Live Code Editor (in maintenance mode)

WARNING: This project is currently in maintenance mode. Please feel free to file bug reports. If determine that the issue is serious enough we will fix it. Please do not submit Pull Requests (PRs). We don't have the resources to review them and they'll be closed.

Overview

Welcome to the GitHub repository for the Live Code Editor project. Live Code Editor is an Open Source project dedicated to giving learners an interactive code learning tool to help teach teach students how to code. This is the live coding environment developed for the Khan Academy Computer Programming curriculum. It gives learners an editor on the left (either ACE or our Blocks-based drag-and-drop editor) and an output on the right (either JS+ProcessingJS, HTML, or SQL). Here's a tour of how it's used on KA.

How it works

For a deep dive into the components of the LiveEditor, read this wiki.

You can also watch these talks that the team has given about the editor:

- John Resig on CodeGenius
- Pamela Fox at ReactConf

You can find various demos in the `demos/` directory, and start playing immediately with the simple demo:

- http://khan.github.io/live-editor/demos/simple/

Who it’s for

Live Editor is Open Source software under an MIT License. This tool is for everyone, particularly for those learning to code.

What the goal is

The goal of Live Editor is to provide a browser based code editor with live output to aid students learning to code see what effects their code has as they practice coding. An emphasis is placed on error handling (and error messages) to help make debugging easier to understand for beginners.

Communication

Live Code Editor has its own Gitter channel (what is Gitter?).

If you want to send a message through Gitter, click the link below

gitter join chat

You can sign in using your GitHub account and start messaging with the Live Editor team.

How to run

In order to run `live-editor` locally you'll have run a local web server. Follow these steps:

1) Download the project to local storage (Refer to this guide)

2) Make sure you have python installed (get Python). You can make sure that python is installed by running the following command from a console:

```
python --version
```

You should see an output similar to:

```
Python X.X.X
```

where X’s are version numbers. If your output differs refer back to Python documentation.

3) Open a console in live-editor folder and run the following command:

```
python -m SimpleHTTPServer
```

You should see the following console output:

```
Serving HTTP on 0.0.0.0 port 8000 ...
```

Open up a browser and navigate to http://0.0.0.0:8000/demos/simple.

How To Build

WARNING: This project requires node ~6 and npm ~3 in order for `npm install` to work correctly. (Learn more about nodejs and npm)

Note: nvm (or node version manager) is a convenient way to manage and switch between various node versions. Learn more about nvm

You can check to see what version of npm you have using the command:

```
npm --version
```

Likewise you can check what version of node you are using:

```
node –version
```

```
python -m SimpleHTTPServer
```

You can use the pre-built copies of everything inside the `build/` directory. If you wish to make some changes and re-build the library from scratch you'll need to install some dependencies. Run the following commands from a console:

```
git submodule update --init --recursive
npm install
npm bower install
bower install
```

If you have any issues installing bower, be sure to check the most up to date documentation (`bower`)

```
# Build the Ace editor files (This is usually *not* needed)
cd bower_components/ace
npm install
node Makefile.dryice.js -nc
```

At this point you can make a fresh build, using Gulp. Type in the console:

```
npm run build
```

If you have an issue with "this.merge" is undefined, then try the command: `rm -rf node_modules/gulp-handlebars/node_modules/handlebars`.

How to test

The tests are in the `/tests` folder. They use Mocha/Chai/Sinon. Gulp typically runs the tests when relevant files change, but you can explicitly run the tests from the console with the command:

```
npm test
```

You can also run single test suites at a time - see gulpfile.js for what suites are available:

```
gulp test_output_pjs_assert
```

You can run the tests in the browser runner by opening the relevant webpage:

http://0.0.0.0:8000/tests/output/sql/

TravisCI also runs those tests when new commits are made.

In order to run the tests that create Worker threads, you'll need to run Chrome with a flag:

```
/Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --allow-file-access-
from-files
```

Please add tests whenever possible for any change that you make or propose.

How you can help

We have many open issues here. The top priority are those marked as regressionbug, since those are things that used to work. After that, the ones marked as browserbug may be the easiest to take on, and there are also plain old bugs. All the issues are tagged according to their environment, pjs, webpage, sql, or if they're generally about the ACE editor, editor. There are also a few bugs specifically about the demo pages, since those can get behind, and requests for more tests, since we can always use more of those!

Some aspects of the editor are in subrepos with their own issue trackers, like structuredjs and structuredblocks, so be sure to poke around those and see if they're more up your bug-fixing alley.

There are also a handful of ideas floating here from our community. You are welcome to take them on, but it's possible we won't merge them if we worry about their effect on the programming experience on Khan Academy, like if they may introduce backwards incompatibilities or performance regressions.

We have no full-time resource working on the editing environment right now, so we will do pull requests when we find ourselves with time between other projects. We thank you for your contribution, even if we may be slow to acknowledge it at times. :)

Pull Requests

When submitting pull request please do the following:

- link to an issue, if no issue exists please create one
- write automated tests for new functionality or bug fixes
- include build files
- run the automated tests
- squash your commits into a single a commit

Please use the provided template for submitting pull requests.

License

Live Code Editor is an Open Source project and falls under the MIT License.

Before we can accept any pull requests you must sign our CLA.