Aurora Documentation

Release 0.0.1

Tundra

Apr 24, 2018

Plugin System

These are the docs for Aurora

CHAPTER 1

Mutations in Aurora

This is a simple tutorial for writing a mutation for the Aurora application. These are the steps you'll need to follow:

1. Setting up npm
2. Writing your mutation
3. Setting up Webpack
4. Publishing your mutation to npm

Mutations allow our users to have ultimate flexibility with their note taking styles. A mutation can change most aspects of Aurora. For instance, if you like taking notes in a cornell style format, you can write a mutation that will allow you to do that. Don't like the way the previews look? You can write a mutation to change that.

1.1 Plugins as NPM Packages

First you'll need to install node if you don't already have it. That can be done with this link: node download

Node comes with a version of npm, but you will need to check if it is the latest version. That can be done with entering this command in your terminal

```
npm install npm@latest -g
```

To test if it updated enter the command

```
npm -v
```

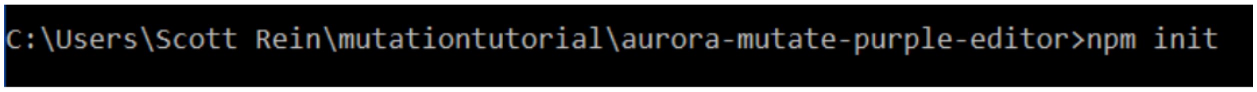
The version should be higher than 2.1.8

Once you have npm downloaded and updated you can then you can start creating your mutation.

Navigate in your file system to where you would like your npm package to be stored. Then create a folder called `aurora-mutate-nameOfMutation`. It is important to have the prefix `aurora-mutate` because that is how it will end up in our mutation store.

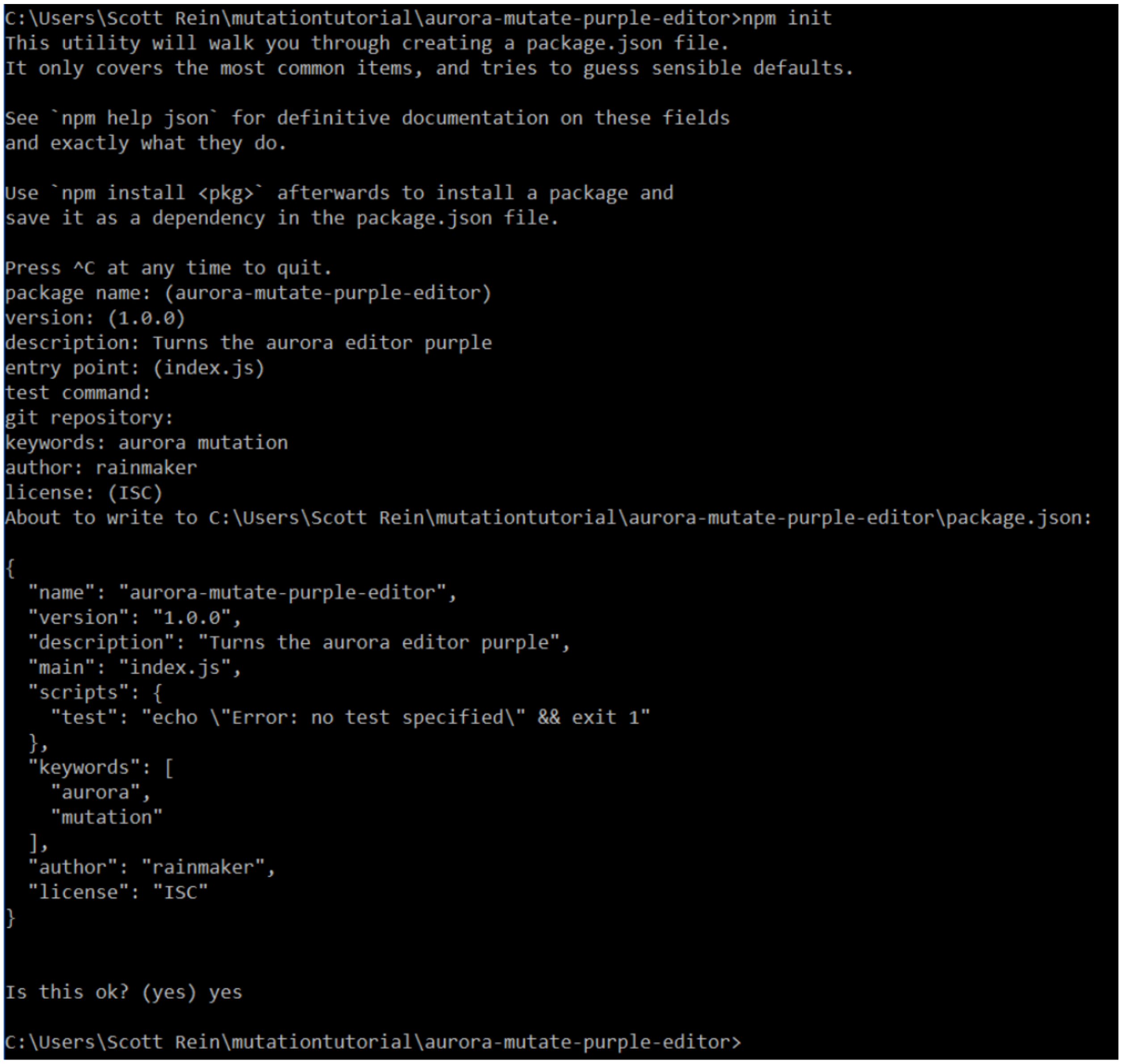
Enter the following command into your command line

```
npm init
```



```
C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>npm init
```

