Statemanagement, Context API

Agenda

- Propdrilling
- Statemanagement
- Context API

Statemanagement

React är jättebra, jättekul och relativt enkelt så länge man arbetar med små och få komponenter….

Men så snart vi bygger mer komplexa applikationer desto svårare blir det att hantera state

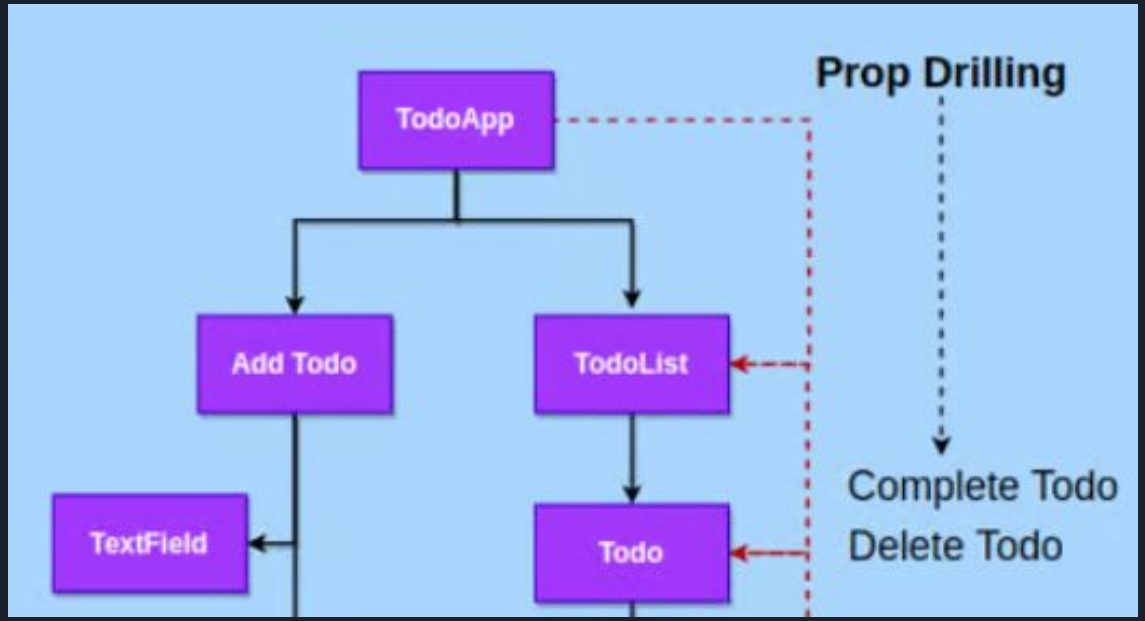
Propdrilling

Demo på propdrilling

https://www.w3schools.com/react/showreact.asp?filename=demo2_react_context1

Propdrilling

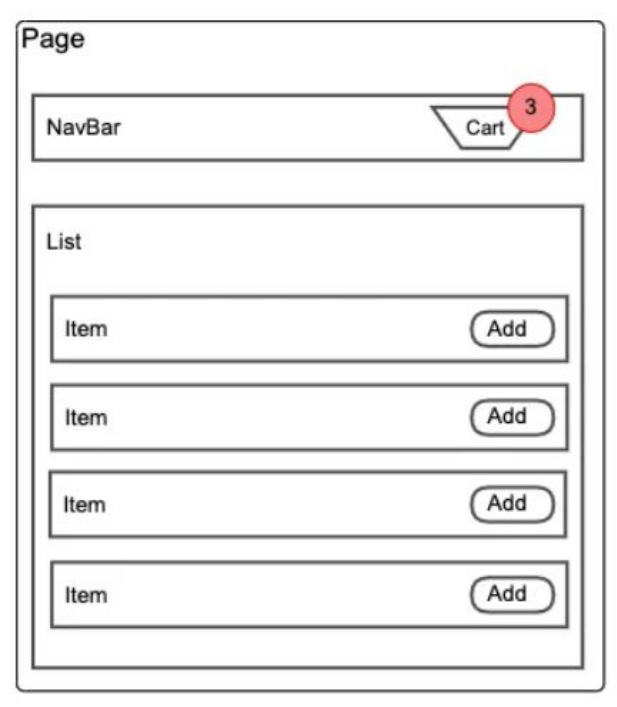
Propdrilling

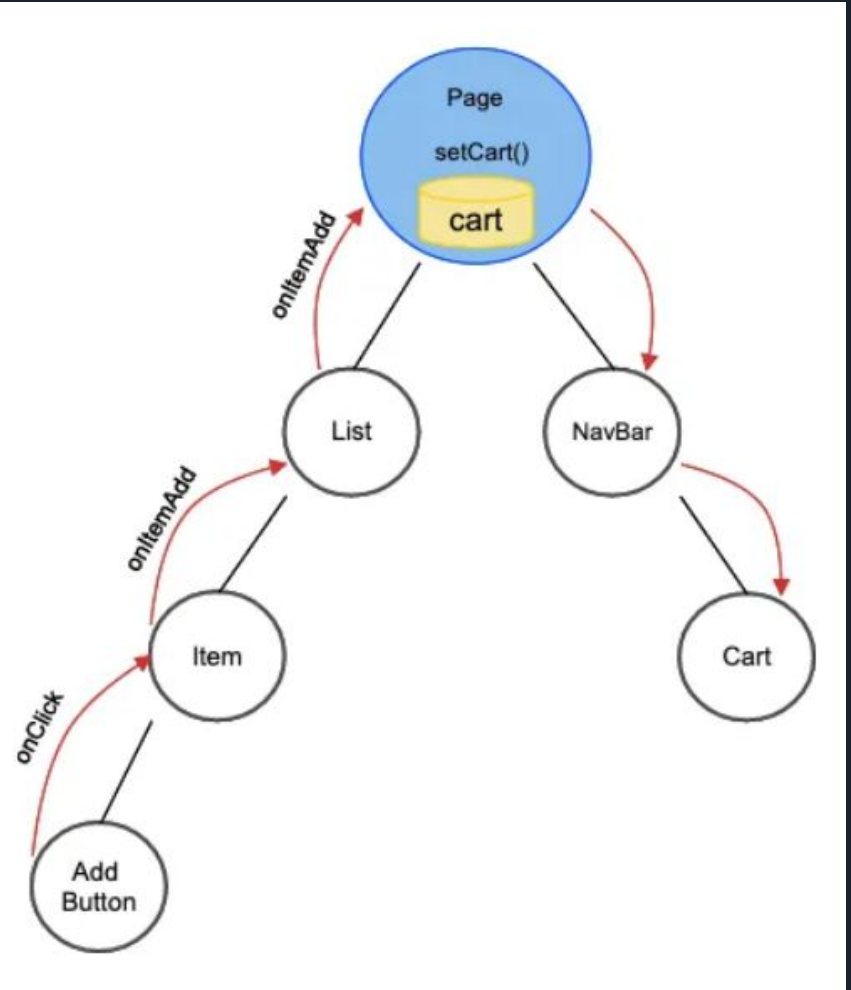


Det vanligaste sättet att hantera state är att ha allt globalt state i förälderkomponenter och skicka ner setters och state till barnen för att ändra

Propdrilling

Demo på propdrilling





Propdrilling

Demo på app med prop drilling

https://phungnc.medium.com/react-from-prop-drilling-to-use-hook-to-pass-data-in-app-3e3caad6a65f

Propdrilling

Övning:

- App
  - Lägg in globalt state (messages = []) en lista med texter
  - Renderar: 2 barnkomponenter
    - AddMessage
    - DisplayMessages
- AddMessage
  - En prop, setMessages som skickas från App
  - Renderar: Ett formulär med en textinput och submit knapp för att lägga till “message” i messages
- DisplayMessages
  - En prop, messages som skickas från App
  - Renderar: En lista med alla meddelanden i messages

Statemanagement

Det går naturligtvis att använda sig av “propdrilling” för att hantera state i en app men det blir svårare ju större applikationen blir och ju mer state man introducerar

Därför behöver vi något nytt för att hantera detta problem…

Statemanagement

Här är några externa bibliotek som är populära att använda tillsammans med React

Statemanagement

