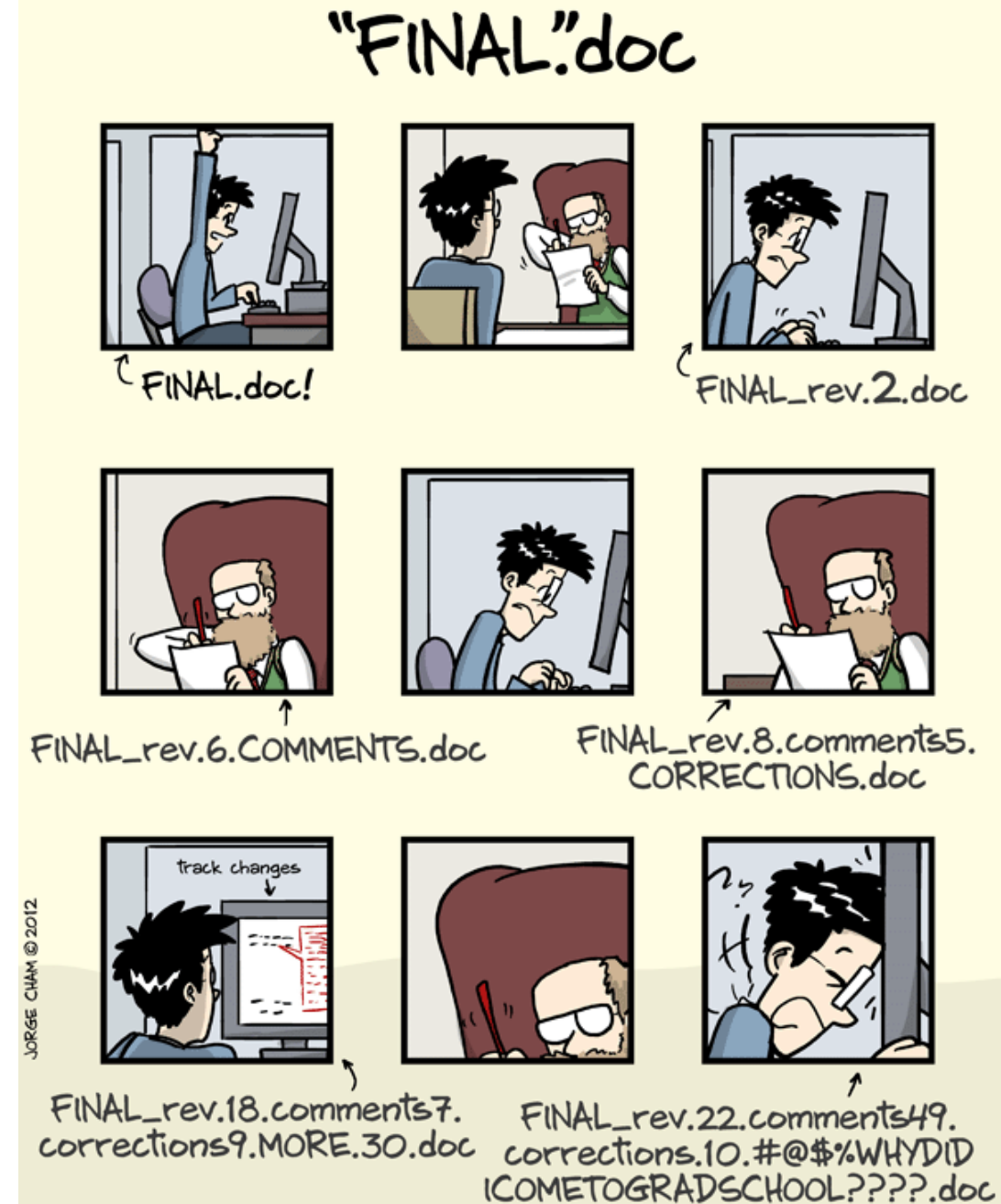


# Git & GitHub

Including project management

Deepak Tanwar  
Bioinformatics Specialist and Trainer  
Swiss Institute of Bioinformatics  
University of Zurich

Email: [deepak.tanwar@uzh.ch](mailto:deepak.tanwar@uzh.ch)  
Bluesky: <http://dktanwar.bsky.social/>



# Learning objectives

What are versions?

Basic Git and GitHub terminologies

Have a GitHub user account

Markdown styling of the text

Git commands: add, commit, diff, branch, checkout, merge, log

# Introduction

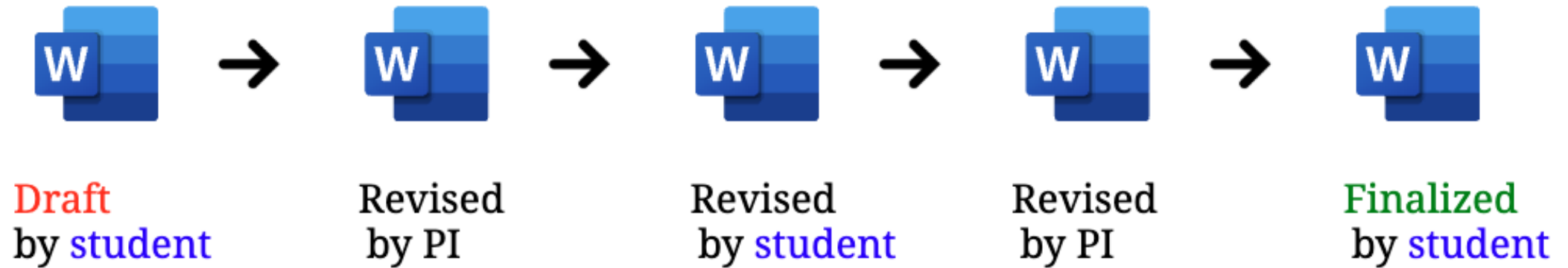
Name

Designation

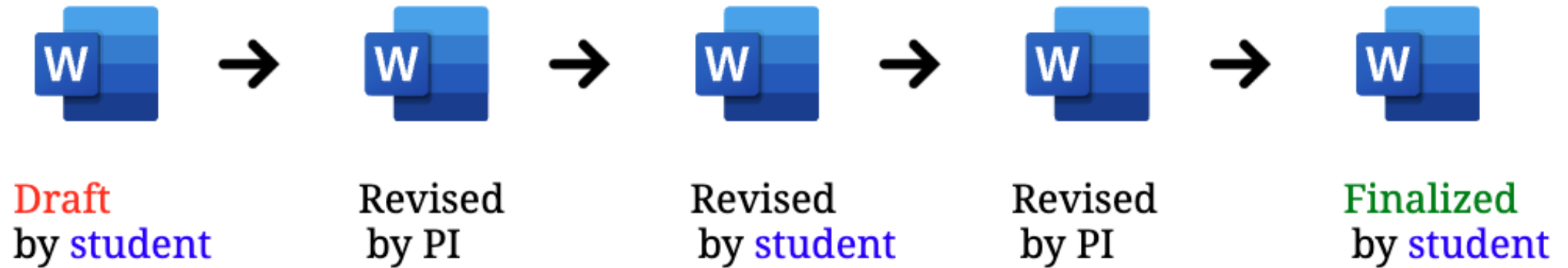
Research group & Institution

Experience with Git and/ or GitHub

# A general example of manuscript revision

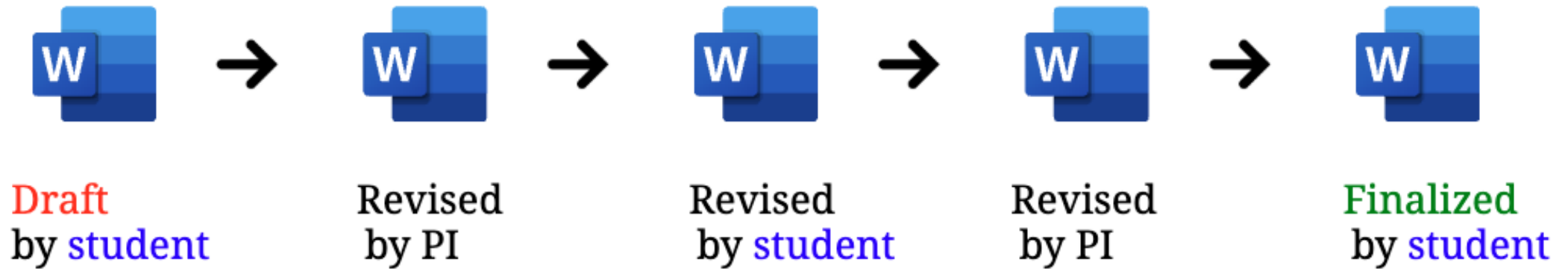


# A general example of manuscript revision



What would you call these files?

