



# Lecture 19: Git!

CSE 374: Intermediate  
Programming Concepts and  
Tools

# QUICK RECAP

# Some “git” commands

- **git init**
  - Create a new empty git repo or convert an existing folder to a git repo
- **git add**
  - Preparing edited files to be saved (committed) to a repo
- **git commit**
  - Records (saves) changes to a repo
  - Accompanied by a short descriptive message
- **git push**
  - Update the remote copy of the repo with the local changes and commits

```
MINGW64:/c/Users/kusha/ta/git x + v
kusha@LAPTOP-UA25NDJJ MINGW64 ~/ta/git_practice (main)
$ touch .gitignore

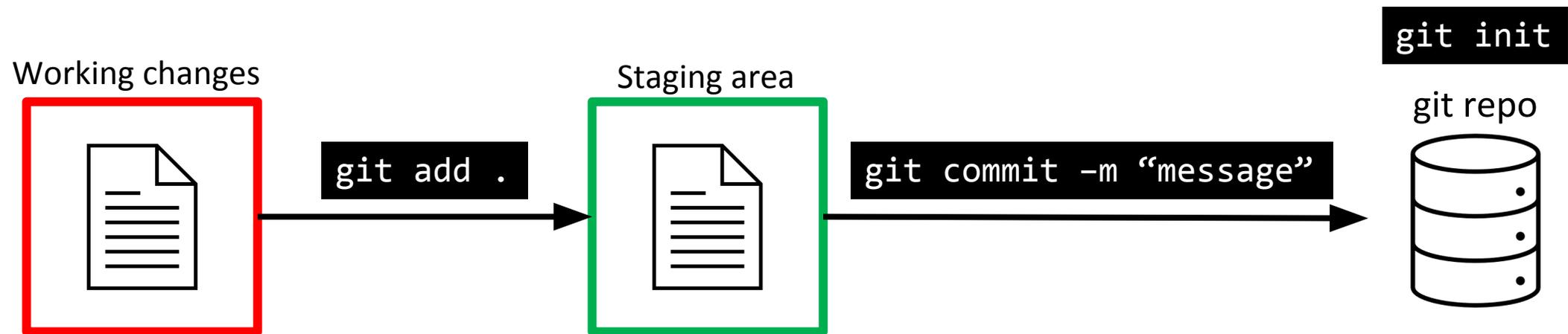
kusha@LAPTOP-UA25NDJJ MINGW64 ~/ta/git_practice (main)
$ git add .gitignore

kusha@LAPTOP-UA25NDJJ MINGW64 ~/ta/git_practice (main)
$ git commit -m "Add gitignore to repo"
[main 46480c4] Add gitignore to repo
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 .gitignore

kusha@LAPTOP-UA25NDJJ MINGW64 ~/ta/git_practice (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 346 bytes | 346.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/kushalj/h/git_practice.git
1d9015d..46480c4 main -> main

kusha@LAPTOP-UA25NDJJ MINGW64 ~/ta/git_practice (main)
$ |
```

# Staging and committing overview



# Inspecting a repository

## ■ git status

- Lists the files which you have changed but not yet committed
  - Working directory
  - Staging area
- Indicates how many commits have made but not yet pushed

## ■ git log

- Shows the commit history
- `git log -graph --oneline`
  - Shows branch info as a graph

```
MINGW64:/c/Users/kusha/Docu x + v
kusha@LAPTOP-UA25NDJJ MINGW64 ~/Documents/GitHub/git_practice (main)
$ git status
On branch main
Your branch is behind 'origin/main' by 1 commit, and can be fast-forwarded.
(use "git pull" to update your local branch)

nothing to commit, working tree clean

kusha@LAPTOP-UA25NDJJ MINGW64 ~/Documents/GitHub/git_practice (main)
$ git log
commit 1d9015d8c3146f3377df31ab2e9d0fa7d8b1f787 (HEAD -> main)
Author: Kushal Jhunhunwalla <22863544+kushaljh@users.noreply.github.com>
Date: Sun Nov 8 01:35:30 2020 -0800

    Add Author to README

commit e93bc2ce898d7420b1381d95f62d46860937da7a
Author: Kushal Jhunhunwalla <22863544+kushaljh@users.noreply.github.com>
Date: Sun Nov 8 01:33:55 2020 -0800

    Initial commit

kusha@LAPTOP-UA25NDJJ MINGW64 ~/Documents/GitHub/git_practice (main)
$ git log -1
commit 1d9015d8c3146f3377df31ab2e9d0fa7d8b1f787 (HEAD -> main)
Author: Kushal Jhunhunwalla <22863544+kushaljh@users.noreply.github.com>
Date: Sun Nov 8 01:35:30 2020 -0800

    Add Author to README
```



