Get To Git

Session 2

Hadeer Ramadan

Agenda

01 Branches

1 Commits

2 Branches

3 Merging

4 Remotes

5 Pull requests

02 Contribution

1 Contribution

2 How to find issues?

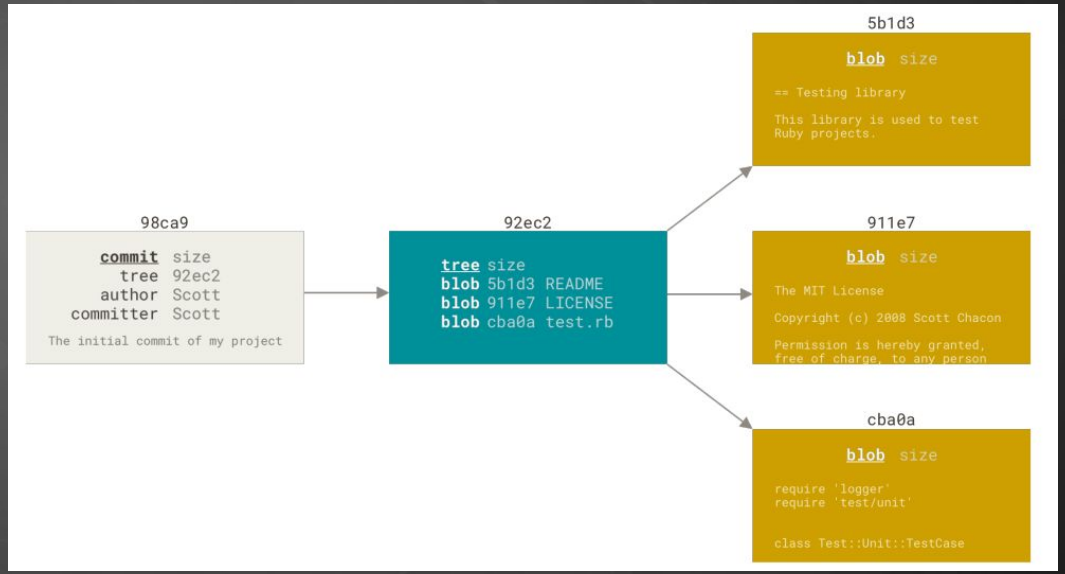
Part One: Branches

Commits

How Git Stores Commits

A commit contains:

- A pointer to commit object
- Information about author and committer
- Pointer to commit parent (if it's not the first commit)

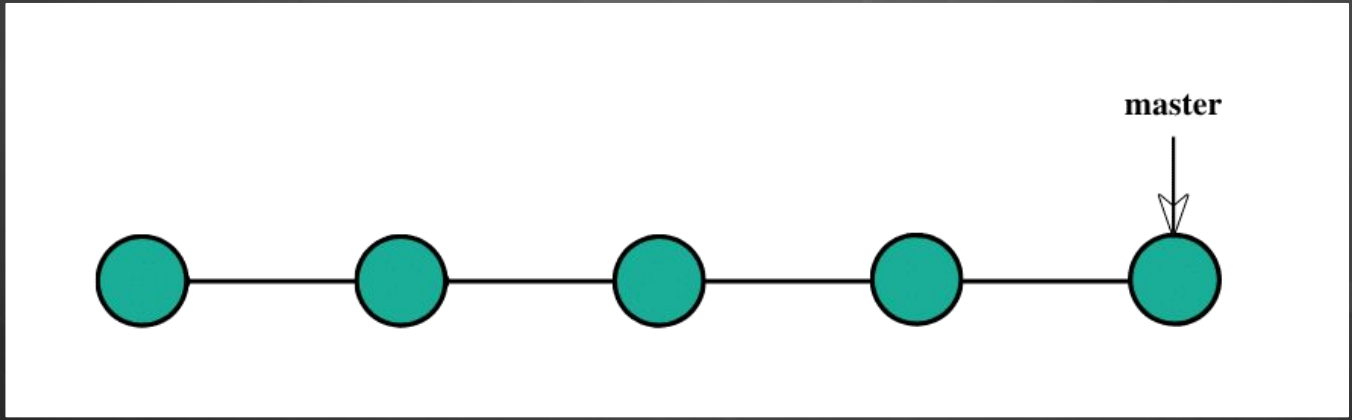


Branches

How Git stores Commit

Master branch:

This is the default branch in Git
Note: GitHub calls the default branch *main*.



Creating New Branch

- When creating new branch, Git creates new pointer that points to start of this branch.
- Git moves this pointer forward every time you commit to the branch.
- To create a new branch:

```
git branch branchName
```

- To create a new branch starting from a specific commit:

```
git branch branchName [startPoint]
```

Switching to Another Branch

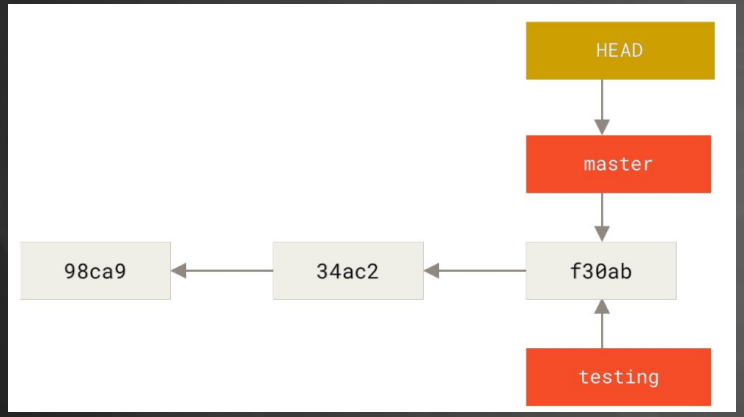
- To switch to another branch:

```
git switch branchName
```

