1. Intro <https://www.youtube.com/watch?v=Vnj2VNQROtI>
2. Github profile important items <https://www.youtube.com/watch?v=prvPVTjVkwQ>
3. Good github repo <https://www.youtube.com/watch?v=qBi8Q1EJdfQ>
4. Interview with Art part 1 <https://www.youtube.com/watch?v=ClLYamtaO-Q>
5. Writing readme with Walter <https://www.youtube.com/watch?v=DQEfT2Zq5_o>
6. Interview with Art part 2 <https://www.youtube.com/watch?v=Vvzl2J5K7-Y>
7. Commit messages with github
8. Participating in open source projects <https://www.youtube.com/watch?v=OxL-gMTizUA>
9. Interview with Art part 3 <https://www.youtube.com/watch?v=M6PKr3S1rPg>
10. Participating in open source project 2 <https://www.youtube.com/watch?v=elZCLxVvJrY>
11. Outro <https://www.youtube.com/watch?v=dps7Ti6Lado> (+cf)
12. Github profile review overview
13. Github profile review details

Commit messages with github

As Art said, commit messages are essential for communicating why your code was changed. This is for your coworkers or collaborators, and also for your future self. Let’s go through some best practices for writing commit messages.

Here is the message format we use here at Udacity. You can find it at [**http://udacity.github.io/git-styleguide/**](http://udacity.github.io/git-styleguide/) for future reference.

type: subject

body

footer

The first line is the subject. This should be a short description of what changed. Nothing like “fixed it” or “did something,” these need to be clear and informative, and try to avoid profanity. The subject should be 50 characters or less, with the first letter capitalized, and end without a period. At Udacity, we also include a short annotation about the type of the commit, if it is a bug fix, a feature, change to the documentation, etc.

The body is next, this is where you give a more detailed description of why you made the change. The body should typically have around 72 characters per line. This is to ensure that the message fits into a terminal window when using git on the command line. You’ll also need to make sure there is a blank line between the subject line and the body. You can also add bullet points, using asterisks or dashes, when you need to make a list.

Some commits don’t require a body in the message. If you fix a typo for example, it’s okay to only have a subject line.

You can also include a footer, typically this will be used to indicate which issues or bugs the commit addresses.

A more fleshed out example looks like this:

feat: Summarize changes in around 50 characters or less

More detailed explanatory text, if necessary. Wrap it to about 72

characters or so. In some contexts, the first line is treated as the

subject of the commit and the rest of the text as the body. The

blank line separating the summary from the body is critical (unless

you omit the body entirely); various tools like `log`, `shortlog`

and `rebase` can get confused if you run the two together.

Explain the problem that this commit is solving. Focus on why you

are making this change as opposed to how (the code explains that).

Are there side effects or other unintuitive consequences of this

change? Here's the place to explain them.

Further paragraphs come after blank lines.

- Bullet points are okay, too

- Typically a hyphen or asterisk is used for the bullet, preceded

by a single space, with blank lines in between, but conventions

vary here

If you use an issue tracker, put references to them at the bottom,

like this:

Resolves: #123

See also: #456, #789

This does come with an exception of course. If you are working on an open source project, be sure to follow the message format for that project. This will make the maintainers happy and increase the chance your pull request is accepted.

**Video: Outro**

Here's the [**checklist**](https://docs.google.com/document/d/1a9AKnNyqfGgdQV5ohPCN5H9ntnEUhMptWMwVBWURCN0/pub?embedded=true) for keeping track of improvements.

## GitHub Profile Project Overview

Your GitHub Profile showcases the projects you have built and informs others how, and how often, you collaborate on code. Demonstrating that you know GitHub best practices will send a signal to other developers and hiring managers that you’re ready to work together on the next great thing. Your repositories should show that you’re serious about the projects you start and that you write clean, well-formatted code. Check out this [**great summary blog post**](http://blog.udacity.com/2015/06/become-github-pro.html) on GitHub and how it can be used for continued learning and your career.

