Determining What To Work On

What To Work On

Let's say you're using some third-party library to help you build a project. What happens if, while you're using this third-party library, you come across a bug or a misspelling? You have the skill to be able to fix it, but you don't have direct access to make modifications to the original library. Well that's not a problem because you know that forking another developer's repository copies it to your account and gives you full access to `git pull` and `git push` to it!

But what are you supposed to do now that you've got full access to a duplicate of the other developer's project. We'll look at this in the next section but if you have forked a project and you have code in your fork that's not in the original project, you can get code into the original project by sending the original project's maintainer a request to include your code changes. This request is known as a "Pull Requests". Again, we'll look at sending and working with pull requests in the next lesson.

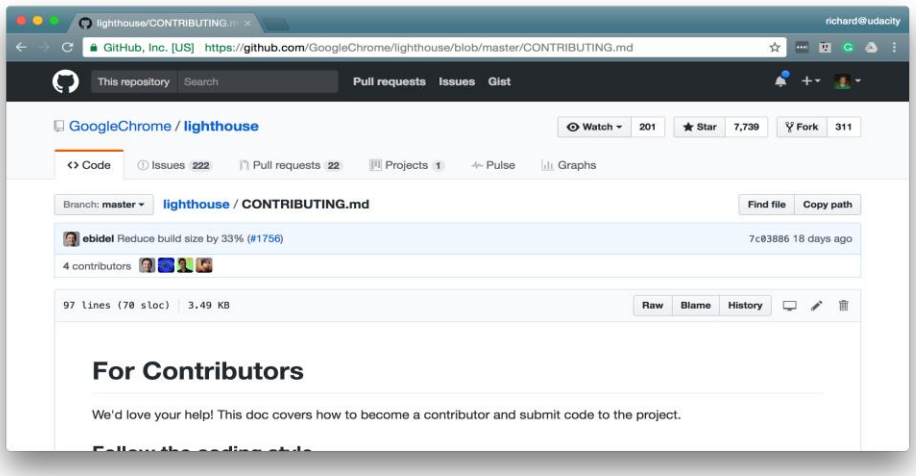
So you know it's possible to get your code in the original project and you know you want to help and fix this spelling/code mistake. So you got something to work on! But how do you go about actually contributing to the project in the way that the original project maintainer will be happy with and will end up actually incorporating your changes?

The first thing you should always look for in a project is a file with the name `CONTRIBUTING.md`.

CONTRIBUTING.md File

The name of the `CONTRIBUTING.md file is typically written in all caps so that it's easily seen. As you could probably tell by its name, this file lists out the information you should follow to contribute to the project. You should look for this file before you start doing development work of any kind.

Let's look at the lighthouse project's contributing file:


lighthouse/CONTRIBUTING.m
richard@udacity
GitHub, Inc. [US] https://github.com/GoogleChrome/lighthouse/blob/master/CONTRIBUTING.md
This repository
Search
Pull requests
Issues
Gist
GoogleChrome / lighthouse
Watch 201
Star 7,739
Fork 311
Code
Issues 222
Pull requests 22
Projects 1
Pulse
Graphs
Branch: master
lighthouse / CONTRIBUTING.md
Find file
Copy path
ebidel Reduce build size by 33% (#1756)
7c03886 18 days ago
4 contributors
97 lines (70 sloc) 3.49 KB
Raw
Blame
History

For Contributors

We'd love your help! This doc covers how to become a contributor and submit code to the project.

Google's Lighthouse project's CONTRIBUTING.md file.

You can see that the top line of the file says:

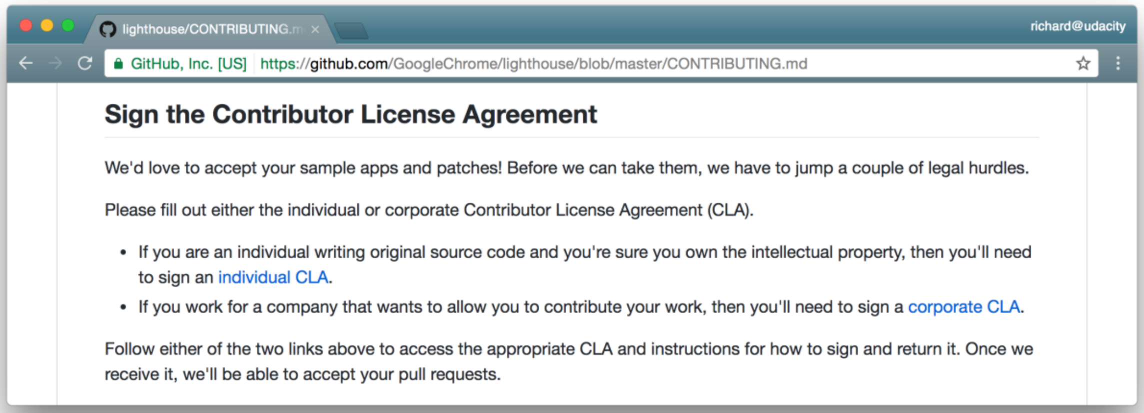
We'd love your help! This doc covers how to become a contributor and submit code to the project.

There are two main sections to this file:

- the "For Contributors" section
- the "For Maintainers" section

Each one of these sections has subsections of its own to instruct readers on how to contribute to and work with this project.

Let's take a look at the section on signing the contributors license. Here's what it looks like at the time of making the course:


lighthouse/CONTRIBUTING.m
richard@udacity
GitHub, Inc. [US]
https://github.com/GoogleChrome/lighthouse/blob/master/CONTRIBUTING.md

Sign the Contributor License Agreement

We'd love to accept your sample apps and patches! Before we can take them, we have to jump a couple of legal hurdles.

Please fill out either the individual or corporate Contributor License Agreement (CLA).

- If you are an individual writing original source code and you're sure you own the intellectual property, then you'll need to sign an individual CLA.
- If you work for a company that wants to allow you to contribute your work, then you'll need to sign a corporate CLA.

Follow either of the two links above to access the appropriate CLA and instructions for how to sign and return it. Once we receive it, we'll be able to accept your pull requests.

The Contributor License Agreement section of Google's Lighthouse project's CONTRIBUTING.md file.

You can see that to be able to contribute to this project you need to sign Google's Contributor License Agreement.

Take a look through the Lighthouse project's contributing file. What file contains the information about the code styles for the Lighthouse project?

Answer: The .eslintrc.js file defines all. We use JSDoc with TypeScript checkJs. Annotations are encouraged for all contributions.

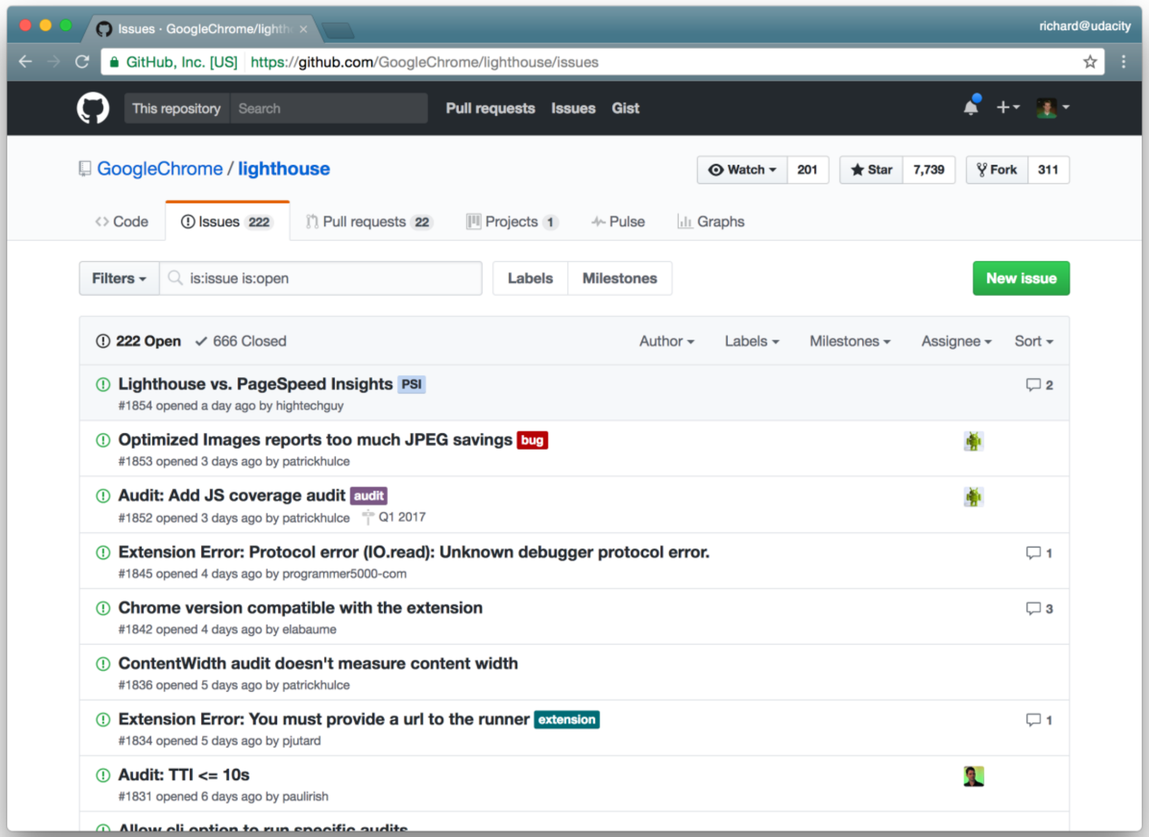
As you can see this contributor file has a ton of information in it. So you should definitely look for a CONTRIBUTING.md file when you want to contribute to a project.

