



What is Git?

Free and open source version control system

What is Version Control system?

- A system that keeps track of our files or projects.
- It allows you to revert selected files to a previous state, **revert** the entire project to a previous state, **compare changes** over time, see who last modified something so that we can know what might be causing a problem, or **what** is the issue, **who** made it, and **when** with the details.



Alice's Code



v1.0.0



v1.0.1



v1.0.2

Bob's code



v1.0.0



v1.0.1



v1.0.2



2 types of VCS

Centralized

Distributed

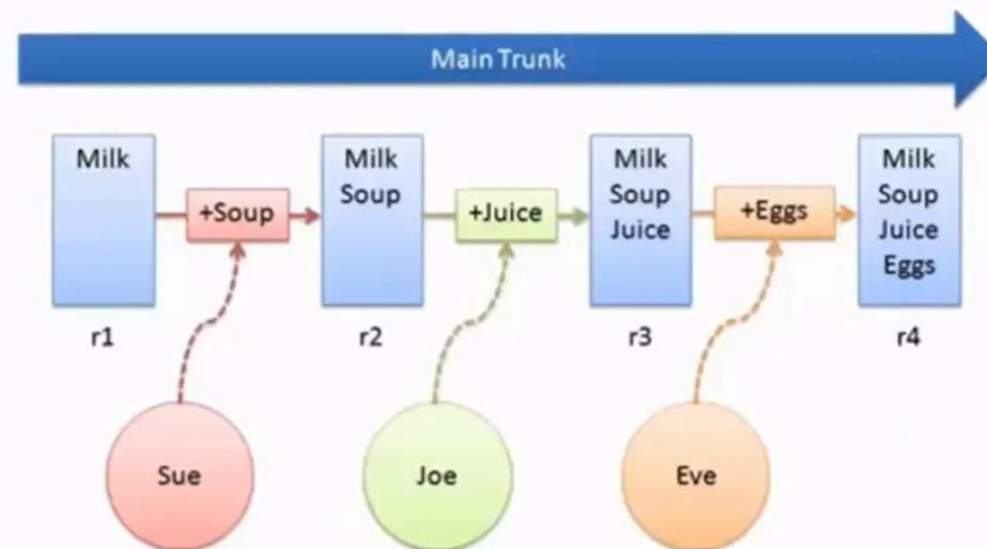


Centralized version control

Helps you backup, track and synchronize files.

Centralized VCS

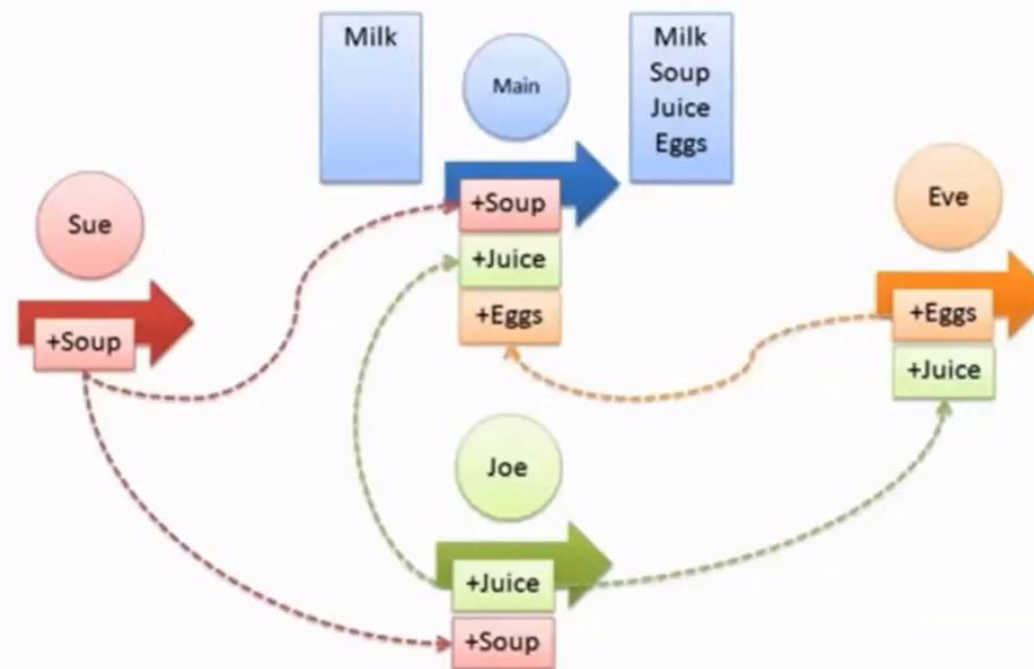
Eg: Subversion &
Team foundation
server





Distributed Version Control Systems

Distributed VCS



Eg: Git & Mercurial



Why Git?

- Free
- Open source
- Scalable
- Super Fast
- Cheap branching and merging



What is GitHub?

GitHub is a **web-based hosting service** for git repositories.

You can use git without Github, but you cannot use GitHub without Git.

