JAVA FULL STACK DEVELOPMENT PROGRAM

VERSION CONTROL & GITHUB

OUTLINE

- Git
  - What is Git?
  - Version Control?
- Tutorial
  - Installation
  - How to use Git
    - Basic Commit Management
    - Advanced Branch & Merge

WHAT IS GIT

- *"Git is a free and **open source distributed version control system** designed to handle everything from small to very large projects with speed and efficiency."*
- Version control system/tool/software

WHAT IS VERSION CONTROL

- Also known as ***source control***
- What can Version Control do?
  - Help us keep track file changes
  - Provide metadata for each file change (date, description)
  - Allow us to switch between different file versions
  - Provide a easy way to rollback file changes
- Minimize the disruption of our code change to all other team members

VERSION CONTROL EXAMPLE

Three month later

Work Experience

Education

Rephrase Sentences

Name	Date modified	Type	Size
MyResume	4/26/2021 10:55 PM	Microsoft Word ...	13 KB
MyResume - Version 1	4/26/2021 10:55 PM	Microsoft Word ...	13 KB
MyResume - Version 2	4/26/2021 10:55 PM	Microsoft Word ...	13 KB
MyResume - Finalized Vesion 1	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
MyResume - Finalized Vesion 2	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
MyResume - This is the final version	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
MyResume - This is the final version 2	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
MyResume - No more changes	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
MyResume - Ok, one more ochange	4/26/2021 10:56 PM	Microsoft Word ...	13 KB
My Last Will	4/26/2021 11:06 PM	Microsoft Word ...	13 KB

VERSION CONTROL – DATA TABLE

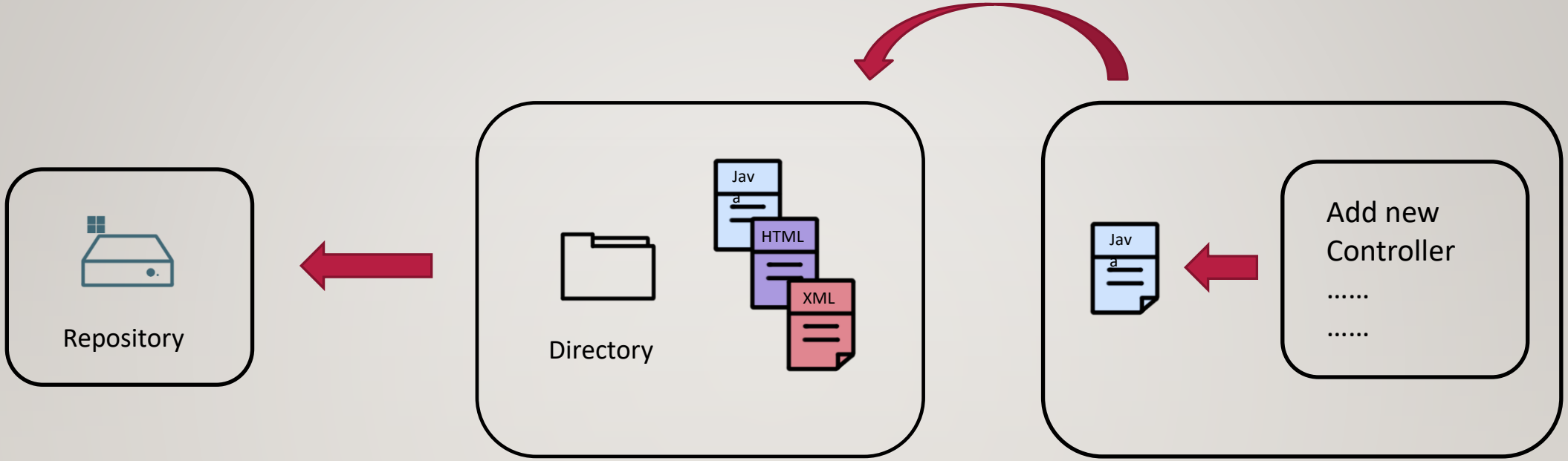
Version	File Name	User	Description	Date	FileLocation
1	MyResume	Landon	First Draft	4/26/2021	C:\git\Landon\Resume\MyResume.docx
2	MyResume - Version 1	Landon	Add work experience	4/27/2021	C:\git\Landon\Resume\MyResume - Version 1.docx
3	MyResume - Version 2	Landon	Add Education	4/28/2021	C:\git\Landon\Resume\MyResume - Version 2.docx
4	MyResume - Finalized Version	Landon	Rephrase sentences	4/29/2021	C:\git\Landon\Resume\MyResume - Finalized Version.docx

