neue fische

School and Pool for Digital Talent

oh oh...
git

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

What is git?

- Open-source project
- Distributed Version Control System
- can be used completely offline

What is GitHub?

- Collaborate effectively online based on git (see below)
- Open-source community and contribution
- Follow others, find and star interesting repositories
- Show others what you did, invite to collaborate and collaborate yourself
- Demonstrate that you are actively working on projects
- Learn from code

Your business card as data scientist

If you want to learn more about git & GitHub, you can watch this video.

»neue fische
School and Pool for Digital Talent

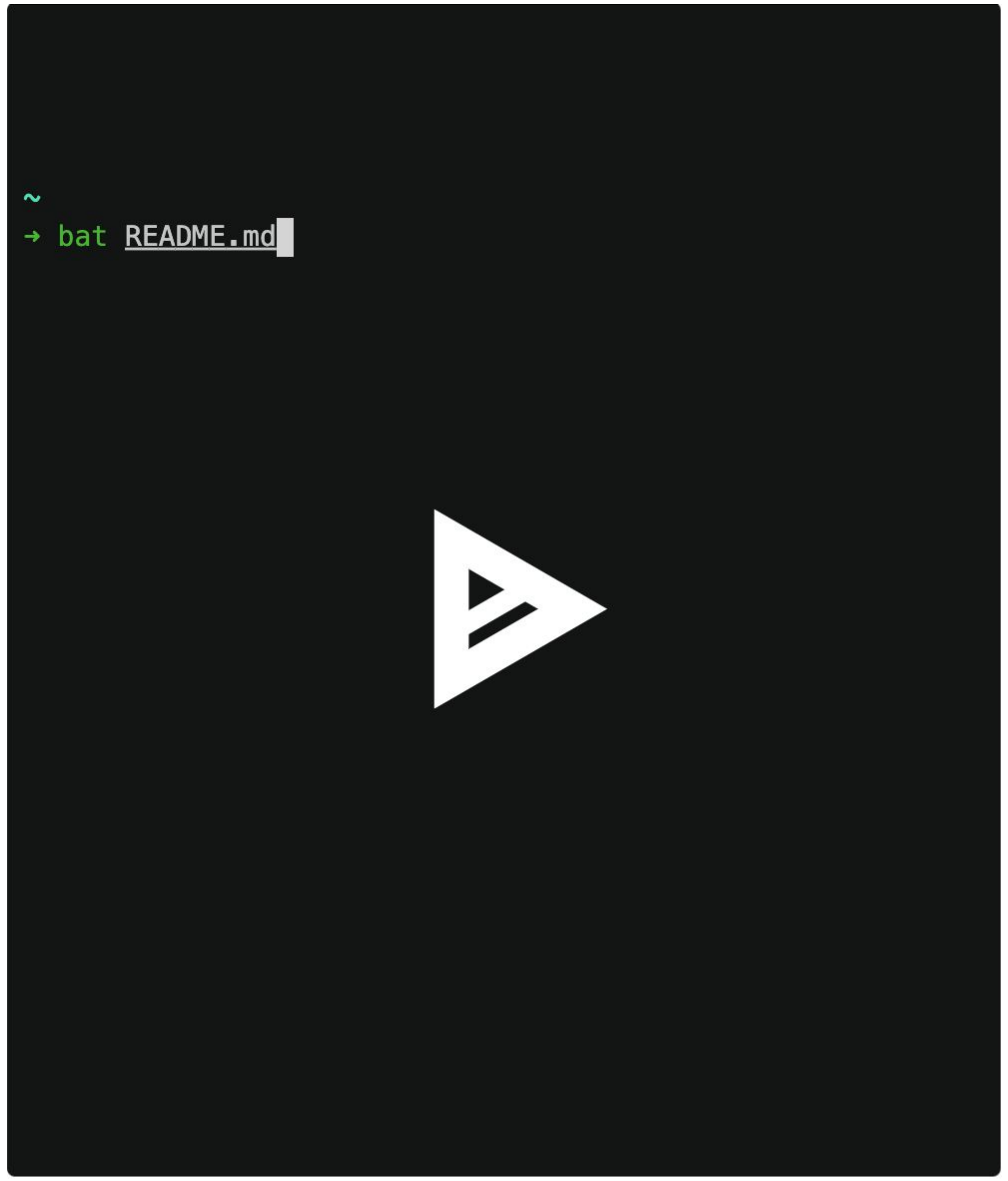
oh oh... git

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

Setting up Git

- Setting the user name globally:
  ```
  git config --global user.name “your name”
  ```
- Setting the email address globally:
  ```
  git config --global user.email “your email”
  ```
- Setting the pull behaviour:
  ```
  git config --global pull.ff only
  ```
- Change default branch name:
  ```
  git config --global init.defaultBranch main
  ```
- Show configuration settings:
  ```
  git config --list
  ```



```
~
➜ bat README.md
```