## Why this GitHub Profile Project?

GitHub is used as a recruiting platform and as a way for potential employers to review your work. It is also a great place to connect with other developers and showcase your skills.

## What will I do?

You will look at your GitHub profile through the lens of a recruiter or hiring manager, focusing on how your profile, projects and code represent you as a potential candidate for a company or collaborator on a project.

To do this, you will work on updating your GitHub profile so that it is professional and ready for the job hunt. You will also work on perfecting your projects so that they are navigable for recruiters and anyone else interested in your work. To do this, you will add README files and properly formatted commit messages.

## How do I Complete this GitHub Profile Project?

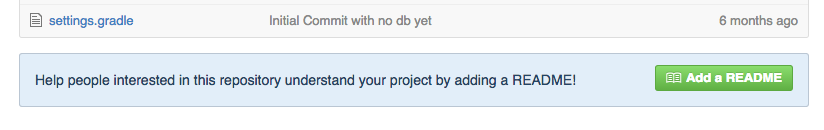
At this point in the Nanodegree, we assume you know the basics of Git and GitHub: how to clone a repository, how to initialize a repository, how to make a commit with a message and how to upload a repository to GitHub. If you do not, check out this short course on [**Version Control**](https://www.udacity.com/course/how-to-use-git-and-github--ud775).

Start by reading over the [**Udacity GitHub Profile checklist**](https://docs.google.com/document/d/1a9AKnNyqfGgdQV5ohPCN5H9ntnEUhMptWMwVBWURCN0/pub?embedded=true) to get an overview of the goals and tasks for the GitHub Profile.

Go through the entire checklist; check each item that you’ve completed. For items that are not completed, fix and update your GitHub profile. We recommend [**this blog post**](http://blog.udacity.com/2015/06/become-github-pro.html)for more details about GitHub as an online platform. We also encourage you to check out the [**Career Resource Center**](http://career.udacity.com/resource-center/hiring_process_portfolio.html) for more guidance on updating your GitHub profile.

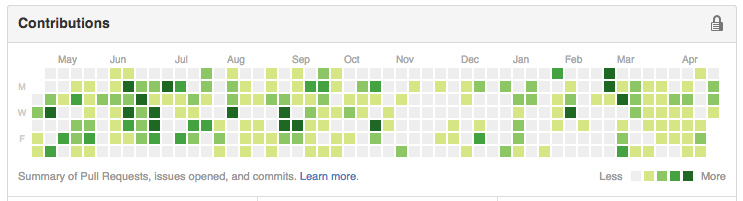
## How to Create a README on GitHub

If you're unfamiliar with how to create a **README** on GitHub, here are some instructions to get you started:

* Click on one of your projects and scroll down below the list of files. If you haven't created a README, you'll see an image like this: 
* Click the **Add a README** button.
* In your README, explain the purpose of the project, any instructions about how to use or view it, and what kind of collaboration is sought. READMEs are written in Markdown; check out [**GitHub's Markdown guide for more information**](https://help.github.com/articles/markdown-basics/).

## The Path to a Perfect Profile

GitHub does a good job of quickly and visually showing your repository activity. Because of this, we highly encourage you as a developer to **show activity**. What you’ve done in this project is the beginning. GitHub highlights your activity for the last year, so we have two pieces of long term advice:

* Commit small, incremental changes, every day. A green commit graph, like the one pictured below, shows that you’re actively adding to your projects. 
* Contribute to open source repositories. This is a huge positive signal for recruiters. It shows that you work well with others and more importantly that coding is a true passion; something that you do in your free time and without a direct monetary reward. If you’re interested in how to get started, check out the [**GitHub guide**](https://guides.github.com/activities/contributing-to-open-source/).

## Next Steps

Once you’ve checked off all of the items, move to the next section to get your profile reviewed.

[**Career Resource Center: Portfolio**](http://career.udacity.com/resource-center/hiring_process_portfolio.html)