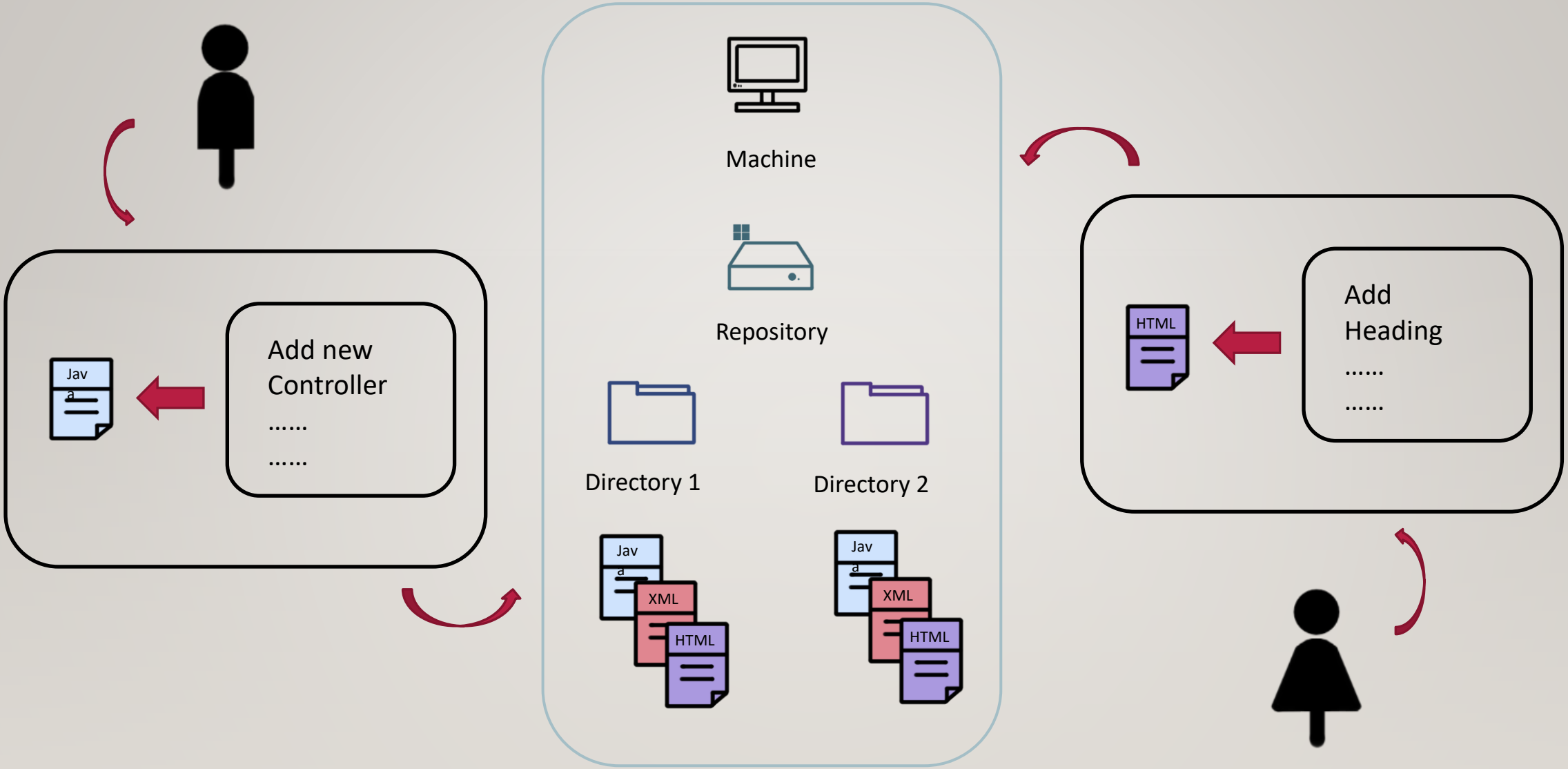
VERSION CONTROL

Version	File Name	User	Description	Date	FileLocation
1	UserController	Jason	Add getUserById() method	3/26/2021	C:\project\java\controller\UserController
2	UserController	Landon	Add getAllUsers() method	3/27/2021	C:\project\java\controller\UserController
3	UserController	Jason	Add getUserByName() method	3/28/2021	C:\project\java\controller\UserController
4	UserController	Landon	Add createNewUser() method	3/29/2021	C:\project\java\controller\UserController

GIT REPOSITORY

- Basic of version control is **Repository**
- **Repository**
  - Works like a database
  - Store all the directories and its files
  - Store the status of these directories
  - Store the history record of all changes users made to the directories


Repository
Directory
Jav
a
HTML
XML
Jav
a
Add new
Controller
……
……


Jav
a
Add new
Controller
……
……
Machine
Repository
Directory 1
Directory 2
Jav
a
XML
HTML
Jav
a
XML
HTML
HTML
Add
Heading
……
……