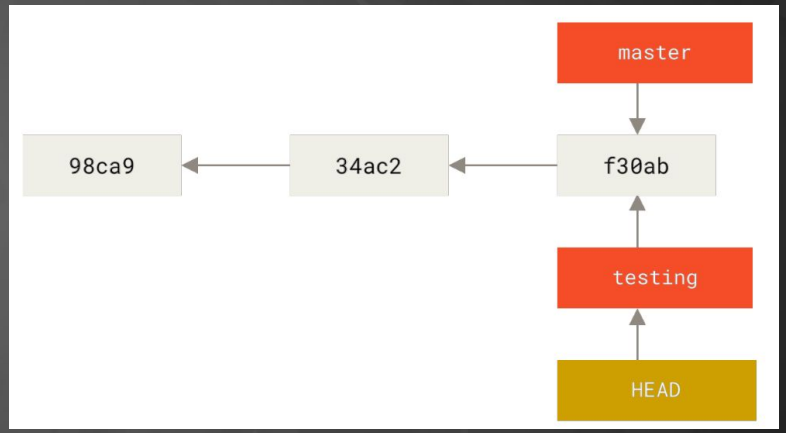
- To create and switch at the same time :

```
git switch -c branchName
```

Switching



git switch
branchName



Git Checkout

- `checkout` is Git's swiss army knife – it has many seemingly unrelated uses.
- In Git version 2, `checkout` functionality was split into two commands: `switch` and `restore`
- Learn more here and here!

Functionality	Using checkout	Using switch/restore
Switch to a branch	`git checkout branchName`	`git switch branchName`
Untrack modified file	`git checkout -- file`	`git restore file`
Creating a branch and switching to it	`git checkout -b branchName [startpoint]`	`git switch -c branchName [startpoint]`

Commit History

- To show commit history for the a given branch:

```
git log branchName
```

- To show all of the branches:

```
git log --all
```

Commit History

- To show all of the branches tree path, add --graph

```
git log --all --graph --oneline
```

Merging

Merging Types

Fast Forward

Three Way Merge

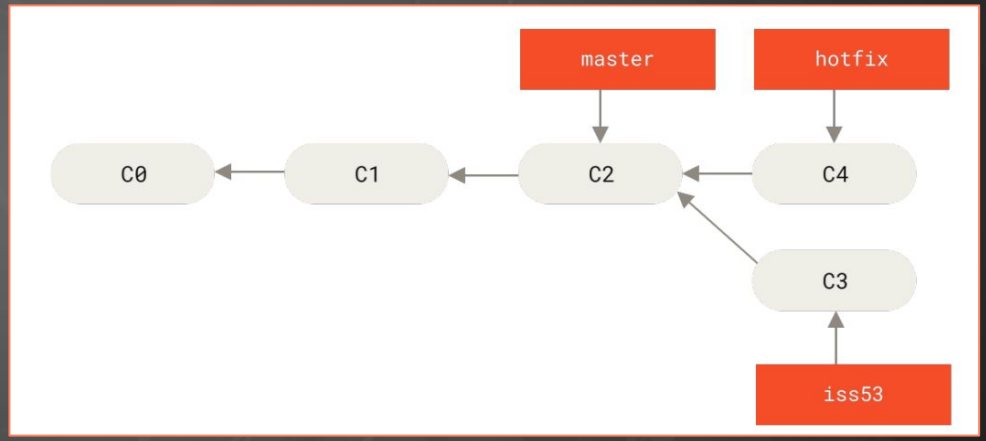
Basic Merging Conflicts

Fast Forward

Fast forward merging is to move the master pointer forward to the last commit in the branch that you merge.

