GitHub

README File – Everything you Need to Know

By **Great Learning Team** - Jul 6, 2021

40420

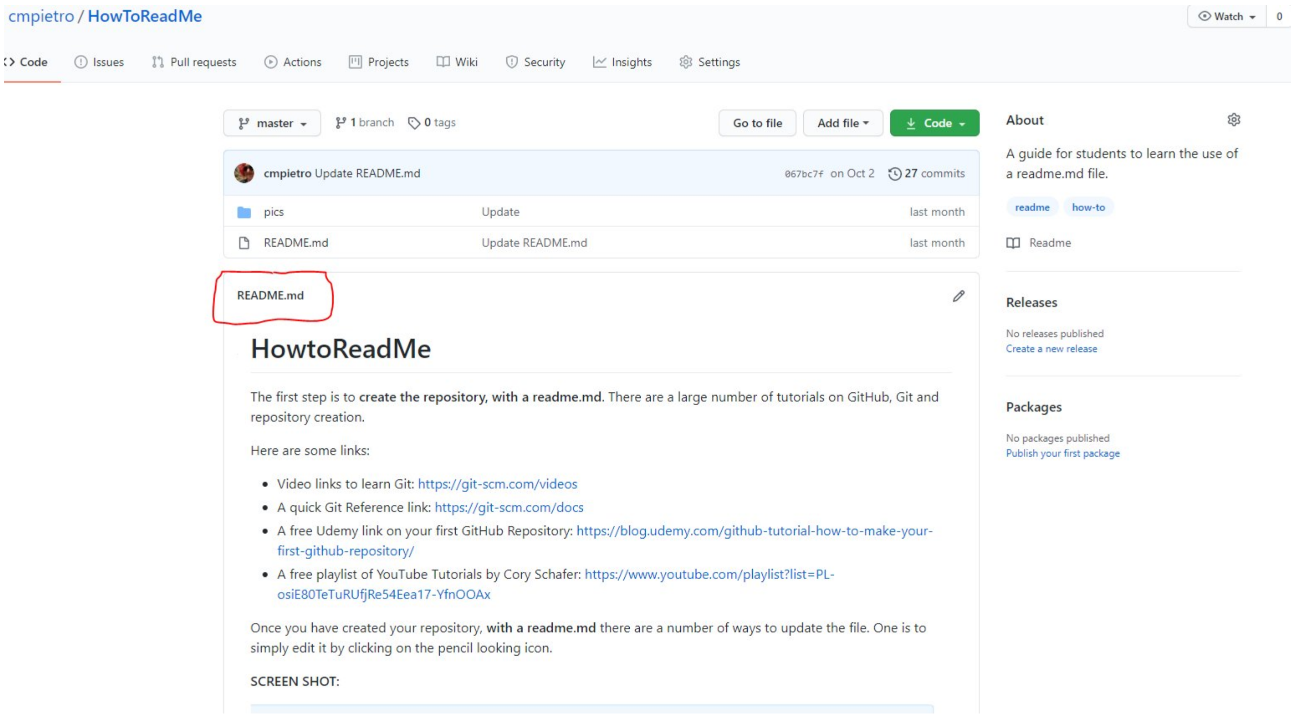
A README file is a text file that describes and launches a project. It comprises information that is frequently needed to grasp the scope of the project. In this blog, we will talk about what a README file is, why is it necessary and how you can write a good README file.

Why is a README File necessary?

Well, as software engineers, we all must have come across the term GitHub somewhere. It is where we upload our code so that it is safe even if our system crashes. It is also a platform where you can get a lot of open-source codes shared by awesome developers for free. One thing which we see all the time but do not give much importance to is the Readme file. First of all, let's understand what a readme file is.

What is a README File?

The Readme file is often the first file that the users read. It is a text file that contains the information for the user about the software, project, code, or game, or it might contain instructions, help, or details about the patches or updates.