# A general example of manuscript revision



What would you call these files?

Versions?

# How would you store different versions?

- manuscript\_FA.docx
- manuscript\_FA\_LA.docx
- manuscript\_FA\_LA\_FA.docx
- manuscript\_FA\_LA\_FA\_LA.docx
- manuscript\_final.docx

# How would you store different versions?

- manuscript\_**FA**.docx
- manuscript\_**FA\_LA**.docx
- manuscript\_**FA\_LA\_FA**.docx
- manuscript\_**FA\_LA\_FA\_LA**.docx
- manuscript\_**final**.docx
- manuscript\_**20221001**.docx
- manuscript\_**20221224**.docx
- manuscript\_**20230107**.docx
- manuscript\_**20230201**.docx
- manuscript\_**final**.docx



# How would you store different versions?

- manuscript\_FA.docx
- manuscript\_FA\_LA.docx
- manuscript\_FA\_LA\_FA.docx
- manuscript\_FA\_LA\_FA\_LA.docx
- manuscript\_final.docx
- manuscript\_20221001.docx
- manuscript\_20221224.docx
- manuscript\_20230107.docx
- manuscript\_20230201.docx
- manuscript\_final.docx

Does anyone store files differently?

# Version control

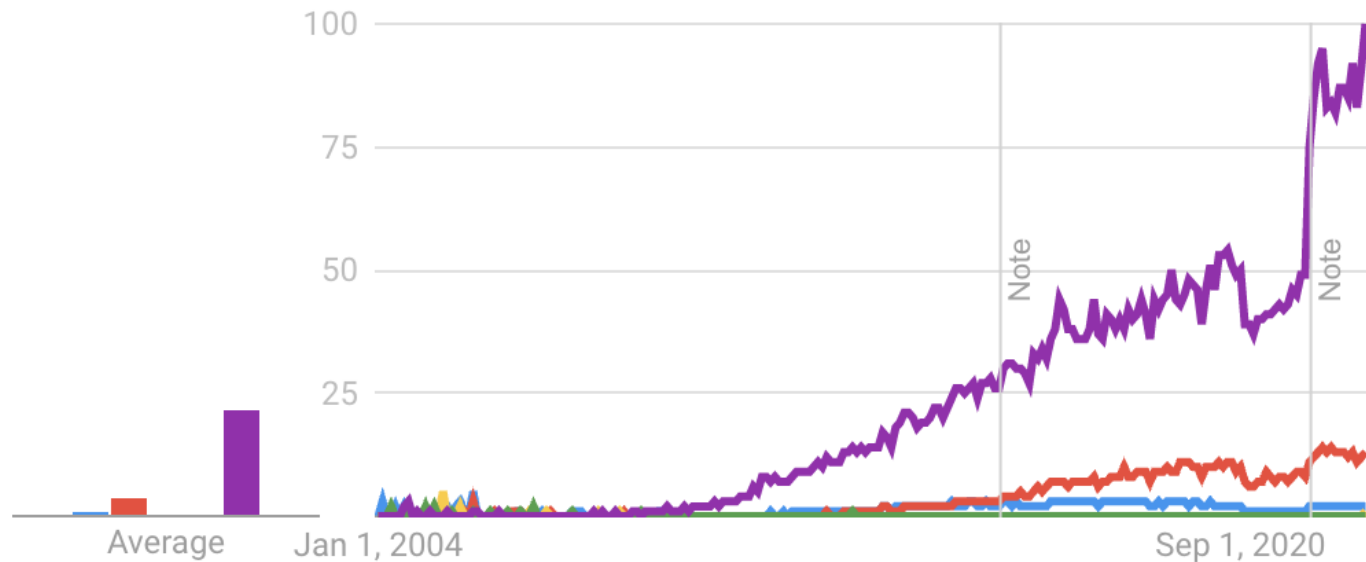
• File	“commit message”	version/commit ID
• manuscript.docx	“first draft”	ac3eveta
• manuscript.docx	“revision 1”	a23eveta
• manuscript.docx	“revision 2”	cc3eveta
• manuscript.docx	“updated figures”	dc3edeta
• manuscript.docx	“removed table”	zc9aveta
• manuscript.docx	“line numbering”	tc5eveta
• manuscript.docx	“bioRxiv submission”	yc8evata
• manuscript.docx	“published”	zz33eeta

# How to host these versions?

Interest over time

Google Trends

● BitBucket ● GitLab ● Gitea ● Renku ● GitHub



Worldwide. 1/1/04 - 2/9/23. Web Search.

# Terminologies

Organization	shared account
Username	your identity
Repository	folder
Issues	track bugs, request features, discuss
Branch	separate line of development
Fork	server-side copy of Repository
Pull request	proposal to merge changes

# Exercise 1

1. Create a **username** on <https://github.com>
2. Check if you belong to any **organization**
3. Create a new **public Repository** with your **username**
4. Go to your home page: <https://github.com/username>
5. Take a screenshot

# Writing style on GitHub: Markdown

# This is heading 1

## This is heading 2

`code`

```r

R code

```

# Writing style on GitHub: Markdown

**\*\*Bold\*\***     \_\_Bold\_\_

*\*Italics\**     \_Italics\_

~~~Strikeout~~~

> quote

# Writing style on GitHub: Markdown

This is a list

- Item 1
- Item 2
- Item 3

This is a numbered list

1. Item 1
2. Item 2
3. Item 3



# Writing style on GitHub: Markdown

This is a task list

- [ ] Item 1
- [ ] Item 2
- [ ] Item 3

[this is a link](<https://github.com>)

# Writing style on GitHub: Markdown

## Making a table