GitHub Issues

If your code change is a simple spelling mistake then you can probably just go ahead and make that change. But if your change is more substantial where it modifies a number of files in a significant way, then you probably want to get approval by the project's maintainer(s) before you start working on it. You definitely don't want to spend a couple hours making changes to the project only to find out that someone else is doing the exact same thing. You'd just be wasting a lot of time and energy duplicating work.

In a CONTRIBUTING.md file it explains *how* your code should be formatted and the steps you should go about to contribute, but how do you know *what* you should contribute? You should talk to the project maintainers directly. GitHub has a fantastic interface for asking questions of the project maintainer in an open way that lets everyone see what's being done with the project.

This is the GitHub Issues interface:



Lighthouse vs. PageSpeed Insights PSI — 2
#1854 opened a day ago by hightechguy

Optimized Images reports too much JPEG savings bug
#1853 opened 3 days ago by patrickhulce

Audit: Add JS coverage audit audit
#1852 opened 3 days ago by patrickhulce Q1 2017

Extension Error: Protocol error (IO.read): Unknown debugger protocol error. — 1
#1845 opened 4 days ago by programmer5000-com

Chrome version compatible with the extension — 3
#1842 opened 4 days ago by elabaume

ContentWidth audit doesn't measure content width
#1836 opened 5 days ago by patrickhulce

Extension Error: You must provide a url to the runner extension — 1
#1834 opened 5 days ago by pjutard

Audit: TTI <= 10s
#1831 opened 6 days ago by paulirish

Allow cli option to run specific audits

The Issues page of the Lighthouse project.