Git	GitHub
Used for Version Control	Used for hosting Git repositories
Installed locally on computer	Cloud based
Tracks changes made to a file	Provides a web interface to view file changes



Local Repository

Every VCS tool provides a private workplace as a working copy. Developers make changes in their private workplace and after commit, these changes become a part of the repository. Git takes it one step further by providing them a private copy of the whole repository. Users can perform many operations with this repository such as add file, remove file, rename file, move file, commit changes, and many more.

Working Directory and Staging Area or Index: An intermediate area where commits can be formatted and reviewed before completing the commit.

push: send a change to another repository (may require permission)

pull: grab a change from a repository

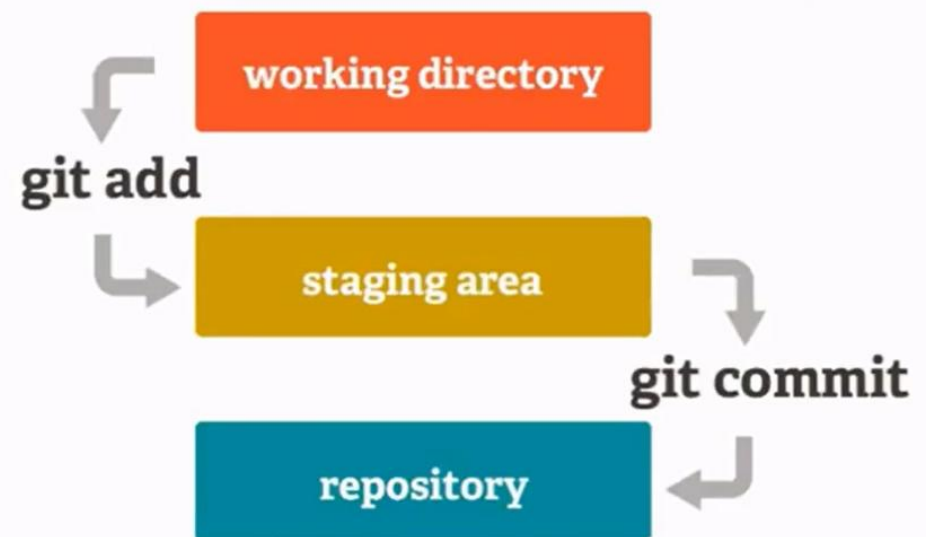


Basic workflow of Git.

Step 1 – You modify a file from the working directory.

Step 2 – You add these files to the staging area.

Step 3 – You perform commit operation that moves the files from the staging area. After push operation, it stores the changes permanently to the Git repository.



Source git-scm.com



Blobs

Blob stands for **B**inary **L**arge **O**bject. Each version of a file is represented by blob. A blob holds the file data but doesn't contain any metadata about the file. It is a binary file, and in Git database, it is named as SHA1 hash of that file. In Git, files are not addressed by names. Everything is content-addressed.

Trees

Tree is an object, which represents a directory. It holds blobs as well as other sub-directories. A tree is a binary file that stores references to blobs and trees which are also named as SHA1 hash of the tree object.



Commits

- Commit holds the current state of the repository. A commit is also named by SHA1 hash.
- Commit object = a node of the linked list.
- Every commit object has a pointer to the parent commit object.
- From a given commit, you can traverse back by looking at the parent pointer to view the history of the commit.
- If a commit has multiple parent commits, then that particular commit has been created by merging two branches.

Git commands



Clone: Bring a repository hosted somewhere like Github into a folder on your local machine

Add: Track your files and changes in Git

Commit: Save your files in git

Push: Upload your commits to a git repo, like GitHub

Pull: Download changes from a remote repository to your local repository.



Already have an account?



LAISHA WADHWA

Welcome to GitHub!
Let's begin the adventure

Enter your email*



Continue

By creating an account, you agree to the [Terms of Service](#). For more information about GitHub's privacy practices, see the [GitHub Privacy Statement](#). We'll occasionally send you account-related emails.



-1:09:27

1x



Adding a new SSH key to your account

Add new SSH key

Generating a new SSH key and adding it to your account

https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

Import bookmarks... Getting Started invoice maker Fractional NFTs netflix_recommendations Advocu - Google Wo... Young Executive Wom... StarkNet Lisbon Hacke... Pesto | Hire developer...

GitHub Docs

Version: Free, Pro, & Team

Search GitHub

Authentication / Connect with SSH / Add a new SSH key

Adding a new SSH key to your GitHub account

To configure your account on GitHub.com to use your new (or existing) SSH key, you'll also need to add the key to your account.

Mac

Windows

Linux

GitHub CLI

Web browser

About addition of SSH keys to your account

You can access and write data in repositories on GitHub.com using SSH (Secure Shell Protocol). When you connect via SSH, you authenticate using a private key file on your local machine. For more information, see "[About SSH](#)."

You can also use SSH to sign commits and tags. For more information about commit signing, see "[About commit signature verification](#)."