oh oh...
git

neue fische
School and Pool for Digital Talent

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

How to collaborate on our Repos

Order to work together on a neuefische repository (in brackets is which person should do the step)

- **Fork the repo** under your github account (*person1*)
- **Grant** person2 **access** to your new repo (*person1*)
- **clone** the repo (*both*)
- **change into** your repo (*both*)
- **create** own **branch** with your name (*both*)
- setup your venv (often written in README how to do so) (*both*)
- **start working** (*both*)

oh oh... git

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

How to switch branches

How to…

- **create** a new branch

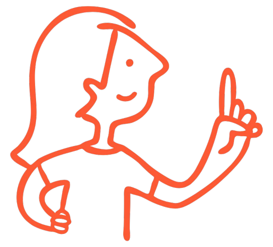
  ```
  git switch -c Protocol_day20
  ```

- **switch** to an existing branch

  ```
  git switch main
  ```

- **list** existing branches (carefull, vim trap! → exit with q)

  ```
  git branch -l
  ```



** If you get an error here - read it carefully. Maybe the solution is already suggested?*

How to commit changes

How to…

- **Update** your branch

```
git pull
```

- Figure out what all **changes**

```
git status
```

- **Add** changes to commit

```
git add FILENAME
```

- **commit** changes (to the currently active branch!)

```
git commit -m ‘Added gif to protocol’
```

- **Upload** changes to github *

```
git push
```

oh oh... git

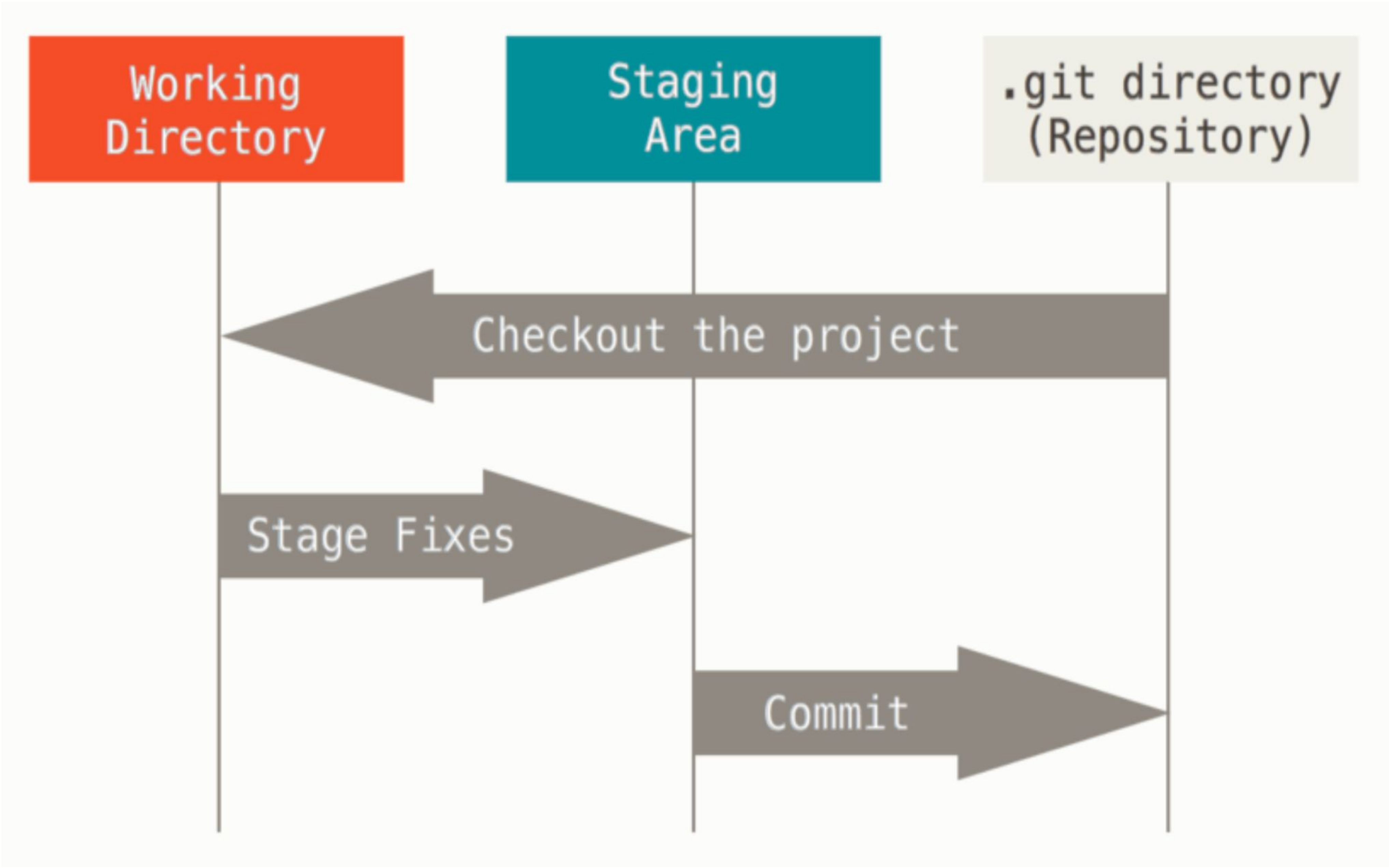
neue fische
School and Pool for Digital Talent

