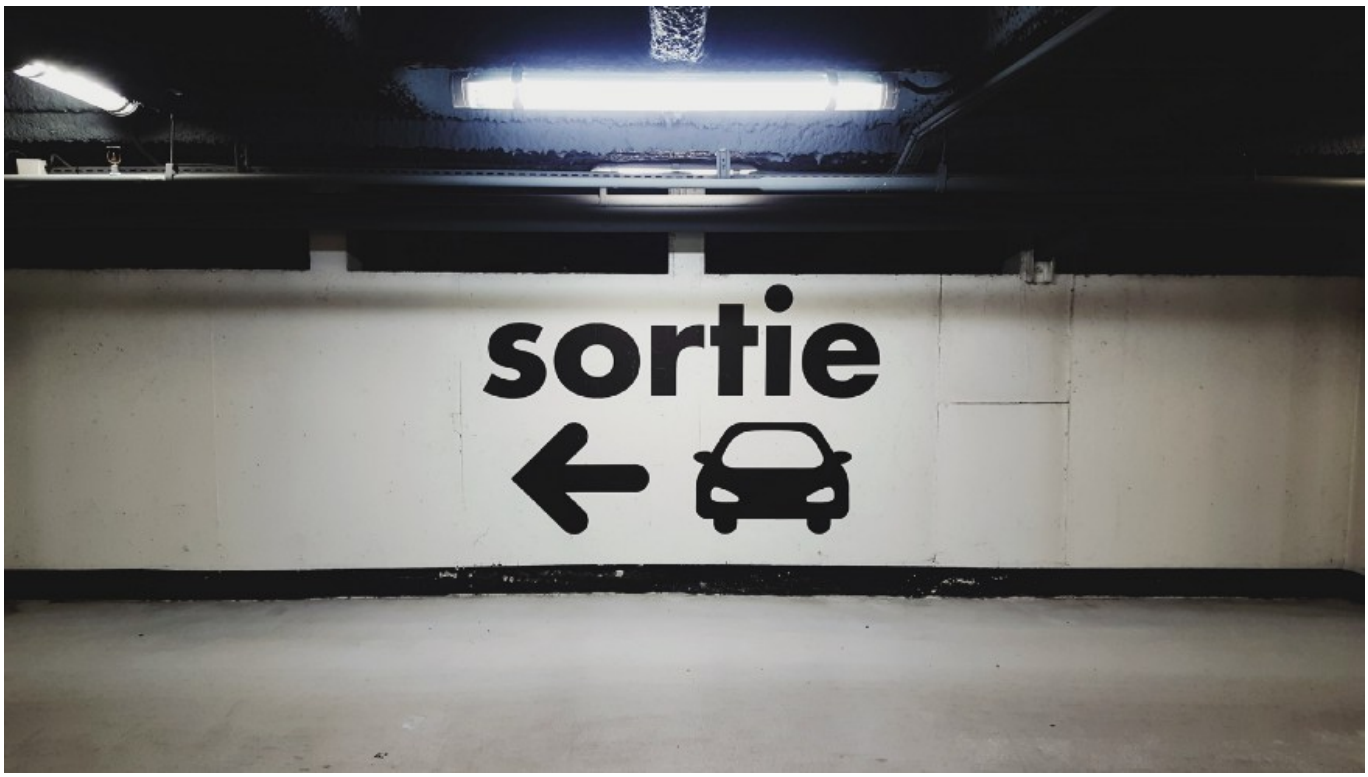

Photo by Lucas Gallone on Unsplash

Files on Git LFS are then available on GitHub with the following label.

In order to pull the repository on another device, simply install ***git-lfs*** on that device (per local repository).

Related Articles

Thank you for reading! If you are interested in data science, check out the following articles:

5 Python features I wish I had known earlier

Python tricks beyond lambda, map, and filter

towardsdatascience.com

Why Sample Variance is Divided by n-1

Explaining high school statistics that your teachers didn't teach

towardsdatascience.com

Originally published on my blog edenau.github.io.

Git Github Programming Technology