

Distributed version control with Git

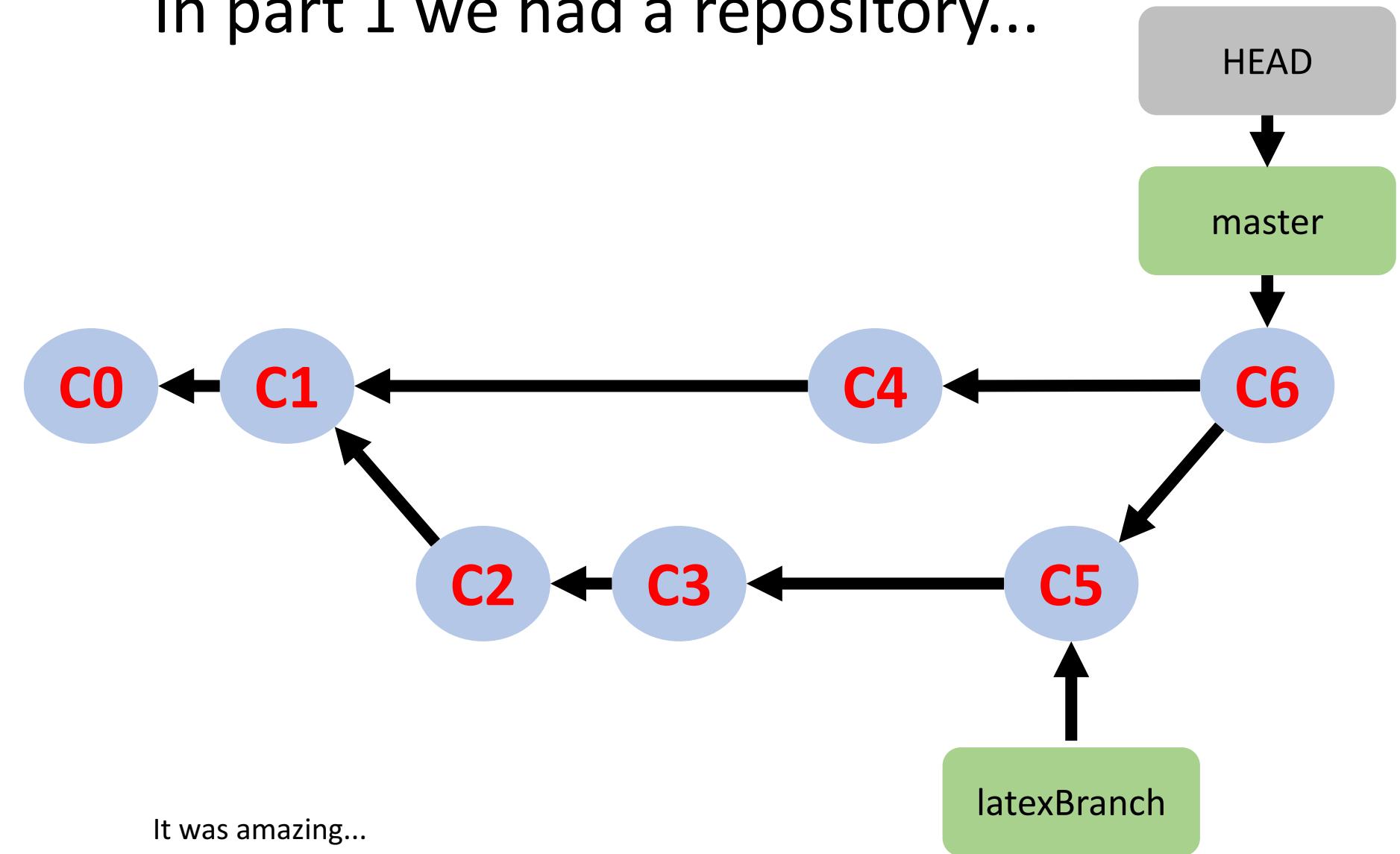


Part 2:
Working with remotes

Daniel Sussman

<https://gitlab.com/dmsussman/workingWithGit/>

In part 1 we had a repository...