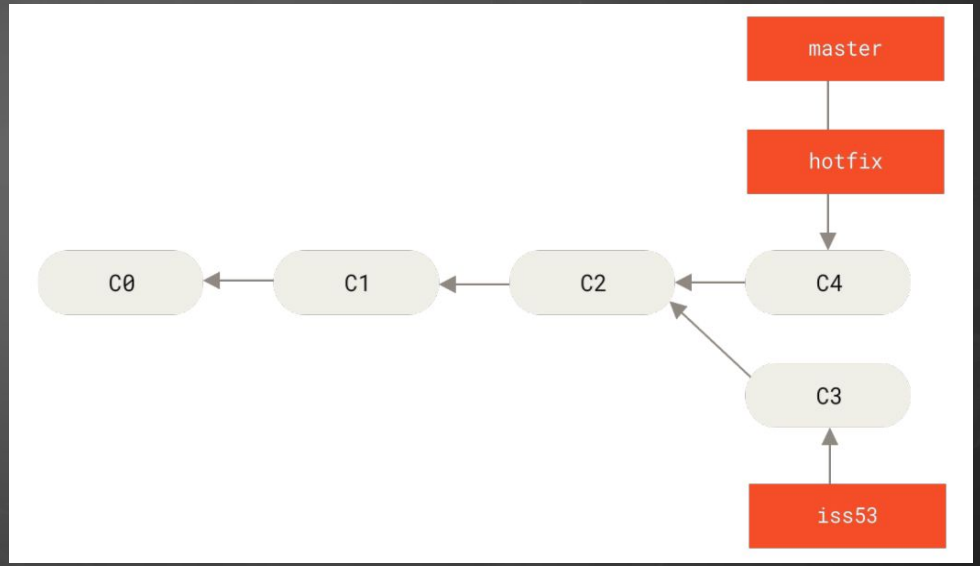
Command :

```
git merge branchName
```



Fast Forward

Now hotfix branch and master point to commit C4



Three Way Merge

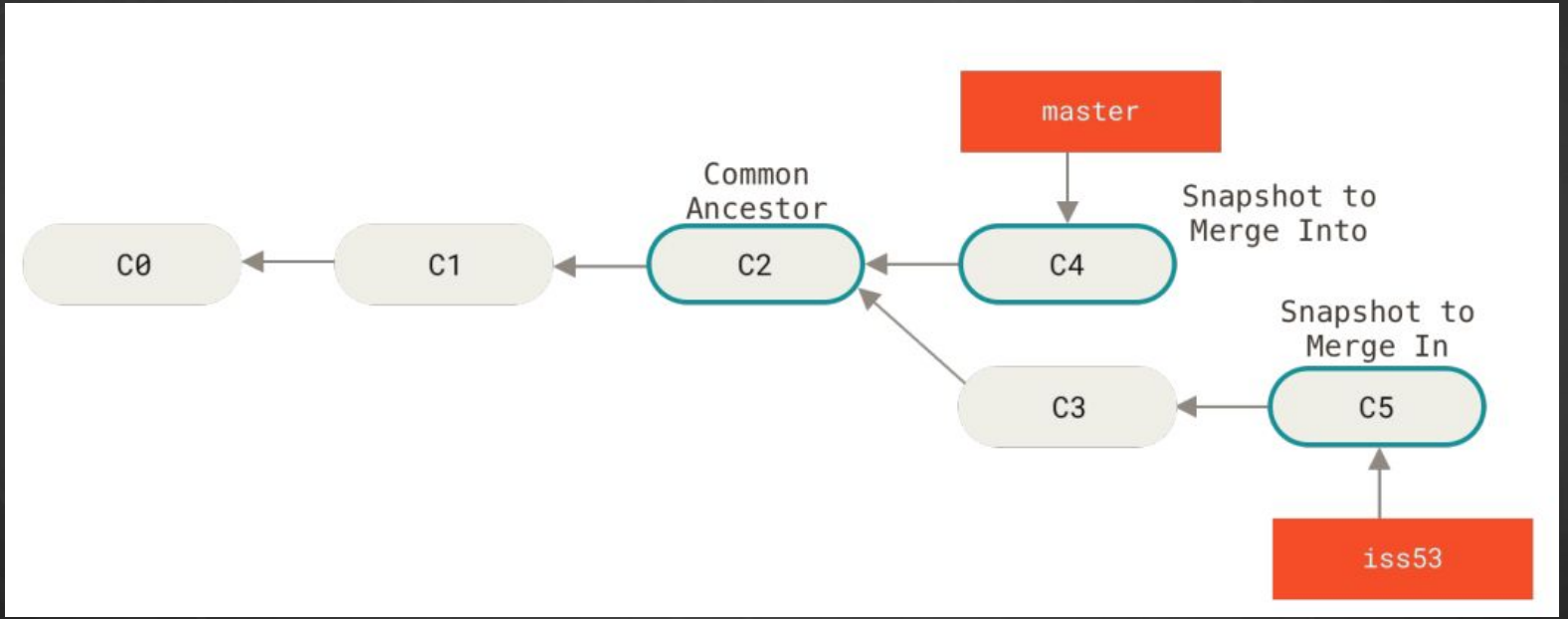
It happens when two branches have diverged and both have their own unique commits since the common ancestor.

Three Way Merge

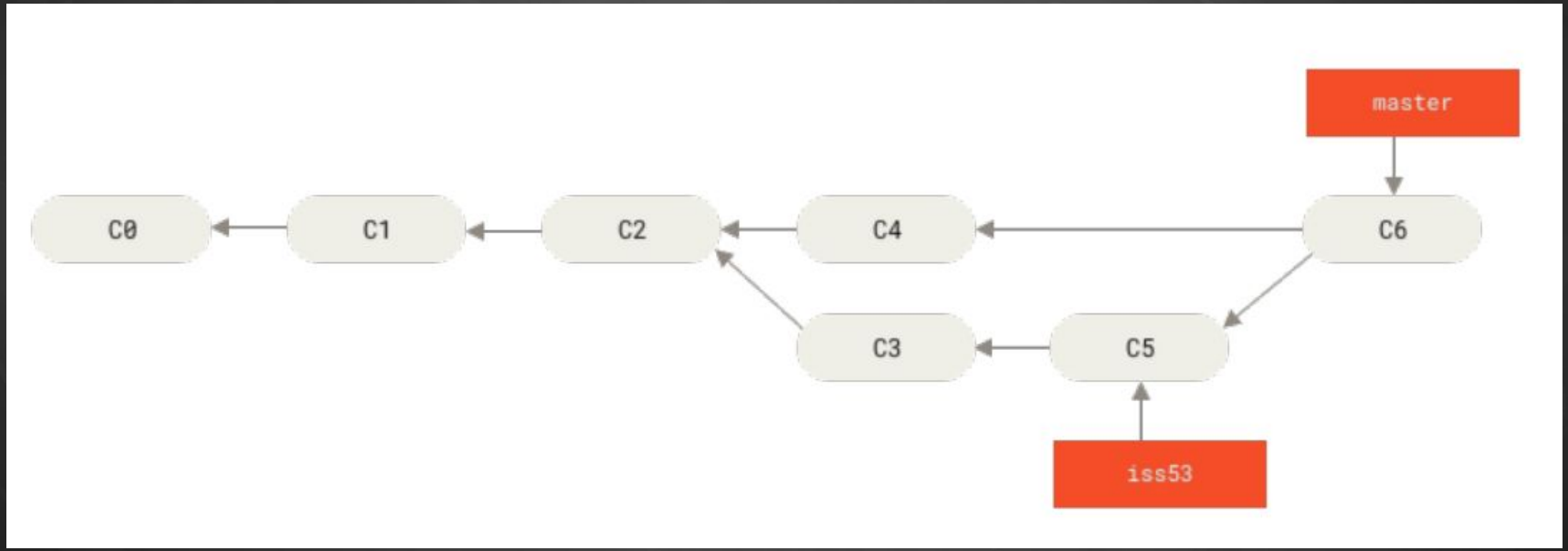
It’s called three ways, as Git does three things

