

The Tech World Has Adopted GitHub

In 2017, [GitHub](#) saw:

- 1.5 million organizations using the platform
- 24 million users across 200 countries, from entrepreneurs to open source contributors to company employees
- 25.3 million active repositories
- 47 million pull requests, which means people and organizations are working together

Software engineers, developers, and data scientists at most companies use GitHub daily to house their work, read each other's documentation, and collaborate across teams and organizations.

Recruiters, hiring managers, and senior engineers search through GitHub for potential job candidates. Your GitHub profile - activity, repos, commits, documentation - provides evidence that you're a skilled engineer.

Udacity graduate Ryan Waite was [hired](#) based on the strength of his GitHub profile.

Resources

We recommend you review these resources to ensure your profile is ready for feedback.

1. [Rubric](#). *Your project will be reviewed by a Udacity Career Reviewer against this rubric.*
2. [Checklist](#). *Based on the project rubric, this is a handy checklist on GitHub best practices.*
3. [Career Resource Center](#). *Find additional tips and guides on developing your GitHub Profile.*

PROJECT SPECIFICATION

Optimize Your GitHub Profile

General

CRITERIA	MEETS SPECIFICATIONS
Student has been and continues to be active on GitHub	<ul style="list-style-type: none">Account has at least three projectsAccount shows knowledge about how to make incremental commitsThe commit graph shows many green squares for the last two weeks (indicating that commits have been pushed regularly)

Personal Profile

CRITERIA	MEETS SPECIFICATIONS
GitHub is professional and allows recruiters to contact the student	<ul style="list-style-type: none">GitHub username is professionalProfile picture is a professional image of studentProfile includes at least one up-to-date links for: 'URL' and/or 'Company' fields and/or 'Contact Email'Profile includes current location

Projects

CRITERIA	MEETS SPECIFICATIONS

Commit style is consistent	<ul style="list-style-type: none"> • Last commit made matches the Udacity Commit Message Style Guide or the student has indicated that they are following another style guide
Documentation is consistent, allowing recruiters to easily understand student's coding practices	<ul style="list-style-type: none"> • Projects have meaningful names • Projects have meaningful descriptions • Most recent three projects have a completed README

Suggestions to Make Your Project Stand Out!

Keep your GitHub profile easy to read. Make sure your last three repositories don't include unnecessary files such as operating system files, editor configurations, etc.


GitHub Profile Checklist



= Optional Udaciousness

Link to associated [GitHub Profile Rubric](#)


General

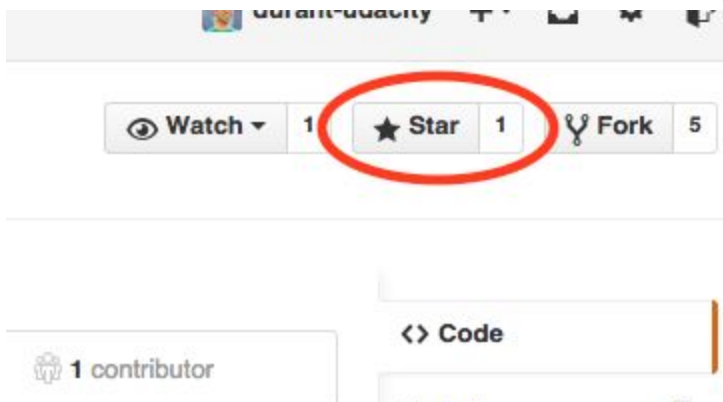
- ☐ I include at least three projects on my GitHub account.
- ☐ My GitHub account shows knowledge about how to make incremental commits
- ☐ My commit graph shows many green squares for the last two weeks. (This indicates that commits have been pushed regularly).
- ☐  I've contributed to an open source project.

Personal Profile

- ☐ My GitHub username is professional.
- ☐ My profile picture is a professional image.
- ☐ My profile includes at least one up-to-date links for: 'URL' and/or 'Company' fields and/or 'Contact Email'.
- ☐ My profile includes my current location.

Projects

- ☐ My last commit made matches the [Udacity Git Commit Message Style Guide](#). If it does not, I indicate that I am following another style guide
- ☐ My projects are forked appropriately.
- ☐ My projects have meaningful names and descriptions.
- ☐ My most recent three projects have a completed README practices.
- ☐  I have starred at least one repository I'd like to keep track of.



Udacity Git Commit Message Style Guide

Introduction

This style guide acts as the official guide to follow in your projects. Udacity evaluators will use this guide to grade your projects. There are many opinions on the "ideal" style in the world of development. Therefore, in order to reduce the confusion on what style students should follow during the course of their projects, we urge all students to refer to this style guide for their projects.