A guide for students to learn the use of a readme.md file.

cmpietro Update README.md	067bc7f on Oct 2	27 commits
pics	Update	last month
README.md	Update README.md	last month

README.md

HowtoReadMe

The first step is to **create the repository, with a readme.md**. There are a large number of tutorials on GitHub, Git and repository creation.

Here are some links:

- Video links to learn Git: https://git-scm.com/videos
- A quick Git Reference link: https://git-scm.com/docs
- A free Udemy link on your first GitHub Repository: https://blog.udemy.com/github-tutorial-how-to-make-your-first-github-repository/
- A free playlist of YouTube Tutorials by Cory Schafer: https://www.youtube.com/playlist?list=PL-osiE80TeTuRUfjRe54Eea17-YfnOOAx

Once you have created your repository, **with a readme.md** there are a number of ways to update the file. One is to simply edit it by clicking on the pencil looking icon.

SCREEN SHOT:

When you create a repository or a project, GitHub gives you the option of a default readme. The default readme file contains the repository name and some basic instructions. The file format is ‘md’, which stands for Markdown documentation. It is a lightweight markup language that can be easily converted to text.

Why is a README File needed?

Github has become the platform where most open-source code is shared as the world is pushing more and more towards open-source projects and code. When sharing your code with the world, a problem that might occur is that they may not particularly understand how to use it or even understand it. So that is where the readme file helps. The readme file is used to explain what is uploaded and how we can install or use it. It even allows the uploader to add images and videos to help the reader navigate through the project. A well-written readme file is more important if you intend to show these projects in your resume. The interviewer might go to your project but might not understand a single piece of code. But if there is a readme file, it will help him/her understand better what the project is for, which coding languages/frameworks were used and how to navigate through that project. A good readme file for your project goes a long way in making an impression in your interview. An unsaid thing about readme files is that it even helps your future self. For example: If, for some reason, a bug was found a few months after you have left the project, and the company or the team asks you to fix it. You are completely out of touch and have no clue where to start looking. A readme file comes in handy in this scenario.

How to write a good README File?

As we have understood that a good readme file is necessary, let's look at writing a good readme file. When writing a readme file, the main focus is it should be brief but also contain all the information needed. In addition, it should have clarity and need to be structured. Some points to help you in writing:

- **Up-to-date**: A readme file should be regularly updated to not conflict with what is uploaded to the repository. If possible, a readme file should contain details about all the version uploads and mention the changes/ updates to the project.
- **Brief and Clear**: A readme file should not extend like an essay. It should only have what is necessary and not anything more. It should be written clearly, keeping in mind the reader who is going to read. It should be written in a simple language so that everyone can understand. It is better to use bullet points or tables to convey your points better.
- **Detailed:** The readme file should not miss out on any information. It should contain all the necessary details necessary to understand what the project is about and how to use it. For example, if there exists a need to install software to run the project. Then the version of the software and technical configurations needed to run the project on their computer should be mentioned in full. If you feel the readme file is getting too long, you can break it up and just show what is in the context of the page or folder of the repository. You can also provide links to tutorials or instructions to configure the system or install the necessary software to make it short.
- **Self Explanatory:** The user shouldn't need to refer to some other article to understand your readme file. It should be self contained, as in the user should be able to understand everything by reading it. So it's best to write everything in layman terms for the common reader to understand because you never know who is going to read it. As developers, we do try to avoid writing the basics

because it’s quite normal. But it’s best to write it as the reader might be a beginner in the field.

What is a good template to write a README File?

There are multiple layouts or formats on the web which will help you in writing a readme file. No matter which template you follow, make sure that it follows all the points listed above. So let’s just look at one of them:

Project Title
A little brief about what the project is about. It should be like a small summary format informing about the main purpose of the project.
Motivation
This section is for letting the reader know why you created this project, the reason behind pursuing such a project, and why you have decided to do it.
Build Status
This basically explains the current build status of the project. If there is a bug /error which needs addressing. This is done so for two different reasons The user understands that this is an issue and does not spend more time figuring if it was a mistake on their part.A developer who is familiar with the issue can suggest some solutions directly without going through the whole code.
Code Style
This lets the users know that you have used a particular code style and helps them when contributing to your project so that the whole project code style stays the same. Some common code styles: standard, xo, etc.
Screenshots
As the saying goes, a picture is equal to a thousand words. Most people will be interested if there is a visual representation of what the project is about. It helps them understand better. A visual representation can be snapshots of the project or a video of the functioning of the project.
Tech/Framework used
This is used to help the reader understand which tech or frameworks have been used to do the project. It helps the reader understand which all tech stack he has to be familiar with to understand the whole project.
Features

This is where you write what all extra features have been done in your project. Basically this is where you try to make your project stand out from the rest.
Code Examples
This is where you try to compress your project and make the reader understand what it does as simply as possible. This should help the reader understand if your code solves their issue.
Installation
If your project needs installation of certain software or configurations to the system. Do mention it in this section as it helps a lot for the reader to use your project. The steps mentioned should be precise and explanatory. If possible, you can add links that can help them better understand how to configure/install the necessary files or softwares.
API reference
If your project is small, then we can add the reference docs in the readme. For larger projects, it is better to provide links to where the API reference documentation is documented.
Tests
This is the section where you mention all the different tests that can be performed with code examples
How to Use?
As I have mentioned before, you never know who is going to read your readme. So it is better to provide information on how to use your project. A step-by-step guide is best suited for this purpose. It is better to explain steps as detailed as possible because it might be a beginner who is reading it.
Contribute
This is where you let them know that they can contribute and help you out. A guideline on how to contribute is also helpful
Credits
Giving proper credit is most important. Mention any links/repos which helped you or inspired you to build this project. It can be a blog, another open source project, or just people who have contributed in building this project.
License
A short description of the license. (MIT, Apache, etc.)

How to make your README file more interesting?

Now that you have understood what to write in a readme. Let's look at how to make it interesting to read. The main issue with readme files is it can look like a terms and conditions page of a product. It is a long page filled with information. This is not at all interesting to the user. So some methods to improve your readme file is by adding images and visual representations. Having images helps it become more colorful and more attractive. It also helps in better understanding for some people. Everyone might not be familiar with all the technical jargon which we use. So showing visual representation helps more in that case. Adding funny-related memes or videos also helps in some ways.

How is README written?

We have all seen what all to write. But we did not discuss how to write it. In the beginning, we mentioned that the readme file is in md format, which stands for markdown documentation. So let's see how we can write it.

Markdown is very similar to text with some syntax written for bold, italic, links etc.

So let's head on with some examples:

- **Headers:**

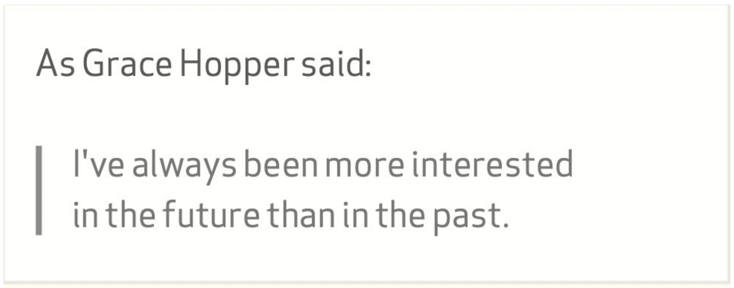
```
#   This is a  <h1> tag.  ##  This is a  <h2> tag. ###### This is a <h6> tag.
Result: This is a <h1> tag.This is a  <h2> tag.This is a <h6> tag.
```

- **Emphasis:**

```
*This text will be italic*_This will also be italic_
**This text will be bold**__This will also be bold__
**You **can** combine them*
Result:This text will be italicThis will also be italic
This text will be boldThis will also be bold
You can combine them
```

- **Block Quotes:**

As Grace Hopper said: > I've always been more interested > in the future than in the present.Result:



As Grace Hopper said:

> I've always been more interested
> in the future than in the past.

