UI DEVELOPMENT WITH STORYBOOK

A STORY ABOUT DEVELOPER HAPPINESS

Edwin Shin

- Partner at TRA
  - Data Science Consultancy
  - Data Collaboration Platform
- eddie@tra.sg

We're Hiring!

https://tra.sg

Storybook is a UI development environment for your UI components. With it, you can visualize different states of your UI components and develop them interactively.

Storybook is like unit-testing for UI development.

Remember why we embrace unit testing?

- It forces us to reflect on the design of our code (e.g. weighing the cost of initial development versus ease of maintenance and refactoring, modularity and re-use)

- Cleaner code, because (re)writing our code to be testable is often easier than writing complicated tests with lots of mocks and simulated state

- Validation that our code is working as intended and assurance that we haven't introduced regressions

Storybook encourages you to develop presentational components in isolation from the context of your full app (with all of its fully-loaded Redux/Relay/Apollo goodness).

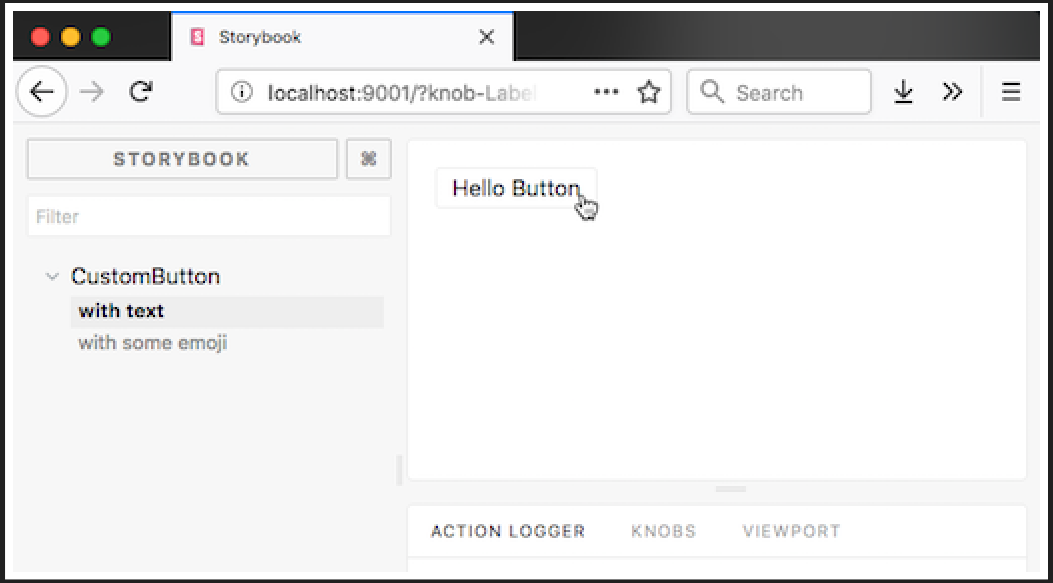
Storybook can run on its own (with hot-reloading), separate from your React app or even produce a static app so that you can collaborate with (or showcase to) non-developers (designers, QA, clients, etc.).

WRITING A STORY

Once you have Storybook installed (`npm install ...`) and configured (`.storybook/config.js`), writing a story is like writing a test for a component:

```
import React from 'react';
import { storiesOf } from '@storybook/react';
import CustomButton from './CustomButton';

storiesOf('CustomButton', module)
  .add('with text', () => (
    <CustomButton onClick={action('clicked')}>Hello Button</Cu
  ))
  .add('with some emoji', () => (
    <CustomButton onClick={action('clicked')}><span role="img"
  ));
```



Similar to Jest, I configure Storybook to load stories that end with a specific suffix (`.stories.js`). So for the component `CustomButton.js`, I'll create my story as `CustomButton.stories.js`:

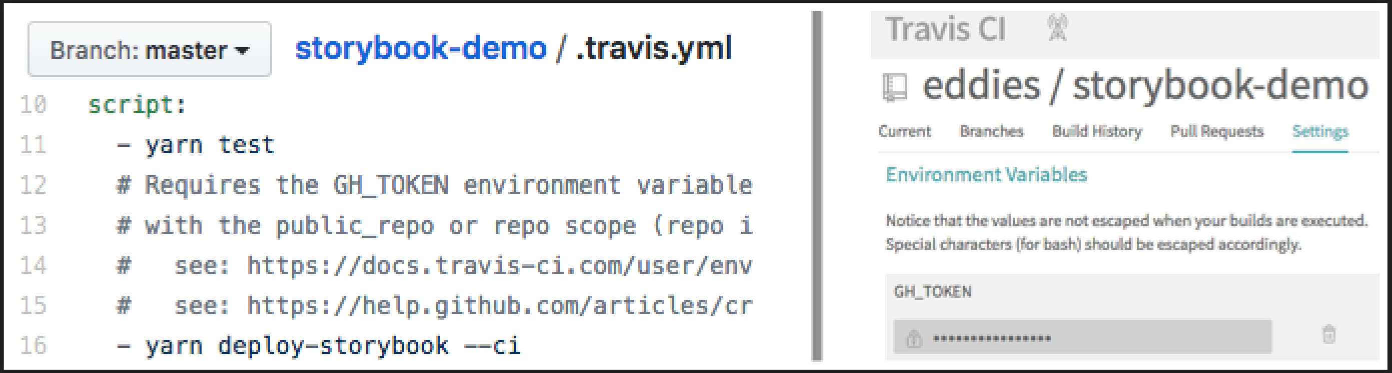
```
src/
└───components/
    └───CustomButton/
        │    CustomButton.js
        │    CustomButton.stories.js
        │    CustomButton.test.js
```

CI & TESTING

Storybook can generate a static app. This means we can add Storybook to your CI (e.g. Travis) build.

Even better, have your CI deploy on every build and now you have an automatic component showcase for every end-of-iteration/sprint review.

In this demo repo, I use storybook-deployer along with Travis to publish Storybook to GitHub Pages.


Branch: master storybook-demo / .travis.yml

```
script:
  - yarn test
  # Requires the GH_TOKEN environment variable
  # with the public_repo or repo scope (repo i
  #   see: https://docs.travis-ci.com/user/env
  #   see: https://help.github.com/articles/cr
  - yarn deploy-storybook --ci
```

Notice that the values are not escaped when your builds are executed. Special characters (for bash) should be escaped accordingly.

GH_TOKEN
••••••••••••••••

STRUCTURAL TESTS WITH STORYSHOTS

StoryShots is an addon that integrates Storybook and Jest Snapshot Testing.

StoryShots isn't a replacement for unit or integration tests. When a snapshot test fails, it just means the story doesn't look like it did before.

When you're actively developing a component you're just going to update the snapshots without much consideration or effort. But StoryShots is most useful in making sure developers don't introduce unintended or breaking changes in the rendered output (i.e. not behavior) of components.

COLLABORATION

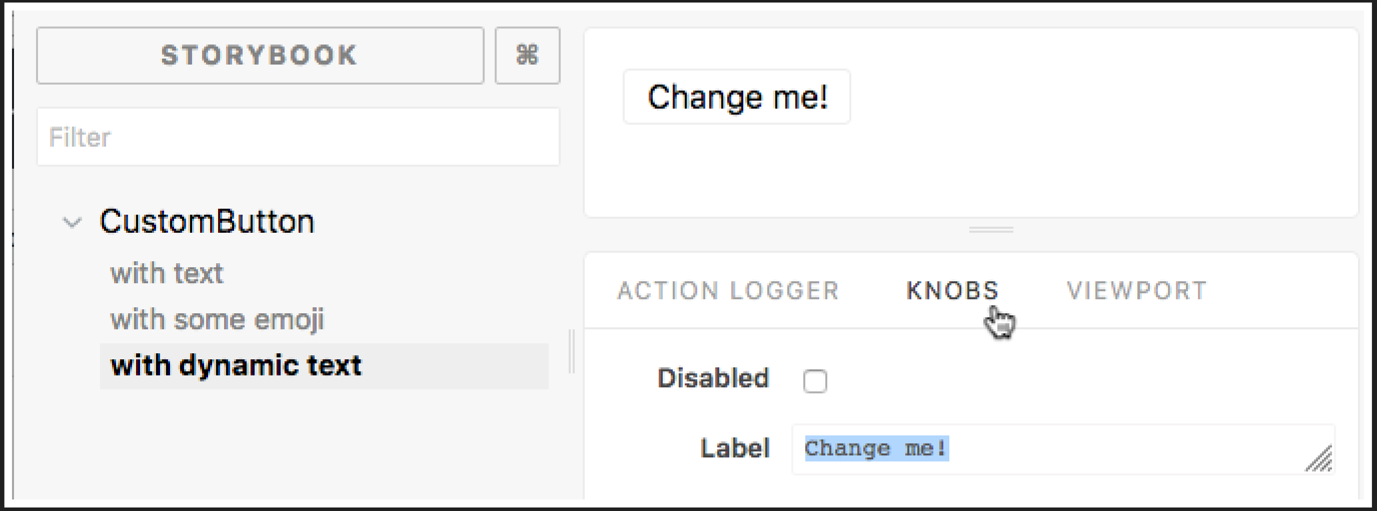
Once you're publishing your stories via CI, it's also easy to adopt a more collaborative workflow where you push a branch and send a link to the static storybook build so you can get feedback on more experimental UI elements independently of getting your middleware and backend all wired up.