Commit Messages

Message Structure

A commit messages consists of three distinct parts separated by a blank line: the title, an optional body and an optional footer. The layout looks like this:

```
type: subject
```

```
body
```

```
footer
```

The title consists of the type of the message and subject.

The Type

The type is contained within the title and can be one of these types:

- **feat:** a new feature
- **fix:** a bug fix
- **docs:** changes to documentation
- **style:** formatting, missing semi colons, etc; no code change
- **refactor:** refactoring production code
- **test:** adding tests, refactoring test; no production code change
- **chore:** updating build tasks, package manager configs, etc; no production code change

The Subject

Subjects should be no greater than 50 characters, should begin with a capital letter and do not end with a period.

Use an imperative tone to describe what a commit does, rather than what it did. For example, use **change**; not changed or changes.

The Body

Not all commits are complex enough to warrant a body, therefore it is optional and only used when a commit requires a bit of explanation and context. Use the body to explain the **what** and **why** of a commit, not the how.

When writing a body, the blank line between the title and the body is required and you should limit the length of each line to no more than 72 characters. This is to ensure that the message fits into a terminal window when using git on the command line. You can also add bullet points, using asterisks or dashes, when you need to make a list.

The Footer

The footer is optional and is used to reference issue tracker IDs.

Example Commit Message

```
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.

Optimize your GitHub profile Module

- Make it a Professional Portfolio.
- Picture:
 - A smiling picture helps recruiters make a connection with you
 - Else have a picture that can capture your personality
- Links:
 - Twitter: so they can see what you are interested in
 - Personal blog maybe
- Name is professional
- Repos:
 - They see you're most popular repo (by number of stars)
 - Also they see the repos you have contributed to indicating your collaboration with open source software. Team work evaluation.
 - Keep clear repo description
- Code:
 - You probably have to show part of your code to recruiters
 - Keep em well commented
 - Make sure they work
- Contribution Chart:
 - Shows activeness
 - And involvement with code
- Will look through recent commits, contribution in projects and README
- The things a HR manager will see in order:

1. Heat activity at the bottom (green stuff): Are they contributing often? Do you love to code?
 2. See the repos candidate had created and contributed. Find where they have done the meatiest work and dig in that.
 3. Have they contributed to their people's repos, do they interact well in discussions etc. Are they good collaborators?
- **ReadMe****: Well written readme is one of the most important parts of a good repository.
 - It contains: What the code is for? How to build/install code? How to contribute?
 - <https://classroom.udacity.com/courses/ud777/lessons/5338568539/concepts/53317786070923>
 - **Commi Messages**:
 - The formatting is important because when you try to figure out what a piece of code is doing or looking for a bug you get **git blame**. Shows committer and commit messages from where you get to know what the code does. This proves clear communication!
 - **Open Source** Project Contribution is important showing that you CAN collaborate. When you get a job you have to work with other people on shared code bases. Recruiters look at issues they have participated in, the comments and if they commented on pull requests or not. **Whether they are patient with people who are newer in the community. Helpful, kind and good to people.
 - How to start contributing to open source:
 - To break the ice with a new project, find some documentation, even the Readme. Clone it, try to build and run it. Maybe pull requests in Readme just about building or running the project. Then move on to fixing bug fixes and adding features. So contributing to documentation is a great way to start.
 - Look at the issues, and maybe solve those. To start, find an issue related to the documentation
 - Star and watch interesting repos. It IS important. Shows your likings.

Apply these best GitHub practices moving forward and attract the attention of recruiters and senior software engineers.

You'll want to:

- Commit every week, at minimum
- Get 2-4 weeks streaks

A personal side-project may grow into something bigger. You can't always predict what will catch the eye of a recruiter or senior software engineer, but when it does, you want to showcase your best work.

Continue building your knowledge of using GitHub and improving your profile, be sure to check out the "GitHub Professional Profile Resources" pdf.

Once you've implemented the necessary changes to make your profile more professional, remember to maintain activity by regularly making commits on GitHub.

Project Resources

1. [Project Rubric](#). *Your project will be reviewed by a Udacity Career Reviewer against this rubric.*
2. [Project checklist](#). *Based on the project rubric, this is a handy checklist on GitHub best practices.*
3. [Career Resource Center](#). *Find additional tips and guides on developing your GitHub Profile.*

Up Next

Continue developing on GitHub! For project ideas, consider [contributing to open source](#).