It was amazing...

Since then, I've put the files and slides
in a subdirectory...

```
[dmsussma@~/repos/introToGitManningGroup$ ls *
 README.md

part1:
chapter1.tex          chapter4.rtf
chapter2.tex          gitIntroPart1.pdf
chapter3.tex          gitIntroPart1.pptx
```

Just working locally is good... but it isn't very distributed...

Why use version control?

Team work:

- * Makes working collaboratively much easier

Individual work:

- * Scientific reproducibility of code
- * Magical time machine for reverting to previous version of code



Remote repositories
will help achieve
some of these aims

Step 0: Make a GitLab account

GitLab – an open source, web-based git repository manager

Alternatives: GitHub, BitBucket...

GitLab.com

GitLab.com offers free unlimited (private) repositories and unlimited collaborators.

- [Explore projects on GitLab.com](#) (no login needed)
- [More information about GitLab.com](#)
- [GitLab.com Support Forum](#)

By signing up for and by signing in to this service you accept our:

- [Privacy policy](#)
- [GitLab.com Terms.](#)

[Sign in](#) [Register](#)

Username or email

Password

Remember me [Forgot your password?](#)

[Sign in](#)

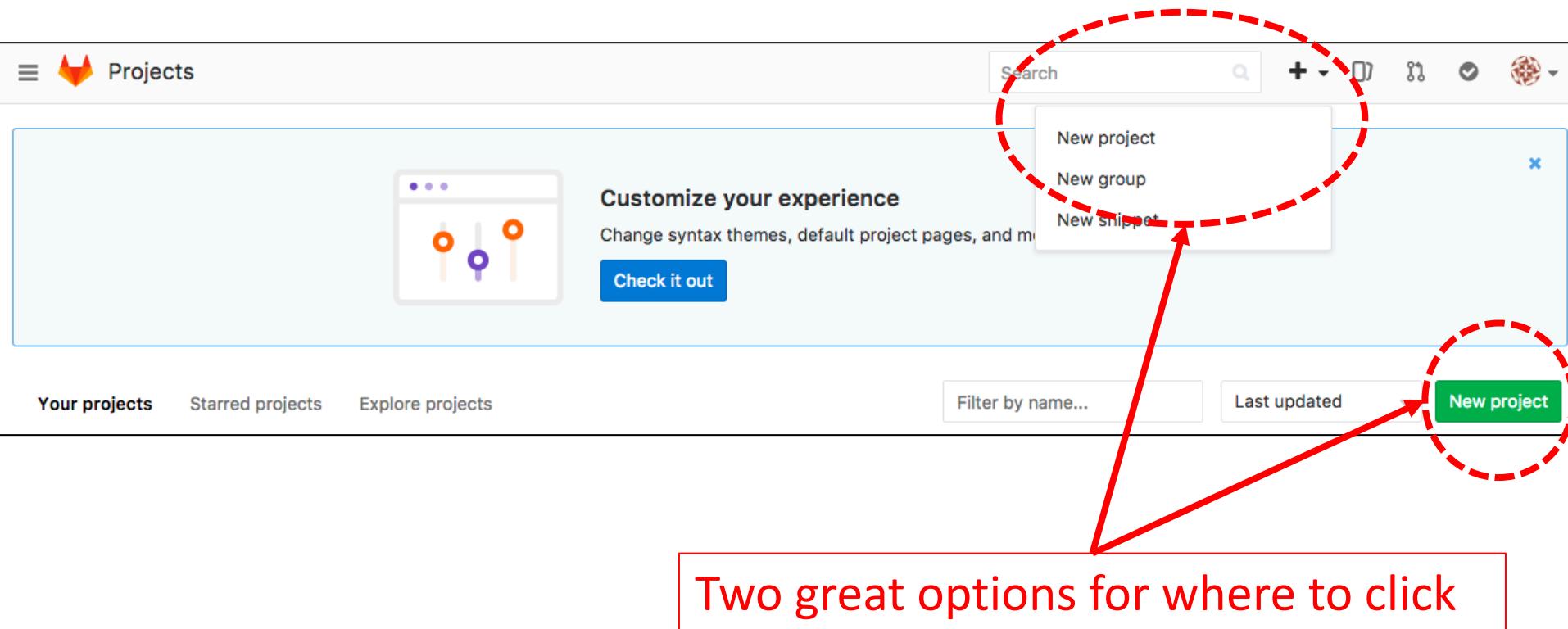
Didn't receive a confirmation email? [Request a new one.](#)

