Ingo Steinke
Posted on 22 Jun 2021 • Updated on 9 Dec 2021

No need for Virtual DOM and Controlled Components

#javascript #react #css #a11y

Building a Reading List web app with Node, Preact, and Tailwind CSS (2 Part Series)

1. MERN-Stack Setup: Building a Reading List Web App with Node, …
2. **No need for Virtual DOM and Controlled Components**

This is a work in progress, last updated on 1st July 2021, part of the series "Building a Reading List web app with Node, Preact, and Tailwind CSS".

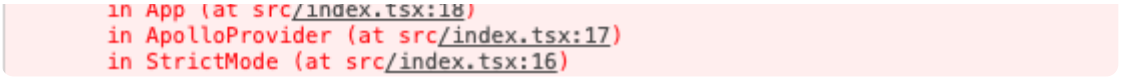
What are Controlled Components

One of the most annoying misfeatures of typical react apps, in my personal view as an old-school web developer, is the concept of "controlled input" or, more generally, "controlled components" where you take the control away from the native web forms to intercept user input and update the form field using React.

Controlled components for no real reason?

There are plenty of posts explaining what Controlled Input is and how to use it, but all of them fail to give me a valid reason **why** (actually they do, they want the SPA to be the "single source of truth" while in my view, the DOM should be the single source of truth at least for any user input.

```
▸Warning: A component is changing an uncontrolled input of type   index.js:1
text to be controlled. Input elements should not switch from uncontrolled to
controlled (or vice versa). Decide between using a controlled or uncontrolled
input element for the lifetime of the component. More info: https://fb.me/rea
ct-controlled-components
    in input (at App.tsx:45)
    in div (created by styled.div)
    in styled.div (at App.tsx:44)
```



Controlled Components in my experience add a minimal delay i.e. sluggishness to the front-end and make working with form elements less straightforward and less accessible, as they basically throw away what the browser already has to offer only to rebuild it in an often worse way.

Styling Form Elements

For example there is a React plugin called react-select. At least I could agree to the purpose that CSS support for form elements used to be quite limited in browsers, especially for dropdowns.

But react-select turned out hard to maintain in the project where we used it, and finally we were happy to replace every dropdown, most of which only had 2 - 3 entries anyway, with checkboxes, for the sake of user experience and to save us more painful work with incompatibilities when updating either the plugin or React itself.

HTML form elements can be styled in an accessible way using only CSS, all that you need to do is copy and modify the styles found in well-written articles like Custom Styling Form Inputs With Modern CSS Features.

Virtual DOM

React uses a virtual Document Object Model to prepare and check any changes before applying them to the actual DOM. Another annoyance for me as an old-school developer who even used to do DOM transformations with XSLT for many years and who still thinks, that the actual DOM should have been the single source of truth, but ...

State Management

... of course in a single page application that receives back-end data and user input asynchronously, there has to be state management to properly handle this influx of data without creating race conditions.

But this is still no reason to have an additional internal representation of any element that might get re-rendered at any time, when browsers can already handle that.

Browsers can already handle that!

Back in time around 2005, "Web 2.0" was a buzz about interactive websites like FlickR and YouTube not only for their social interactions, but also for using AJAX to update parts of a website without reloading the whole page.

Ten years before, there had been framesets for a similar experience, but with AJAX and modern stylesheets, Web 2.0 got the job done in a visually appealing way and it still does.

There are JavaScript promises, async and await, and the Fetch API and of course, web forms can be styled using CSS much better than ever before.

Browsers can already handle that!

Enter Preact

An appealing aspect about Preact is its minimalism. There is no virtual DOM, no unnecessary code and no unnecessary rendering, at least this is the goal that the Preact developers strive for.

TO BE CONTINUED

More content will be added here as you see me commit and push and update this article...

Building a Reading List web app with Node, Preact, and Tailwind CSS (2 Part Series)

1 MERN-Stack Setup: Building a Reading List Web App with Node, …

2 **No need for Virtual DOM and Controlled Components**

Discussion (0)

Code of Conduct • Report abuse

Ingo Steinke

Web Development, Sustainability, Art and Music, Nature and Travel, Sustainability

LOCATION
Germany

WORK
Creative Web Developer at Ingo Steinke

JOINED
21 Sep 2019

More from Ingo Steinke

Animated Gradient Text Color
#webdev #showdev #css #tutorial

Aspect ratio: no need for container units!
#css #html #webdev #todayilearned

CSS :has(.parent-selectors)
#css #webdev #todayilearned #javascript