CENTRALIZED VS. DISTRIBUTED

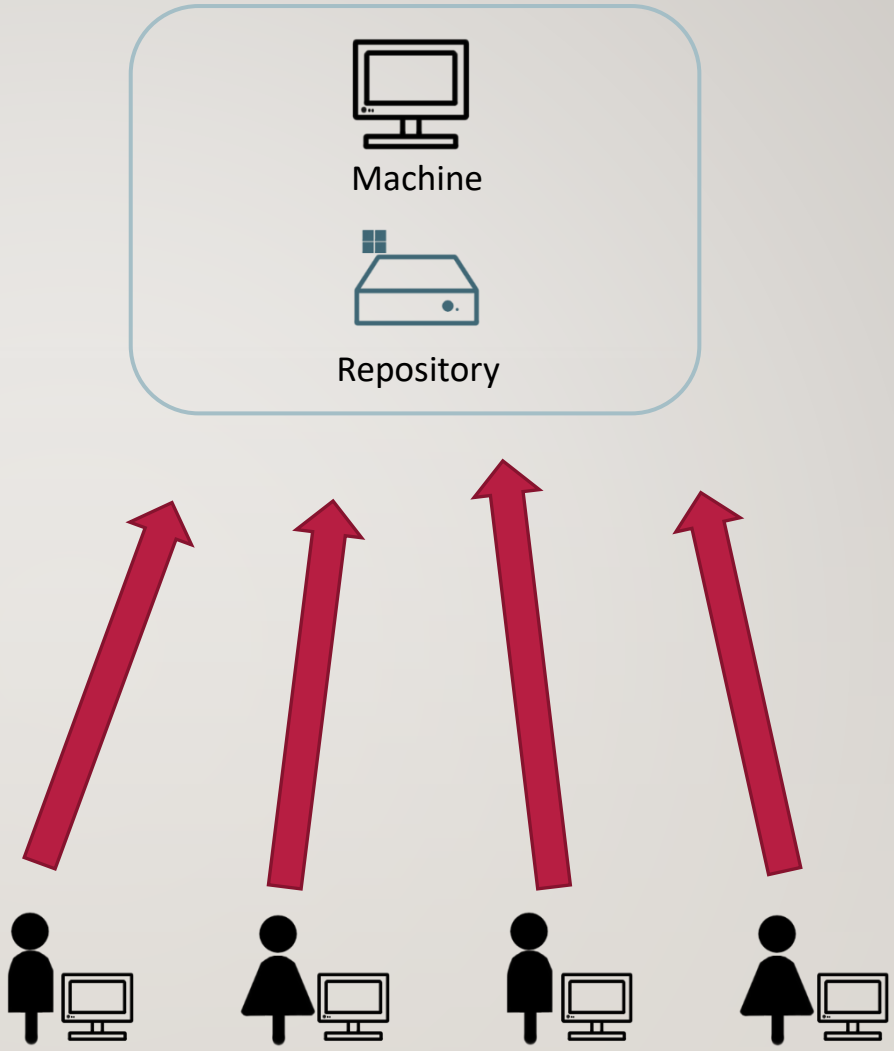
- Centralized
  - Store all the files in a single server (*single source of truth*)
  - All users must connect to the server before submitting or downloading files (*network required*)
  - Multiple users cannot work on the same file at the same moment (*order matters*)
  - CVS, SVM

Pros

- Transparent Operation

Cons

- Single Point of Failure



CENTRALIZED VS. DISTRIBUTED

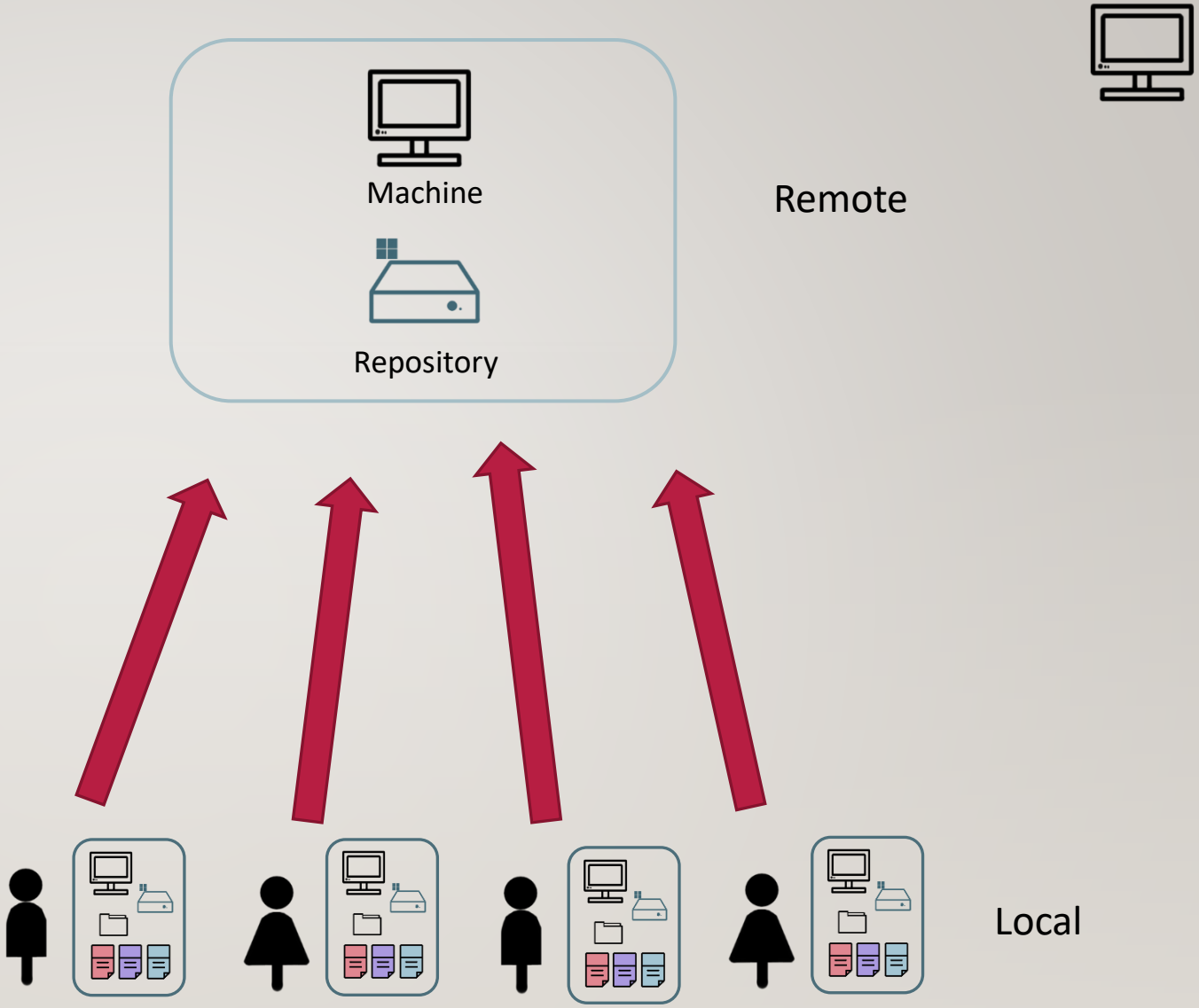
- Distributed
  - Each machine hold a copy code repository (distributed version control system)
  - Use remote code repository for file exchange (act as central server)
  - Git