# Working with remote

# git commands for interaction with remote

## ▪ **git clone**

- Cloning is the process of creating a working copy of the remote or local repository by passing the following command.
- **git clone username@git\_server\_hostname:/path\_of\_repository**

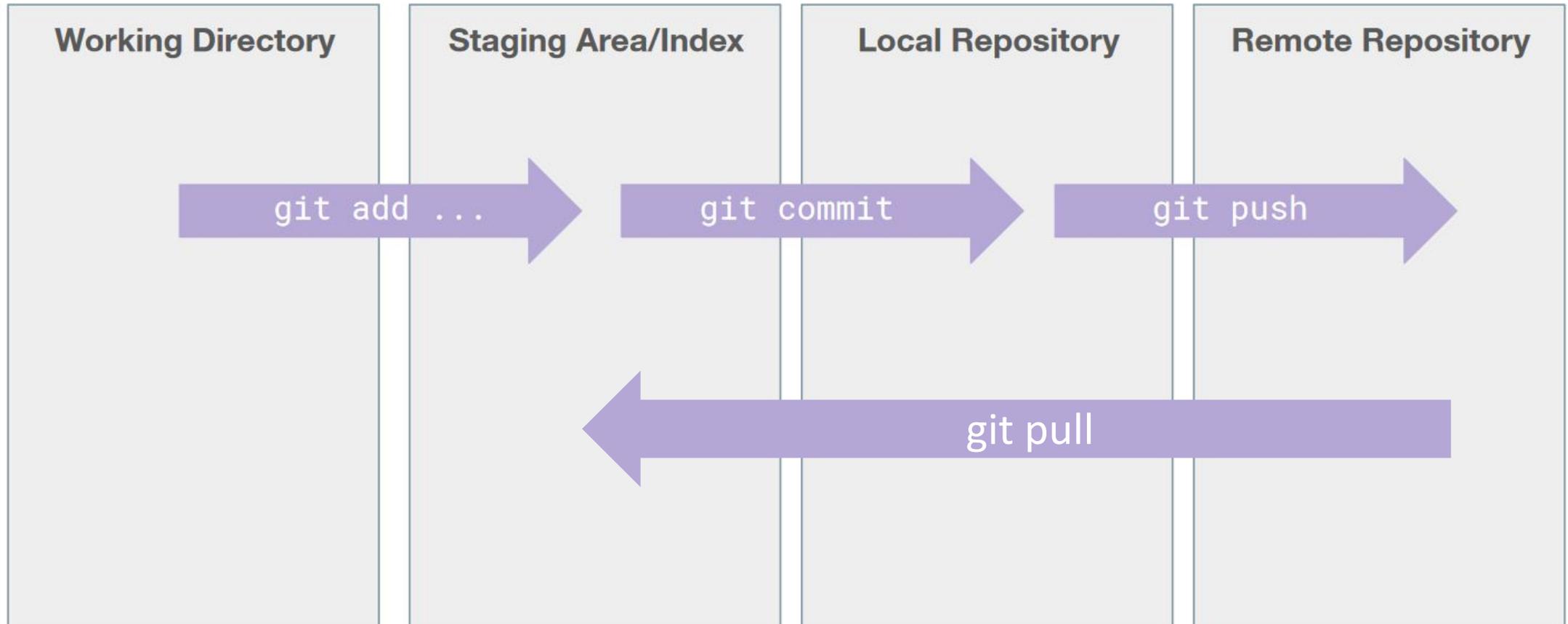
## ▪ **git pull**

- If we have already cloned the repository and need to update local (only code) respect to the remote server
- **git pull origin main**

## ▪ **git fetch**

- Fetching is the process of updating (only git information) the local git structure and information from remote repository
- **git fetch**

# Updating changes using pull



**NOTE:** There are way more git commands than what is listed here - this is a simplified model to get us started.

# What's next!

- Branching, checkout
- Merging, Merge (Pull) Requests
- Conflicts
- Interacting with a Git Server (GitHub / GitLab)

# Skipping files using .gitignore

- As a recap, generally we should not have these files in a git repo:
  - Object files (i.e. .class files, .o files) and executables
  - Huge media files (e.g. videos)
  - Credentials and system files (e.g. .DS\_Store in Mac)
- Its tedious to mention all the files you want to add into a repo every time you add your changes (and ensure you skip the others)
- To skip above files you can create a file in the root of the git repo folder called `.gitignore`
- This file should contain patterns in file names you would like to skip being added
- <https://github.com/github/gitignore>

```
1 # OS generated files
2 **/.DS_Store
3
4 # Compiled source
5 *.com
6 *.class
7 *.dll
8 *.exe
9 *.o
10 *.so
11
12 # Logs and databases
13 *.log
14 *.sql
15 *.sqlite
16
```