```
| Column 1 | Column 2 | Column 3 |  
|-----|-----|-----|  
| row 1 a | row 1 b | row 1 c |  
| row 2 a | row 2 b | row 2 c |
```

# Exercise 2

1. Create a file: **README.md** in your username **Repository**
2. Write your introduction into sections: Background, Research project, Affiliation, Education
3. In the Education section, make a table of your studies with university and year
4. Save your **README.md** file with a proper commit message
5. Go to your home page: <https://github.com/username>
6. Compare it with previously taken screenshot

# Exercise 3

1. Create an **issue** in your **repository**:  
<https://github.com/username/repository/issues/new>
2. Give it a title: updating **README** file
3. Write a description: what else you would like to update in the **README** file
4. **Assign** it to yourself, add relevant **labels** and type

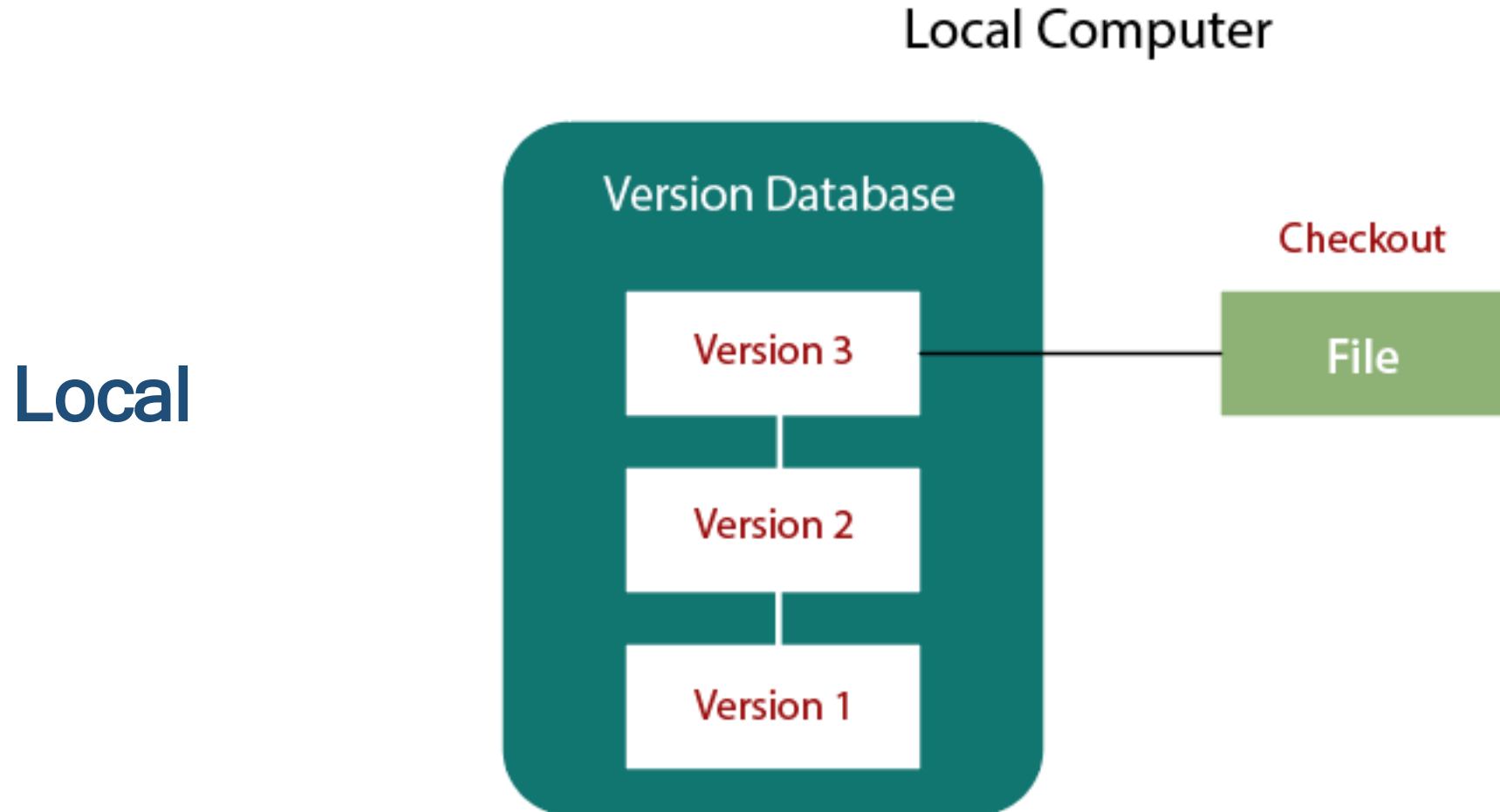
# Exercise 4

1. Go to [https://github.com/DQBM-SIB/intro\\_git\\_github](https://github.com/DQBM-SIB/intro_git_github)
2. Fork this repository under your username
3. Create a branch with your first name