Pros

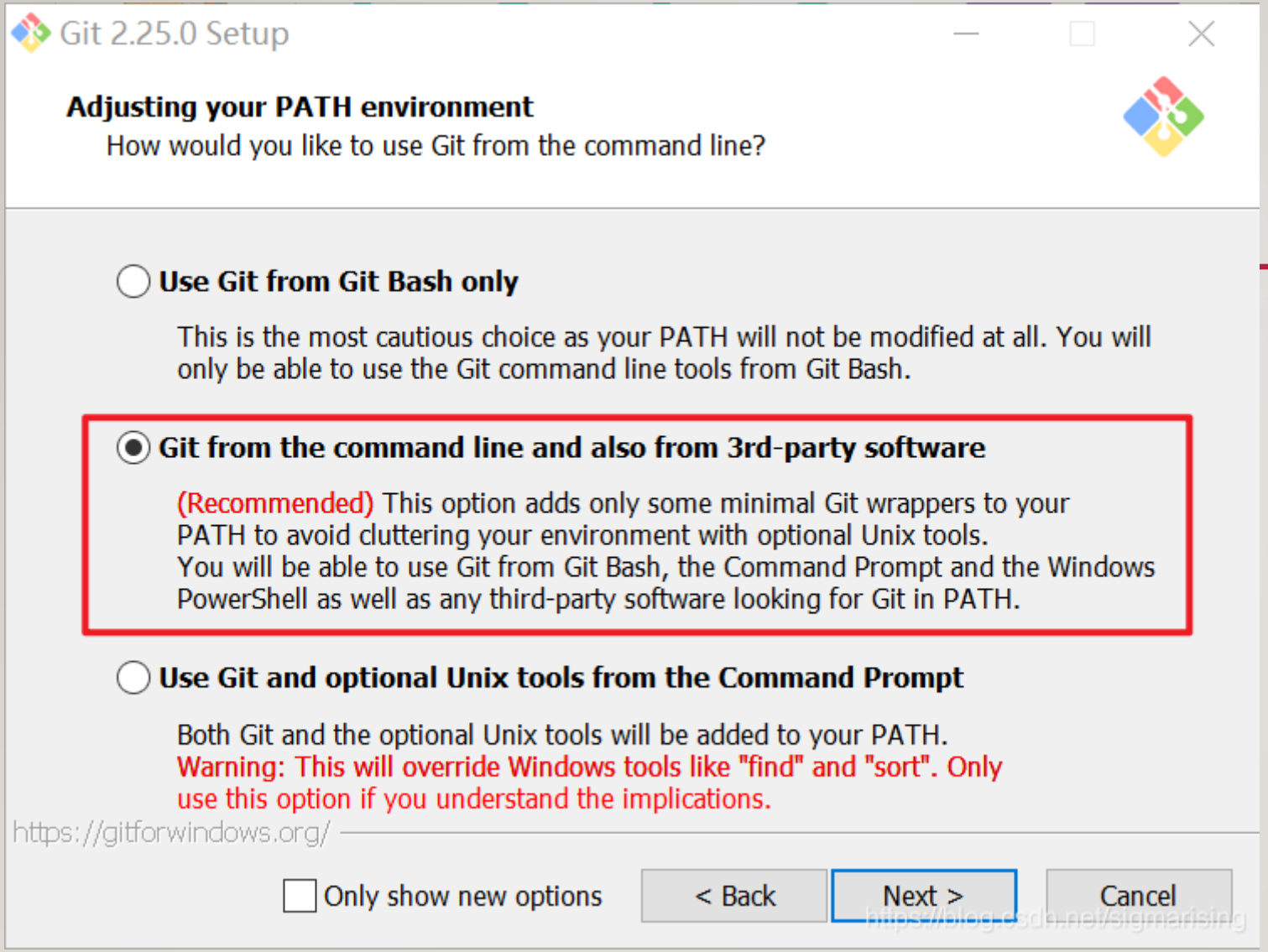
- More flexible
- Not single point of failure