Working with a Git server

**GitHub**

# GitHub profile

kushalj (Kushal Jhunjunwalla) x +

https://github.com/kushalj

Why GitHub? Team Enterprise Explore Marketplace Pricing Search Sign in Sign up

Overview Repositories 23 Projects Packages

**Kushal Jhunjunwalla**  
kushalj

Follow ...

1 follower · 0 following · 0 stars

**Highlights**  
\* Arctic Code Vault Contributor

**Pinned**

- BlockChain-In-HealthCare**  
Python 4 stars 1 fork
- CameraApp**  
Java
- Project\_News\_Search**  
INFO 201 Project  
R

112 contributions in the last year

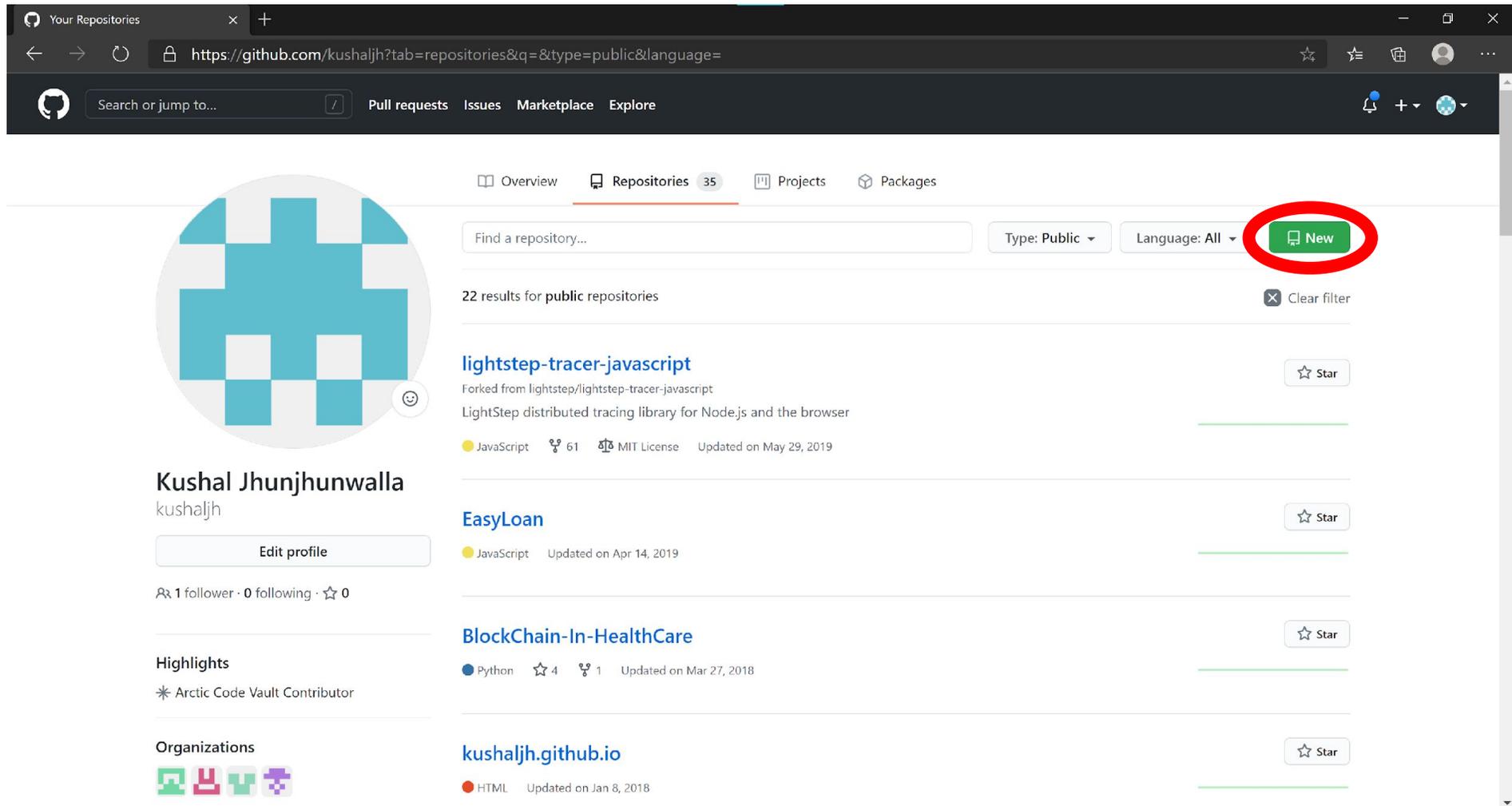
	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
Mon													
Wed													
Fri													

