



(/)

Curriculum

SE Foundations ^

Average: 108.76% v

0x03. Git

Git

Code versioning

Github

By: Guillaume

Weight: 1

 Project over - took place from Feb 7, 2023 6:00 AM to Feb 8, 2023 6:00 AM☒ An auto review will be launched at the deadline

In a nutshell...

- **Auto QA review:** 26.0/26 mandatory & 11.0/11 optional
- **Altogether: 200.0%**
 - Mandatory: 100.0%
 - Optional: 100.0%
 - Calculation: $100.0\% + (100.0\% * 100.0\%) == 200.0\%$

Concepts

For this project, we expect you to look at these concepts:

- Authenticating Git (/concepts/100035)
- Struggling with the sandbox? Try this: Using Docker & WSL on your local host (/concepts/100039)
- Source code management (/concepts/22)
- Right-engineering, right-documenting (/concepts/6)
- Git and Github cheat sheet - Everything in less than 30 seconds (/concepts/57)

Resources

Read or watch:

Help

- Resources to learn Git (/rltoken/fq9RPIPS5ldYzOWTvwbdFQ)
- (/). • About READMEs (/rltoken/R7MeJ8alpK3JoVF8w24wiQ)
- How to write a Git commit message (/rltoken/FYsjjR-97Hk4NJtgqzWdtQ)

Resources for advanced tasks (Read only after finishing the mandatory tasks):

- Learning branching (/rltoken/tN8ZJ0yWubOP6jdcitrFw)
- Effective pull requests and other good practices for teams using GitHub (/rltoken/mjpQ9OCU0Dz-DFxZjASEJg)

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/jj0uPL9hiKF10KCH4u620A), **without the help of Google**:

General

- What is source code management
- What is Git
- What is GitHub
- What is the difference between Git and GitHub
- How to create a repository
- What is a README
- How to write good READMEs
- How to commit
- How to write helpful commit messages
- How to push code
- How to pull updates
- How to create a branch
- How to merge branches
- How to work as collaborators on a project
- Which files should and which files should not appear in your repo

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

- A README.md file at the root of the alx-zero_day repo, containing a description of the repository
- A README.md file, at the root of the folder of *this* project (i.e. 0x03-git), describing what this project is about



- **Do not use GitHub's web UI**, but the command line to perform the exercise (except for operations (/) that can not possibly be done any other way than through the web UI). You won't be able to perform many of the task requirements on the web UI, and you should start getting used to the command line for simple tasks because many complex tasks can only be done via the command line.
- Your answer files should only contain the command, and nothing else

More Info

Basic usage

At the end of this project you should be able to reproduce and understand these command lines:

```
$ git clone <repo>
$ touch test
$ git add test
$ git commit -m "Initial commit"
$ git push origin main
```

Quiz questions

Great! You've completed the quiz successfully! Keep going! ([Show quiz](#))

Tasks

0. Create and setup your Git and GitHub account

mandatory

Score: 100.0% (Checks completed: 100.0%)

Step 0 - Create an account on GitHub [if you do not have one already]

You will need a GitHub account for all your projects at ALX. If you do not already have a github.com account, you can create an account for free here (</rltoken/1vpH3ScWYjfgZD0J59jusA>)

Step 1 - Create a Personal Access Token on Github

To have access to your repositories and authenticate yourself, you need to create a Personal Access Token on Github.

You can follow this tutorial (</rltoken/coQUsDnam4suGaXSvinQxA>) to create a token.

Once it's created, you should have a token that looks like this:



Tokens you have generated that can be used to access the [GitHub API](#).

(/)

Make sure to copy your new personal access token now. You won't be able to see it again!

✓ ghp_IqIMN0ZH6z0wIEB4T9A2g4EHMy8Ji42q4HA5

Enable SSO ▾

Delete

Step 2 - Update your profile on the Intranet

Update your Intranet profile by adding your Github username here (/rltoken/9dthiwx9AyxxUzV_QPIRWQ)

If it's not done **the Checker won't be able to correct your work**

Social presence

Twitter username

Github username

Step 3 - Create your first repository

Using the graphic interface on the github website (/rltoken/1vpH3ScWYjfgZD0J59jusA), create your first repository.