This command creates a package.json file and it will prompt you to enter in values for the fields of the `package.json`. The default values are fine to use.

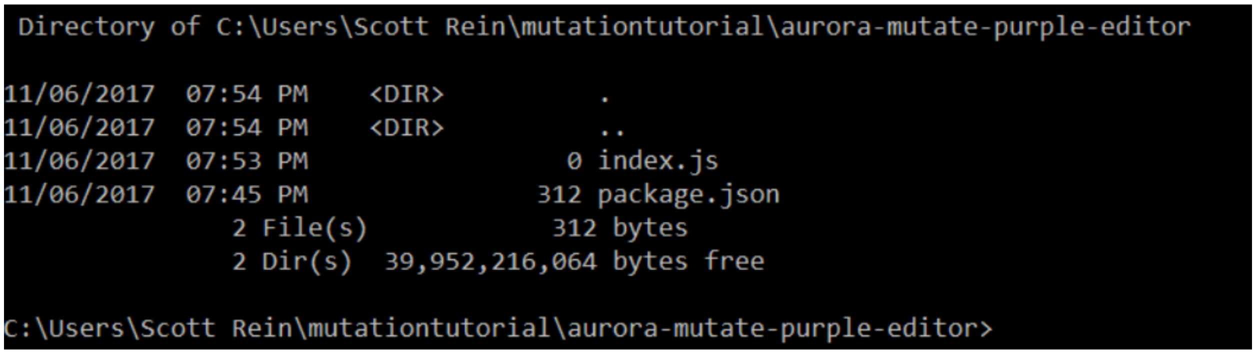


```
C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (aurora-mutate-purple-editor)
version: (1.0.0)
description: Turns the aurora editor purple
entry point: (index.js)
test command:
git repository:
keywords: aurora mutation
author: rainmaker
license: (ISC)
About to write to C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor\package.json:

{
  "name": "aurora-mutate-purple-editor",
  "version": "1.0.0",
  "description": "Turns the aurora editor purple",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "aurora",
    "mutation"
  ],
  "author": "rainmaker",
  "license": "ISC"
}


Is this ok? (yes) yes

C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>
```