4. In your named branch, edit README file of this repo and add your GitHub username
5. Create a pull request for original repository DQBM-SIB/intro\_git\_github in main branch with appropriate message
6. After pull requests are merged, check different commit messages

# Exercise 5

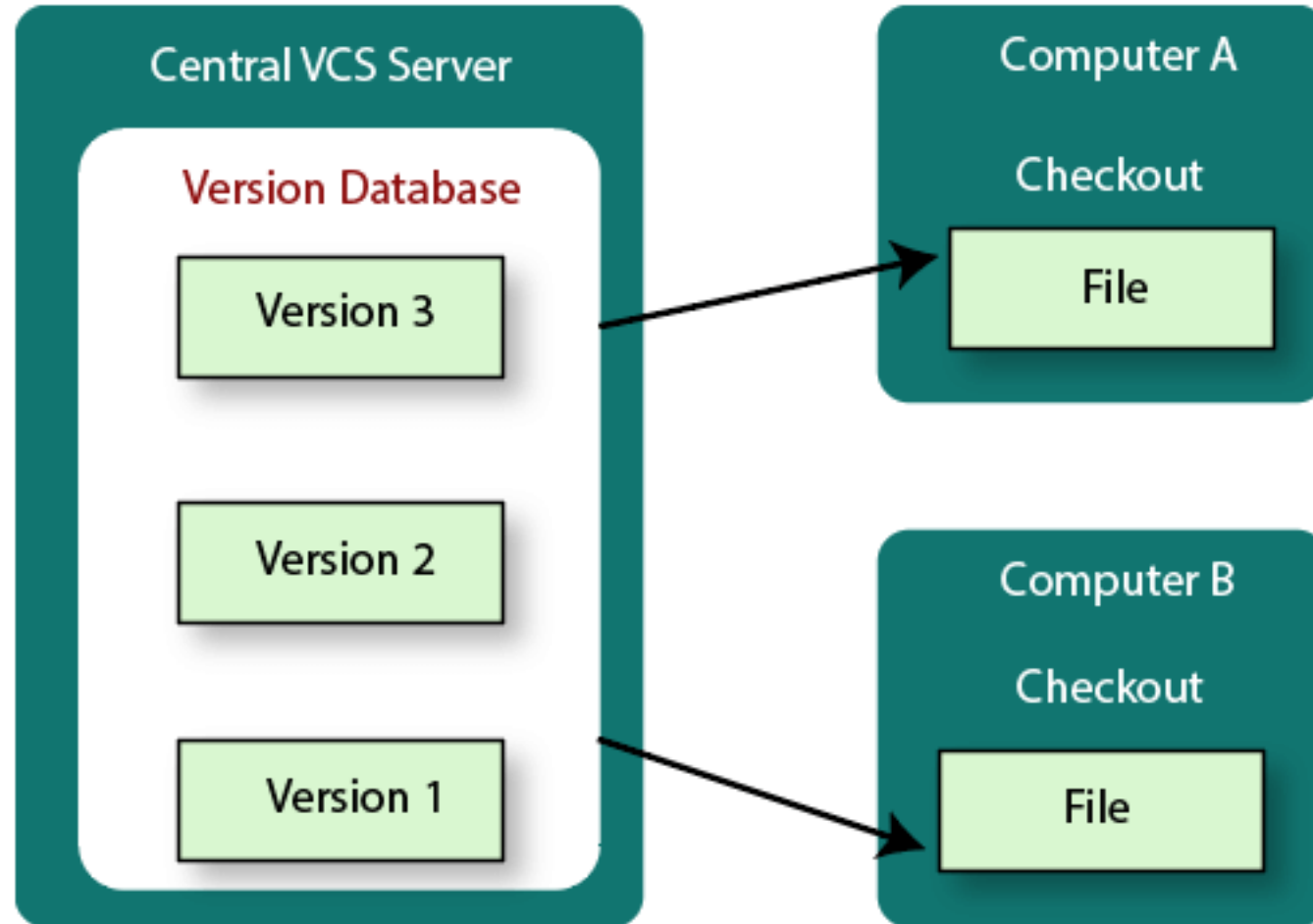
1. Go to [https://github.com/DQBM-SIB/intro\\_git\\_github](https://github.com/DQBM-SIB/intro_git_github)
2. Click on Projects
3. Execute three projects in groups

# Overview of Version Control System



# Overview of Version Control System

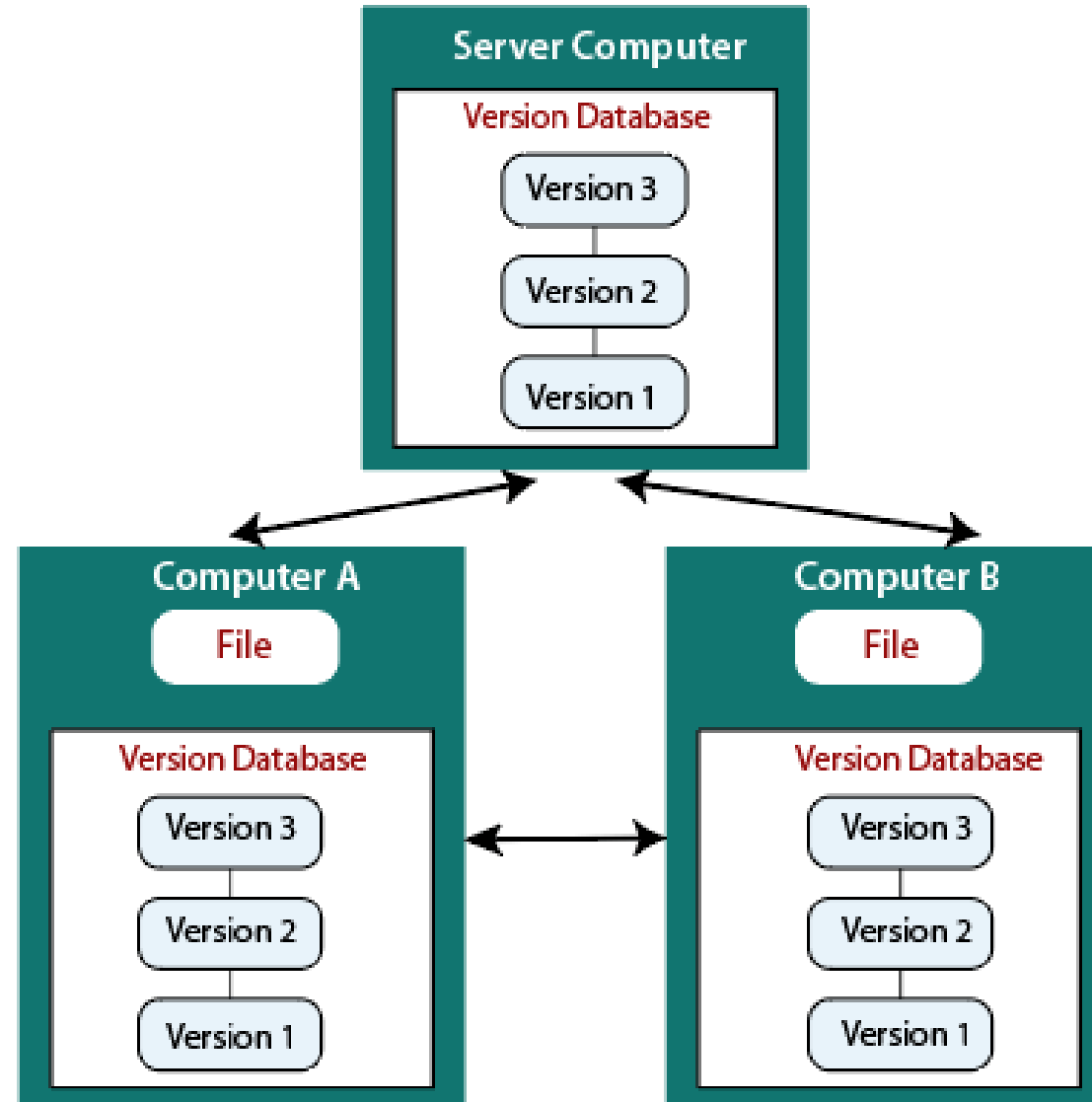
Central





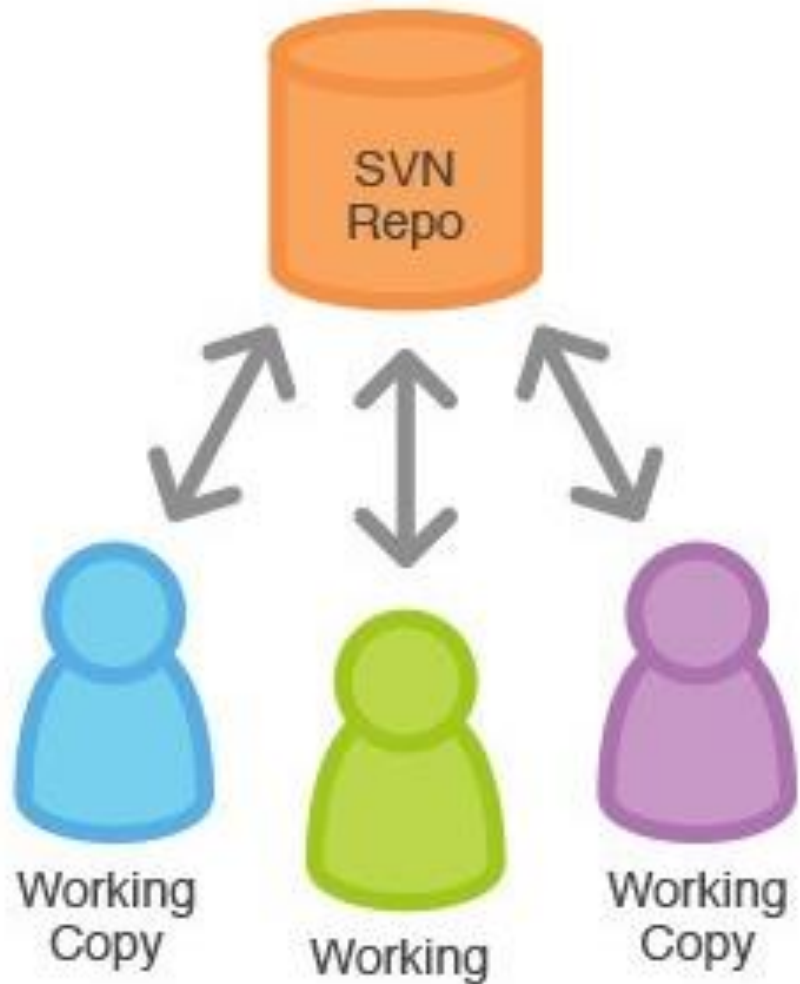
# Overview of Version Control System

Distributed

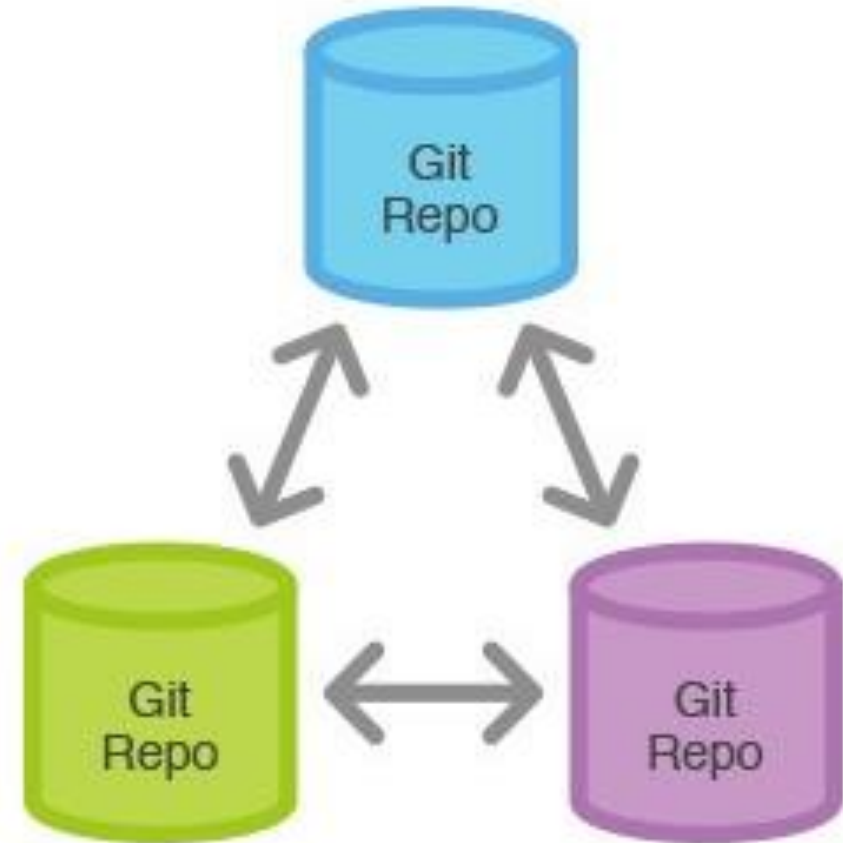


# The two main VCS

Central-Repo-to-Working-Copy  
Collaboration



Repo-to-Repo  
Collaboration



# Git is more trending than svn

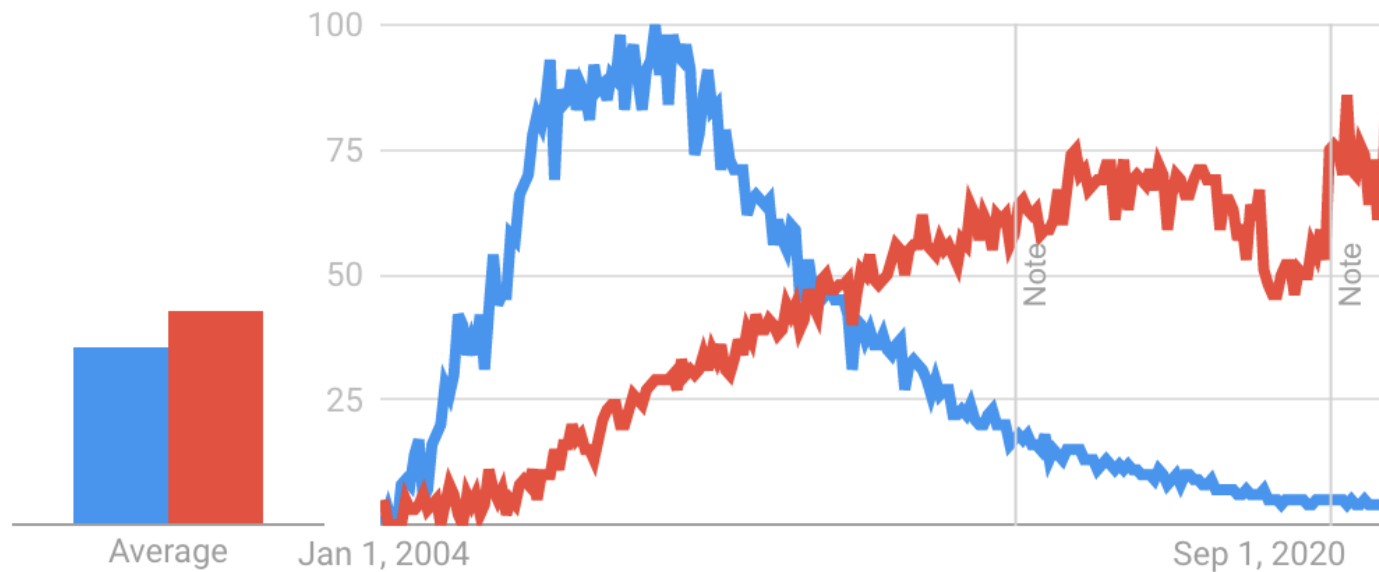
svn: 2000

git: 2005

Interest over time

Google Trends

● svn ● git



Worldwide. 1/1/04 - 2/9/23. Web Search.

# Quiz 1-3

# Getting started with git

Is git installed in your system?