Learn how we count contributions. Less More

Contribution activity 2020

October November 2020

# Creating a new repo on GitHub



The screenshot shows a web browser window displaying the GitHub profile page for user 'kushalj'. The browser's address bar shows the URL: `https://github.com/kushalj?tab=repositories&q=&type=public&language=`. The page header includes navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The main content area is divided into two columns. The left column features the user's profile picture (a blue and white grid), the name 'Kushal Jhunjunwalla', the username 'kushalj', an 'Edit profile' button, and statistics: '1 follower · 0 following · ☆ 0'. Below this are sections for 'Highlights' (listing 'Arctic Code Vault Contributor') and 'Organizations' (with icons for various organizations). The right column shows the 'Repositories' tab with 35 items. A search bar is present with the text 'Find a repository...'. Filter buttons for 'Type: Public' and 'Language: All' are visible. A green 'New' button with a repository icon is circled in red. Below the filters, there are 22 search results for public repositories. The first result is 'lightstep-tracer-javascript', which is a forked repository from 'lightstep/lightstep-tracer-javascript'. It is described as a 'LightStep distributed tracing library for Node.js and the browser' and is written in JavaScript. Other results include 'EasyLoan' (JavaScript), 'BlockChain-In-HealthCare' (Python), and 'kushalj.github.io' (HTML). Each repository listing includes a 'Star' button.

# Creating a new repo on GitHub

Create a New Repository

https://github.com/new

Search or jump to... Pull requests Issues Marketplace Explore

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \* Repository name \*

kushaljh / cse374\_hw3 ✓

Great repository names are short and memorable. Need inspiration? How about [redesigned-octo-spork?](#)

Description (optional)

HW3 Repo for CSE 374

Public  
Anyone on the internet can see this repository. You choose who can commit.

Private  
You choose who can see and commit to this repository.

Initialize this repository with:  
Skip this step if you're importing an existing repository.

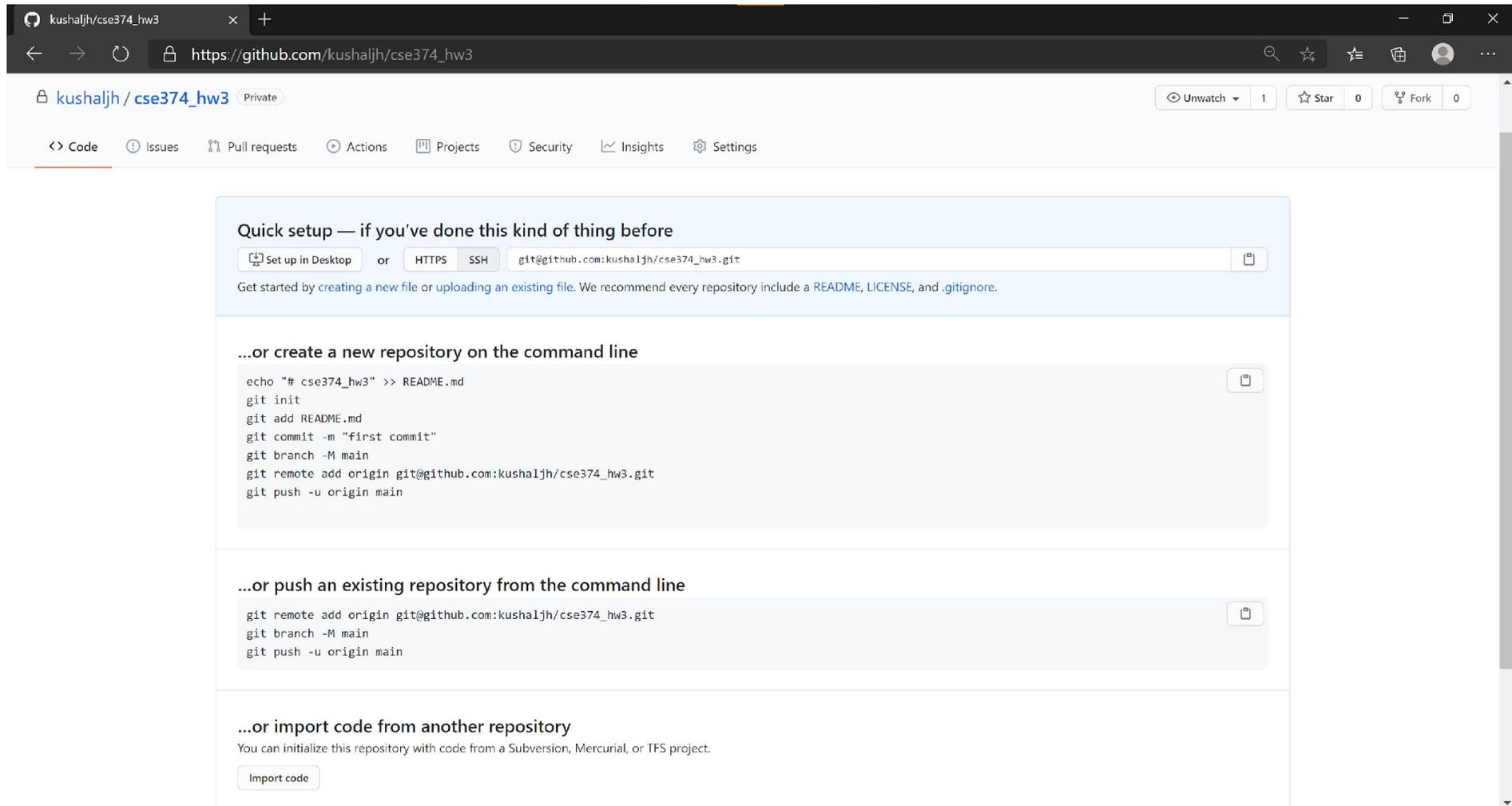
Add a README file  
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore  
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license  
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