ADDONS

Storybook has a number of addons developed by Storybook maintainers and the community at large.

I find Knobs and Viewport to be especially useful.

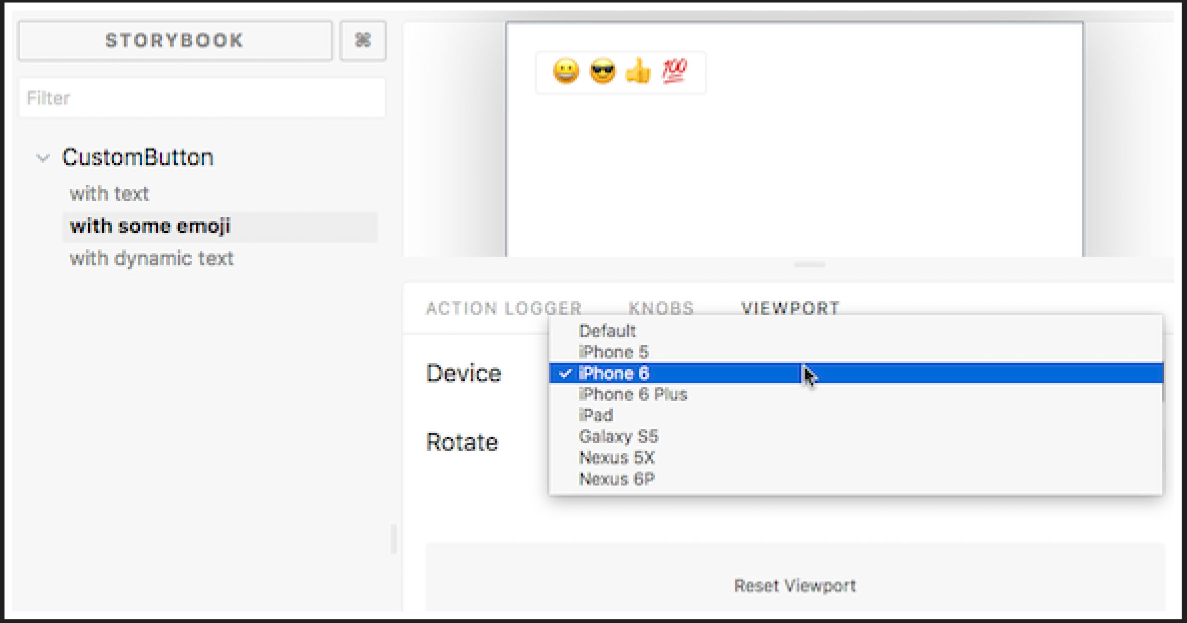
Knobs lets you edit React props within the Storybook UI.



Disabled ☐

Label `Change me!`

Viewport helps you build responsive components by displaying your stories in various sizes and layouts. In v4 (in alpha as of this writing), viewports will support custom sizes.



EXAMPLES

- Airbnb Dates
- Uber React-Vis
- this repo