Sign in with    

Remember me

Let's set up a new project

“Project” being code for “repository”



Let's set up a new project

The screenshot shows the GitLab 'New project' creation interface. Key elements include:

- Create from template**: A section with icons for 'Blank', 'Ruby on Rails', 'Spring', and 'NodeJS Express'. The 'Blank' icon is circled in red.
- Import project from**: Options for GitHub, Bitbucket, Google Code, Fogbugz, Gitea, git Repo by URL, and GitLab export. An 'OR' button is between the template and import sections.
- Project path**: A field containing `https://gitlab.com/ dmsussman`, with the user part (`dmsussman`) highlighted in yellow.
- Project name**: A field containing `workingWithGit`, highlighted in yellow.
- Description**: A section titled 'Project description (optional)' with a 'Description format' input area.
- Visibility Level**: A section with three options:
 - Private**: Project access must be granted explicitly to each user.
 - Internal**: The project can be accessed by any logged in user.
 - Public**: The project can be accessed without any authentication.A red dashed circle highlights this entire section.
- Create project**: A green button at the bottom left.
- Cancel**: A button at the bottom right.

And *push* our existing repository to the cloud!

Assuming we're already in the project directory:

```
$ git remote add origin https://gitlab.com/dmsussman/workingWithGit.git  
$ git push -u origin --all
```

GitLab provides step by step instructions for exactly this

Existing Git repository

```
cd existing_repo  
git remote add origin https://gitlab.com/dmsussman/  
git push -u origin --all  
git push -u origin --tags
```

Comment: google any git-related question, and you will find the answer

There is now an equivalent repository...in the cloud!



A screenshot of a GitLab repository page for 'workingWithGit'.

The repository details:

- User icon: A pink circle with a white letter 'W'.
- Name: workingWithGit
- Description: A repository to house a gentle introduction to working with git
- Statistics: 2 stars, 0 forks, 0 issues.
- Clone URL: git@gitlab.com:dmsussman/workingWithGit
- Actions: Unstar, Fork, SSH, Download, Global dropdown.

Repository navigation:

- Branches: master
- Path: workingWithGit /
- Actions: History, Find file, Download.

Recent commit:

- Commit message: move additional info up in the presentations
- Author: Daniel Sussman
- Date: committed about 17 hours ago
- Hash: b35340fb

File list:

Name	Last commit	Last Update
part1	move additional info up in the presentations	about 17 hours ago
README.md	added repository readme and presentation files	about 18 hours ago

Let's make a change and *git push* it

Specifies untracked files or *patterns of files* to ignore

```
$ vim .gitignore  
$ git add .  
$ git commit -m "added a gitignore file"  
$ git push
```

Before

The screenshot shows a GitLab repository page for 'workingWithGit'. At the top, there's a summary bar with 'Unstar' (2), 'Fork' (0), 'SSH' (git@gitlab.com:dmsussman/work), and a 'Global' dropdown. Below this is a navigation bar with 'Files (16.4 MB)', 'Commits (11)', 'Branches (2)', 'Tags (0)', 'Readme', 'Add Changelog', 'Add License', 'Add Contribution guide', and 'Set up CI'. A dropdown menu shows 'master' selected. The main area displays a commit history table:

