

ToiletPaper #161

Preact – Eine leichtgewichtige Alternative zu React

Autor: Philipp Miller / Software Engineer / Automotive

✗ Problem

React ist großartig, aber auch groß. Es versucht den DOM so weit wie möglich zu abstrahieren, um Entwickler*innen vor browserspezifischen Kompatibilitätsproblemen zu bewahren – selbst wenn diese inzwischen durch verbesserte Standards geradegezogen wurden. Beispielsweise werden die vom Browser generierten Events in React nicht einfach durchgereicht, sondern durch eigene Objekte ersetzt. Das schlägt sich auch in der Performance und Bundle-Größe nieder.

✓ Alternative

Preact [1] baut die API von React weitgehend treu nach – auf den ersten Blick ist es von React kaum zu unterscheiden. Die üblichen Hooks stehen zur Verfügung (plus ein sehr bequemer `useErrorBoundary`-Hook). Viele 3rd-party Librarys für React funktionieren einfach so mit Preact (z. B. Material-UI [4]).

Welche Unterschiede [2] am wichtigsten sind, hängt vom Anwendungsfall ab. Hier sind einige:

- Preact ist komplett in TypeScript geschrieben [3] – die Types selber sind dabei häufig unterschiedlich zu denen in React. Beispielsweise gibt es anstelle von `ReactDOM` und `FunctionComponent<P>` andere Typen wie `ComponentChildren` und `VNode<P>`.
- Event-Listener bekommen als Parameter browser-native Event-Objekte. Das hat auch zur Folge, dass diese nicht durch Portale bubbled.
- Die Imports sind etwas anders strukturiert (siehe Beispiel).

➔ Beispiel

```
import { Fragment, h, render } from "preact"
import { useState } from "preact/hooks"

const App = () => {
  const [count, setCount] = useState(0)
  const increment = () => setCount(n => n + 1)
  return (
    <>
      <h1>Clicks: {count}</h1>
      <button onClick={increment}>Add one</button>
    </>
  )
}

render(<App />, document.getElementById("app"))
```

+ Vorteile

- Gute Dokumentation, die beim Umstieg hilft.
- Klein. Zur Veranschaulichung habe ich eine minimale Hello-World App gebaut:

React	Preact
129.23 KiB gzip: 41.68 KiB	11.08 KiB gzip: 4.54 KiB

- Weniger DOM-Abstraktion, was nur ein Problem sein sollte, wenn man besonders alte Browser unterstützen muss.
- In modularem TypeScript implementiert.

Nachteile

- Die API ist nicht komplett identisch (vor allem bei den Types). Das kann ein bisschen Umlernen erfordern. Einige Features, die für mehr React-Kompatibilität nötig sind (z. B. in 3rd-party Librarys), sind über das `preact/compat` package verfügbar.
- Möglicherweise Kompatibilitätsprobleme mit komplexeren Libraries, die auf React-Internas zugreifen. Gerade für die größeren Libraries gibt es aber bereits häufig Tutorials oder Adapter, sofern diese benötigt werden (z. B. `mobx-preact`).

[1] <https://preactjs.com>

[2] <https://preactjs.com/guide/v10/differences-to-react>

[3] <https://preactjs.com/guide/v10/typescript>

[4] <https://twitter.com/preactjs/status/1152267975078154240>