Putting it All Together: Components and Props

New Attempt

Due No Due Date   **Points** 1   **Submitting** a website url

FORK (https://github.com/learn-co-curriculum/react-hooks-component-props-mini-project/fork)

(https://github.com/learn-co-curriculum/react-hooks-component-props-mini-project)

(https://github.com/learn-co-curriculum/react-hooks-component-props-mini-project/issues/new)

Learning Goals

- Create components that return JSX
- Use props to make components dynamic
- Transform lists of data into lists of components

Overview

Now that you've learned how to work with components in React, it's time to build something and put those skills to use! Your goal for this lab is to make a *static site* in React to practice building components, writing JSX, and passing down data as props.

We'll be creating a personal blog site, similar to **Dan Abramov's Overreacted (https://overreacted.io/)** :

(?) **Help**

There is some starter code available in `src/components/App.js` . There is also some data in `data/blog.js` that is being imported into `App` so you can pass it down to the components that need it.

Deliverables

Have a look at the components below and draw out a component hierarchy so you can determine how to pass data down as props.

Header

Make a `Header` component as a child of `App` . It should return:

- a `<header>` element with the following elements inside:
  - an `<h1>` with the *name* of the blog, passed as a prop

⑦ Help

About

Make an `About` component as a child of `App`. It should return:

- an `<aside>` element with the following elements inside:
  - an `<img>` element, with the `src` set to an *image* passed as a prop
  - the `<img>` element should use this placeholder image as a *default value* for the prop if no prop is passed in: "**https://via.placeholder.com/215** **(https://via.placeholder.com/215)** "
  - the image should also be accessible! Give it an `alt` attribute of "blog logo"
  - a `<p>` element, with the *about* text for the blog passed in as a prop

ArticleList

Make an `ArticleList` component as a child of `App`. It should return:

- a `<main>` element with the following components inside:
  - an array of `Article` components (one component for each *post* passed down as a prop to `ArticleList`)
  - make sure to assign a unique `key` prop to each `Article`

Article

Make an `Article` component as a child of `ArticleList`. It should return:

- an `<article>` element, with the following elements inside:
  - an `<h3>` element displaying the *title* of the article, passed as a prop
  - a `<small>` element displaying the *date* of the article, passed as a prop
  - a *default value* of "January 1, 1970" should be used if no date is passed as a prop
  - a `<p>` element displaying the *preview* of the article, passed as a prop

Bonus Feature: 'Minutes to Read'

You'll notice in the original **Overreacted** **(https://overreacted.io/)** site, there's a 'minutes to read' indicator next to each article.

If the article takes less than 30 minutes to read:

- For every 5 minutes (rounded up to the nearest 5), display a coffee cup emoji. For example, if the article takes 3 minutes to read, you should display "☕ 3 min read". If the article takes 7 minute, you should display "☕☕ 7 min read".

If the article takes 30 minutes or longer to read:

- For every 10 minutes (rounded up to the nearest 10), display a bento box emoji. For example, if the article takes 35 minutes to read, you should display "🍱🍱🍱🍱 35 min read". If the article takes 61 minutes to read, you should display "🍱🍱🍱🍱🍱🍱🍱 61 min read".

There aren't tests for this feature, so you'll have to rely on running the code in the br ⑦ **Help** your implementation works!

? Help