- Name: alx-zero_day
- Description: I'm now a ALX Student, this is my first repository as a full-stack engineer
- Public repo
- No README, .gitignore, or license

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Owner *

Repository name *

MYSUPERUSERNAME ▾

/ alx-pre_course ✓

Great repository names are short and memorable. Need inspiration? How about [laughing-computing-machine?](#)

Description (optional)

☒ Public

Anyone on the internet can see this repository. You choose who can commit.

☐ Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more](#).

☐ Add .gitignore

Choose which files not to track from a list of templates. [Learn more](#).

☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more](#).

Create repository



Step 4 - Open the sandbox

On the intranet, just under the task, click on the button  and `run` to start the machine.

Once the container is started, click on  Webterm to open a shell where you can start work from.

Step 5 - Clone your repository

On the webterm of the sandbox, do the following:

- Clone your repository

```
root@896cf839cf9a:/# git clone https://{YOUR_PERSONAL_TOKEN}@github.com/{YOUR_USERNAME}/alx-zero_day.git
Cloning into 'alx-zero_day'...
warning: You appear to have cloned an empty repository.
```

Replace {YOUR_PERSONAL_TOKEN} with your token from step 1

Replace {YOUR_USERNAME} with your username from step 0 and 1

Pro-Tip: On windows, use CTRL + A + V to paste in the web terminal

Step 6 - Create the README.md and push the modifications

- Navigate to this new directory. Tips (/rltoken/qSP5HcBSSIL0U23PdIGKLw)

```
root@896cf839cf9a:/# cd alx-zero_day/
root@896cf839cf9a:/alx-zero_day#
```

- Create the file README.md with the content My first readme. Tips (/rltoken/_SSotigVtvVNThTVLJKZPQ)

```
root@896cf839cf9a:/alx-zero_day# echo 'My first readme' > README.md
root@896cf839cf9a:/alx-zero_day# cat README.md
My first readme
```

- Update your git identity

```
root@896cf839cf9a:/alx-pre_course# git config --global user.email "you@example.com"
root@896cf839cf9a:/alx-pre_course# git config --global user.name "Your Name"
```

- Add this new file to git, commit the change with this message "My first commit" and push to the remote server / origin



```
root@896cf839cf9a:/alx-zero_day# git add .
root@896cf839cf9a:/alx-zero_day# git commit -m 'My first commit'
[master (root-commit) 98eef93] My first commit
1 file changed, 1 insertion(+)
create mode 100644 README.md
root@896cf839cf9a:/alx-zero_day# git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 212 bytes | 212.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/{YOUR_USERNAME}/alx-zero_day.git
* [new branch]      master -> master
```

Good job!

You pushed your first file in your **first repository** of the **first task** of your **first ALX School project**.

You can now check your repository on GitHub to see if everything is good.

Repo:

- GitHub repository: alx-zero_day
- File: README.md

☒ Done![Help](#)[Check your code](#)[Get a sandbox](#)[QA Review](#)

1. Repo-session

mandatory

Score: 100.0% (Checks completed: 100.0%)

Create a new directory called 0x03-git in your alx-zero_day repo.

Make sure you include a not empty README.md in your directory:

- at the root of your repository alx-zero_day
- AND in the directory 0x03-git

And important part: **Make sure your commit and push your code to Github - otherwise the Checker will always fail.**

Repo:

- GitHub repository: alx-zero_day

☒ Done![Help](#)[Check your code](#)[Get a sandbox](#)[QA Review](#)

2. Coding fury road

mandatory

Score: 100.0% (Checks completed: 100.0%)

For the moment we have an empty project directory containing only a `README.md`. It's time to code!

