Context

Нижче наведені ілюстративні приклади!

<https://reactjs.org/docs/context.html>

Context provides a way to pass data through the component tree without having to pass props down manually at every level

Контексти призначені для передачі даних дочірнім компонентам без потреби ланцюгової передачі даних по дереву елементів від джерела даних до адресата.

Використання одного об’єкта контексту

|  |  |
| --- | --- |
|  | Приклад. Передача даних від |
| Створення об’єкта контексту  const контекст = React.createContext(знач.за замовчуванням); | const ThemeContext = React.createContext('light'); |
|  | // ----- Базовий елемент з даними ------  class App extends React.Component {  render() {  return ( |
| // Задання даних у контексті  < контекст .Provider value = значення\_що\_передаємо >  . . . дочірні елементи . . .  </ контекст.Provider >  дочірні елементи оновлюються кожного разу, коли змінюється значення у контексті | <ThemeContext.Provider value="dark">  <Toolbar />  </ThemeContext.Provider>  );  }  }  // ------ Проміжковий елемент ----- |
|  | function Toolbar(props) {  return (  <div>  <ThemedButton />  </div>  );  } |
| // «Приєднання» до контексту (підписка на контекст) шляхом ініціалізації  // поля contextType  static contextType = контекст ; //Якщо всередині класу  або ж  клас . contextType = контекст ; //Якщо за межами класу  (у прикладі можна було підписатися на контекст  ThemedButton. contextType = ThemeContext;) | // ------ Цільовий елемент ------  class ThemedButton extends React.Component {  static contextType = ThemeContext; |
| // Використання даних, які знаходяться у **value**  this.context | render() {  return <Button theme={this.context} />;  }  } |

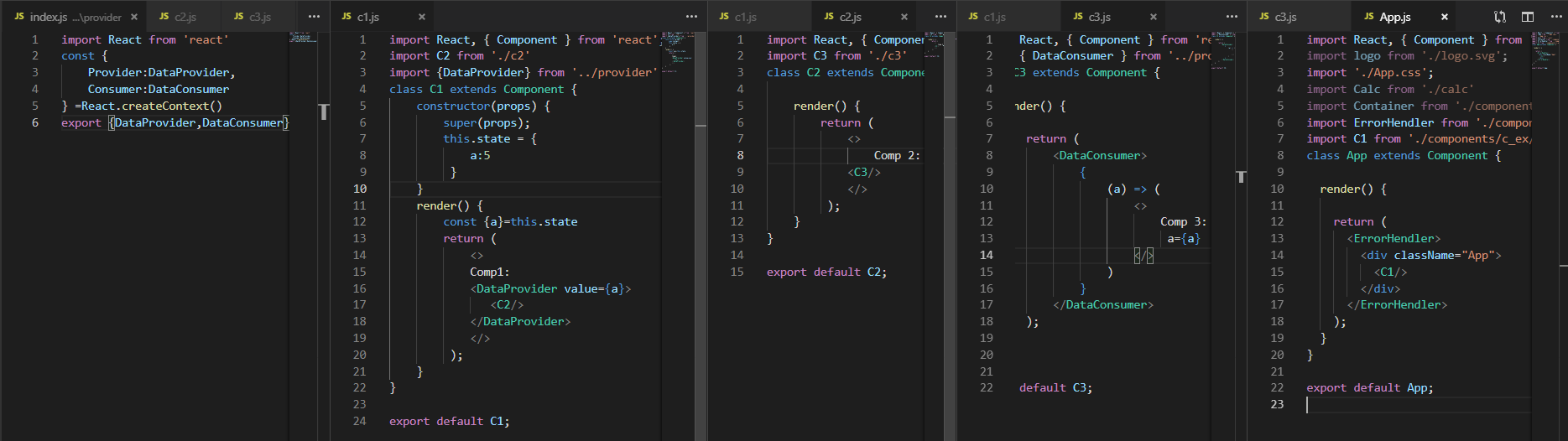
Для доступу до даних контексту можна також використати Context.Consumer

|  |  |
| --- | --- |
|  |  |
| В якості дочірнього елемента необхідно вказати функцію (як аргумент приймає значення контексту, а повертає компонент)  < контекст . Consumer>  { value =>  . . . використання контексту . . .  }  </ контекст .Consumer> | const ThemeContext = React.createContext('light');  // ----- Базовий елемент з даними ------  class App extends React.Component {  render() {  return (  <ThemeContext.Provider value="dark">  <Toolbar />  </ThemeContext.Provider>  );  }  }  // ------ Проміжковий елемент -----  function Toolbar(props) {  return (  <div>  <ThemedButton />  </div>  );  }  // ------ Цільовий елемент ------  class ThemedButton extends React.Component {  render() {  **< ThemeContext . Consumer>**  { value =>  return <Button theme={value} />;  }  **</ ThemeContext.Consumer>**  }  } |

