Lesson Tuesday

Introduction to Programming (/introduction-to-programming) / Git, HTML and CSS (/introduction-to-programming/git-html-and-css) / Writing a README

Text Cheat sheet

Every GitHub repository should include a `README.md` file that provides any visitor to the repository with detailed information about the project. The name README is self-explanatory — it's a document that users should read to get more information about the contents of a project. The most common format for READMEs is Markdown, which is why our README files will include the `.md` extension. Markdown is perfect for READMEs because it provides simple formatting — which is really all we need for a README.

Why Add a README?

Imagine you find an object covered in little screens and buttons. There are no instructions on how to use it or what it is. It might be an advanced alien technology or it might be a piece of junk. You have no idea — and you never figure out how to operate it. It could have some amazing functionality but as far as the user is concerned, it's worthless (except perhaps as a paperweight).

In order to use the object, you really need to know what it does — and how to use it. Then it becomes useful. The same is true for code — and that's exactly what a README does. It explains what a project is for and how to use it. Without a README, your project will

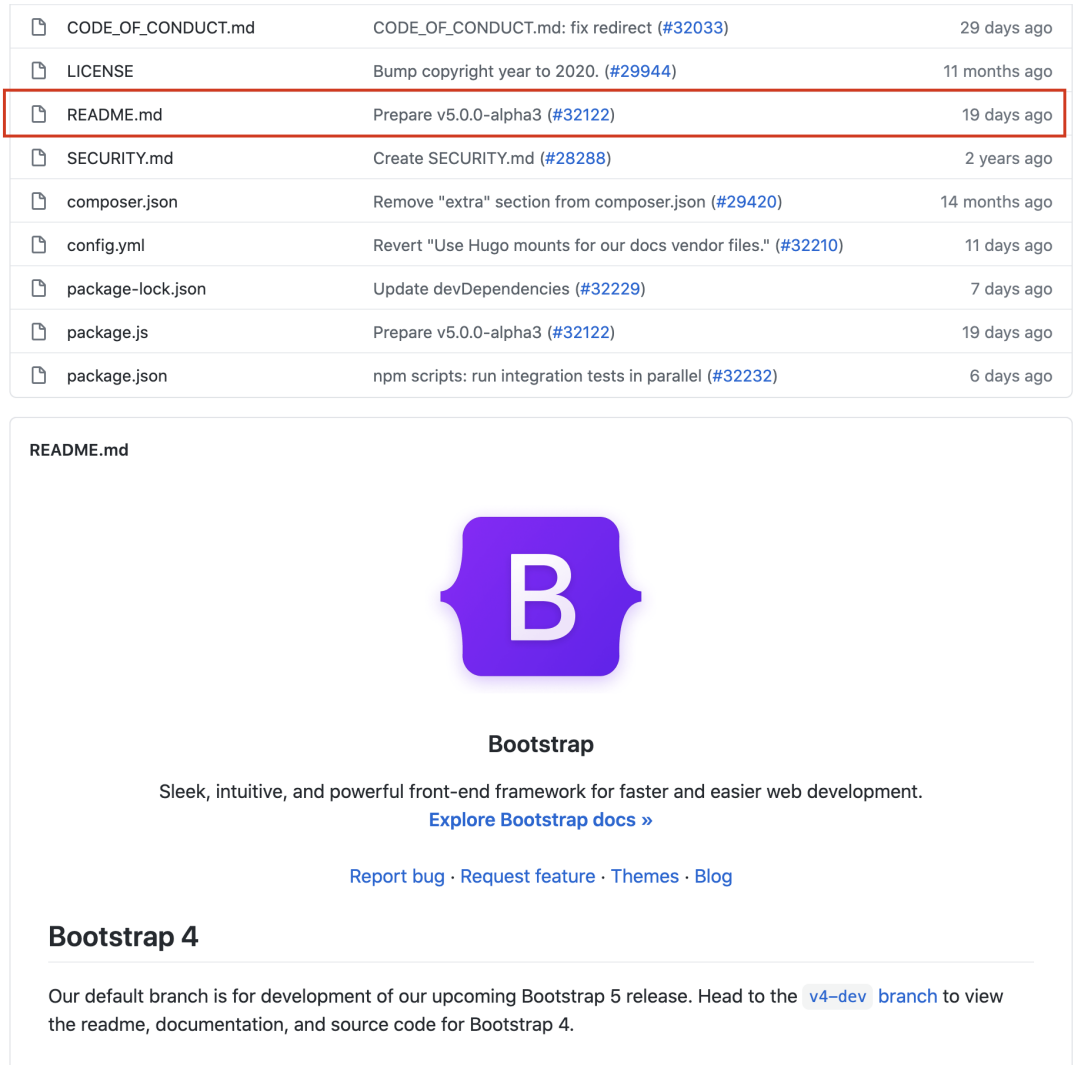
likely be ignored — whether that's by friends, family, classmates, or potential employers. They won't know how to use it or what it's for. It could be a project that you've polished, put lots of work into, and that could potentially showcase a fantastic project — but no one will know unless it includes a good README.