# Creating a new repo on GitHub



The screenshot shows the GitHub interface for a new repository named 'kushaljh/cse374\_hw3'. The page is titled 'Quick setup — if you've done this kind of thing before' and provides several options for creating a new repository:

- Quick setup — if you've done this kind of thing before:** This section offers a 'Set up in Desktop' button, an 'or' separator, and 'HTTPS' and 'SSH' options. The SSH URL is `git@github.com:kushaljh/cse374_hw3.git`. Below this, it says 'Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.'
- ...or create a new repository on the command line:** This section provides a list of terminal commands to create a new repository and push it to GitHub:

```
echo "# cse374_hw3" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:kushaljh/cse374_hw3.git
git push -u origin main
```
- ...or push an existing repository from the command line:** This section provides a list of terminal commands to push an existing repository to GitHub:

```
git remote add origin git@github.com:kushaljh/cse374_hw3.git
git branch -M main
git push -u origin main
```
- ...or import code from another repository:** This section states 'You can initialize this repository with code from a Subversion, Mercurial, or TFS project.' and includes an 'Import code' button.

# Adding local files to the GitHub server

```
Ubuntu
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git status
fatal: not a git repository (or any parent up to mount point /mnt)
Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ ls
clint.py empty.txt man_glob.txt myfgrep.c rick.txt
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git init
Initialized empty Git repository in /mnt/c/Users/kusha/ta/cse374_hw3/.git/
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git add .
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git commit -m "Add starter files"
[master (root-commit) 4b018c4] Add starter files
 5 files changed, 3353 insertions(+)
 create mode 100644 clint.py
 create mode 100644 empty.txt
 create mode 100644 man_glob.txt
 create mode 100644 myfgrep.c
 create mode 100644 rick.txt
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git remote add origin https://github.com/kushaljh/cse374_hw3.git
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git push -u origin master
Username for 'https://github.com': kushaljh
Password for 'https://kushaljh@github.com':
Counting objects: 7, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 40.68 KiB | 2.91 MiB/s, done.
Total 7 (delta 0), reused 0 (delta 0)
To https://github.com/kushaljh/cse374_hw3.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ |
```