- Finding the common ancestor of the two branches. This ancestor is called “merge base.”
- Finding the last two snapshots.
- Creating new commit that points to the last two snapshots

Three Way Merge



Three Way Merge



Basic Merging Conflict

If you changed the same part of the same file in the two branches you're merging, Git won't be able to merge them cleanly. It will leave this task for you!

How Does Git Mark Conflicts?

Git adds standard conflict resolution markers to the files that have conflicts, so you can open them manually and resolve those conflicts.

```
#include <iostream>
<<<<<<< HEAD
using namespace std;

=======


int add(int a, int b);
>>>>>>> feature
```

How To Solve The Conflict?

- **First,**

  Choose the part that you want to save in your new version or merge between them.

- **Second,**

  Remove conflict resolution markers

- **Finally,**

  Commits your files.

Git Branch <options>:

It can be used to view branches

`-v`

view every commit that branch points to

`-d <branch>`

delete merged branch

`-D <branch>`

delete not merged branch

`--merged`

view branched that have merged with current branch

`--no-merged`

view branched that have not merged with current branch

Hands on

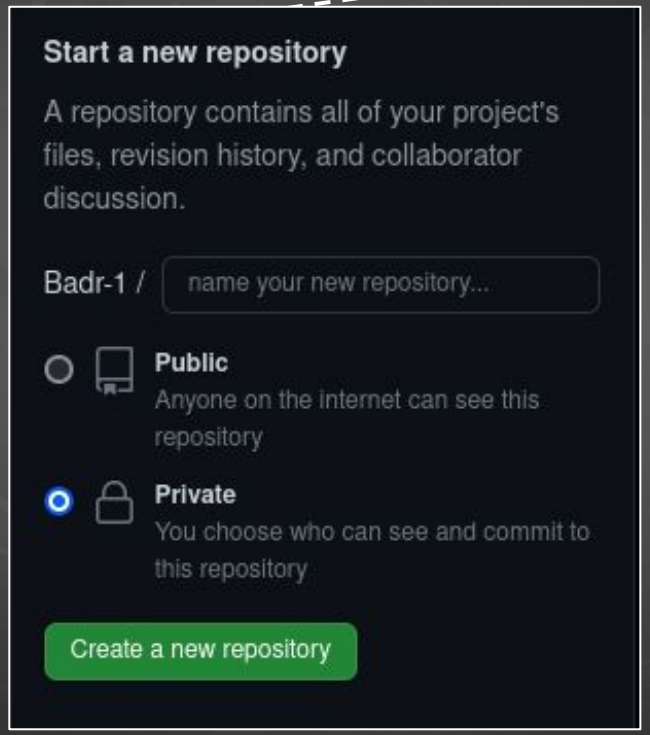
1. Create a "project" directory and navigate inside it.
2. Add a README.txt file in it and write your name in it
3. Initalize a git repository.
4. Create two branches, iss1 and feat.
5. In branch feat, create “CONTRIBUTING.md” file
6. In branch iss1, modify your name in to be in capital and commit changes
7. In master branch, replace first line with “Project Features.” and commit changes
8. Merge feat in master
9. Merge iss1 in master (We want to keep all changes)