Cons

- Need more efforts to manage change history and authentication

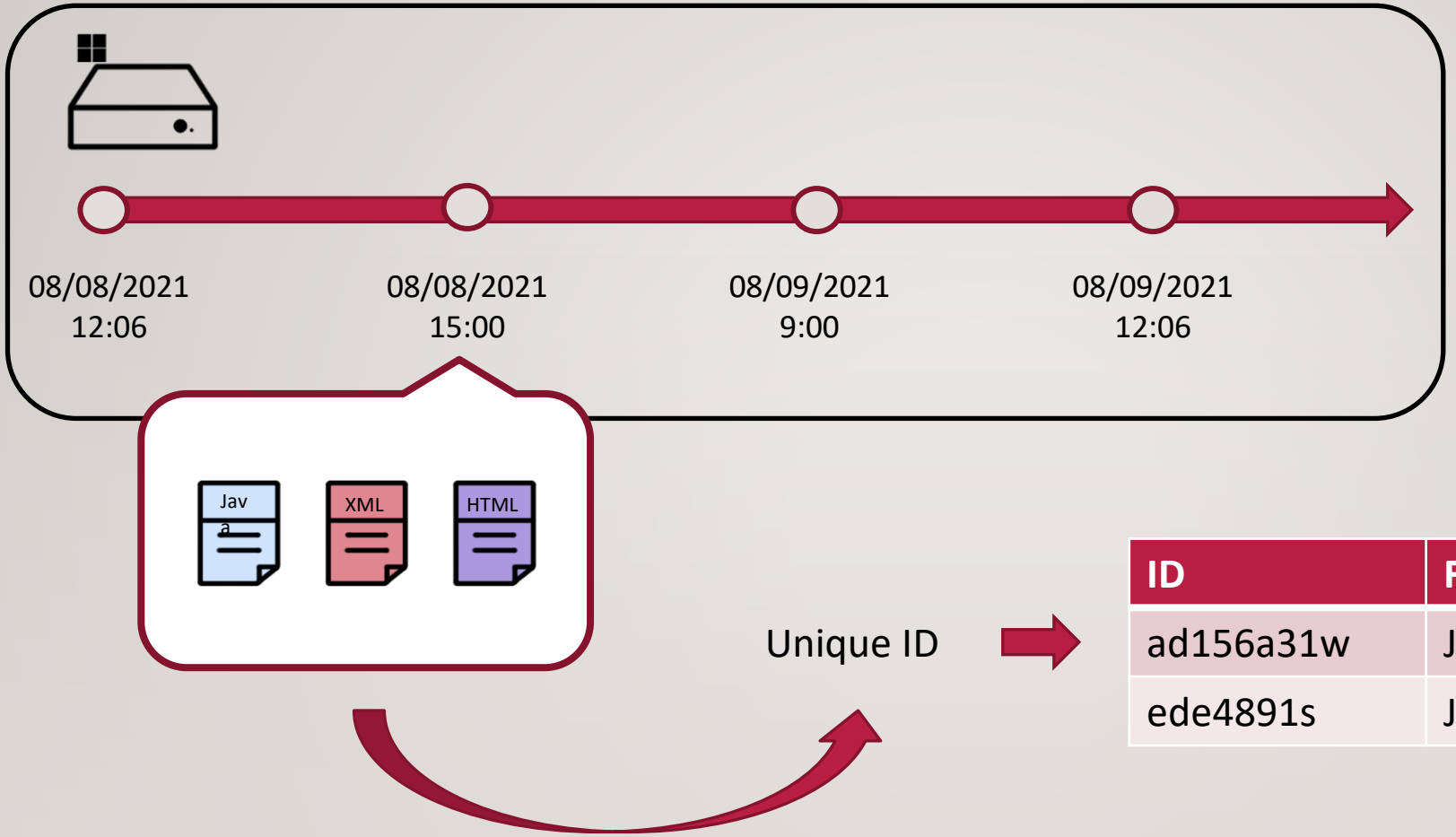


GIT TOOL


Git 2.25.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Only show new options
< Back
Next >
Cancel