# Checking the changes on GitHub

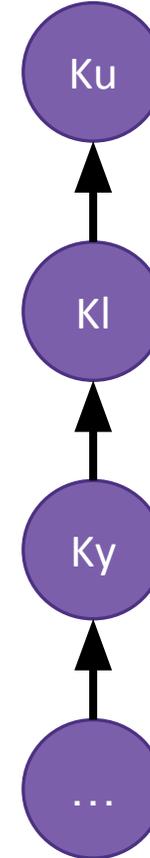
The screenshot shows a web browser displaying the GitHub repository page for 'kushaljh/cse374\_hw3'. The browser's address bar shows the URL 'https://github.com/kushaljh/cse374\_hw3'. The repository page includes a navigation bar with links for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Security', 'Insights', and 'Settings'. The repository name 'kushaljh / cse374\_hw3' is shown as 'Private'. On the right side, there are buttons for 'Unwatch', 'Star' (1), and 'Fork' (0). The main content area shows the 'master' branch with 1 branch and 0 tags. A commit by 'kushaljh' is listed with the message 'Add starter files', commit hash '4b018c4', and '5 minutes ago' with '1 commit'. Below the commit, a table lists files: 'clint.py', 'empty.txt', 'man\_glob.txt', 'myfgrep.c', and 'rick.txt', each with a file icon and the text 'Add starter files' and '5 minutes ago'. At the bottom of the commit list, there is a blue box with the text 'Add a README with an overview of your project.' and a green 'Add a README' button. On the right side, there are sections for 'About' (HW3 Repo for CSE 374), 'Releases' (No releases published), 'Packages' (No packages published), and 'Languages' (Python 97.9%, C 2.1%). The footer of the page includes the GitHub logo, copyright information '© 2020 GitHub, Inc.', and various links like 'Terms', 'Privacy', 'Security', 'Status', 'Help', 'Contact GitHub', 'Pricing', 'API', 'Training', 'Blog', and 'About'.



# Branching

# Collaboration – the ideal case

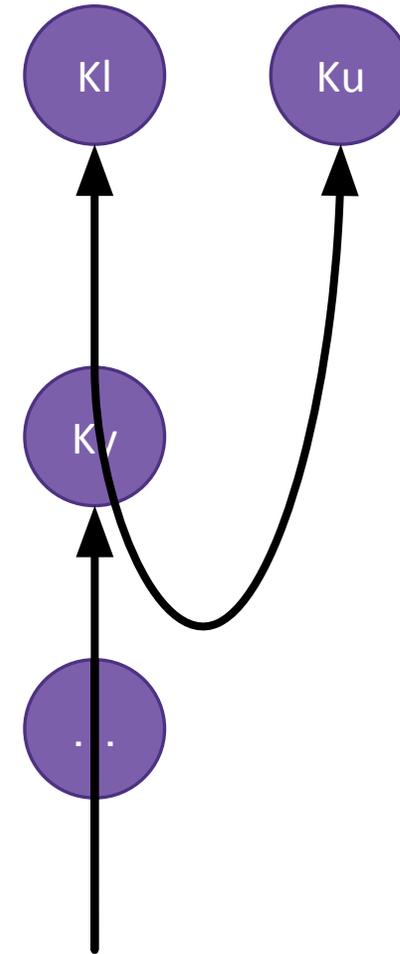
- 3 Collaborators
- Kasey makes a change and updates the repo (git push)
- Kalyani gets Kasey's changes (git pull), edits some files and updates the repo (git push)
- Kushal gets the latest changes that include both Kasey and Kalyani's changes (git pull)
- This goes on...



But ...

# Collaboration – the reality

- 3 Collaborators
- Kasey makes a change and updates the repo
- Kalyani and Kushal get Kasey's changes, edit some files and Kalyani updates the repo while Kushal is still working on some edits
- And here you see different people having different version histories



# Error!!

```
kushaljh@LAPTOP-UA25NDJJ:/mnt/d/kusha/OneDrive - UW/Documents/GitHub/git_practice$ git push
Username for 'https://github.com': kushaljh
Password for 'https://kushaljh@github.com':
To https://github.com/kushaljh/git_practice.git
 ! [rejected]          main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/kushaljh/git_practice.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
kushaljh@LAPTOP-UA25NDJJ:/mnt/d/kusha/OneDrive - UW/Documents/GitHub/git_practice$ |
```

# Error!!

```
kushaljh@LAPTOP-UA25NDJJ:/mnt/d/kusha/OneDrive - UW/Documents/GitHub/git_practice$ git push
Username for 'https://github.com': kushaljh
Password for 'https://kushaljh@github.com':
To https://github.com/kushaljh/git_practice.git
! [rejected]          main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/kushaljh/git_practice.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. Integrate the remote changes (e.g.
hint: 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
kushaljh@LAPTOP-UA25NDJJ:/mnt/d/kusha/OneDrive - UW/Documents/GitHub/git_practice$ |
```