Det finns även ett sätt som har ökat i popularitet de senaste åren, som inte använder sig av externa bibliotek utan använder sig av Reacts inbyggda Context API

Context API

Demo på Context API

https://www.w3schools.com/react/showreact.asp?filename=demo2_react_context2

Context API

Steg för att skapa och använda Context API

1. Skapa ett “Context”
2. Skapa en “Provider” med hjälp av ditt context
3. Konsumera ditt context varifrån som helst i appen

Context API

1. Skapa ett “Context”

```
import { createContext } from "react";

const MyContext = createContext("defaultValue");
```

Context API

2. Skapa en “Provider” med hjälp av ditt context

```
const MyContext = createContext("defaultValue");

<MyContext.Provider value="Hello, World!">
  <YourApp />
</MyContext.Provider>;
```

Context API

3. Konsumera ditt context varifrån som helst i appen

```
import { useContext } from "react";
import { MyContext } from "./App"

const value = useContext(MyContext);
```

Context API

Demo på Context API där Context separeras till en egen fil så att den kan importeras på olika ställen

Exportera

- Provider komponent (med text)
- Context (för att kunna konsumera text)

Context API

Övning

- Skapa en fil (TextContext.js)
  - const TextContext = createContext()
  - exportera TextContext och Provider
    - export default TextContext.Provider
    - export { TextContext]
- Tillför en Provider runt App
  - import TextContextProvider from “./TextContext.js”
  - <TextContextProvider value=”test”> ... </TextContext.Provider>
- I någon barnkomponent konsumera värde
  - import { TextContext } from “./TextContext.js”
  - const text = useContext(TextContext)

Context API

Demo av föregående app med Contex API

https://codesandbox.io/embed/react-context-ooh65?fontsize=14