Name	Last commit	Last Update
part1	move additional info up in the presentations	about 17 hours ago
README.md	added repository readme and presentation files	about 18 hours ago

After

The screenshot shows the same GitLab repository page after a 'git push' command. The commit history table has been updated:

Name	Last commit	Last Update
part1	got rid of an accidental swap file	about a minute ago
.gitignore	added a gitignore file	2 minutes ago
README.md	added repository readme and presentation files	about 18 hours ago

Getting new local copies: *git clone*

It is easy to clone new copies:

```
$ git clone https://gitlab.com/dmsussman/workingWithGit.git  
      newDirectory
```

This copies the repository at the specified location and puts it in a new directory (called, in this case “newDirectory”)

Work on multiple machines, use as a backup, ...

Let's get collaborative!

Project Repository Registry Issues 0 Merge Requests 0 Pipelines Wiki Snippets **Members** Settings

Users were successfully added.

Project members

You can add a new member to workingWithGit or share it with another group.

Add member Share with group

Select members to invite

Staniel Sussman X

Choose a role permission

Guest
 Reporter
 Developer
 Master

Expiration date

Add to project Import

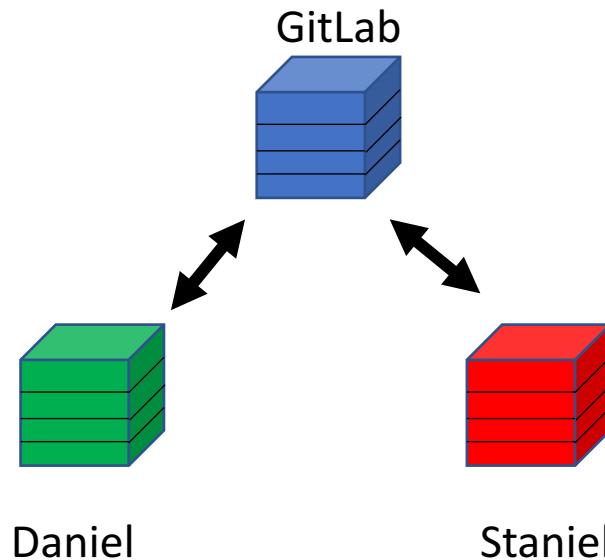
I just searched for Master and I found Staniel Sussman (almost) completely added to the repository

Staniel *git clones* the repository

Plot twist: “Staniel Sussman” is also me, using the Manning Group GitLab account.

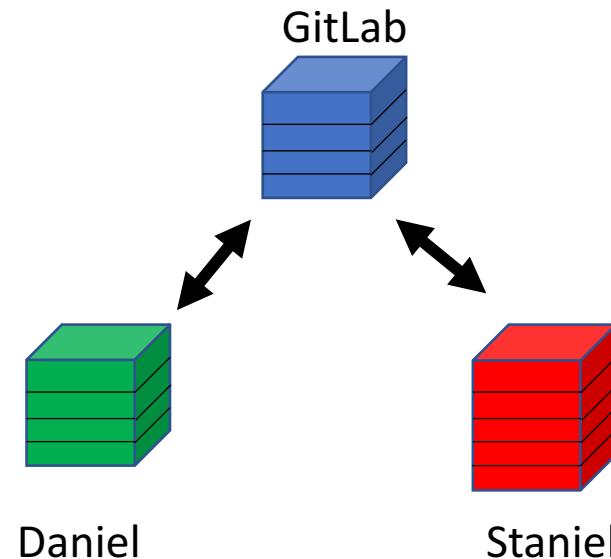
```
[dmsussma@~/repos/garbage]$ git clone https://ManningGroup@gitlab.com/dmsussman/workingWithGit.git
Cloning into 'workingWithGit'...
[Password for 'https://ManningGroup@gitlab.com':
remote: Counting objects: 51, done.
remote: Compressing objects: 100% (48/48), done.
remote: Total 51 (delta 19), reused 0 (delta 0)
Unpacking objects: 100% (51/51), done.
[...]
```

