

 **ai2-education-fiep-turma-4 / git-katas** Publicforked from [eficode-academy/git-katas](#)**Code**


Pull requests

Actions

Wiki

Security

Insights

 master ▾

...

**git-katas / basic-staging /**

This branch is 1 commit ahead of eficode-academy:master.

 **Contribute** ▾**JKrag** ...on 15 Apr 2020 

..



README.md

2 years ago



setup.ps1

3 years ago



setup.sh

2 years ago

## Git Kata: Basic Staging

This kata will examine the staging area of git.

In git we are working with three different areas:

- The working directory where you are making your changes
- The staging area where all changes you have added through `git add` will stay
- The repository where every commit ends up, making your history. To put your staged changes in here you issue the `git commit` command.

A file can have changes both in the working directory and staging area at the same time. These changes do not have to be the same.

We will also work with `git reset` to reset the staged changes of a file, and `git checkout` to return a file to a previous state.

## Setup:

---

1. Run `source setup.sh` (or `.\setup.ps1` in PowerShell)

## The task

---

You live in your own repository. There is a file called `file.txt` .

1. What's the content of `file.txt` ?
2. Overwrite the content in `file.txt` : `echo 2 > file.txt` to change the state of your file in the working directory (or `sc file.txt '2'` in PowerShell)
3. What does `git diff` tell you?
4. What does `git diff --staged` tell you? why is this blank?
5. Run `git add file.txt` to stage your changes from the working directory.
6. What does `git diff` tell you?
7. What does `git diff --staged` tell you?
8. Overwrite the content in `file.txt` : `echo 3 > file.txt` to change the state of your file in the working directory (or `sc file.txt '3'` in PowerShell).
9. What does `git diff` tell you?
10. What does `git diff --staged` tell you?
11. Explain what is happening

### README.md

---

10. Run `git reset HEAD file.txt` to unstage the change
14. What does `git status` tell you now?
15. Stage the change and make a commit
16. What does the log look like?
17. Overwrite the content in `file.txt` : `echo 4 > file.txt` (or `sc file.txt '4'` in PowerShell)
18. What is the content of `file.txt` ?
19. What does `git status` tell us?
20. Run `git checkout file.txt`
21. What is the content of `file.txt` ?
22. What does `git status` tell us?

## Useful commands

---

- `git add`
- `git commit`
- `git commit -m "My lazy short commit message"`

- `git reset`
- `git checkout`
- `git log`
- `git log -n 5`
- `git log --oneline`
- `git log --oneline --graph`
- `git reset HEAD`
- `git checkout`

## Aliases

---

You can set up aliases as such: `git config --global alias.lol 'log --oneline --graph --all'` This might be useful to you.

---