For that reason, a good README is an essential part of any project. We can accurately say that a project is only as good as its README. So make sure you always include an excellent README for *every* project. All passing code reviews must have a README.

Location

When we create a `README.md` file, it should always be stored at the top level of the project directory. There are two reasons for this.

The most important reason is that when we push a repository to GitHub, GitHub will *automatically* take the file named `README.md` and add its contents to the front page of the repository. For instance, take a look at the repository for Bootstrap (https://github.com/twbs/bootstrap), a popular open source toolkit for styling websites that we will be learning about later in this section. If you scroll down the page, you'll see that the README is visible — and if you look at the file structure of the project, you'll also see that `README.md` is in the top level of the directory. You can also see this in the image below:



In this image, we can see that the `README.md` file is one of many in the top level of the project repository. It was updated just a few weeks before this photo was taken. A README isn't just something that we add to a project and forget about. Instead, it's **a living document** that often changes as the project changes. We can also see that the README is automatically rendered beneath the file directory. GitHub will take care of this for us as long as the file is named `README.md` and is stored in the top level of the project.

There's another reason to store a README in the top level of our project. Even if we weren't using GitHub, we still want instructions for using the project to be easily accessible. For instance, let's say we are opening a project on our desktop. The instructions for what the project is and how it works should be readily accessible — that is, in the top level of the project.

Content

READMEs vary widely from one repository to the next. A project that has many users (like Bootstrap) will often include detailed documentation on how to use the product in the README. At this point, your projects won't be widely-used applications, which means your READMEs don't need to be quite that extensive. However, even if your project is basic, that's not a good reason to skimp on your README. Your READMEs should always include the following information no matter how basic the project:

- **Application name**: The name of the project
- **Names of contributors:** If you are working on a project on your own, you will be the only contributor. However, if you are working with a pair or in a group, you will need to list all contributors.
- **Brief description of the project**: What does the project *do*? This should be a quick summary of what the project can be used for. Here are some examples:
    - Bootstrap: "Sleek, intuitive, and powerful front-end framework for faster and easier web development."
    - React: "React is a JavaScript library for building user interfaces."
    - A student HTML project: "Demonstration site showcasing HTML, CSS, and jQuery."
- **Technologies used**: You should always include a section that describes the technologies used in the project. This serves a number of purposes:
    - You can better showcase your skills by listing the technologies used. This can get a potential employer's attention.
    - This can help other users determine whether they want to use or modify the project. For instance, if a project uses Ruby, a Ruby programmer might want to clone and modify that project. On the other hand, a programmer that doesn't use Ruby might choose another project instead.

- **Longer description of the project as needed**: This includes any extra details you might need to share about the project such as how to use it or specific features.
- **Complete setup/installation instructions**: At this point, your setup and installation instructions will be minimal. However, later in the program, they will often be more involved. These instructions are necessary so that other users can actually download and use your project. The best way to generate installation instructions is to clone your project and then walk through all the steps necessary to get the project running, documenting them along the way.
- **Known bugs**: If there are any bugs in your application or features that are incomplete or not working correctly, they should be included in this section. This section can help users determine whether they should use the project. Adding known bugs can also be helpful for your future self. For instance, if you return to a project that you haven't worked on for a while, it's helpful to have a quick refresher on what's not working. Finally, adding information about known bugs can allow others to help you identify and fix those bugs — including your classmates and instructors.
- **License information with a copyright and date**: A standard license such as an MIT license (https://opensource.org/licenses/MIT) allows others to use and modify your project.

README Template

Here is an example bare-bones README:

Northwest Gardener

By Jasmine Smith and Willow Jones

A simple JavaScript interface for planning a garden.