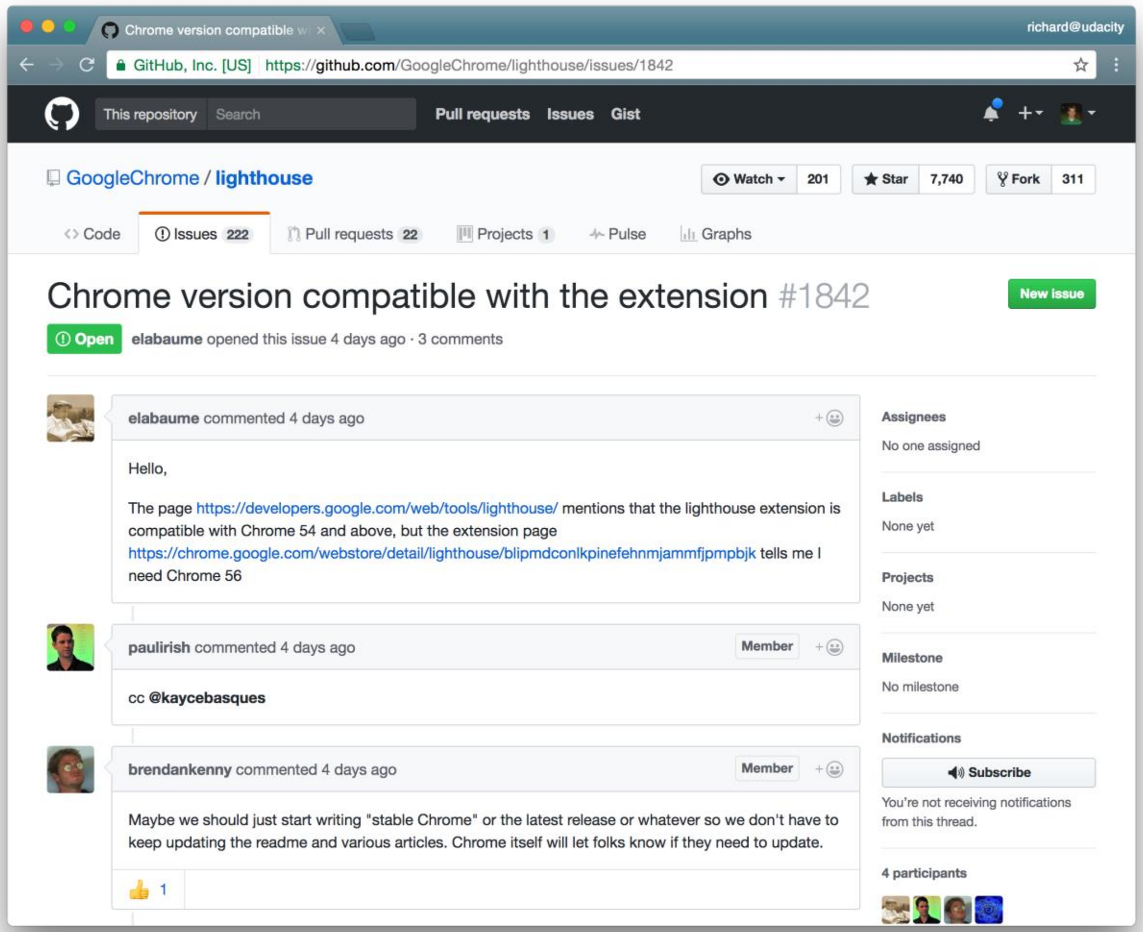
Now, "issues" doesn't mean that there's actually a bug, it can just be any change that needs to be made to the project. GitHub's issue tractor is quite sophisticated. Each issue can:

- have a label or multiple labels applied to it

- can be assigned to an individual
- can be assigned a milestone (for example the issue will be resolved by the next major release)

But probably one of the most important aspects of the issue tracker is that each issue can have its own comments, so a conversation can form around the issue.

Check out this issue that has a number of comments:



Chrome version compatible with the extension #1842

Open elabaume opened this issue 4 days ago · 3 comments

elabaume commented 4 days ago

Hello,

The page https://developers.google.com/web/tools/lighthouse/ mentions that the lighthouse extension is compatible with Chrome 54 and above, but the extension page https://chrome.google.com/webstore/detail/lighthouse/blipmdconlkpinefehnmjammfjpmpbjk tells me I need Chrome 56

paulirish commented 4 days ago Member

cc @kaycebasques

brendankenny commented 4 days ago Member

Maybe we should just start writing "stable Chrome" or the latest release or whatever so we don't have to keep updating the readme and various articles. Chrome itself will let folks know if they need to update.

👍 1

Assignees: No one assigned

Labels: None yet

Projects: None yet

Milestone: No milestone

Notifications: Subscribe

You're not receiving notifications from this thread.