```
$ git config --global user.name "Your Name"
```

```
$ git config --global user.email "Your email  
address"
```

# git commands

|                  |                  |                   |                    |                  |                 |                    |
|------------------|------------------|-------------------|--------------------|------------------|-----------------|--------------------|
| add              | citool           | diff-tree         | index-pack         | multi-pack-index | repack          | status             |
| am               | clean            | fast-export       | init               | mv               | replace         | strip-space        |
| annex            | clone            | fast-import       | instaweb           | name-rev         | request-pull    | submodule          |
| annex-shell      | column           | fat               | interpret-trailers | notes            | rerere          | svn                |
| annotate         | commit           | fetch             | lfs                | p4               | reset           | switch             |
| apply            | commit-graph     | fetch-pack        | log                | pack-objects     | restore         | symbolic-ref       |
| archimport       | commit-tree      | filter-branch     | ls-files           | pack-redundant   | revert          | tag                |
| archive          | config           | fmt-merge-msg     | ls-remote          | pack-refs        | rev-list        | unpack-file        |
| bisect           | count-objects    | for-each-ref      | ls-tree            | patch-id         | rev-parse       | unpack-objects     |
| blame            | credential       | format-patch      | mailinfo           | prune            | rm              | update-index       |
| branch           | credential-cache | fsck              | mailsplit          | prune-packed     | send-email      | update-ref         |
| bundle           | credential-store | gc                | media              | pull             | send-pack       | update-server-info |
| cat-file         | cvsexportcommit  | get-tar-commit-id | merge              | push             | sh-i18n         | var                |
| check-attr       | cvssimport       | gitk              | merge-base         | quiltimport      | shortlog        | verify-commit      |
| check-ignore     | cvssserver       | gitweb            | merge-file         | range-diff       | show            | verify-pack        |
| check-mailmap    | daemon           | grep              | merge-index        | read-tree        | show-branch     | verify-tag         |
| checkout         | describe         | gui               | merge-one-file     | rebase           | show-index      | whatchanged        |
| checkout-index   | diff             | hash-object       | mergetool          | reflog           | show-ref        | worktree           |
| check-ref-format | diff-files       | help              | merge-tree         | remote           | sh-setup        | write-tree         |
| cherry           | diff-index       | http-backend      | mktag              | remote-gcrypt    | sparse-checkout |                    |
| cherry-pick      | difftool         | imap-send         | mktree             | remote-tor-annex | stash           |                    |

# git commands

|                  |                  |                   |                    |                  |                 |                    |
|------------------|------------------|-------------------|--------------------|------------------|-----------------|--------------------|
| <b>add</b>       | citool           | diff-tree         | index-pack         | multi-pack-index | repack          | <b>status</b>      |
| am               | clean            | fast-export       | <b>init</b>        | mv               | replace         | stripspace         |
| annex            | <b>clone</b>     | fast-import       | instaweb           | name-rev         | request-pull    | submodule          |
| annex-shell      | column           | fat               | interpret-trailers | notes            | rerere          | svn                |
| annotate         | <b>commit</b>    | fetch             | lfs                | p4               | <b>reset</b>    | switch             |
| apply            | commit-graph     | fetch-pack        | <b>log</b>         | pack-objects     | restore         | symbolic-ref       |
| archimport       | commit-tree      | filter-branch     | ls-files           | pack-redundant   | revert          | tag                |
| archive          | config           | fmt-merge-msg     | ls-remote          | pack-refs        | rev-list        | unpack-file        |
| bisect           | count-objects    | for-each-ref      | ls-tree            | patch-id         | rev-parse       | unpack-objects     |
| blame            | credential       | format-patch      | mailinfo           | prune            | rm              | update-index       |
| <b>branch</b>    | credential-cache | fsck              | mailsplit          | prune-packed     | send-email      | update-ref         |
| bundle           | credential-store | gc                | media              | <b>pull</b>      | send-pack       | update-server-info |
| cat-file         | cvsexportcommit  | get-tar-commit-id | <b>merge</b>       | <b>push</b>      | sh-i18n         | var                |
| check-attr       | cvsimport        | gitk              | merge-base         | quiltimport      | shortlog        | verify-commit      |
| check-ignore     | cvsserver        | gitweb            | merge-file         | range-diff       | show            | verify-pack        |
| check-mailmap    | daemon           | grep              | merge-index        | read-tree        | show-branch     | verify-tag         |
| <b>checkout</b>  | describe         | gui               | merge-one-file     | rebase           | show-index      | whatchanged        |
| checkout-index   | <b>diff</b>      | hash-object       | mergetool          | reflog           | show-ref        | worktree           |
| check-ref-format | diff-files       | <b>help</b>       | merge-tree         | <b>remote</b>    | sh-setup        | write-tree         |
| cherry           | diff-index       | http-backend      | mktag              | remote-gcrypt    | sparse-checkout |                    |
| cherry-pick      | difftool         | imap-send         | mktree             | remote-tor-annex | stash           |                    |

# Creating a Repository

Make a directory