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

The Three Stages

- **modified:**
  file was changed but not committed yet
- **staged:**
  file was marked in its current version to go into the next commit
- **committed:**
  file/data is stored in local database



The Workflow

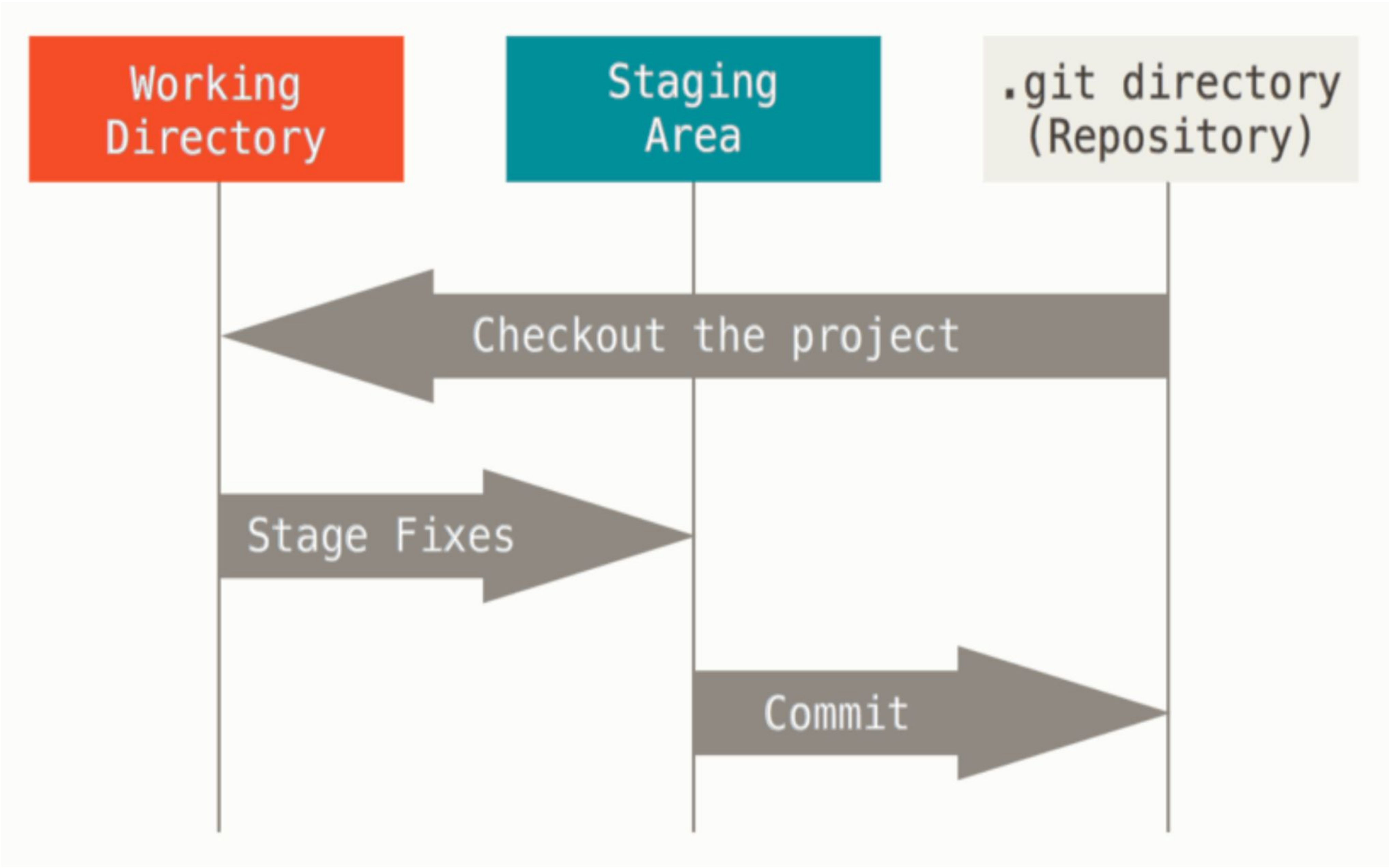
1. Modify files in working tree/directory