4 participants

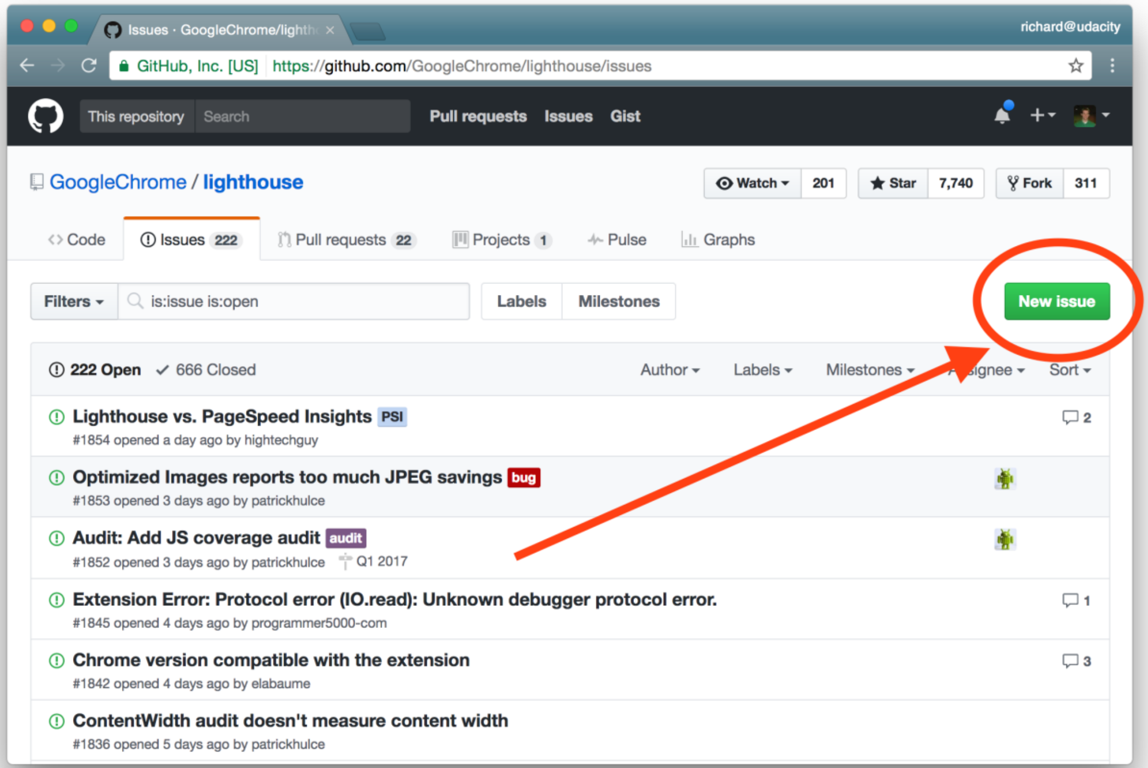
The first few comments of an issue dealing with compatibility of Chrome and the Lighthouse extension.

Another thing that's nice about issues is:

- they let you subscribe to an issue so you'll be notified of new comments and code changes
- you can communicate back and forth with a project maintainer on a specific change

Before you contribute anything to a file, check out the instructions in CONTRIBUTING.md. Then check out the project's issues and look to see if there's anything that's similar to what you want to contribute. If there is, then subscribe to that issue and read the existing conversation to see if you can help.

If you've looked through the list of issues and don't see one that similar to what you want to do, then you can create a new issue of your own. On every page of the GitHub issues interface, you'll find the new issue button:



Lighthouse vs. PageSpeed Insights PSI
#1854 opened a day ago by hightechguy 2

Optimized Images reports too much JPEG savings bug
#1853 opened 3 days ago by patrickhulce

Audit: Add JS coverage audit audit
#1852 opened 3 days ago by patrickhulce Q1 2017

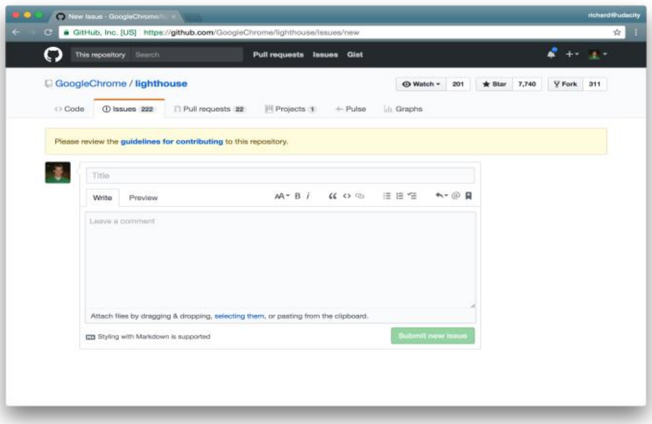
Extension Error: Protocol error (IO.read): Unknown debugger protocol error.
#1845 opened 4 days ago by programmer5000-com 1

Chrome version compatible with the extension
#1842 opened 4 days ago by elabaume 3

ContentWidth audit doesn't measure content width
#1836 opened 5 days ago by patrickhulce

The new issue button on the Lighthouse project's issue page.

Clicking on that button brings you to create a new issue



Please review the guidelines for contributing to this repository.