In this article

About addition of SSH keys to your account

Prerequisites

Adding a new SSH key to your account

Further reading

https://www.google.com/imgres?imgurl=https://thumbs.dreamstime.com/b/young-executive-woman-profile-icon-be...&w=800&h=800&q=woman executive icon&client=firefox-b-d&ved=2ahUKEwie7PmMj736AhVHi9gFHfXdDCIQMygZegUIARCPAg

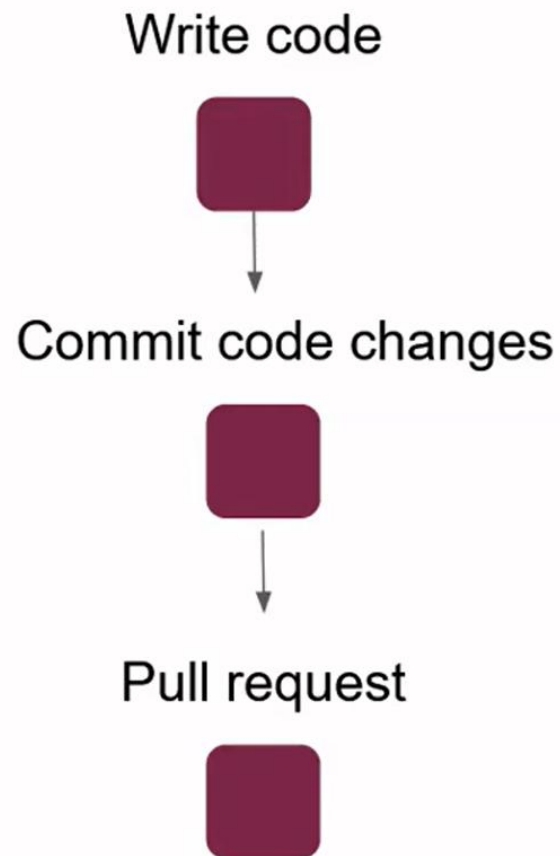
01:25

12°C Haze

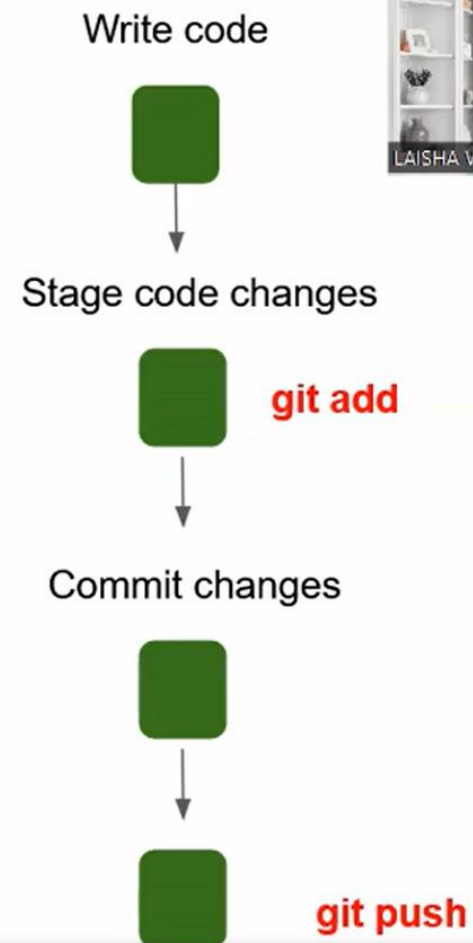
ENG

1x-12

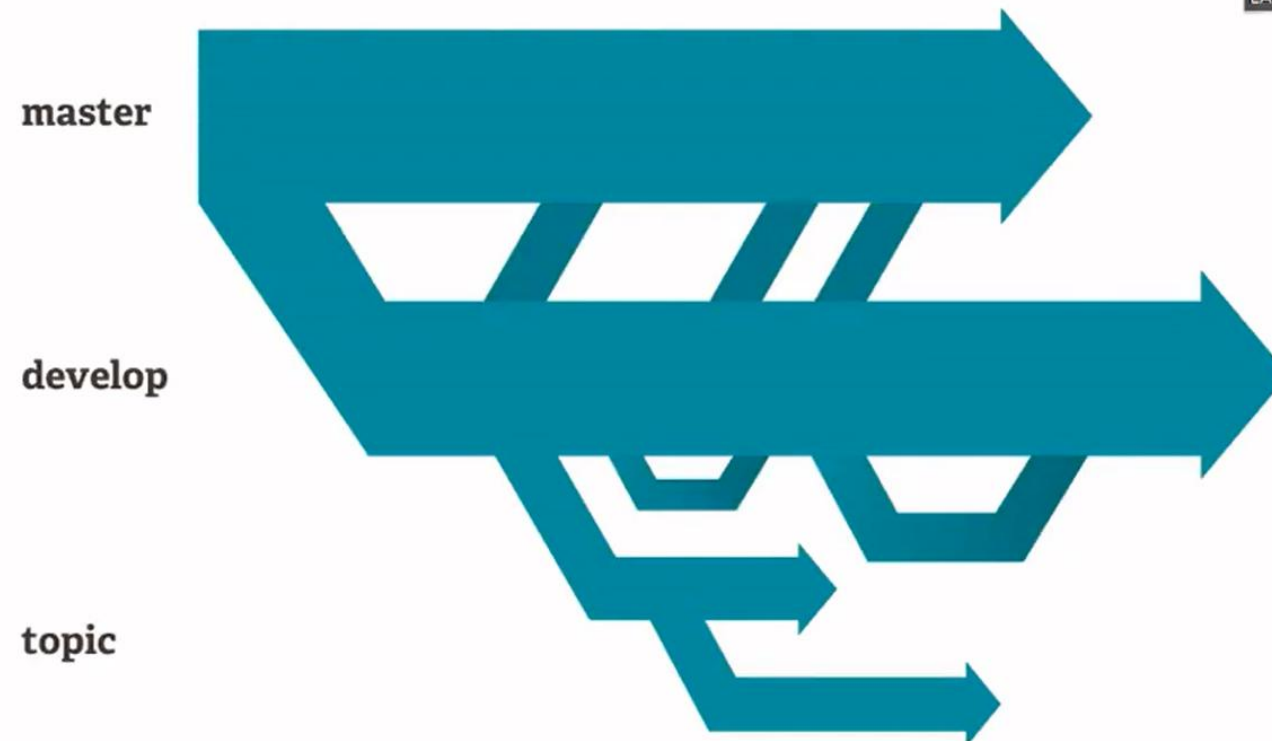
GitHub workflow

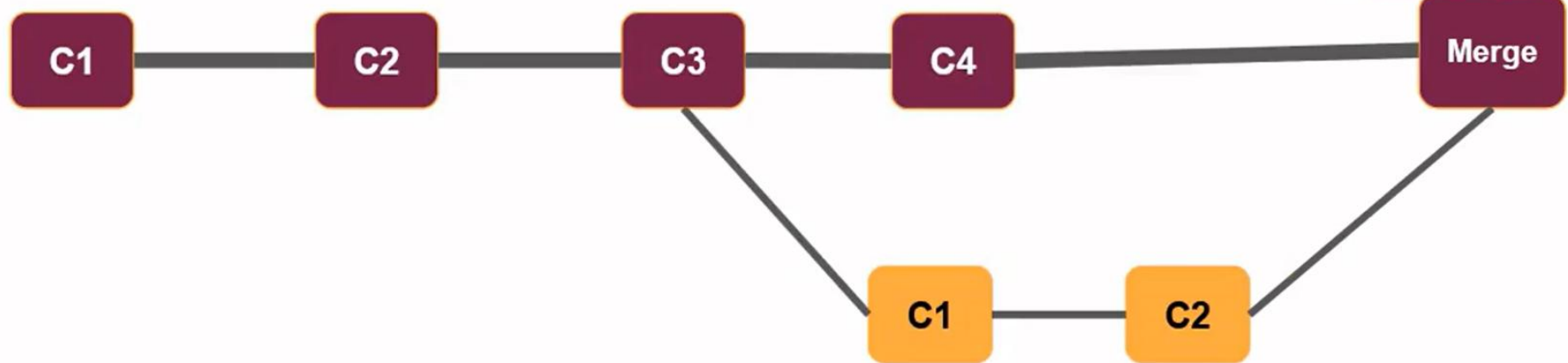


Local Git Workflow



Git Branching



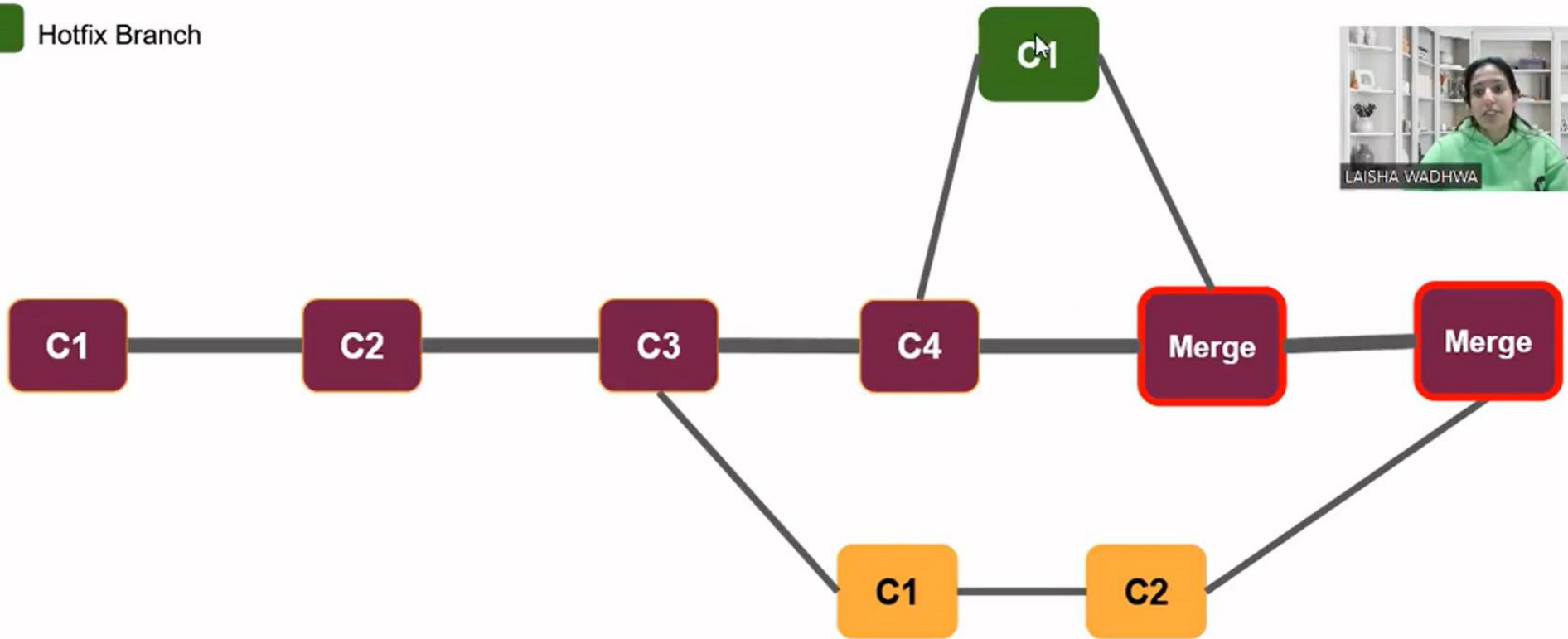


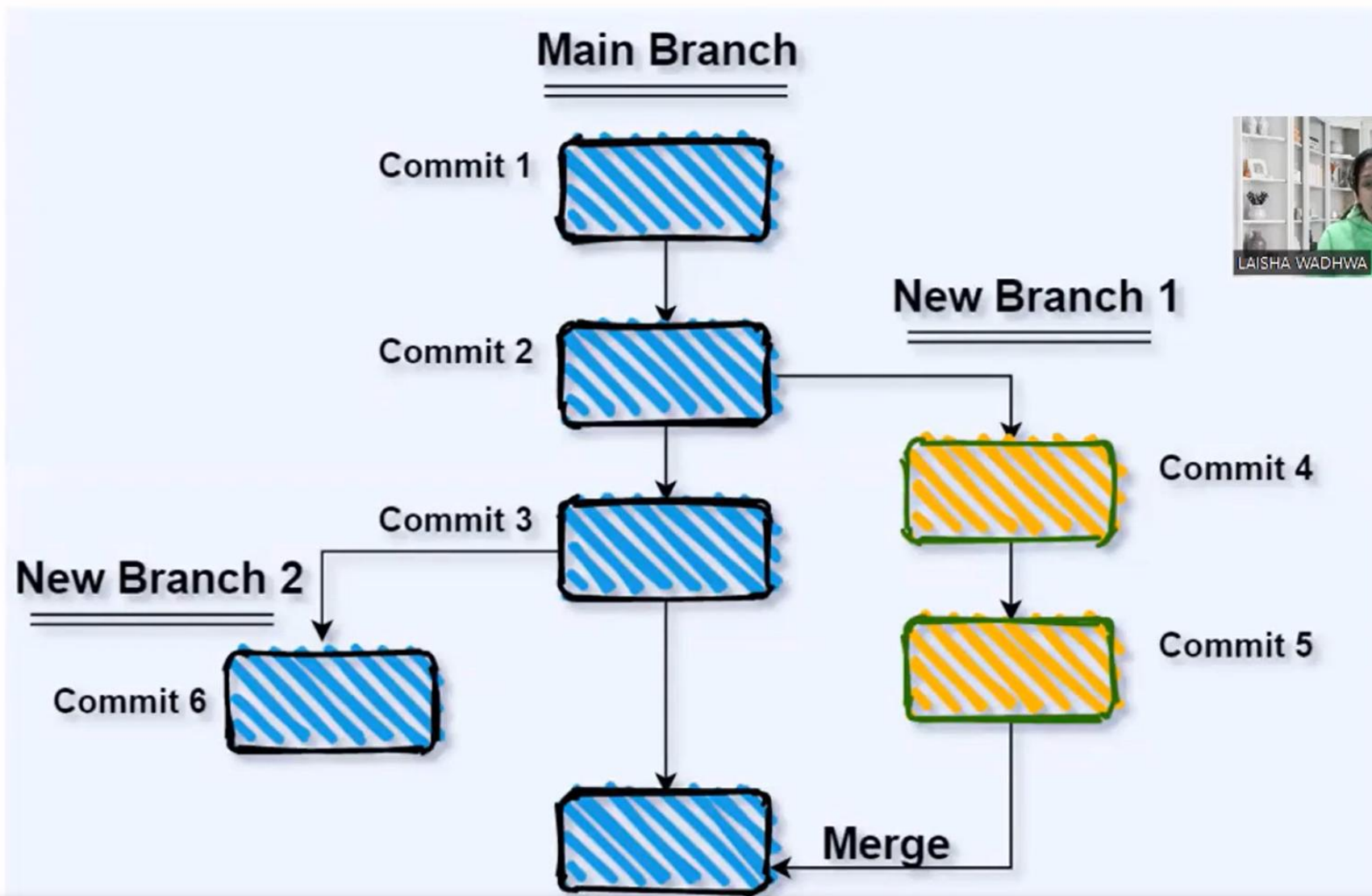
Master Branch



Feature Branch

- Master Branch
- Feature Branch
- Hotfix Branch





MINGW64:/c/Users/Laisha/Documents/repositories/demo-repo

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

```
$ ls
README.md  index.html
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

```
$ git branch
* main
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

```
$ git checkout -b feature_update_files
Switched to a new branch 'feature_update_files'
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

```
$ git branch nch
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

```
$ git branch
* feature_update_files
  main
  nch
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

```
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

```
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

```
$ git branch
  feature_update_files
* main
  nch
```

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

```
$ git |
```



MINGW64: c:/Users/Laisha/Documents/repositories/demo-repo

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ ls
README.md index.html
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ vi README.md
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ git status
On branch feature_update_files
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md
```

no changes added to commit (use "git add" and/or "git commit -a")