- **Lists:**

Unordered Lists:
Item 1Item 2Item 3Item 4

Order Lists:
Item 1Item 2Item 3Item 4Item 5

- **Images:**

![GitHub Logo](https://d1m75rqqgidzqn.cloudfront.net/images/logo.png) Format: ![Alt Text](url)

- **Links:**

http://github.com – automatic! [GitHub](http://github.com)

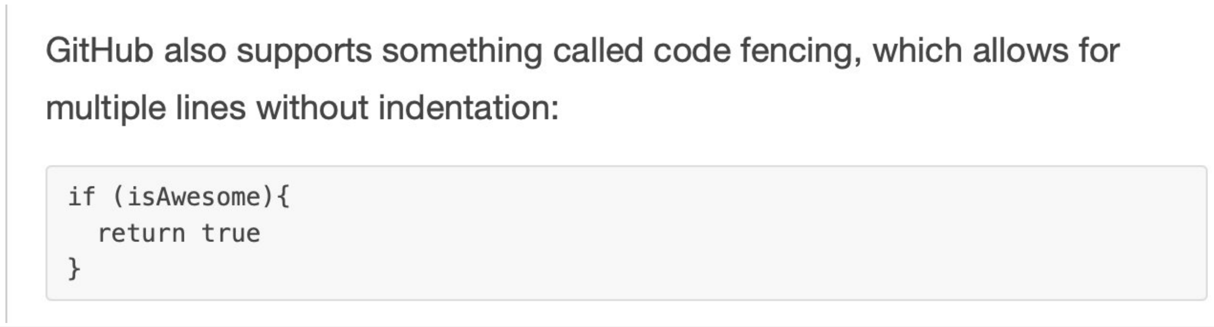
- **Extras**

1. \ – backslash
2. ‘ – backtick
3. * – asterisk
4. _ – underscore

5. {} – curly braces
6. [] – square brackets
7. () – parenthesis
8. # – hash mark
9. + – plus sign
10. – – minus sign
11. . – dot
12. ! – exclamation mark

- **Code**

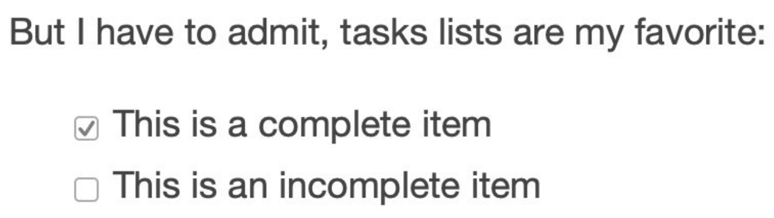
Github also supports something called code fencing, which allows for multiple lines without indentation:“`if (isAwesome){ return true}Result:


GitHub also supports something called code fencing, which allows for multiple lines without indentation:

```
if (isAwesome){
  return true
}
```

- **Task Lists**

But I have to admit, tasks lists are my favourite:
– [x] This is a complete item– [ ] This is an incomplete item
Result:

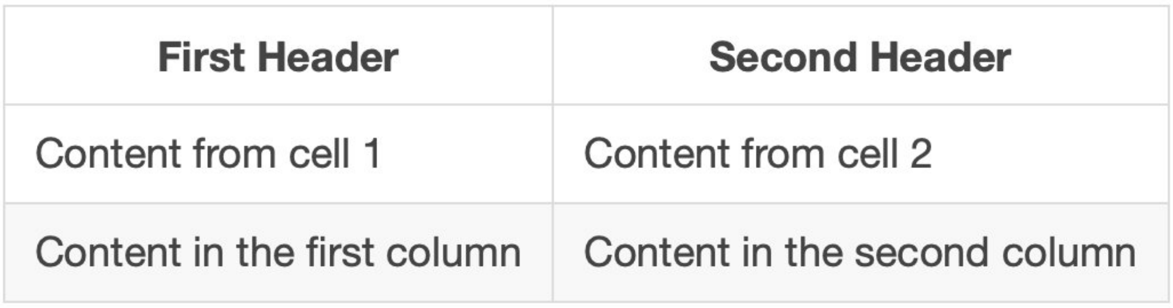

But I have to admit, tasks lists are my favorite:

- [x] This is a complete item
- [ ] This is an incomplete item

- **Tables**

First Header | Second Header———— | —————Content from cell 1 | Content

from cell 2Content in the first column | Content in the second column
Result:



First Header	Second Header
Content from cell 1	Content from cell 2
Content in the first column	Content in the second column

- **SHA References**

If we enter a commit SHA-1 hash, it will be automatically converted to a link by Github.