COMMIT

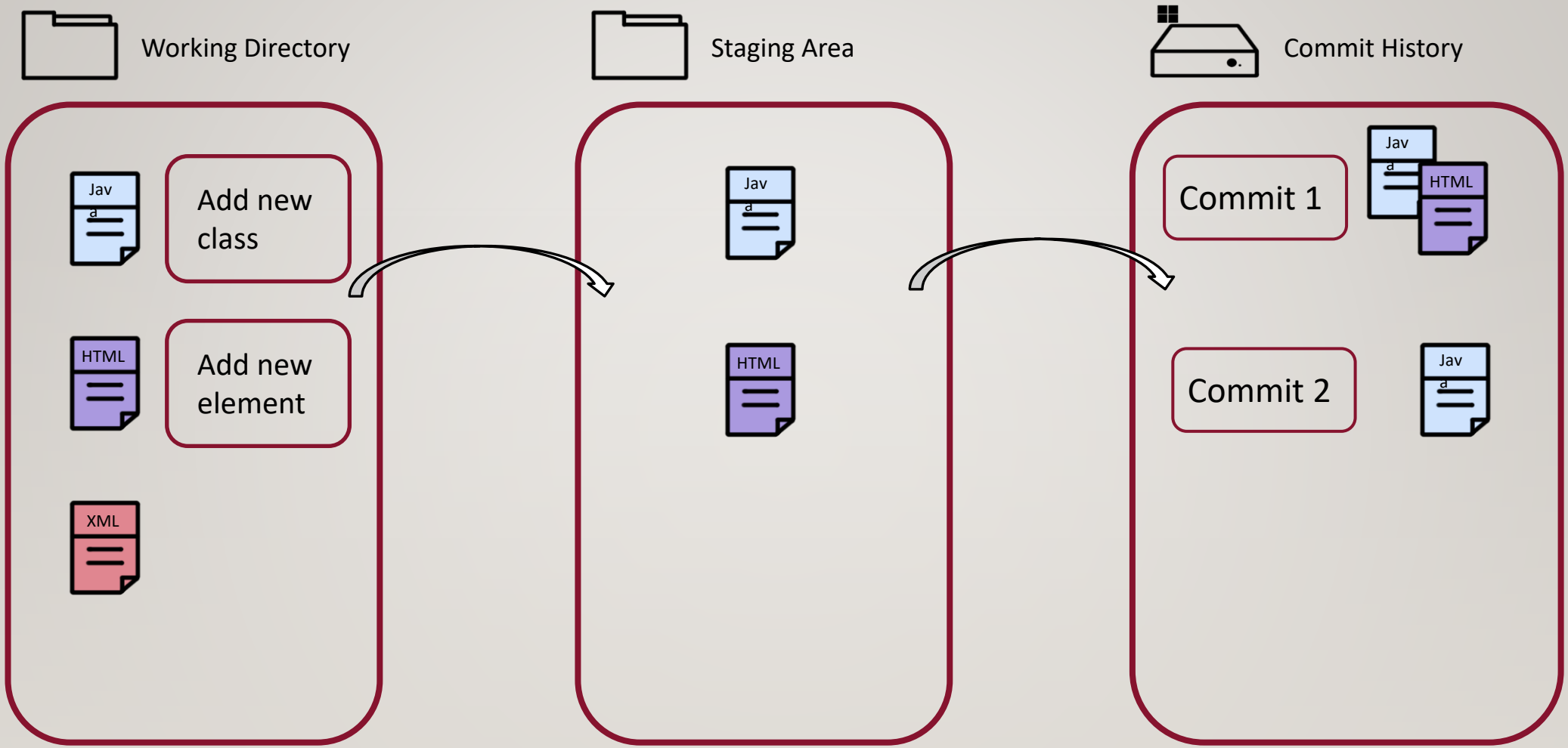
- Git commit
  - A record in git database
  - Represent a snapshot in certain timeline of the git project



ID	Files
ad156a31w	Java
ede4891s	Java, HTML

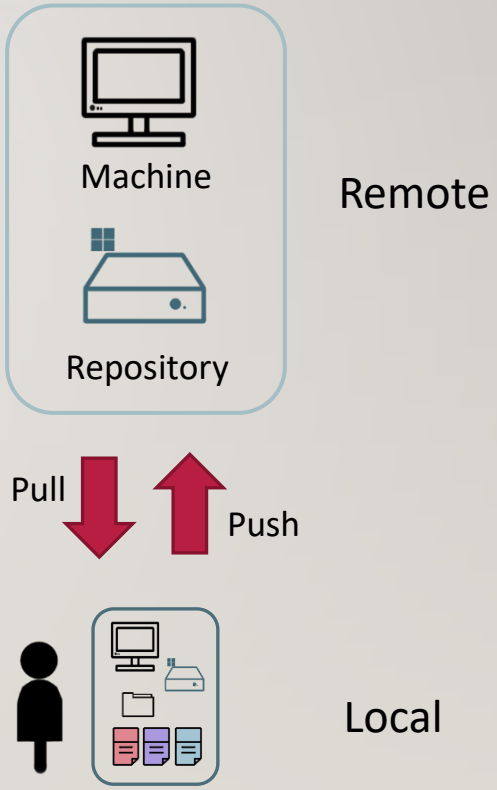
GIT FEATURES

- How does Git create new commit?
  - ***Working Directory*** – disk area where we store all the files
  - ***Staging Area*** – cache memory, temporary hole the files from workspace
  - ***Commit History*** – files submitted from staging area will be stored here, keeps the history versions of all the files
- Work flow
  - Make file changes in ***Working Directory***
  - Submit those files into ***Staging Area***
  - Gather all the files in staging area and create a commit in ***Commit History***


Working Directory
Jav
d
Add new class
HTML
Add new element
XML
Staging Area
Jav
d
HTML
Commit History
Commit 1
Jav
d
HTML
Commit 2
Jav
d

GIT REMOTE REPOSITORY

- Remote repository work as a file exchange center
  - We can ***push*** our changes from local repo to remote repo
  - We can ***pull*** the changes from remote repo to our local repo
- Local Area Network: GitLab
- Internet: GitHub
- GitHub – public remote service, open source



GIT COMMAND

GIT COMMAND

- Two generic type
  - Version control related
  - Configuration related
- Configuration
  - Account setup – let git know who we are
    - git config --global user.name {name}
    - git config --global user.email {email}