Technologies Used

- JavaScript
- HTML
- CSS
- jQuery

Description

This browser application allows users to create vegetable, flower, and fruit gardens. Users can create a grid and then block out sections of the grid for each plant.

Simply enter the dimensions of your garden in the handy form, then add custom shapes and labels for your plants!

Setup/Installation Requirements

- Clone this repository to your desktop.
- Navigate to the top level of the directory.
- Open `js/index.html` in your browser.

Known Bugs

- The tomato icon is currently broken.
- When refreshing the page, the garden is reset. We plan to add local storage for temporary persistence.

License

MIT

This screenshot shows a very basic README that incorporates all of the required items. It doesn't look fancy but it does use basic Markdown such as headings and bullets to make it more readable. A README that looks like this is perfectly fine for now. However, keep in mind that your portfolio projects (and any projects that you are really proud of) should have more visually appealing READMEs. We will cover tips for creating great READMEs later in the program. For now, just make sure you get used to adding the basics above.

Remember, it's worth taking the time to make your README look nice because it will be the first thing anyone will see in your repositories. If people see that you have a messy or incomplete README, they will assume that your code is also messy or incomplete. But if you have a README with clean formatting and

clear setup instructions, your users will have more confidence in trying out your software — and in trying you out, too, if they're an employer looking to hire!

Below, we've included the basic template used in the Northwest Gardener README above. You can copy and paste the text below into a new file and replace the relevant parts that are mentioned in curly brackets with information about your project. Don't include the curly brackets in your README — instead, replace with your individualized content. You are welcome to add additional sections and content to your README as needed — just make sure that at a minimum, your READMEs include the content in this template.

```
# _{Application Name}_

#### By _**{List of contributors}**_

#### _{Brief description of application}_

## Technologies Used

* _List all_
* _the major technologies_
* _you used in your project_
* _here_

## Description

_{This is a detailed description of your application. Give
as much detail as needed to explain what the application do
es as well as any other information you want users or other
developers to have.}_

## Setup/Installation Requirements

* _This is a great place_
* _to list setup instructions_
* _in a simple_
* _easy-to-understand_
* _format_

_{Leave nothing to chance! You want it to be easy for poten
tial users, employers and collaborators to run your app. Do
I need to run a server? How should I set up my databases? I
s there other code this application depends on? We recommen
d deleting the project from your desktop, re-cloning the pr
oject from GitHub, and writing down all the steps necessary
to get the project working again.}_

## Known Bugs

* _Any known issues_
* _should go here_
```

```
## License

_{Let people know what to do if they run into any issues or
have questions, ideas or concerns.  Encourage them to conta
ct you or make a contribution to the code.}_

Copyright (c) _date_ _author name(s)_
```

Licensing

As we mentioned earlier, an **open source** license details how others can use your code. MIT (https://en.wikipedia.org/wiki/MIT_License) and GPL (https://en.wikipedia.org/wiki/GNU_General_Public_License) are the most common licenses. An MIT license means your code is free to use by anyone and you are not liable for any problems in the software. Rails and jQuery are examples of software that use an MIT license. GPL also indicates free usage of the code but when used, the resulting work MUST be open source. Linux, Git and WordPress all use GPL.

For additional details on choosing a license for your code, visit GitHub's Choose a License site (http://choosealicense.com/).

Expectations

Moving forward, you're expected to include a detailed `README.md` file for every project you create here at Epicodus. This ensures that anyone who views your portfolios can quickly understand what you've created.

Towards the end of the program, all full-time students will spend a full day preparing materials for internship interviews and their future job search. In addition to writing cover letters, preparing a LinkedIn profile, and practicing interview questions, students are also required to ensure all their GitHub repos have detailed

READMEs, as seen in this assignment (https://www.learnhowtoprogram.com/internship-and-job-search/applying-for-internships-and-jobs/sprucing-up-github). Make sure to create a README for each project now so you don't have to go back and add them later.

Also, the independent projects you turn in at the end of each section are *required* to include a README. If they don't, they won't be marked as passing.

For more information on creating a README, check out Make a README (https://www.makeareadme.com/).

Previous (/introduction-to-programming/git-html-and-css/markdown)
Next (/introduction-to-programming/git-html-and-css/practice-markdown-by-writing-readmes)

Lesson 31 of 64
Last updated March 24, 2023

disable dark mode

(http://www.epicodus.com)