- **Issues references within a repository**

Any reference to any issue or pull request is automatically converted to a link.

- **UserName mentions**

If there is a need to mention someone who contributed in any format to the project or reported an issue, then you can easily mention that person using the **'@'** symbol followed by the name. You can also refer to teams by using the @ keyword.

- **Automatic URL conversion**

Any URL link which is entered is converted directly into a clickable link.

- **Strikethrough**

Any text or word wrapped in **~~word~~** will appear as crossed out.

- **Emoji**

Now we can enter emojis as well to express better.

Some common emojis are as follows:

SNo.	ShortCode	Emoji		ShortCode	Emoji
1	:grinning:	😀	8	:smiley:	😃
2	🙂	😄	9	😁	😁
3	:laughing: or :satisfied:	😆	10	:sweat_smile:	😅
4	:rofl:	□	11	:joy:	😂
5	:slightly_smiling_faces:	🙂	12	:upside_down_face:	🙃
6	😉	😉	13	:blush:	😊
7	:innocent:	😇			

Now that we have seen how to write in the md file . Let's look at how the readme file is intended for different users.

1. **For end-users:** A readme file answers questions about installing the versions of software needed for the project to run. The configuration steps are to be done so that everything runs smoothly. The readme file also answers questions basically based on which framework and coding language was used. It also helps the user understand why a certain framework was used instead of another. How the owner of the project got inspired, the method by which he solved the issue, and much more.
2. For our own development work: A readme file provides two good use cases. First, if it is written prior to starting the development, it helps the developer by acting as a roadmap or tracker so that he is aware of what is left to do in the project life cycle. Second, it helps the developer understand the code that he might have written a few years back better when there is a need for it to be revisited and de-bugged. This helps a lot since it takes a lot of time to understand the code when you have to scrape through millions of lines of code that you might have written a few years back. The readme file helps you locate where each section of the code is and helps you find the piece of code you were looking for. It is a great time saver when the issue is critical and time-bound.
3. **For other developers:** The readme file serves a similar purpose to other developers going through the code and helps them understand where each section of code is located and which page contains the logic. This helps when there is an issue in the code, and another developer is trying to solve the issue. He can go through the code easily and find where the issue is. Likewise, if

another developer is trying to enhance the code, then a good readme file goes a long way in helping that person.

So to wrap it all up. First, we have gone through the definition of the readme. The format in which it is written. Common syntax to be used, which will help you in writing one. Things to keep in mind while writing a readme file. A common template to follow and understand what to write and what not to write in a readme file. And last, we saw how the readme file is useful for different types of readers/users.

So we hope this has helped you get a little bit of knowledge on the Readme file. To learn more about programming and other related concepts, check out the courses on Great Learning Academy.

Have a Great Learning!

Previous article
The way Great Learning Imparted Training was Really Good- Prakash Oraon, PGP- AIML

Next article
NGX Bootstrap – Everything you Need to Know

Great Learning Team

Great Learning's Blog covers the latest developments and innovations in technology that can be leveraged to build rewarding careers. You'll find career guides, tech tutorials and industry news to keep yourself updated with the fast-changing world of tech and business.

RELATED ARTICLES MORE FROM AUTHOR

GitHub

GitHub vs GitLab: The key differences

GitHub

Git Vs. Github

GitHub

Git Cheat Sheet

LEAVE A REPLY

Comment:

Name:*

Email:*

Website:

☐ Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

What is Analysis of Variance?

Python Tutorial For Beginners

Palindrome in Python

Factorial Program in Python