Break 🎉

15 minutes

Remotes

Creating New Repository on GitHub

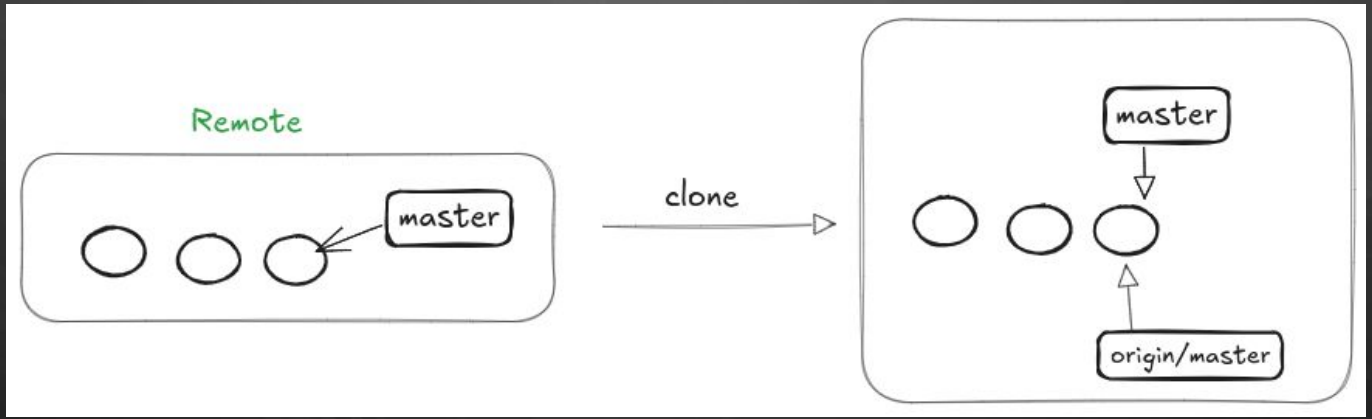


Creating New Repository on GitHub

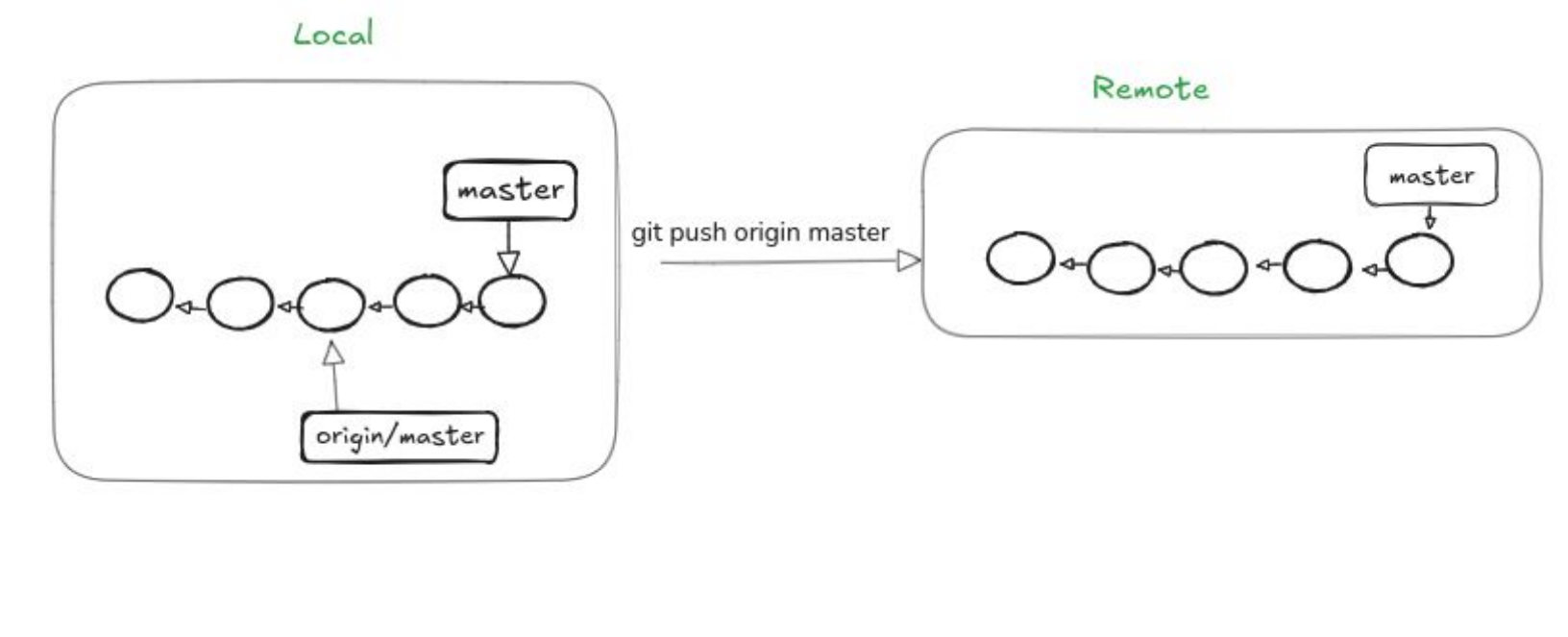
```
echo "# demo" >> README.md
git init
git add README.md
git commit -m "First commit"
git branch -M main
git remote add origin URL
git push -u origin main
```

Cloning Remote Repo

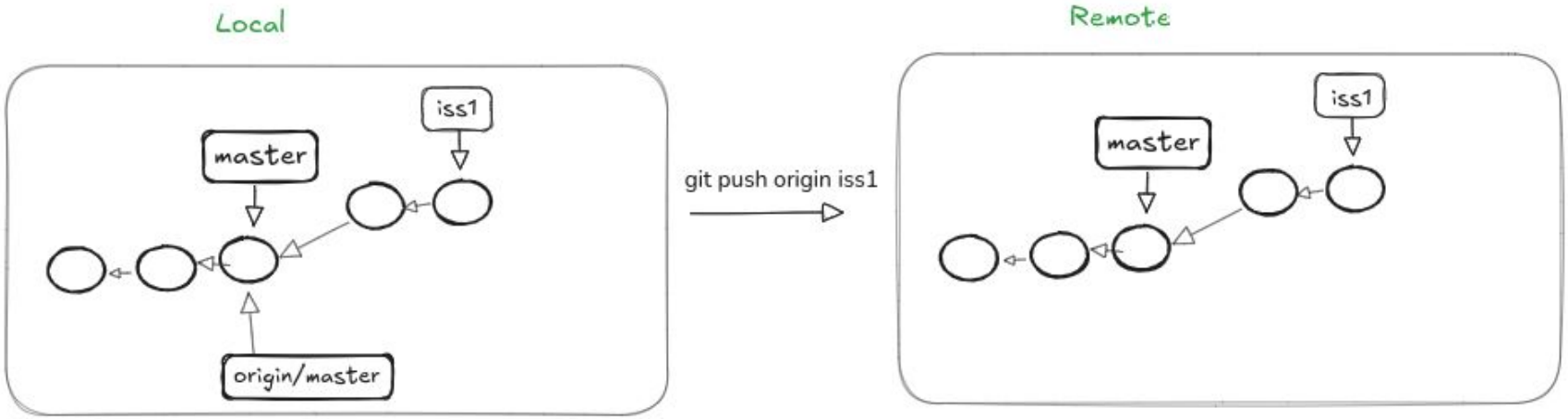
- `git clone url <dir-name>`
- It creates a local repo from master/main branch of remote repo
- To see remotes run : **git remote <options>**



git push <remote-name> <branch-name>



git push <remote-name> <branch-name>



Get Updates

`git fetch <remote> <branch>`

- it get updated to your repo but it doesn't merge in master/main branch
- to merge changes use : **`git merge`**
- it's good to review some updates before merging it