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ git add README.md
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ git status
On branch feature_update_files
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ git commit -m "Updated README file"
[feature_update_files 3223d42] Updated README file
1 file changed, 4 insertions(+)
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ vi README.md
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ |
```



MINGW64:/c/Users/Laisha/Documents/repositories/demo-repo

index 460553d..c08c96f 100644

--- a/README.md

+++ b/README.md

@@ -4,7 +4,3 @@ Adding line.

Adding another line to the file!

hello world!

-## Development Environemnt

-

-Make sure you have javascript availbale

-Run index.html in browser

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git checkout feature_update_files

Switched to branch 'feature_update_files'

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

\$ git status

On branch feature_update_files

nothing to commit, working tree clean

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

\$ git push

fatal: The current branch feature_update_files has no upstream branch.

To push the current branch and set the remote as upstream, use

git push --set-upstream origin feature_update_files

To have this happen automatically for branches without a tracking

upstream, see 'push.autoSetupRemote' in 'git help config'.

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

\$ git push -u origin feature_update_files

Enumerating objects: 5, done.

Counting objects: 100% (5/5), done.

Delta compression using up to 4 threads

Compressing objects: 100% (3/3), done.

Writing objects: 100% (3/3), 425 bytes | 425.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

remote:

remote: Create a pull request for 'feature_update_files' on GitHub by visiting:

remote: https://github.com/laishawadhwa/demo-repo/pull/new/feature_update_files

remote:

To <https://github.com/laishawadhwa/demo-repo.git>

* [new branch] feature_update_files -> feature_update_files

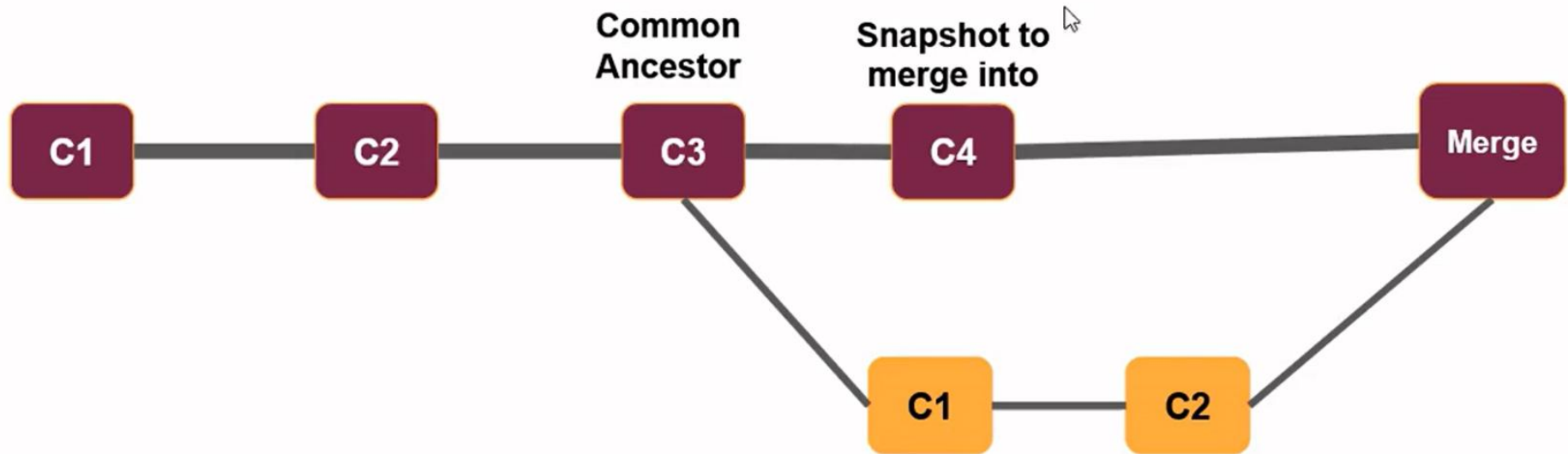
branch 'feature_update_files' set up to track 'origin/feature_update_files'.



Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (feature_update_files)

\$



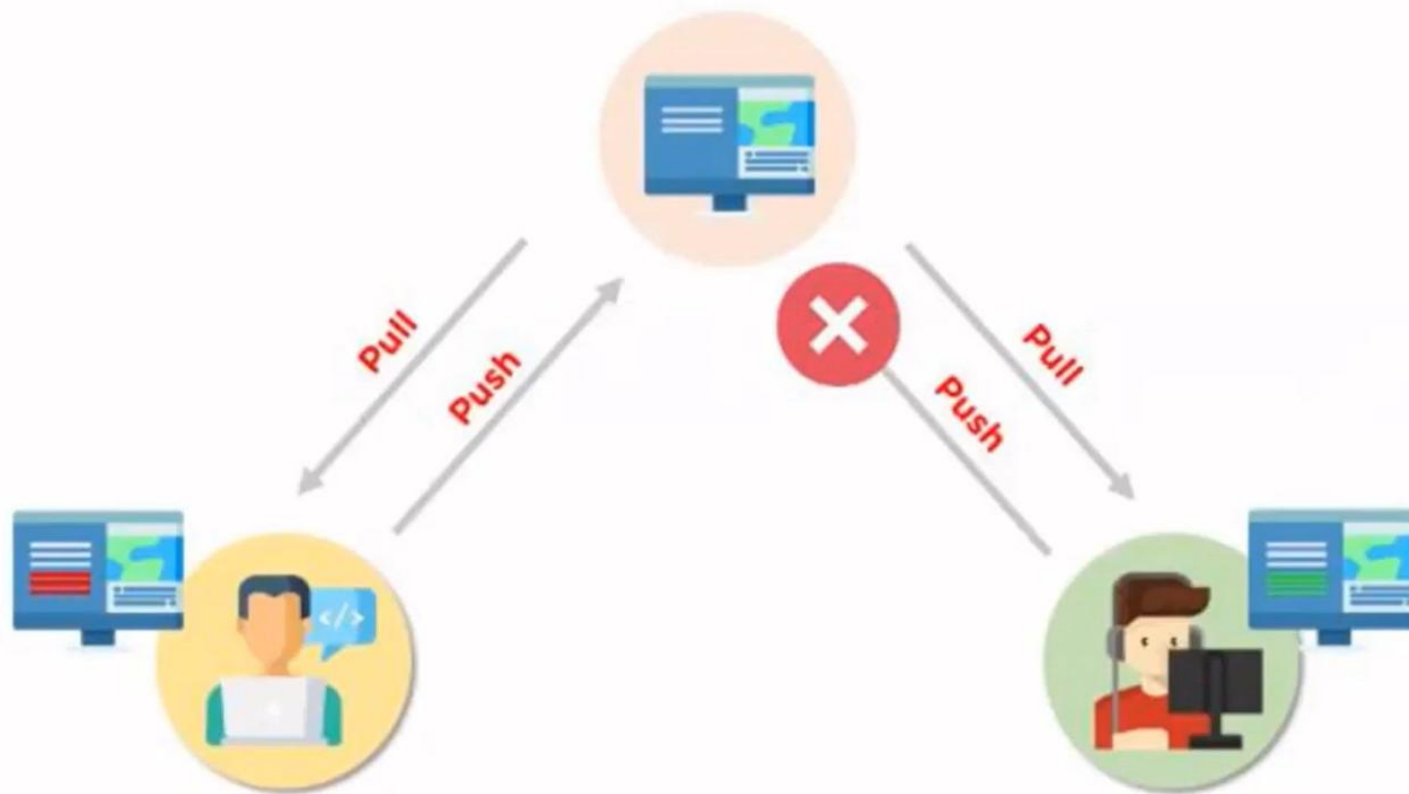
Merging



-  Master Branch
-  Feature Branch

Snapshot to
merge in

What is a Git Merge Conflict?





Merge Conflicts

While starting the merge process

If there are changes in the working directory's stage area for the current project, merging won't start.

In this case, conflicts happen due to pending

During the merge process

When there is a conflict between the local branch and the branch being merged.

Git resolves as much as possible, but there are things that have to be resolved

changes that need to be stabilized

manually in the conflicted files

Merge Conflicts



Occasionally, this process doesn't go smoothly.

If you changed the same part of the same file differently in the two branches you're merging, Git won't be able to merge them cleanly. If your fix for feature branch modified the same part of a file as the hotfix branch, you'll get a merge conflict that looks something like this:

```
git merge iss53
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the
result.
```



Git hasn't automatically created a new merge commit. It has paused the process while you resolve the conflict. If you want to see which files are unmerged at any point after a merge conflict, you can run `git status`:

```
$ git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")

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

    both modified:    index.html

no changes added to commit (use "git add" and/or "git commit -a")
```


Anything that has merge conflicts and hasn't been resolved is listed as
Git adds standard conflict-resolution markers to the files that have conflict
can open them manually and resolve those conflicts. Your file contains a section
that looks something like this:



```

<<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=====
<div id="footer">
  please contact us at support@github.com
</div>
>>>>>> feature_branch:index.html
```




The version in HEAD (your master branch, because that was what you checked out when you ran your merge command) is the top part of that block (everything above the =====), while the version in your feature_branch looks like everything in the bottom part. In order to resolve the conflict, you have to either choose one side or the other or merge the contents yourself. For instance, you might resolve this conflict by replacing the entire block with this:

```
<div id="footer">
please contact us at email.support@github.com
</div>
```



```
$ git status
On branch master
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
  modified:   index.html
```



git commit



Git commands to resolve conflicts

- **git log --merge:** produce the list of commits that are causing the conflict.
- **git diff:** Identify the differences between the states repositories or files.
- **git checkout:** Used to undo the changes made to the file, or for changing branches.
- **git reset --mixed:** Used to undo changes to the working directory and staging area.
- **git merge --abort:** Helps in exiting the merge process and returning back to the state before the merging began.
- **git reset:** Used at the time of merge conflict to reset the conflicted files to their original state.

MINGW64:/c/Users/Laisha/Documents/repositories/demo-repo

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ git branch -d feature_update_files
warning: deleting branch 'feature_update_files' that has been merged to
'refs/remotes/origin/feature_update_files', but not yet merged to HEAD
Deleted branch feature_update_files (was 3223d42).
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ git branch
* main
  nch
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$
```



MINGW64: c:/Users/Laisha/Documents/repositories/demo-repo

README.md index.html

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
$ vi index.html
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
```

```
$ git diff
```

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

```
diff --git a/index.html b/index.html
```

```
index d87d760..d21b524 100644
```

```
--- a/index.html
```

```
+++ b/index.html
```

```
@@ -1,2 @@
```

```
<p>Hello</p>
```

```
+<p>World</p>
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
```

```
$ git status
```

On branch hotfix

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: index.html

no changes added to commit (use "git add" and/or "git commit -a")

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
```

```
$ git commit -am "Added new line in index.html"
```

warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

```
[hotfix eba34af] Added new line in index.html
```

```
1 file changed, 1 insertion(+)
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
```

```
$ git checkout master
```

error: pathspec 'master' did not match any file(s) known to git

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
```

```
$ git checkout main
```

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
```

```
$ ls
```

README.md index.html

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
```

```
$ vi index.html
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
```

```
$ git |
```



MINGW64:/c/Users/Laisha/Documents/repositories/demo-repo

Switched to branch 'main'

Your branch is up to date with 'origin/main'.

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ ls

README.md index.html

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ vi index.html

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git checkout hotfix

error: Your local changes to the following files would be overwritten by checkout:
index.html

Please commit your changes or stash them before you switch branches.

Aborting

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git status

On branch main

Your branch is up to date with 'origin/main'.

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git restore <file>..." to discard changes in working directory)

modified: index.html

no changes added to commit (use "git add" and/or "git commit -a")

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git commit -ma "Updates index.html with a new line called universe"

error: pathspec 'Updates index.html with a new line called universe' did not match any file(s) known to git

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git commit -am "Updates index.html with a new line called universe"

[main f24ada2] Updates index.html with a new line called universe

1 file changed, 1 insertion(+)

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)

\$ git checkout hotfix

Switched to branch 'hotfix'

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)

\$ git merge main

Auto-merging index.html

CONFLICT (content): Merge conflict in index.html

Automatic merge failed; fix conflicts and then commit the result.

Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix|MERGING)

\$



MINGW64:/c/Users/Laisha/Documents/repositories/demo-repo

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ git commit -ma "Updates index.html with a new line called universe"
error: pathspec 'Updates index.html with a new line called universe' did not match any file(s) known to git
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ git commit -am "Updates index.html with a new line called universe"
[main f24ada2] Updates index.html with a new line called universe
1 file changed, 1 insertion(+)
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (main)
$ git checkout hotfix
Switched to branch 'hotfix'
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix|MERGING)
$ vi index.html
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix|MERGING)
$ git diff master
fatal: ambiguous argument 'master': unknown revision or path not in the working tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...']'
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix|MERGING)
$ git diff main
diff --git a/index.html b/index.html
index c507f17..b10afb6 100644
--- a/index.html
+++ b/index.html
@@ -1,2 +1,3 @@
 <p>Hello</p>
+<p>World</p>
 <p>Universe</p>
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix|MERGING)
$ git commit -am "Resolved merge conflict"
[hotfix 8c1f0b6] Resolved merge conflict
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
$ git merge main
Already up to date.
```

```
Laisha@DESKTOP-VEGAS52 MINGW64 ~/Documents/repositories/demo-repo (hotfix)
$
```