GIT COMMAND – INITIALIZE LOCAL REPO

- Each application in Git has its own version storage called **Repository**
- Repository stores all the versions of files, allow us to switch between different versions
- There are two ways to create git repository
  - Clone an existing one from remote repo to local
  - Create a brand new repo in local
- Initialize new local repository
  - Command: *git init*
  - Give git permission to manage our project
  - Create a ".git" folder with all the configuration files

GIT COMMAND – INSPECT REPO

- Check current repo status
- Command: *git status*

```
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

  - On branch master – show which branch we are currently on
  - No commits yet – we did not commit any file to our version control system
  - Nothing to commit – we do not have any file that need to be committed

GIT COMMAND – INSPECT REPO

- Once we add new files or modify/delete existing files, we will have untracked files
  - Git will tell us we have untracked files/changes that need to be taken care of

```
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Example.txt

nothing added to commit but untracked files present (use "git add" to track)
```

GIT COMMAND – COMMIT FILE TO STAGING

- To save our modification to the code repository, we first need to add the changes to staging area
- Command: *git add {filename}* / *git add .*

```
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   Example.txt
```

- File name will turn green and displayed in “Changes to be committed section”.

GIT COMMAND – REMOVE FROM STAGING

- If we found something is wrong with current modification and we do not want to save the file into version history, we can also remove the file from staging area
- Command: git rm --cached {filename}
- File will be removed from staging, will not be stored in version history
- Remove command has many variation; self-study

GIT COMMAND – COMMIT TO REPOSITORY

- Command
  - Specific file: git commit {filename} –m “{logging message}”
  - All files: git commit –m “{logging message}”
- Git will submit file/files from staging area to the repository and create a version history
- Logging is a short description of the current commit, it helps document the changes and helped other user understand the content (**what if we didn’t put – m?**)

```
$ git commit Example.txt -m "Sample"
[master (root-commit) c109591] Sample
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 Example.txt
```

GIT COMMAND – MODIFY FILE

- Git maintains our repository files based on lines
- Git does not know the detail of our changes
- When we modify files, git will delete the original line and add a new line instead
- This is true even if we only add or delete on character to/from the line