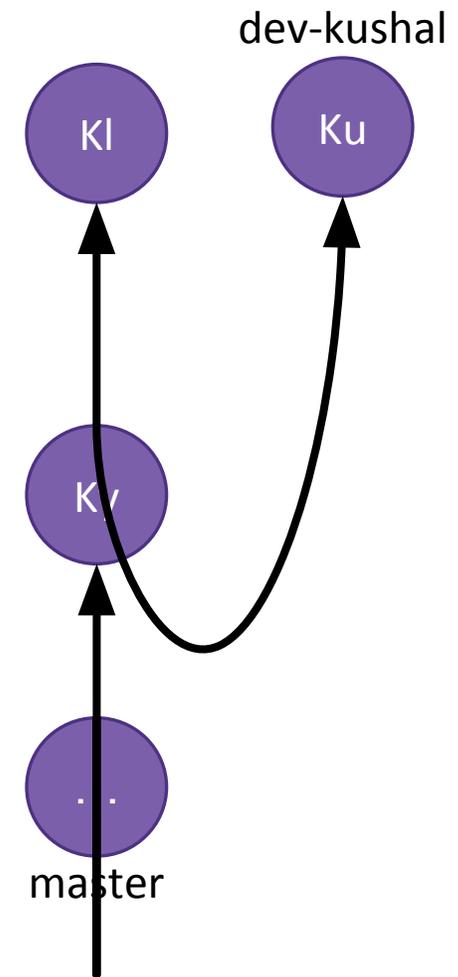


**ONE DOES NOT SIMPLY**

**PUSH TO MASTER**

# Git Branches

- The scenario mentioned in the previous slides would be tough to solve if we used a single history across all collaborators.
- Instead you can start your own history at any point by “branching” out of the main history for a repo
- The master / main branch which is central history and hence the source of truth for the whole project
- We create new version histories based on the main branch and give each of these a branch name

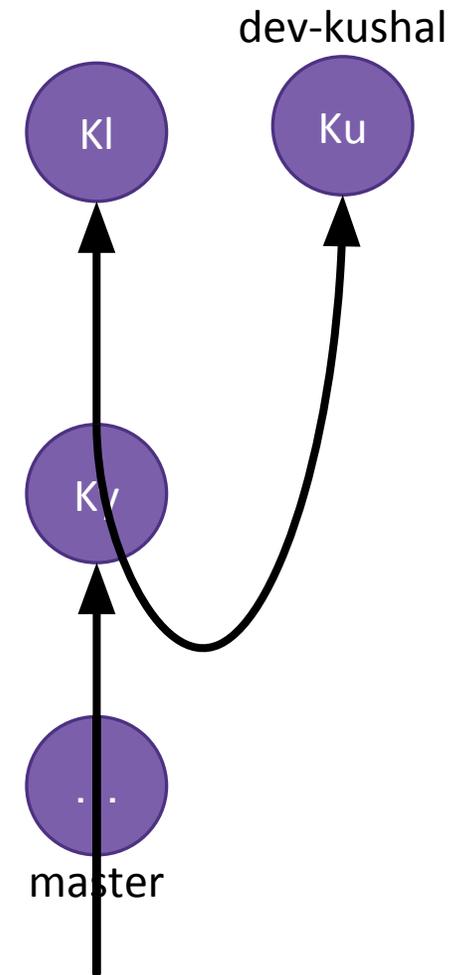


# Git Branches

- Kushal's improved workflow using branches!
- Kushal creates a new branch called `dev-kushal` when she starts working
  - `git branch dev-kushal`
  - `git checkout dev-kushal`