`git pull <remote> <branch>`

- it fetch and merge at the same time

Tracking remote branch

```
git checkout --track remote/branch
```

1. Creates new local branch
2. Link it with the remote branch

- To see what tracking branches you have set up:
  `git branch -vv`

RECAP

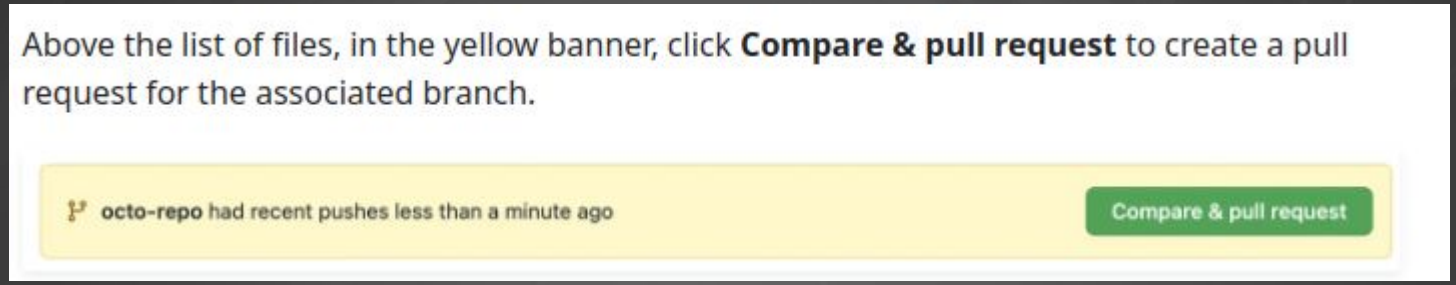
- Branches
- Merging
- Remotes
- Fetch / Pull
- Push

Pull Request

Pull Request

- A pull request is a proposal to merge a set of changes from one branch into another.
- In a pull request, collaborators can review and discuss the proposed set of changes before they integrate them into the main codebase.
- Pull requests display the differences, or diffs, between the content in the source branch and the content in the target branch.

Creating a Pull Request



- How to create a pull request

Updating a Pull Request

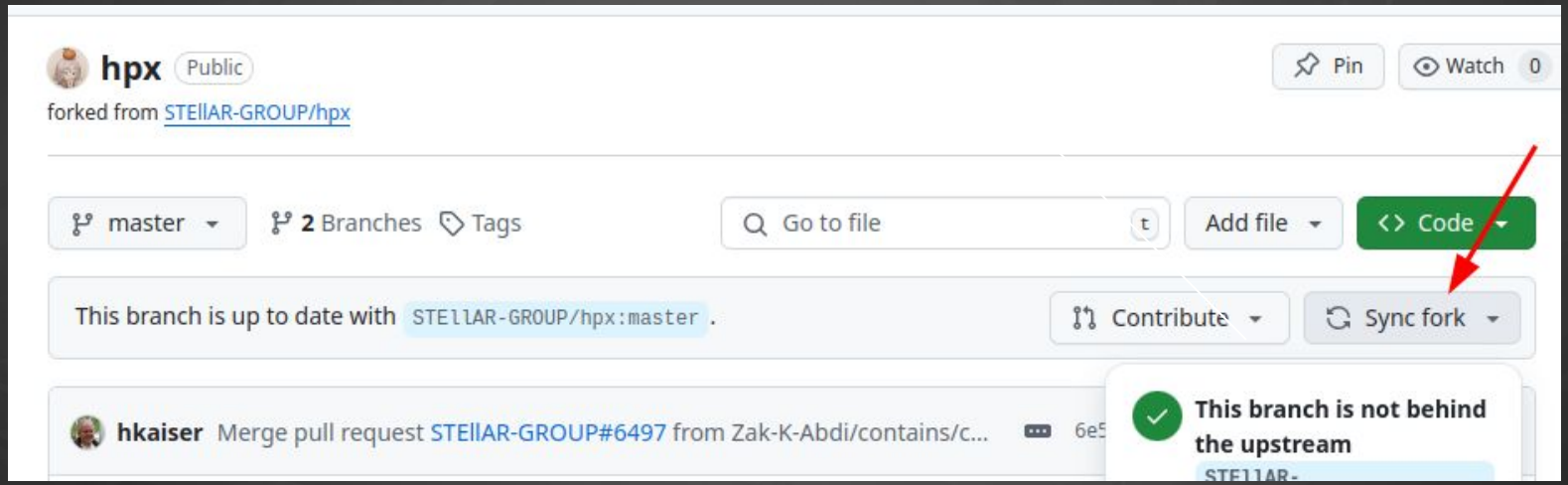
- When creating a Pull Request in branch, there is link that created between this PR and your branch

- Therefore, all you have to do is make changes in the branch and push it there, and it will synchronize with Pr.