```
$ cd ~/Downloads
```

```
$ mkdir planets
```

```
$ cd planets
```



# Creating a Repository

Initialize git

```
$ git init
```

# Creating a Repository

View git repository

```
$ ls -a
```

# Tracking changes in a git repository

Make a file with planet names

```
$ nano planets.txt
```

# Tracking changes in a git repository

Add planet names

Mercury

Venus

Earth

Mars

Jupiter

Saturn

Uranus

Neptune

# Tracking changes in a git repository

Close the file

`ctrl + x --> press Y --> press Enter`

# Tracking changes in a git repository

Check status of the repository

```
$ git status
```

# Tracking changes in a git repository

Add changes to git

```
$ git add --all
```

# Tracking changes in a git repository

Check status of the repository again

```
$ git status
```



# Tracking changes in a git repository

Commit changes to git

```
$ git commit -m "added planets list"
```

# Tracking changes in a git repository

Check status of the repository again

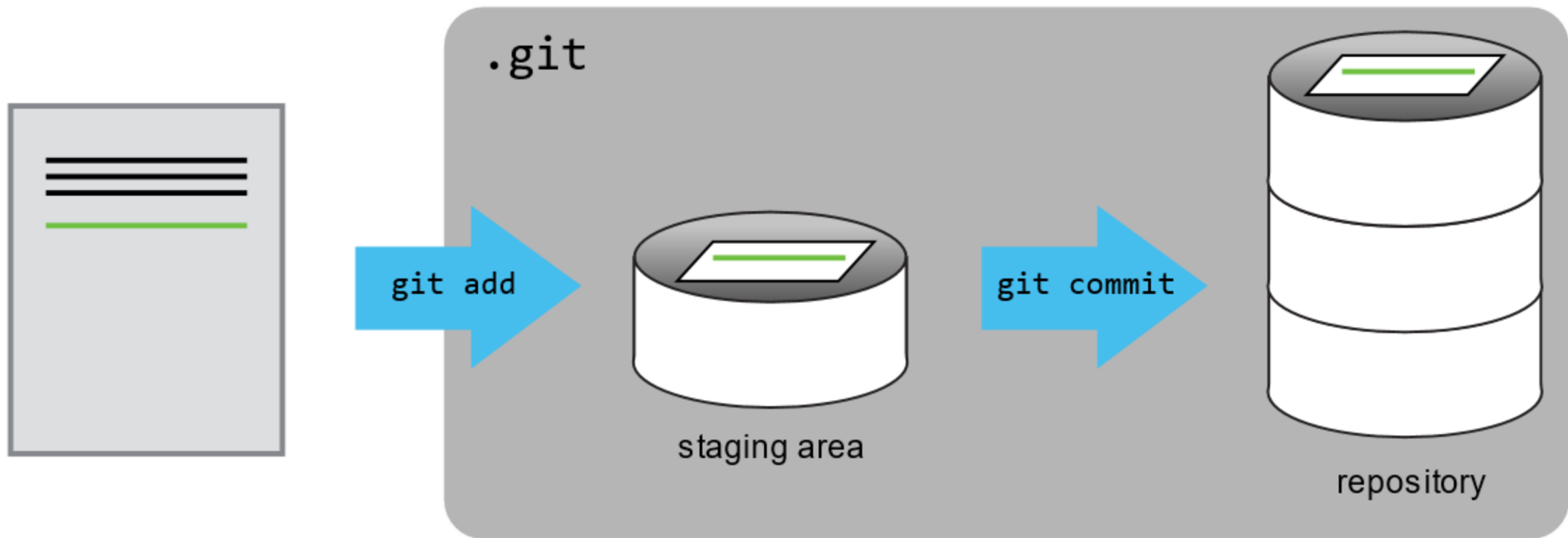
```
$ git status
```

# Tracking changes in a git repository

Check log of your activity