Attach files by dragging & dropping, selecting them, or pasting from the clipboard.

Styling with Markdown is supported

The new issue page for the Lighthouse project. A notice about reviewing the guidelines for contributing displays above the form.

New Issue Page

One really cool thing about the new issue page is that, if the project has a CONTRIBUTING.md file, it will display a notification at the top of the page recommending that you check out the guidelines on how to contribute to the project. Clicking on the "guidelines for contributing" link takes you to the CONTRIBUTING.md file.

The GitHub issues interface support markdown so when you create your issue you can use Markdown to format it and exactly the way you want by including links, images, bulleted lists, and code blocks.

💡 Learn Markdown! 💡

From the README file, to the New Issue Page, to commenting, Markdown is incredibly important! If you're new to Markdown, we discuss everything about it in our Writing READMEs course. It's short, so why not take an hour to learn this incredible skill!

Just like crafting a descriptive commit message, you want to create an issue with an informative title that explains briefly what you want to do. Then, in the comments section, provide plenty of detail on what the change is, or why you think it's needed, or how this will make the project better.

Typically, the project's maintainer has a full-time job and works on their project on the side, so give them some time to respond to your issue before you dive in and start making your changes. Once the project maintainer has given you the go-ahead it's time to start working on the changes you want to contribute back to the project.

Topic Branches

The best way to organize the set of commits/changes you want to contribute back to the project is to *put them all on a topic branch*. Now what do I mean by a **topic branch**? Unlike the master branch which is the default branch that holds all of the commits for your entire project, a topic branch host commits for just a single concept or single area of change.

For example if there is a problem with the login form for logging into the website, then a branch name to address this specific issue could be called:

- login
- login-bug
- signup-bug
- login-form-bug
- etc.

There are plenty of names that can be used for a topic branch's name. You just want to use a clear descriptive name for the branch so that if, for example, you list out all of the branches you can immediately see what changes are supposed to be in a branch just by its name.

Question 2 of 2

One of the branches in the Lighthouse project is named add-a11y-tests. Do you think that's a good name for a topic branch?

(Hint - a11y stands for "accessibility". In the word "accessibility", there are eleven letters between the a and the y, so it gets shortened to just a11y!)

Answer: Yes

One thing to keep in mind is that sometimes a project has specific requirements on what to name your topic branch. For example, if a branch is going to be addressing bug fixes, then many projects require a `bugfix- prefix`. Going back to our branch that was dealing with a bug with the login form, it would have to be named something like `bugfix-login-form`. So definitely check out the `CONTRIBUTING.md` file to see if they provide instructions on what you should name your topic branches.

Best Practices

Write Descriptive Commit Messages

While we're talking about naming branches clearly that describe *what* changes the branch contains, I need to throw in another reminder about how critical it is to write clear, descriptive, commit messages. The more descriptive your branch name and commit messages are the more likely it is that the project's maintainer will not have to ask you questions about the purpose of your code or have dig into the code themselves. The less work the maintainer has to do, the faster they'll include your changes into the project.

Create Small, Focused Commits

This has been stressed numerous times before but make sure when you are committing changes to the project that you make smaller commits. Don't make massive commits that record 10+ file changes and changes to hundreds of lines of code. You want to make smaller, more frequent commits that record just a handful of file changes with a smaller number of line changes.

Think about it this way: if the developer does not like *a portion* of the changes you're adding to a massive commit, there's no way for them to say, "I like commit A, but just not the part where you change the sidebar's background color." A commit can't be broken down into smaller chunks, so make sure your commits are in small enough chunks and that each commit is focused on altering just one thing. This way the maintainer can say I like commits A, B, C, D, and F but not commit E.

Update The README

And lastly if any of the code changes that you're adding drastically changes the project you should update the README file to instruct others about this change.

Recap

Before you start doing any work, make sure to look for the project's CONTRIBUTING.md file.

Next, it's a good idea to look at the GitHub issues for the project

- look at the existing issues to see if one is similar to the change you want to contribute
- if necessary create a new issue
- communicate the changes you'd like to make to the project maintainer in the issue

When you start developing, commit all of your work on a topic branch:

- do not work on the master branch
- make sure to give the topic branch clear, descriptive name

As a general best practice for writing commits:

- make frequent, smaller commits
- use clear and descriptive commit messages
- update the README file, if necessary