- Create these directories at the root of your project: `bash`, `c`, `js`
- Create these empty files:
 - `c/c_is_fun.c`
 - `js/main.js`
 - `js/index.js`
- Create a file `bash/alx` with these two lines inside: `#!/bin/bash` and `echo "ALX"`
- Create a file `bash/school` with these two lines inside: `#!/bin/bash` and `echo "School"`
- Add all these new files to git
- Commit your changes (message: "Starting to code today, so cool") and push to the remote server


Repo:

- GitHub repository: `alx-zero_day`
- Directory: `0x03-git`
- File: `bash/alx`, `bash/school`, `c/c_is_fun.c`, `js/main.js`, `js/index.js`

☒ Done!

Help

Check your code

 Get a sandbox

QA Review

3. Collaboration is the base of a company

mandatory

Score: 100.0% (Checks completed: 100.0%)

A branch is like a copy of your project. It's used mainly for:

- adding a feature in development
- collaborating on the same project with other developers
- not breaking your entire repository
- not upsetting your co-workers

The purpose of a branch is to isolate your work from the main code base of your project and/or from your co-workers' work.

For this project, create a branch `update_script` and in this branch:

- Create an empty file named `bash/98`
- Update `bash/alx` by replacing `echo "ALX"` with `echo "ALX School"`
- Update `bash/school` by replacing `echo "School"` with `echo "The school is open!"`
- Add and commit these changes (message: "My personal work")



- Push this new branch Tips (/rltoken/tN31xFfDVctixZZ350tDGw)

(/)

Perfect! You did an amazing update in your project and it's isolated correctly from the **main** branch.

Ho wait, your manager needs a quick fix in your project and it needs to be deployed now:

- Change branch to `main`
- Update the file `bash/alx` by replacing `echo "ALX"` with `echo "ALX School is so cool!"`
- Delete the directory `js`
- Commit your changes (message: "Hot fix") and push to the origin

Ouf, hot fix is done!

Repo:

- GitHub repository: `alx-zero_day`
- Directory: `0x03-git`
- File: `bash/alx`, `bash/school`, `bash/98`

☒ Done![Help](#)[Check your code](#)[>_ Get a sandbox](#)[QA Review](#)

4. Collaboration: be up to date

mandatory

Score: 100.0% (Checks completed: 100.0%)

Of course, you can also work on the same branch as your co-workers and it's best if you keep up to date with their changes.

For this task – **and only for this task** – please update your file `README.md` in the main branch from GitHub.com. It's the **only time** you are allowed to update and commit from GitHub interface.

After you have done that, in your terminal:

- Get all changes of the main branch locally (i.e. your `README.md` file will be updated)
- Create a new file `up_to_date` at the root of your directory and in it, write the git command line used
- Add `up_to_date` to git, commit (message: "How to be up to date in git"), and push to the origin

Repo:

- GitHub repository: `alx-zero_day`
- Directory: `0x03-git`
- File: `README.md`, `up_to_date`

☒ Done![Help](#)[Check your code](#)[>_ Get a sandbox](#)[QA Review](#)

5. HAAA what did you do???

#advanced

Score: 100.0% (Checks completed: 100.0%)

Collaboration is cool, but not really when you update the same file at the same time...

To illustrate that, please merge the branch `update_script` to `main`: "Cool, all my changes will be now part of the main branch, ready to be deployed!"

HHHHHHHAAAAAAAAA

```
CONFLICT (content): Merge conflict in bash/alx
```

As you can see, you have conflicts between two branches on the same file.

Your goal now is to resolve conflicts by using the version of the branch `update_script`, and push the result to the origin.

At the end, you should have all your work from the branch `update_script` (new file and two updated files) and all latest `main` commits (new files, delete folder, etc.), *without* conflicts.


Repo:

- GitHub repository: `alx-zero_day`
- Directory: `0x03-git`

☒ Done!

Help

Check your code

 Get a sandbox

QA Review

6. Never push too much

#advanced

Score: 100.0% (Checks completed: 100.0%)

Create a `.gitignore` file and define a rule to never push `~` files (generated by Emacs). Tips (`/rltoken/sEKIVnR2IEL0sN4jc26hhA`)


Repo:

- GitHub repository: `alx-zero_day`
- Directory: `0x03-git`
- File: `.gitignore`

☒ Done!

Help

Check your code

 Get a sandbox

QA Review



(/)

Copyright © 2024 ALX, All rights reserved.

