Slot Machine

Slot Machine test app. Live here

First Run

Using docker

The recommended option is to use Docker.

```
// Launch App
docker-compose up
```

If you are not familiar with Docker, or don't have it setup, use plain old NPM and Node

```
// Install dependencies
npm i
npm i -g now

// Launch App
now dev
```

Running Tests

Add appropriate selenium webdriver to your PATH before running end-to-end tests. Executables for windows are included in the 'webdrivers' folder (for chrome and firefox). Simply add that folder to your PATH if you are on windows. if you use a mac or linux (or use a different browser), you can get versions for your machine/browser here

To allow selenium take control of your chrome browser make sure you have this key on your registry: Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Google\Chrome\MachineLevelUserCloudPolicyEnrollmentToken. If you need help adding this key see this guide

NB: These tests are far from extensive or complete and only serve the purpose of highlighting what can be done.

```
// Unit tests
npm test

// For coverage report, run
npm run test:coverage

// UI tests (with Selenium)
// Remember to download and add appropriate selenium webdriver to your PATH.
now dev // If the server isn't running already
```

```
npm run test:ete
```

Notes

Why React?

This application involves simple UI components that need to communicate with one another as their states might depend on their siblings. The application is lightweight, single-paged (at least for now), and doesn't require a backend (at least for now). React is perfect for this, as it makes managing state and UI updates efficient and performant. It also makes component isolation easy.

Why NextJS?

Next.JS is a React framework that adds power to the React dev ecosystem. It brings in SSR, HMR amongst other powerful features by default. This allows developers to focus on the application logic itself. I will admit though, that for an application like this one, NextJS is a bit of an overkill. However, it integrates well with NOW and it's serverless architecture. NOW is where I host most of my applications including Slot Machine. That seamless integration was enough reason to bring NextJS into the picture, but it also sets a great foundation should further development be need for this (I might decide to take things a bit further 😃).

Why NOW?

Serverless architecture is the future. It is simply amazing. You should try it 😃

Why Web Animations API

As you will notice, I use the Web Animations API to animate the spinning reel motion. Alternative approaches like CSS Keyframes, Javascript scrolling or packages like react-spring were either too limited (requiring too much setup/trickery) or too expensive (overkill for the task). The Web Animations API, though relatively new, is native and has fair browser compatibility.

NB: When implemented, the Element.getAnimations() method will replace the setTimeout() method used in the slot component.

Code

- I tend to avoid too many code comments. However, you will find key notes on certain blocks of the code. Everything else should be relatively easy to follow.
- I use ESLint for linting and have included a copy of my ESLint config file in the git repo.
- I develop on a wide screen, so my lines are usually more than 80 characters long.
- I use **{}** if there's more that one line in the block.
- Semi-colons are not necessary, but i still use them. Habits from Java 😃
- I user ternary operators sparingly.

Improvements?

Definitely! There's always room. Plus, I'm doing this on vacation, I can barely focus - almost didn't do it at all 😀

Author

Adombang Munang Mbomndih (Bomdi)

dzedock@yahoo.com

Website