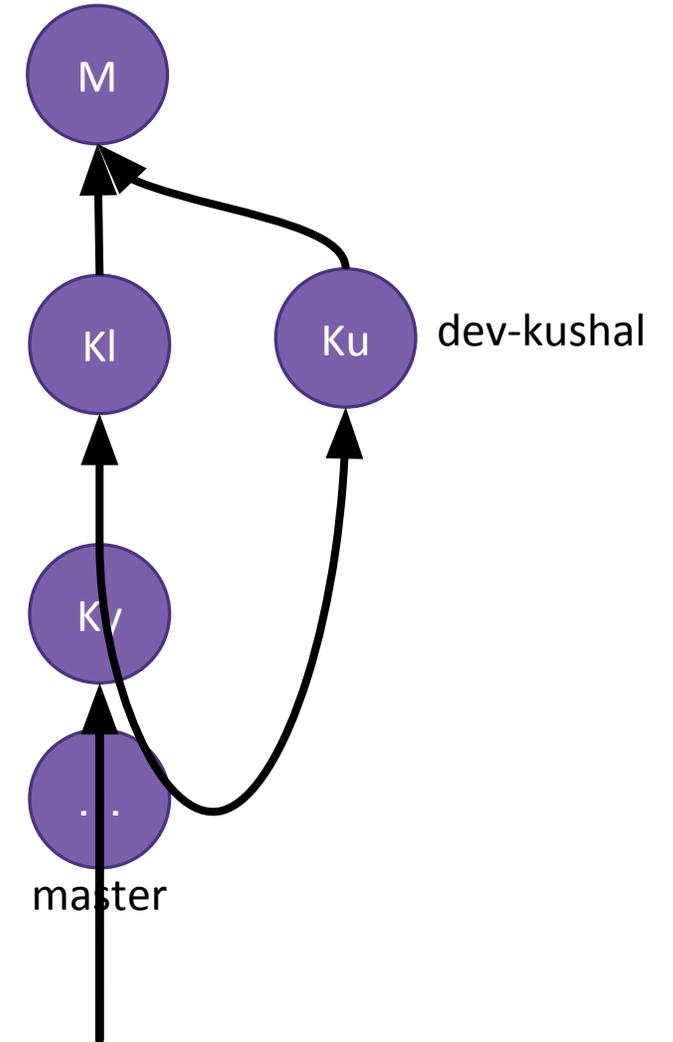
**OR**

  - `git checkout -b dev-kushal`
- Now she makes commits on this branch until she is ready to update master



# Git Merge

- After Kushal is done with her work, she would like the changes from her branch `dev-kushal` be reflected in the `master` branch
- Steps:
  - `git checkout master`
  - `git merge dev-Kushal`
- If there are no changes that are made in the same lines by Kalyani and Kushal there are no conflicting differences, and the merge is good to go
- If not, **WE HAVE A MERGE CONFLICT!**
- Note: this is not the only workflow used for merging



PULL before you start working!!











# Checking the status

```
Ubuntu × + ▾ — □ ×
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git status
On branch master
Your branch is up to date with 'origin/master'.

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 added:   file.txt

no changes added to commit (use "git add" and/or "git commit -a")
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ |
```

# Adding resolved file and saving

```
Ubuntu [x] [+] [v] [ ] [x]
(use "git merge --abort" to abort the merge)

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

    both added:    file.txt

no changes added to commit (use "git add" and/or "git commit -a")
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git add file.txt
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git status
On branch master
Your branch is up to date with 'origin/master'.

All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:

    modified:   file.txt

kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ |
```

# Conflict resolved!!

```
Ubuntu [x] [+] [v] [ ] [x]
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git add file.txt
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git status
On branch master
Your branch is up to date with 'origin/master'.

All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:

  modified:   file.txt

kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git commit -m "Complete merge"
[master ad8ee9b] Complete merge
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ git status
On branch master
Your branch is ahead of 'origin/master' by 2 commits.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
kushaljh@LAPTOP-UA25NDJJ:/mnt/c/Users/kusha/ta/cse374_hw3$ |
```

# Merge (Pull) Requests

- This was nice to know but we generally do not merge branches into master locally
- We use a Git server – GitHub / GitLab – to create a request to merge a feature branch into master
- **We must ensure that everyone on the project agrees to what is present in the master branch as this is our source of truth that everyone shares**
- The workflow is as follows:
  - Create a new branch
  - Add and commit changes to the branch and push it to GitHub
  - Create a Pull Request on GitHub
  - Other collaborators look at the PR and leave their feedback (this is generally called a **Code Review**)
  - Fix issues and then merge

# Sample Pull Request

Update README by tcsian · Pull Request

https://github.com/gcc-mirror/gcc/pull/52/files

gcc-mirror / gcc Mirror  
mirrored from git://gcc.gnu.org/git/gcc.git

Code Pull requests 25 Actions Projects Wiki Security Insights

Update README #52

Open tcsian wants to merge 1 commit into gcc-mirror:master from tcsian:master

Conversation 0 Commits 1 Checks 0 Files changed 1 +2 -2

Changes from all commits File filter... Jump to... 0 / 1 files viewed Review changes

4 README

16	16
17	17
18	18
19	19
20	20
21	21
22	22

ProTip! Use **n** and **p** to navigate between commits in a pull request.

© 2020 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About



Working with branches & more

**DEMO**

# IN CASE OF FIRE



1. **git commit**



2. **git push**



3. **git out!**

# Useful resources

- Try Git (resources and tutorial)
- The Git Cheat Sheet
- Stack Overflow's definitive guide for beginners
- When you are terribly stuck with git
  - DO NOT PANIC! Even experienced developers get stuck with git issues
  - <https://ohshitgit.com/>
  - <https://stackoverflow.com/questions/tagged/git>