1 TEALS AP CS A Curriculum — How to Contribute

1.1 Repository Location

The curriculum's source code is hosted on Github at https://github.com/TEALSK12/apcsa-public. Restricted instructor-only content is hosted at https://github.com/TEALSK12/apcsa-instructor.

Did you catch an error in the TEALS AP CS A curriculum, or perhaps have an idea for an addition that would make it better? If so, we'd love to see what you've got!

1.2 Methods of Contributing

We can accept changes and suggestions to this repo in a bunch of different ways, so here they are from least involved to most.

1.2.1 Casual Communication

You can make comments via email to Kenney Chan. If you have a GitHub account, though, it's better to submit an issue, as there's a higher guarantee that it won't get lost.

1.2.2 Submit an Issue

The best way to make sure your feedback is recognized, tracked, and handled is to submit an issue on GitHub. You'll need a GitHub account to do this.

1.2.3 Submit a Pull Request (PR)

This is the most involved route, but more powerful. It works for all kinds of issues, from fixing typos to making radical changes in curriculum. Of course, if you decide on something massive, make sure to vet the idea with us first.

If you're doing something small or obvious, and you're comfortable, feel free to just go straight to a pull request. For anything else, you might want to first submit an issue. In the issue, mention that you're willing to do the work yourself. If the idea is sound, we'll give you the green light before you commit any effort or time.

1.3 Previewing Locally

To view this GitBook locally for previewing changes, first install the gitbook-cli tool:

npm install gitbook-cli -g

Then from the repository root run:

gitbook serve

For complete instructions, see the GitBook website.

> Note: this method of previewing won't contain all the TEALS styling on the main site, but will give you a rough approximation before making a submission.

1.4 How to Submit a Pull Request (Advanced)

The following steps outline the easiest way to submit a PR:

1. Create a local working clone of the `apcsa` repository. If you haven't done this before, just go to the main code page and hit the "Clone or download" button. Explaining how to use Git is outside the scope of this page, so you'll need to know how to do this already.

2. Create and checkout a feature branch for your work. (The `master` branch is protected, and you won't be able to submit changes there.)

3. Make your changes in the feature branch. Make sure you follow the style guidelines as you do so.

4. Push your branch up to the GitHub repo. For example, if you work on the command line, and your feature branch is named "moar-glitter", then you'd do this:

   git push origin —set-upstream moar-glitter

   This just needs to happen once to establish the connection between your local branch and a branch of the same name up on GitHub. After that, if you have more changes to add, you can just `git push` while working in your feature branch.

5. When you're done and ready for review (and all of your commits have been pushed), head to the main code page. There you should see your branch in a highlighted box with a "Compare & pull request" button on the right. If not, you can just hit the "New pull request" button. If you did the latter, there will be two buttons to control which branch is going to merge into which. In this case, the base branch should be `master`, and the compare branch should be `moar-glitter`. Once that's set, be sure to enter a descriptive comment about the pull request. If there's an associated issue, include the issue number, prefixed with a `#` character. For example, if this is a fix for issue 137, include `#137` in the comments.

6. Finally, hit the "Create pull request" button. This will automatically notify the reviewers. Several things could happen:

   - A reviewer approves your PR and merges it into `master`. Gitbook will see the changes and automatically build an update with your changes within a few minutes.

   - A reviewer rejects your PR. This shouldn't happen if you got approval in a submitted issue. If not, you should get a clear explanation why the change was not accepted.

   - A reviewer may point out problems with your PR, and request changes before approving. You'll need to monitor your PR to catch this. One aid is that GitHub tracks notifications in the upper right corner of the web pages (the bell icon), and should alert you to change requests.

7. After the dust has settled, *please delete your feature branch* from the GitHub repo. You may of course choose to keep it in any local clones, but this will keep the primary repo clean of branch clutter. The one exception to this is if your feature is long-running, and you plan to issue a string of PRs as work progresses.

Hopefully, all went well, and you've helped to make our curriculum even better — *thank you!*