**Використання декількох контекстів**

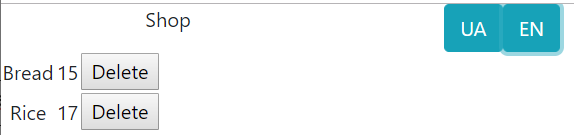
|  |  |
| --- | --- |
|  |  |
| В цьому випадку вкладаємо один тег Context.Consumer в середину інших тегів Context.Consumer  < **контекст 1** .Consumer>  { значення контекста 1 => (  <**контекст 2** .Consumer>  { значення контекста 2 => (  . . використання значень з  контекстів . . .  )}  </ контекст 2 .Consumer >  )}  </контекст 1 .Consumer> | const ThemeContext = React.createContext('light'); **//Створення 1-го**  const UserContext = React.createContext( { name: 'Guest', }); **//створення 2-го**  class App extends React.Component {  render() {  const {signedInUser, theme} = this.props;  // App component that provides initial context values  return (  **<ThemeContext.Provider** value={theme}> **//Підписка на контекст 1**  **<UserContext.Provider** value={signedInUser}> **//Підписка на контекст 2**  <Layout />  **</UserContext.Provider>**  **</ThemeContext.Provider>**  );  }  }  function Layout() {  return (  <div>  <Sidebar />  <Content />  </div>  );  }  // A component may consume multiple contexts  function Content() {  return (  **<ThemeContext.Consumer>**  {theme => (  **<UserContext.Consumer>**  {user => (  <ProfilePage user={**user**} theme={**theme**} />  )}  **</UserContext.Consumer>**  )}  **</ThemeContext.Consumer>**  );  } |

Приклад. Покроково ілюструюємо використання контекста



|  |  |
| --- | --- |
| Крок 1  import React from 'react'  const {  Provider: назва провайдера,  Consumer: назва консюмера  } =React.createContext()  export { назва провайдера, назва консюмера} | Створюємо об’єкти провайдера і консюмера  import React from 'react'  const {  Provider:DataProvider,  Consumer:DataConsumer  } =React.createContext()  export {DataProvider,DataConsumer} |
| Крок 2. За допомогою провайдера заносимо дані у контекст  < назва провайдера value={ дані }>    Дочірні компоненти,  яким передаються дані  </ назва провайдера > | У компоненті, де знаходяться дані (які хочемо передати) за допомогою провайдера передаємо дані у контекст  import React, { Component } from 'react';  import C2 from './c2'  import {DataProvider} from '../provider'  class C1 extends Component {  constructor(props) {  super(props);  this.state = {  a:5 🡨 Значення, яке хочемо передати  }  }  render() {  const {a}=this.state  return (  <>  Comp1:  <DataProvider value={a}> 🡨 Передаємо дані у контекст  <C2/>  </DataProvider>  </>  );  }  }    export default C1; |
| Крок 3. У компоненті, де потрібні дані з контексту вилучаємо дані з використанням консюмера  < назва консюмера >  {  (**value**) => (  <>  Використання **value** з контексту  </>  )  }  </ назва консюмера > | import React, { Component } from 'react';  import { DataConsumer } from '../provider'  class C3 extends Component {  render() {  return (  <DataConsumer>  {  (value) => (  <>  Comp 3:  a={value}  </>  )  }  </DataConsumer>  );  }  }  export default C3; |
|  |  |
|  |  |