Three independent but linked copies



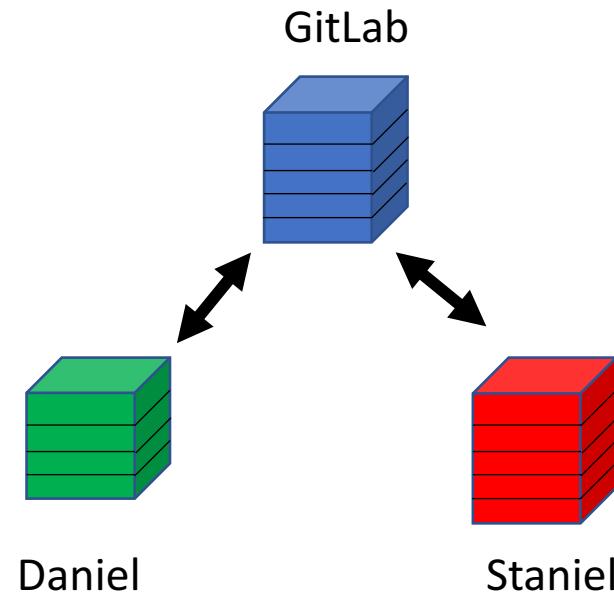
Staniel edits the repository

```
$ mkdir part2  
$ vim part2/file1.txt  
$ git add .  
$ git commit -m "starting part2"
```



Staniel pushes the changes

```
$ git push
```



Daniel's repository *does not* auto-update

```
[dmsussma@~/repos/introToGitManningGroup$ls *
 README.md
```

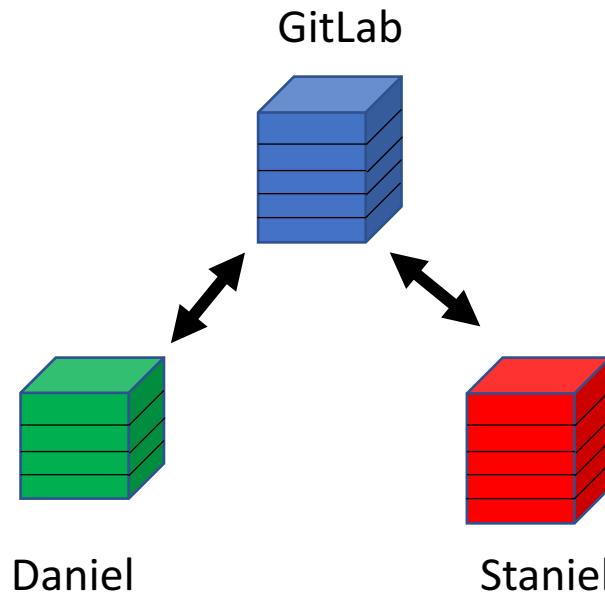
part1:

chapter1.tex
chapter2.tex

chapter3.tex
chapter4.rtf

gitIntroPart1.pdf
gitIntroPart1.pptx

```
[dmsussma@~/repos/introToGitManningGroup$git status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working tree clean
```



Daniel *git pulls* changes from repository

```
[dmsussma@~/repos/introToGitManningGroup$git pull
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 1), reused 0 (delta 0)
Unpacking objects: 100% (4/4), done.
From https://gitlab.com/dmsussman/workingWithGit
  82019cb..a5ce1ff  master      -> origin/master
Updating 82019cb..a5ce1ff
Fast-forward
  part2/file1.txt | 1 +
  1 file changed, 1 insertion(+)
  create mode 100644 part2/file1.txt
[dmsussma@~/repos/introToGitManningGroup$ls *
 README.md

part1:
chapter1.tex          chapter3.tex          gitIntroPart1.pdf
chapter2.tex          chapter4.rtf           gitIntroPart1.pptx

part2:
file1.txt
```



Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel tries to push his changes?



Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel tries to push his changes?

```
$ vim part2/file1.txt
```

```
$ git add .
```

```
$ git commit -m "file1 in part2 edited by Stan"
```

```
$ git push
```



Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel tries to push his changes?

```
$ vim part2/file1.txt
```

```
$ git add .
```

```
$ git commit -m "file1 in part2 edited by Dan"
```



Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel tries to push his changes?

```
$ git push
```



```
dmsussma@~/repos/introToGitManningGroup$git push
To https://gitlab.com/dmsussman/workingWithGit.git
 ! [rejected]      master -> master (fetch first)
error: failed to push some refs to 'https://gitlab.com/dmsussman/workingWithGit.
git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel tries to pull?

```
$ git pull
```

A close-up of the Muppet character Beaker, showing his orange, spiky hair and large white eyes, looking slightly worried.

```
[dmsussma@~/repos/introToGitManningGroup$git pull
remote: Counting objects: 4, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 1), reused 0 (delta 0)
Unpacking objects: 100% (4/4), done.
From https://gitlab.com/dmsussman/workingWithGit
 7889268..478d09e  master      -> origin/master
Auto-merging part2/file1.txt
CONFLICT (content): Merge conflict in part2/file1.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel checks the status?

\$ git status

```
[dmsussma@~/repos/introToGitManningGroup$git status
On branch master
Your branch and 'origin/master' have diverged,
and have 1 and 1 different commits each, respectively.
(use "git pull" to merge the remote branch into yours)
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)

    both modified:  part2/file1.txt
```



Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel looks at that file?

```
$ vim part2/file1.txt
```



```
introduction — vim part2/file1.txt
1 <<<<< HEAD
2 Staniel has struck first. Even so, Daniel is better.
3 =====
4 Staniel has struck first.
5
6 Also, Staniel is the best.
7 >>>>> 478d09e2fe29c22aed756a78f5e6e647eb48b31c
```

Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel *resolves the conflict however he wants*

```
$ vim part2/file1.txt
```



```
introToGitManningGroup — vim part2/file1.txt
1 Daniel wrote this file in the first place;
2 this argument is ridiculous.
3 |
```

Managing conflicts

Suppose:

- (1) Staniel edits a file and pushes the changes
- (2) Daniel edits *the same file* without pulling
- (3) Daniel pushes the changes to the repository

```
$ git add .  
$ git commit -m "conflict managed"  
$ git push
```



```
[dmsussma@~/repos/introToGitManningGroup$ git push  
Counting objects: 8, done.  
Delta compression using up to 4 threads.  
Compressing objects: 100% (6/6), done.  
Writing objects: 100% (8/8), 731 bytes | 0 bytes/s, done.  
Total 8 (delta 2), reused 0 (delta 0)  
To https://gitlab.com/dmsussman/workingWithGit.git  
478d09e..bcc741f master -> master
```

Managing conflicts

All of the attempted commits, even the one Daniel couldn't push, become part of the repository's history...

```
[dmsussma@~/repos/introToGitManningGroup$git lg2
* bcc741f - Wed, 13 Sep 2017 12:21:41 -0400 (5 minutes ago) (HEAD -> master, origin/master)
| \ conflict managed - Daniel Sussman
| * 478d09e - Wed, 13 Sep 2017 12:09:56 -0400 (17 minutes ago)
| | file1 in part2 edited by Stan - Daniel Sussman
* | 5730c52 - Wed, 13 Sep 2017 12:11:15 -0400 (16 minutes ago)
| / file1 in part2 edited by Dan - Daniel Sussman
* 7889268 - Wed, 13 Sep 2017 12:02:29 -0400 (24 minutes ago)
| file1 in part 2 edited - Daniel Sussman
* a5ce1ff - Wed, 13 Sep 2017 11:31:52 -0400 (55 minutes ago)
| starting part2 - Daniel Sussman
* 82019cb - Wed, 13 Sep 2017 09:29:27 -0400 (3 hours ago)
| got rid of an accidental swap file - Daniel Sussman
```

...so collaborate nicely