Fork and Pull Request

- When user doesn’t have access to remote repo of the project
- It is primarily similar to open source projects.
- It also helps to maintain the cleanliness and safety of the main project.
- Use someone else's project as a starting point for your own idea.
- So now you make a pull request from branch in forked repo to branch in original repo
- Fork a repository

Updating Forked Repository



Part Two: Contribution

Last Session

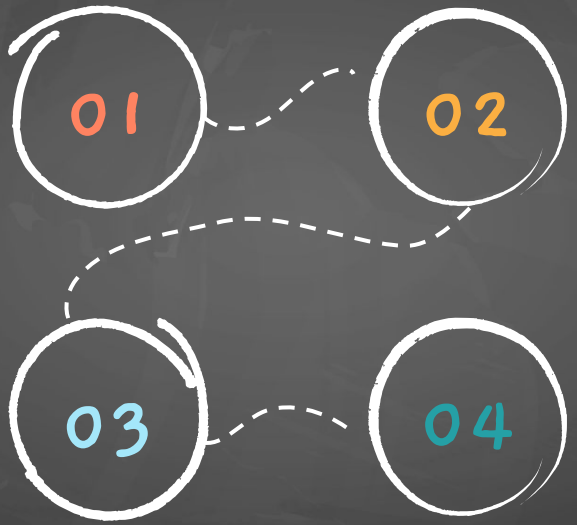
What Is Open Source?

Why Contribute?