```
$ git log
```

# Recap

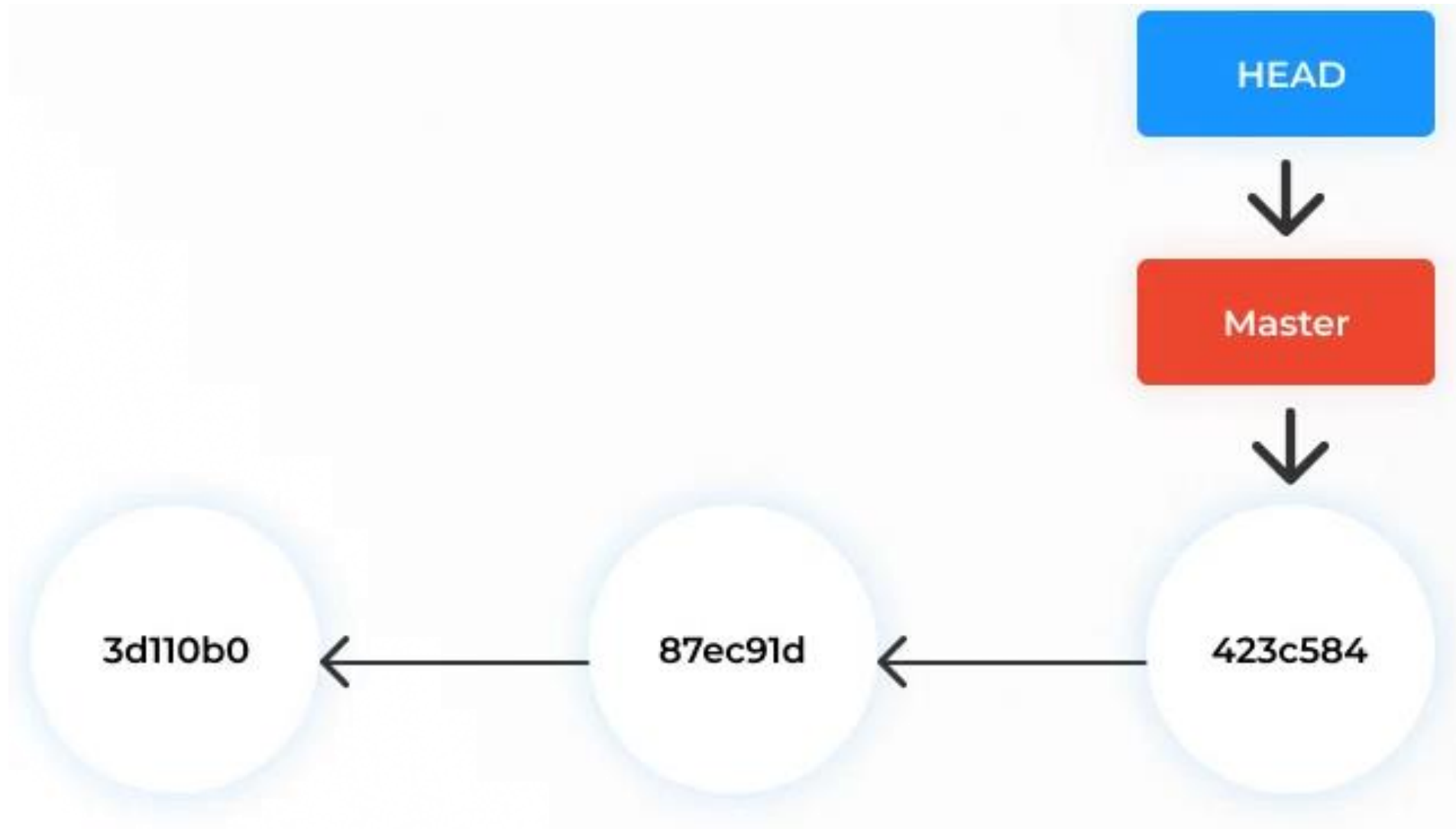


# Exercise 6

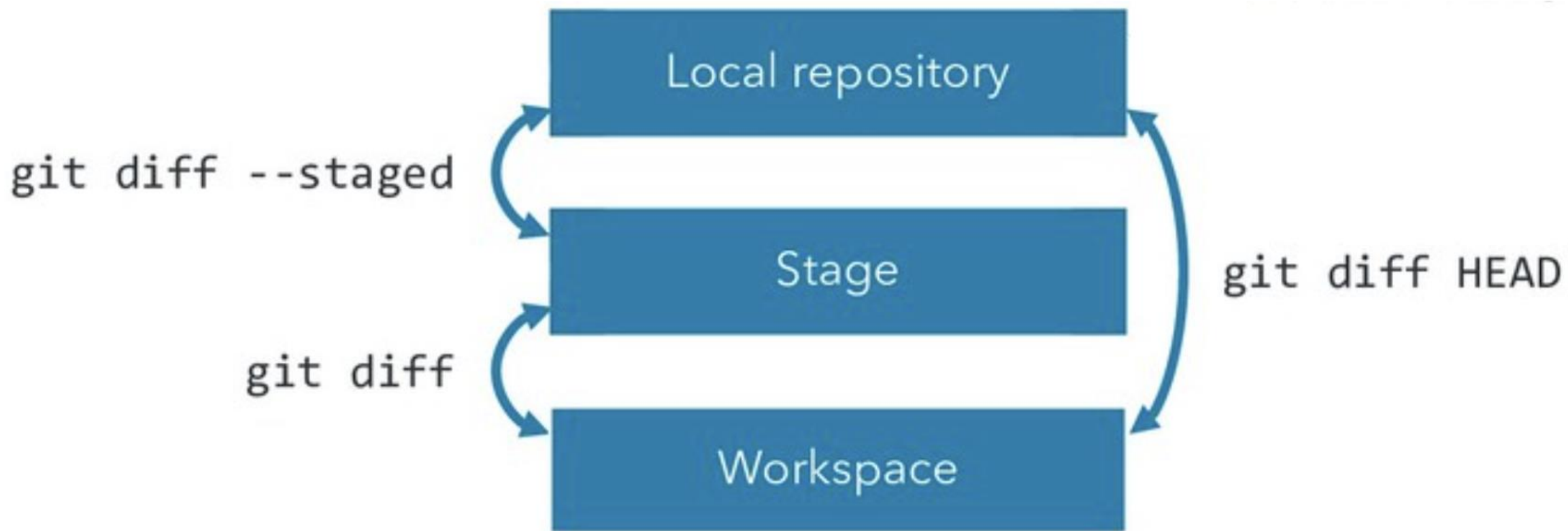
1. Make a directory, naming it as your GitHub user name 1 (ex.: dktanwar1)
2. Initiate this directory as a git repository
3. Make a file: README.md
4. Write a brief introduction about yourself
5. Add the file to git
6. commit the changes with a meaningful message
7. Make a new file: education.md
8. List your education history
9. Add the file to git and commit the changes with a meaningful message
10. Copy content of education.md at the end of README.md
11. Add the changes to git and commit the changes with a message
12. Delete the file education.md
13. Add the changes to git and commit the changes with a message
14. Check the log of your git activity

# Quiz 4-9

# HEAD



# git diff



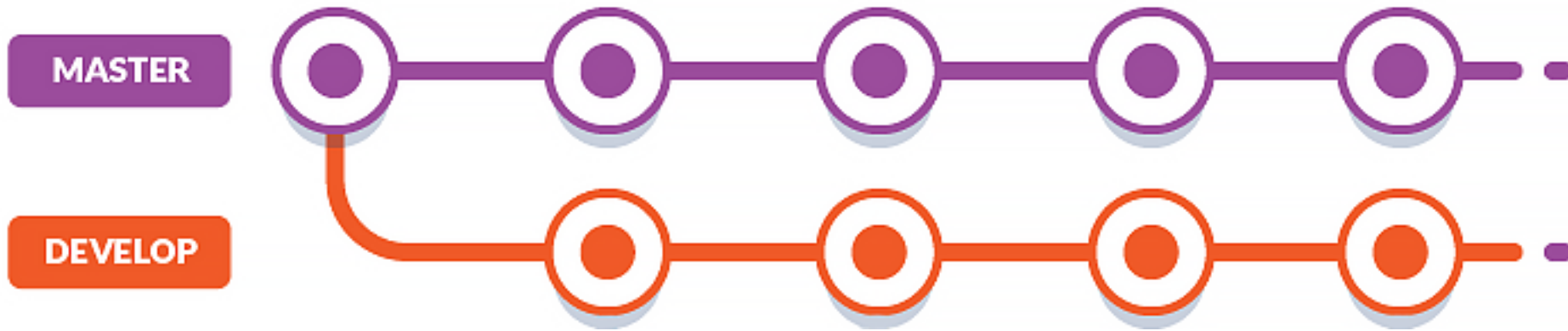


# git diff

**Diffing is a function that takes two input data sets and outputs the changes between them**

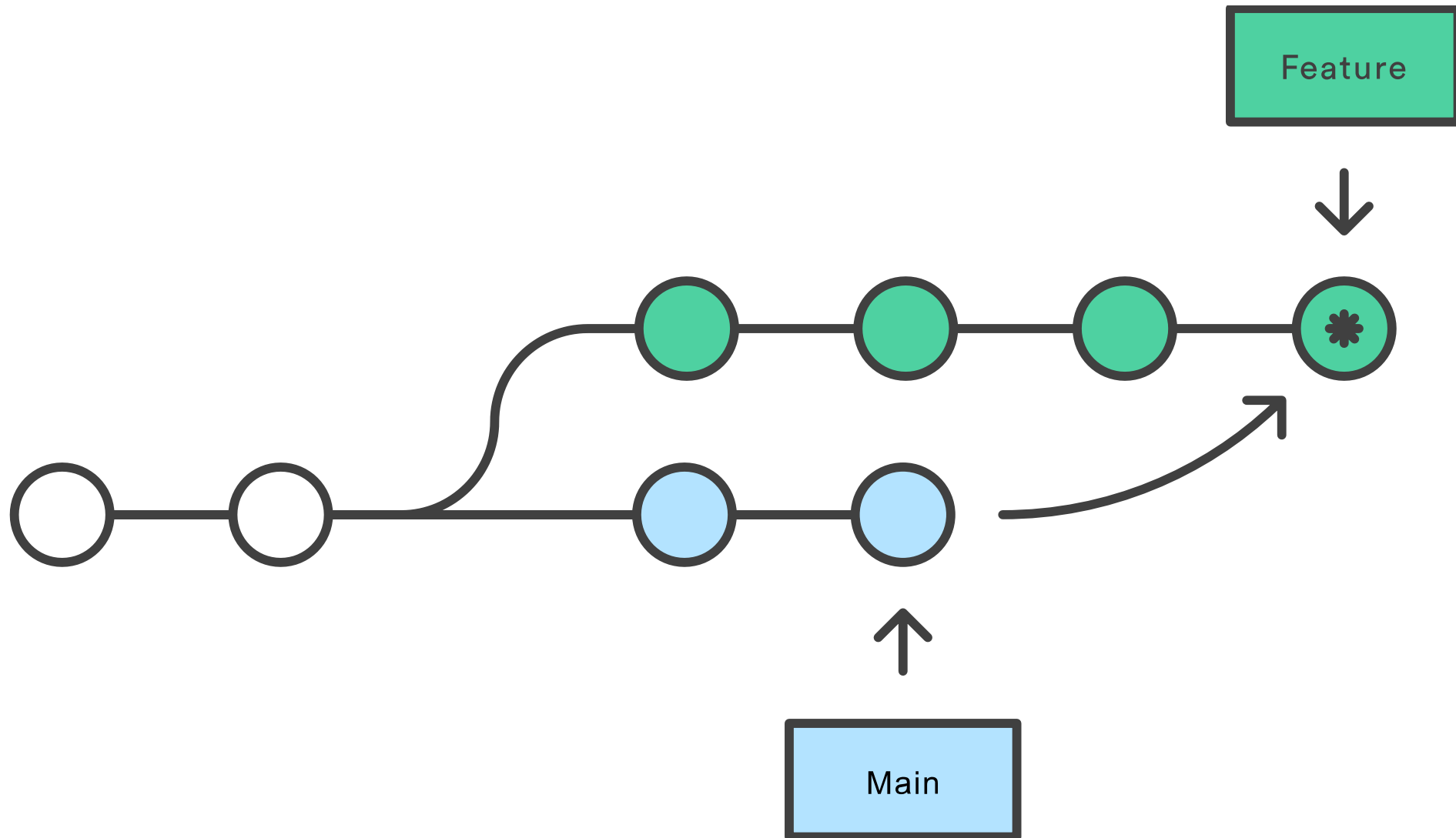
1. Open README.md file and add your hobbies
2. Run `git diff`
3. Run `git diff --staged`
4. Run `git diff` on 2 different commits

# git branch and git checkout

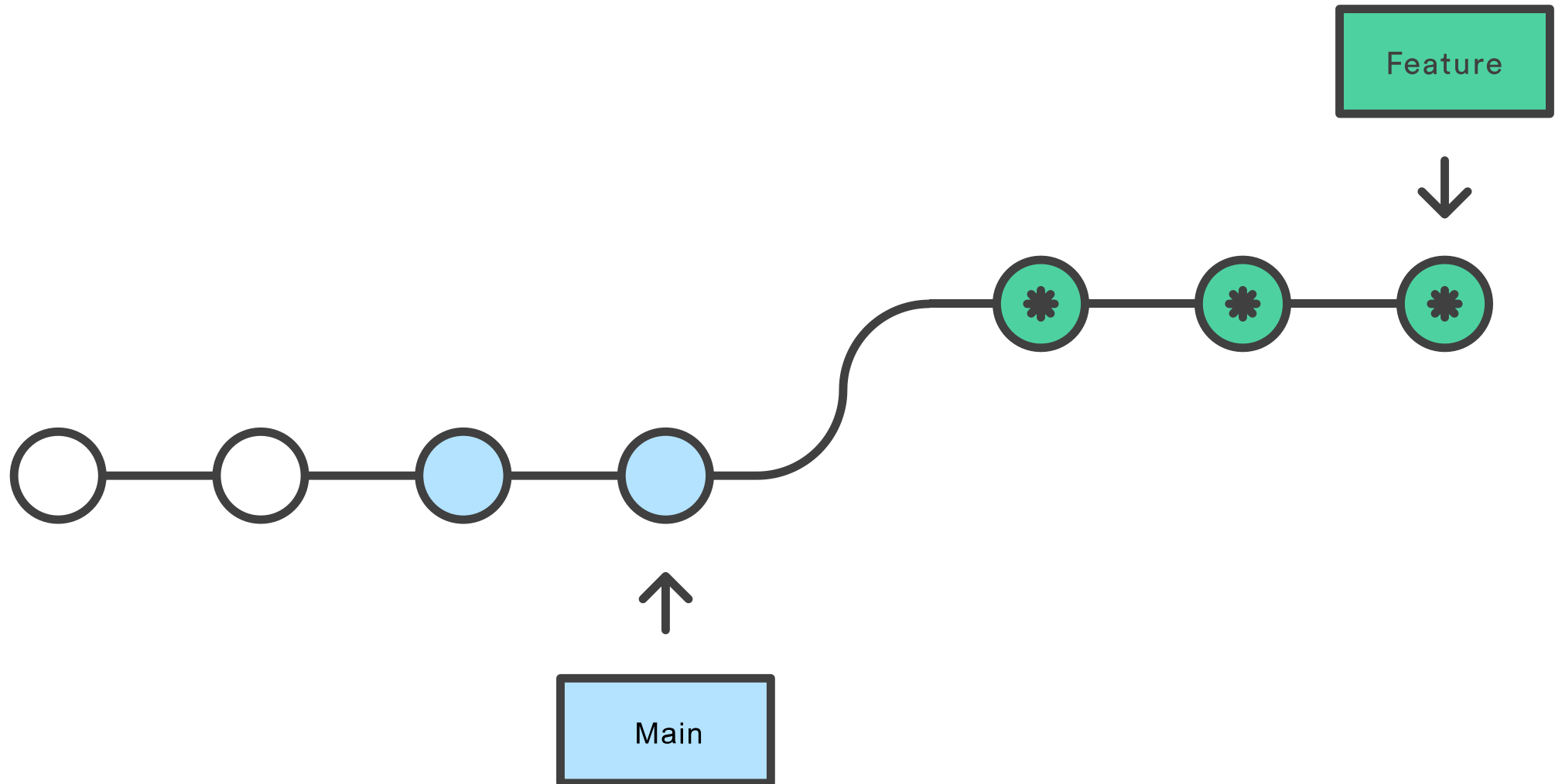


1. Run `git branch` to check on which branch you are in.
2. Make a `new branch` called dev
3. Go to the `dev branch`

# git checkout and git merge



# git checkout and git merge



# git checkout and git merge

1. Make a [new file hobbies.md](#), add it, and commit it
2. Go to the [main branch](#)
3. Check for the file hobbies.md
4. Merge [dev with main](#)

# Exploring git history

How do you check history of your commits?