If you need more help with npm check out their tutorial: npm tutorial

1.2 Creating the Mutation

After you create the package.json file, then you can create your mutation. Create another file in the same folder that your `package.json` is in. It needs to be titled the same as what you put in the `entry point` field when creating your `package.json`. In this running example that file is `index.js`.



```
 Directory of C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor

11/06/2017  07:54 PM    <DIR>          .
11/06/2017  07:54 PM    <DIR>          ..
11/06/2017  07:53 PM                 0 index.js
11/06/2017  07:45 PM               312 package.json
               2 File(s)            312 bytes
               2 Dir(s)  39,952,216,064 bytes free

C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>
```

Your mutation will need to be written in React, which is a framework for javascript. If you do not already have an editor that works well with React, you can get Atom or VScode.

Here is an example on how to mutate the editor of Aurora to turn it purple

```
import React from "react";
import styled from "styled-components";

const purple = `
  color: #8B008D;
`;

function purpleEditor(Editor) {
  return class extends React.Component {
    render() {
      return (
        <Editor style={purple}>{this.props.children}</Editor>;
      );
    }
  };
}

module.exports.mutations = {
  Editor: purpleEditor
};
```

Here is an example on how to mutate the frame of Aurora to replace it with a flying cat gif

```
import React from "react";
import styled from "styled-components";

const EVERYWHERE = styled.img`
  position: fixed;
  top: 0;
  left: 0;
  min-width: 100%;
  min-height: 100%;
`;

function AddKitty(Frame) {
  return class extends React.Component {
    render() {
      return (
        <Frame>
          <EVERYWHERE src="https://media.giphy.com/media/VxbvpfaTTo3le/giphy.gif" />
          {this.props.children}
        </Frame>
      );
    }
  };
}

module.exports.mutations = {
  Frame: AddKitty
};
```

Here is the github repository for this example: flying cat

There are other examples in this repository that you can look at.

The import statements used are to import react and styled components. React is the language used to write the mutation and styled components allow us to change how the components(Editor and Frame) look. The constants (`const`) are used to style the components using css. If you want a more in depth guide for writing React check this out:

https://reactjs.org/docs/hello-world.html

For more information about the React technique used to write mutations check this out:

https://reactjs.org/docs/higher-order-components.html

1.3 Building with Webpack

Webpack is a bundler, and we're going to use it to bundle your mutation once you have written it. You can install Webpack by entering this onto your command line

`npm install webpack -g`

Next you'll have to create a `webpack.config.js` file. This file will tell webpack what it should do. Here is an example of what it should look like.

```
const path = require("path");

module.exports = {
  entry: "./index.js", //tells webpack where to start
  output: {
    filename: "./build/bundle.js", //tells webpack the output
    libraryTarget: "commonjs2"
  },
  module: {
    loaders: [
      { test: /\.js$/, loader: "babel-loader", exclude: /node_modules/ },
      { test: /\.jsx$/, loader: "babel-loader", exclude: /node_modules/ }
    ]
  }
};
```

Next, you'll have to install some babel dependencies by running the following command on your command line:

```
npm install --save-dev babel-core babel-preset-env babel-preset-react
babel-loader
```

Then, you'll have to bundle in react and styled components by running the following command:

`npm install --save react styled-components`

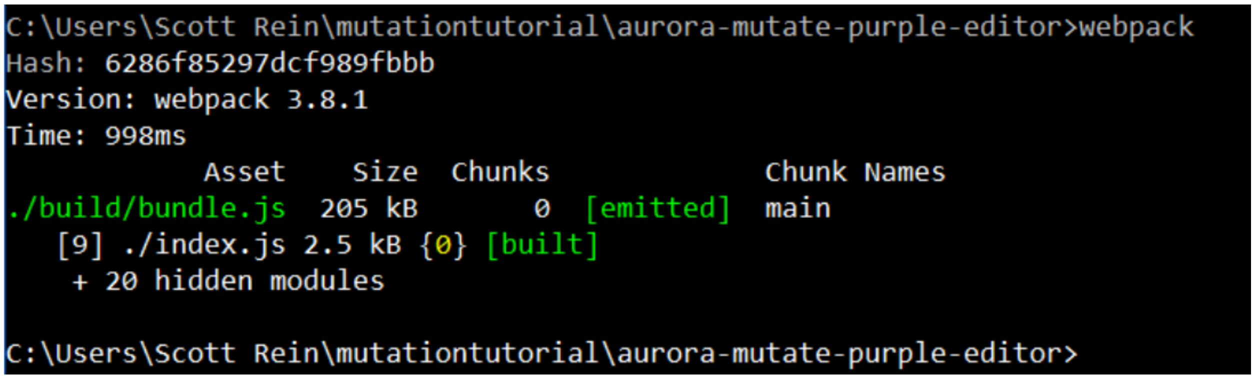
Your `package.json` file should now look something like this

```
{
  "name": "aurora-mutate-purple-editor",
  "version": "1.0.0",
  "description": "Turns the aurora editor purple",
  "main": "build/bundle.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "aurora",
    "mutation"
  ],
  "author": "rainmaker",
  "license": "ISC",
  "dependencies": {
    "react": "^16.1.0",
    "styled-components": "^2.2.3",
    "webpack": "^3.8.1"
  },
  "devDependencies": {
    "babel-core": "^6.26.0",
    "babel-loader": "^7.1.2",
    "babel-preset-env": "^1.6.1",
    "babel-preset-react": "^6.24.1"
  }
}
```

After you write your config file and run the previous npm commands, you'll have to create a `.babelrc` file in your main mutation folder. The contents of that file should be exactly as follows.

```
{
  "presets": ["env", "react"]
}
```

Then run the command `webpack` on your command line. This will create a build folder and a bundle.js file.

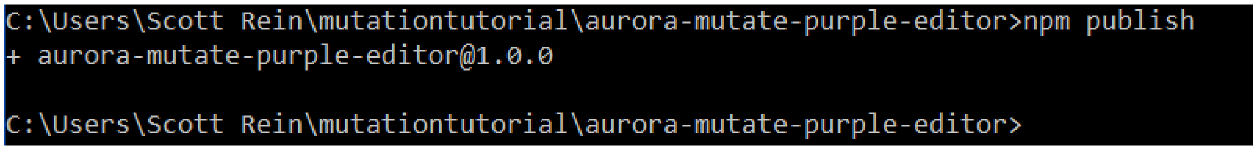


```
C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>webpack
Hash: 6286f85297dcf989fbbb
Version: webpack 3.8.1
Time: 998ms
            Asset    Size  Chunks             Chunk Names
./build/bundle.js  205 kB       0  [emitted]  main
   [9] ./index.js 2.5 kB {0} [built]
    + 20 hidden modules

C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>
```

1.4 Publishing your npm package

To publish a npm package you have to register yourself as a user. If you have already created your account on the npm website use the command `npm login` to login to your account. If you have not created an account yet use the command `npm adduser` to create your npm account. The next step is to publish your npm package by using the command `npm publish`. This command will publish everything in the current directory, so make sure you are in the right spot before running this command.

```
npm adduser //creates a new account for npm
npm login   //logs onto npm
npm publish //publishes your npm package
```



```
C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>npm publish
+ aurora-mutate-purple-editor@1.0.0

C:\Users\Scott Rein\mutationtutorial\aurora-mutate-purple-editor>
```

1.4.1 Updating your npm package

After publishing your package you can also update it by using the `npm version <update-type>` command, where `<update-type>` is either patch, major, or minor. Then run `npm publish` to publish your changes.

```
npm version <update-type>   //updates your npm package
npm publish                 //publishes your updated package
```

1.5 See in the Aurora Mutations Store

Once you publish your npm package your mutation will be available for all of the Aurora users to add to their version of the application!