Hacktoberfest

Steps To Contribute

Read documents, contributing guidelines and README files



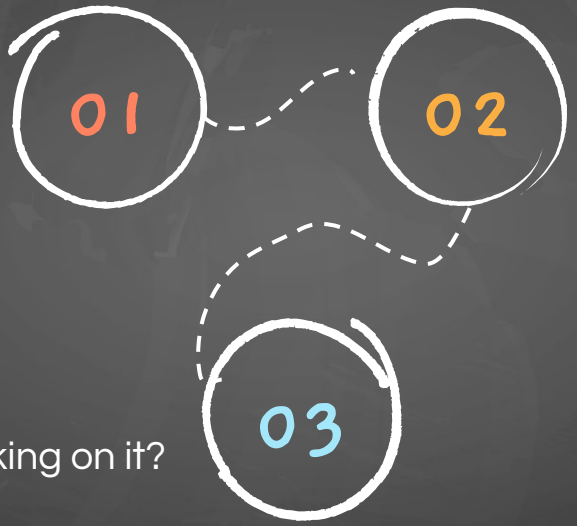
Fork and make a change

Finding an issue or open one

Create your pull request

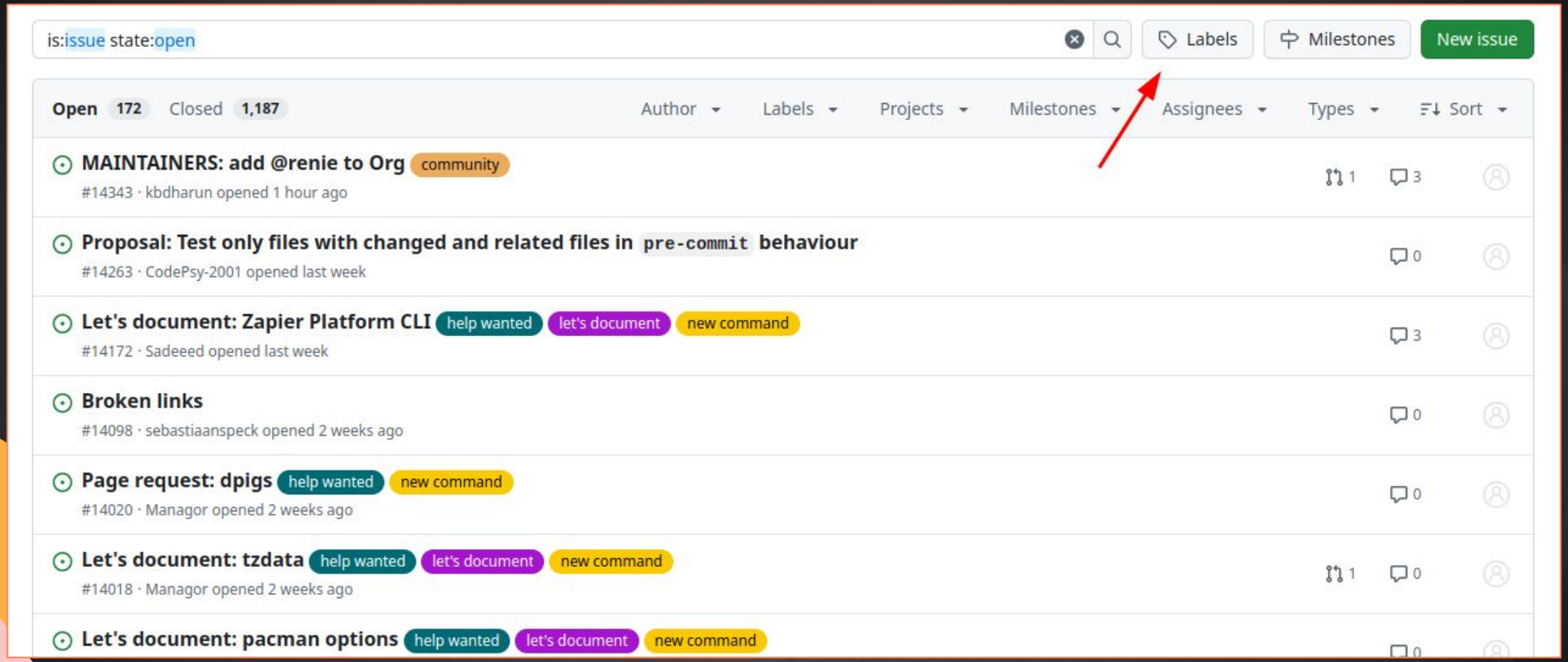
Choosing An Issue

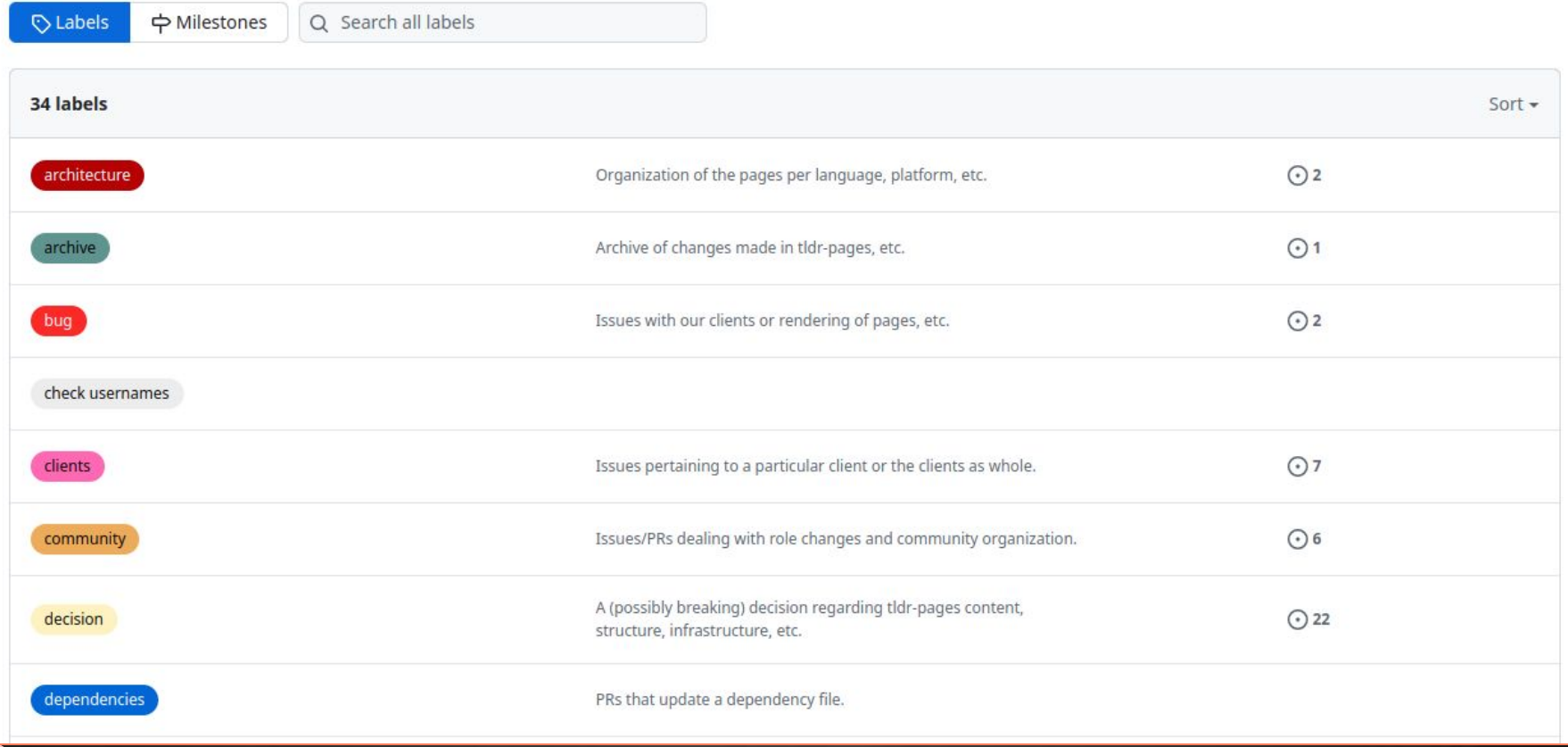
What is required?



Is it available to work on?

Is there someone working on it?


is:issue state:open
Labels
Milestones
New issue
Open 172
Closed 1,187
Author
Labels
Projects
Milestones
Assignees
Types
Sort
MAINTAINERS: add @renie to Org
community
#14343 · kbdharun opened 1 hour ago
1
3
Proposal: Test only files with changed and related files in pre-commit behaviour
#14263 · CodePsy-2001 opened last week
0
Let's document: Zapier Platform CLI
help wanted
let's document
new command
#14172 · Sadeeed opened last week
3
Broken links
#14098 · sebastiaanspeck opened 2 weeks ago
0
Page request: dpigs
help wanted
new command
#14020 · Managor opened 2 weeks ago
0
Let's document: tzdata
help wanted
let's document
new command
#14018 · Managor opened 2 weeks ago
1
0
Let's document: pacman options
help wanted
let's document
new command
0


Labels
Milestones
Search all labels
34 labels
Sort
architecture
Organization of the pages per language, platform, etc.
2
archive
Archive of changes made in tldr-pages, etc.
1
bug
Issues with our clients or rendering of pages, etc.
2
check usernames
clients
Issues pertaining to a particular client or the clients as whole.
7
community
Issues/PRs dealing with role changes and community organization.
6
decision
A (possibly breaking) decision regarding tldr-pages content, structure, infrastructure, etc.
22
dependencies
PRs that update a dependency file.

Where To Find Issues?

- Finder
- GitHub-Hacktoberfest
- Contribhub-Hacktoberfest
- GoodFistIssue

Finally, you can search in github with your favourite topics and languages!

Let’s create our first contribution 🎉

Resources

- ProGit
- Get To Git Playlist
- Livestream about open source by Abdelrahaman Awad
- More about GSoC (Google Summer of Code)

It is easy to shoot your foot off with Git, but also easy to revert to a previous foot and merge it with your current leg."

— Jack William Bell

Thank You!