```
$ git commit -m "Second Commit" Example.txt
[master 9646035] Second Commit
 1 file changed, 1 insertion(+)
```

GIT COMMAND – CHECK VERSION HISTORY

- Git allow us to check version history
- Command:
  - git reflog
  - git log
- Git also provides a visualized version of the log:
  - git log --graph --oneline --all (better experience when dealing with branch)

GIT COMMAND – SWITCH VERSION

- Git allow us to switch between different versions
- Command:
  - git reset --hard {version number}
- Git manage version using pointer, pointer will point to the current version
- When we switch between different versions, git move the pointer to the version number and load the corresponding files

HOW DOES IT WORK

Pointer	Version ID	File Name	User	Description	Date	Location
	1	UserController	Jason	Add getUserById() method	01/01/2022/	C:\project\Usercontroller
Head	2	UserController	Jason	Add getUserById() method	01/01/2022/	C:\project\Usercontroller

GIT REMOTE

GIT REMOTE

- What is remote repository?
  - A common repository shared by team members
  - Used to exchange version history and file changes
  - Stored on a code hosting service like GitHub
  - Team member can submit, fetch and clone info from remote server
- How clone Git repository?
  - *git clone {repo address}*

GIT REMOTE – OPERATIONS

- Interact with remote service
  - Submit local changes
    - git push
  - Fetch remote changes
    - git pull
    - Git rebase

GIT CONFLICT

- Happens when two developers make changes on the same file, in the same location
- Git does not know which change it should take as the version to use
- Demo
- Resolve conflict
  - Open the files in editor and manually fix the conflict
  - Commit the change

AVOID CONFLICT

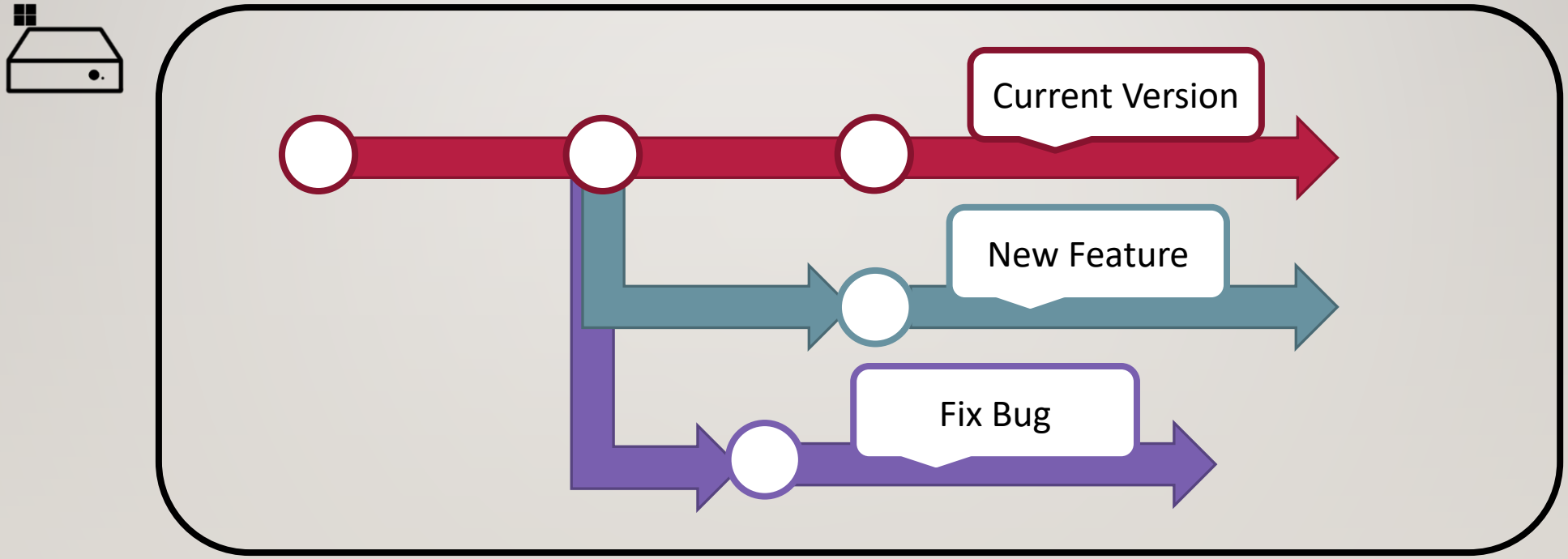
- Git conflict happens when multiple team members work on the same file
- To avoid conflict:
  - Split task to each member will work on different parts
  - When you need to make changes on other files, always notify your team member

GIT BRANCH

GIT BRANCH

- What is branch?
  - A feature available in most modern version control system
  - An independent line of development.
- Why do we need branch?
  - Isolate the development of new feature from existing code base
  - Prevent unstable code to be merged into main code base

GIT BRANCH – EXAMPLE



HOW DOES IT WORK

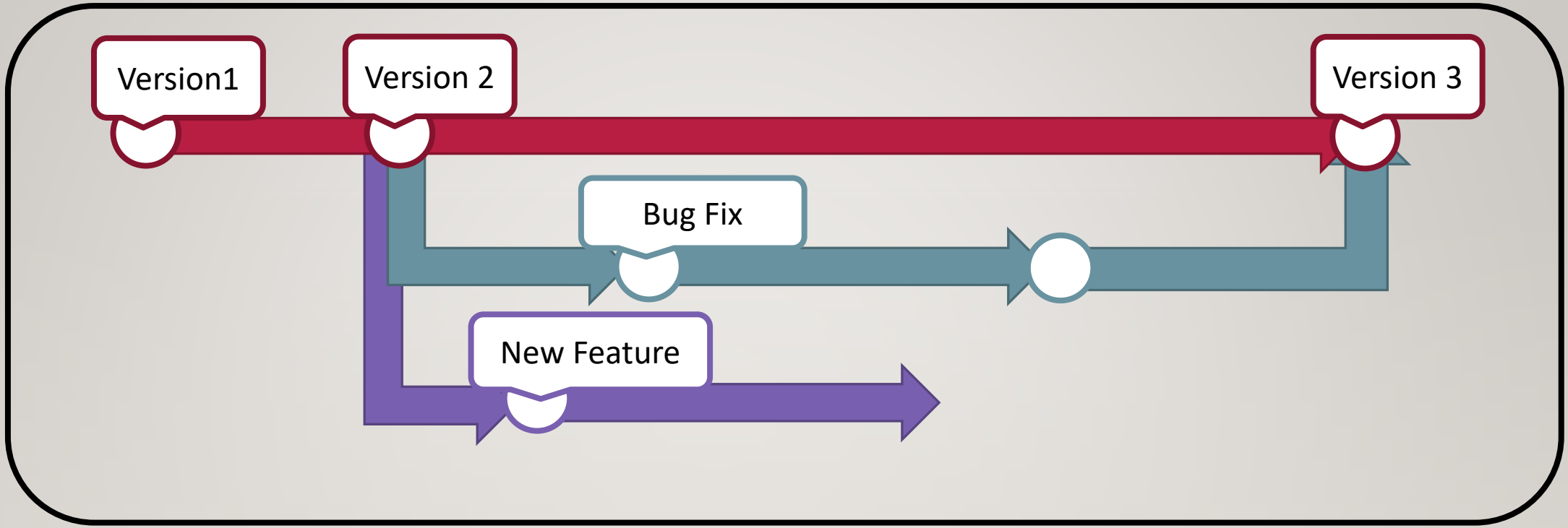
Pointer	Branch	Version ID	File Name	User	Description	Date	Location
		1	UserController	Jason	Add getUserById() method	01/01/2022/	C:\project\Usercontroller
	Master	2	UserController	Jason	Add getUserById() method	01/01/2022/	C:\project\Usercontroller
Head	Branch1	3	UserController	Jason	Add getUserById() method	01/01/2022/	C:\project\Usercontroller

GIT BRANCH – COMMAND

- Inspect Branch
  - git branch -v
- Create Branch
  - git branch {name}
- Switch between Branches
  - git checkout {name}

MERGE BRANCH

- When development is finished and tested, we would like to merge our changes back into our main branch
- Merge Branches
  - git merge {name}
  - If merge A to B, switch to branch B first, then call “git merge A”
  - Merge may also cause conflict


Version1
Version 2
Version 3
Bug Fix
New Feature

QUESTIONS