Cours, leçons, tutoriels, etc

- Linux
  - Cours guidé
- Git
- Know-How | Write a nice README.md

Linux

Cours guidé

https://learning.lpi.org/fr/learning-materials/101-500/103/103.1/103.1_01/

Git

Guide Merge Request

https://mediashare.fr/post/procedure-de-merge-request

Know-How | Write a nice README.md

https://matthiasott.com/notes/how-to-readme https://links.kalvn.net/shaare/YUH_nQ

https://outils.warriordudimanche.net/outils/readmetoaster/

What’s in a README?

But what information should you put into the README of, let’s say, a website project? I’ve seen (and written) quite a few short and incomplete READMEs, so coming up with a good one surely is an art in itself.

The answer is, as always: it depends. It depends on the project itself, the team size, or the time and budget you have. So there really is no right or wrong here. There are, however, a few things almost every README has. Here’s a list of things you might consider to include:

Name and Description

What is this project about? What does the project do? Why is it useful? What problem does it solve? Who is it for?

Installation

How do you install the project and what do you need for it? Mention all dependencies that need to be installed first. Ideally, you also provide version numbers. (I’m looking at you, Node.js…)

Configuration and Usage

Instructions for how to configure, run, and use the project. For example, you can include the commands needed to start the development environment or any other useful and important commands.

Deployment

Instructions for how to deploy the project to a production environment, including any server requirements and commands used. And, in case you are using a CI/CD pipeline, for example, how do any automated processes work?

Testing

Any unit or integration tests people can run to assure that everything’s working as expected? Any frameworks or commands that are needed here? And are there any tests in your deployment pipeline that ensure that no errors make it into the live site?

How to contribute

Often overlooked, but so important: how do you contribute to the project? Are there any guidelines you should adhere to, for example, related to the coding syntax? But also: how is the approval process? Do people commit their changes to Git branches? Do they write pull requests? Who can merge and push, and so on? And who is the point of contact for the project in case you have questions?

License

If applicable, and especially if the project is publicly available, provide license information.

Roadmap

What are you planning to add in the future? Any known bugs or missing features?

Changelog

What did change with which version?

Authors and Acknowledgments

If people helped to build the project or provided useful contributions like features or bug fixes, or also if your project is built on other people’s open source code, mention them and – thank them!

~

I’m sure that I’m still missing some things. So if there is anything you would add, write a quick response blog post. ;) But whatever you include in your next README (template?), keep in mind that a good README is concise, complete, and up to date. It doesn’t necessarily have to be short. If there’s a lot to say and know about a project, then write it down. But always have the people in mind who will read the file. Keep it as short and simple as possible and also consider that not everyone might know what you know. So better include a few bits of information or links too many to make sure everyone will be able to get started and successfully work with your project.

For more useful tips and resources about READMEs, also have a look at the GitHub docs, this article by Binyamin Green, Make a README, and Awesome READMEs.