```
$ git log
```

```
$ git log --oneline
```

Go to the content of specific commit.

```
$ git checkout a499ea4
```

# Ignore files in git

1. Make a folder called test
2. In the test folder, creates files: test1.txt, test2.txt, test3.txt
3. Check status
4. Make a file, [.gitignore](#) and write test/
5. Check [status](#)

# git reset

1. Make a file called animal.md (`touch animal.md`)
2. Check `status`
3. Add file (`git add animal.md`)
4. Remove file from adding (`git reset animal.md`)
5. Add file again, and then `commit` it
6. Remove file from commit (`git reset --soft HEAD~1`)
7. Repeat step 5
8. Remove file completely (`git reset --hard HEAD~1`)



# Quiz 10-16

# Exercise 7

1. Go to the directory of your GitHub user name1 (ex.: dktanwar1)
2. Open README.md file and add links to your professional profiles (LinkedIn?)
3. Use `diff` command to compare current state of repo with `HEAD`
4. `Add` the file to git and `compare` changes of local repo (git database) with `staged` changes
5. `Commit` the change.
6. `Compare` the first commit of your git repo with the last commit
7. Make a `new branch` and give it your first name
8. Check `how many branches` you have and then `go to your name branch`
9. Make a `new` file: test.md and write your favorite color in it
10. `Add and commit` the changes with a message
11. Go back to main branch and check if test.md file is there. If not, `merge` the your name branch with main branch
12. Make a new file: test2.md and `ignore` this file from git

**Thank you!**