2. Selectively stage those changes that should be part of the next commit

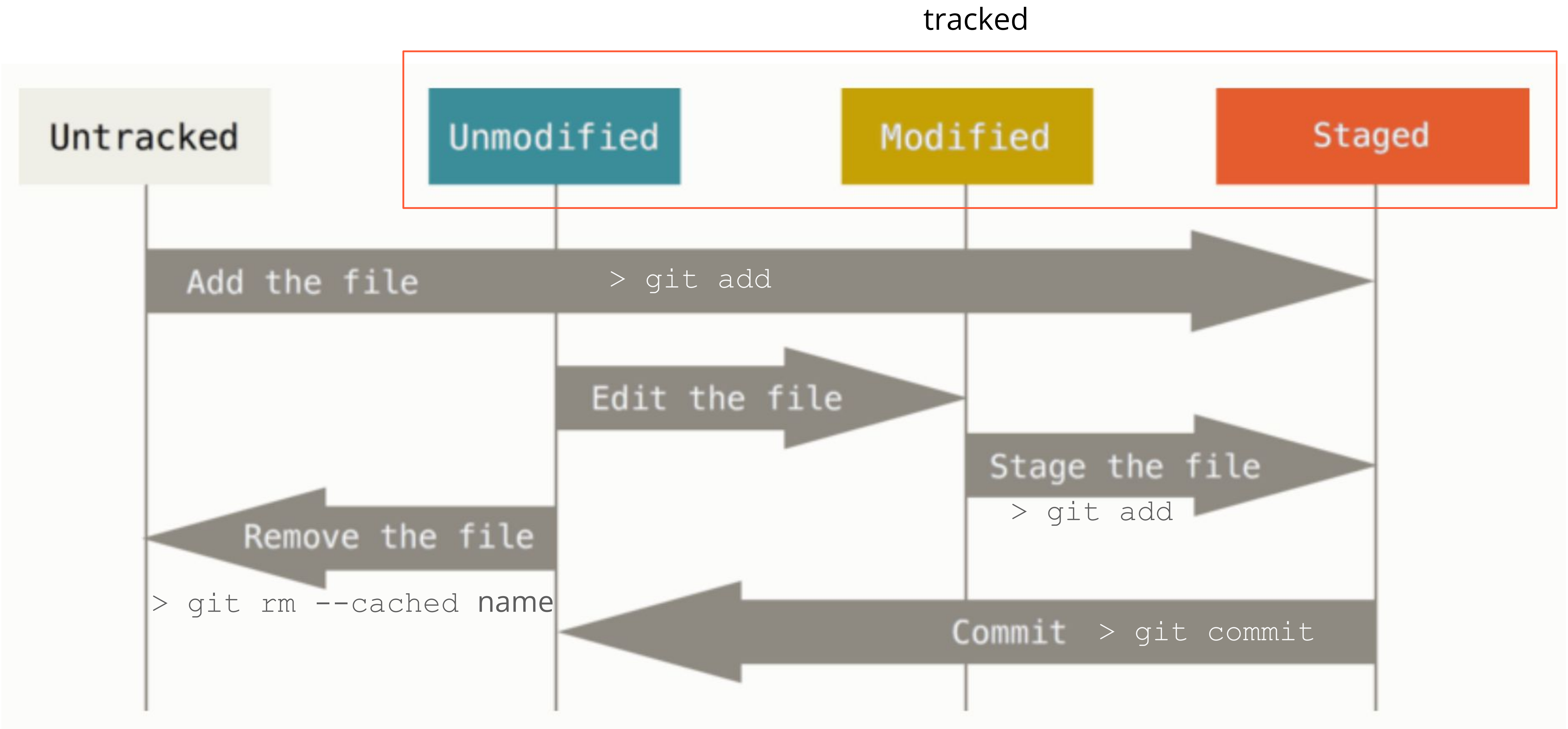
   ```
   git add <filename1> <filename2>
   ```

3. Commit the files as they are in the staging area and store that **snapshot** to your git repository

   ```
   git commit -m “message”
   ```



The Status of your Files



oh oh...
git

→ Configure git for terminal

→ Fork and clone
→ Branch and commit
→ Stages of git
→ From git to github and back

Resources

Git Basics - https://www.youtube.com/watch?v=_OZVJpLHUaI

Reference Manual - https://www.git-scm.com/docs

Pro Git Book - https://www.git-scm.com/book/en/v2

and the most common things
you will need will be explained
in this link:

oh shit, git.

neue fische
School and Pool for Digital Talent

Exercises!

→

Exercise 1

Using Git Locally

1. In your terminal: create a new folder and navigate into it
2. Turn it into a git repository with `git init` and create a file (can be any file type)
3. Explore the different stages and commands:
   a. Track a file
   b. Stage and commit a file
   c. Modify a commited file and commit it again
   d. Stage a file and unstage it again
   e. Create a new file, track it and untrack it again

Between every command run `git status`. It will not only show you the effects of your command but also suggest all the commands you will need to unstage or untrack a file.

Exercise 2

Using Git with Remote Repository

1. Create and clone new repo on Github
2. Create new branch
3. Commit and push changes to new branch
4. Open a pull request (pr)
5. Review code and approve pull request
6. Merge branch into main / Deal with merge conflicts (if one occurs)
7. Synchronize branches
8. Open a pull request, approve and merge

Exercise 2

Let’s work in pairs.

1. New Repo
   - ONE: creates new public repo on github with README.md, .gitignore and Licence (MIT)
   - Invites partner as collaborator
   - BOTH: Clone it to your machine

2. New Branches
   - BOTH: create and switch to a new branch locally: `git switch -c` <new_branch>

Exercise 2

3. Commit Changes (BOTH)
    - Create a new file and add, commit and push your change
    - To push a local branch to a new remote branch use:

      ```
      git push --set-upstream origin <branch_name>
      ```

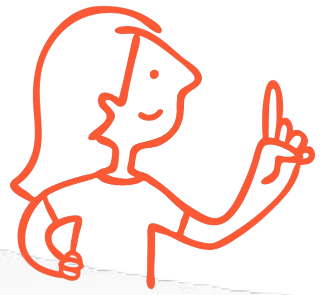
4. Open Pull Request
    - ONE: opens pull request and adds partner as reviewer

5. Code Review
    - TWO: reviews changes and approves pull request

Exercise 2

6. Merge Branch
   - ONE: merges approved pull request into main branch and deletes remote branch

7. Syncing Branches Locally
   - Now branch of TWO will be behind main branch
     → switch to main branch: `git switch main`
     → pull changes: `git pull`
     → switch to your branch: `git switch <branch_name>`
     → merge new changes from main branch in your branch: `git merge main`
     → add commit message (vim will open automatically) if necessary and push again



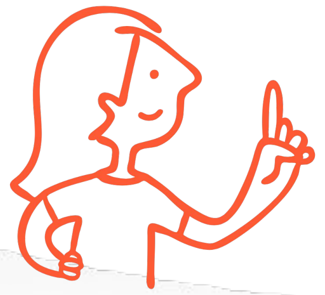
Merging should be done by the person who did the pull request!

Exercise 2

8. Second pull request
    - TWO: opens pull request for his branch
    - ONE: reviews changes and approves pull request
    - TWO: merges approved pull request into main branch and deletes remote branch

Last but not least…

- BOTH: switch to main branch locally and pull changes



Merging should be done by the person who did the pull request!