Приклад. Реалізація можливості зміни мови інтерфейсу (змінюємо написи на кнопках)

|  |  |
| --- | --- |
| Створюємо об’єкти провайдера і консюмера |  |
| Опис компонента, у якому через контекст передається значення (використовуємо провайдер) | import React, { Component } from 'react';  import ProdList from '../prod-list'  import { LangProvider } from '../providers/lang-provider' 🡨Підключення провайдера  class Shop extends Component {  constructor(props) {  super(props);  this.state = {  lang: 'ua' 🡨 Поточна мова інтерфейсу  }  this.languages = { 🡨 Масив з мітками для кнопок  ua: { remBtnLabel: 'Видалити' },  en: { remBtnLabel: 'Delete' },  }  }  render() {  const products = [{ 🡨 Масив продуктів (для прикладу)  id: 1,  title: 'Bread',  price: 15  },  {  id: 2,  title: Rice,  price: 17  },  ]  return (  <>    <div className='container'>  <div lassName='row'>  <div className='col-8'>  Shop  </div>  <div className='col-2'> КНОПКИ ДЛЯ ПЕРЕМИКАННЯ МОВИ  <button className='btn btn-info'  onClick={() => this.setState({ lang: 'ua' })}>  UA  </button>  <button className='btn btn-info'  onClick={() => this.setState({ lang: 'en' })}>  EN  </button>  </div>  </div>  <div className='row'>  <LangProvider value={this.languages[this.state.lang]}> 🡨ПЕРЕДАЧА МОВИ У ПРОВАЙДЕР  <ProdList products={products} />  </LangProvider>  </div>  </div>  </>  );  }  }  export default Shop; |
| Використання значення з контекста (консюмера) | import React from 'react';  import {LangConsumer} from '../providers/lang-provider' 🡨 Підключаємо модуль з косюмером  const Product = ({product:{id,title,price, onDel}}) => {  return (  <LangConsumer> 🡨 Додаємо консюмер (через який забираємо значення)  { ({remBtnLabel})=>(  <tr key={id}>  <td>{title}</td>  <td>{price}</td>  <td><button onClick={()=>onDel(id)}>{remBtnLabel}</button></td>  </tr>  )}  </LangConsumer>  );  }    export default Product; |
|  |  |
|  |  |
|  |  |

Приклад. Створити онлайн магазин (через контекст передаємо розмір знижки).

|  |  |
| --- | --- |
| Компоненти | Товар (Product)  Список товарів (ProductTable)  Магазин (Shop)  Передбачити можливість надання скидок на усі товари (передача через контекст) |
|  | **//-------- Список товарів ------------**  **const** products=[  {  **title**:**"Prestigio SmartBook"**,  **src**:**`https://i2.rozetka.ua/goods/9002686/prestigio\_psb141c01bfh\_db\_cis\_images\_9002686551.jpg`**,  **description**:**"Prestigio SmartBook"**,  **price**:10000  },  {  **title**:**"Ноутбук HP 255 G6 "**,  **src**:**`https://i2.rozetka.ua/goods/2104309/copy\_hp\_250\_g6\_2hg31es\_597705c3e6f77\_images\_2104309039.jpg`**,  **description**:**`Акционный кредит 0.01% на 10 месяцев! Дарим 100 грн за распаковку  Экран 15.6” (1366x768) HD, матовый / AMD Dual-Core A6-9220 (2.5 - 2.9 ГГц) / RAM 8 ГБ / HDD 1 ТБ / AMD Radeon R4 / без ОД / LAN / Wi-Fi / Bluetooth / веб-камера / Windows 10 Home / 1.86 кг / черный  Подробнее: https://rozetka.com.ua/hp\_255\_g6\_2hg32es/p19644135/`**,  **price**:20000  },   {  **title**:**"Ноутбук Dell Inspiron 15 3567"**,  **src**:**`https://i2.rozetka.ua/goods/6999557/dell\_inspiron\_3567\_35fi34h1ihd\_wbk\_images\_6999557226.jpg`**,  **description**:**`Экран 15.6" (1920x1080) Full HD, глянцевый с антибликовым покрытием / Intel Core i3-7020U (2.3 ГГц) / RAM 4 ГБ / HDD 1 ТБ / Intel HD Graphics 620 / DVD+/-RW / LAN / Wi-Fi / Bluetooth / веб-камера / Windows 10 Home / 2.3 кг / черный Подробнее: https://rozetka.com.ua/dell\_inspiron\_3567\_35fi34h1ihd\_wbk/p51954204/`**,  **price**:30000  },  ];  **//============================================================ const** DiscountContext=React.createContext(0); //🡨 Створення контексту  //-----------------------------------------------------------------  **class** Product **extends** Component { //🡨 Компонент «Товар»   constructor(props) {  **super**(props);  }  **static** *contextType*=DiscountContext; //🡨 Приєднання до контексту  render() {  **return** (  <**div style=**{{**display**:**"inline-block"**,**width**:**"310px"**, **border**:**"2px solid black"**, **float**:**"left"**}}>  <**h1**>{**this**.**props**.**data**.**title**}</**h1**>  <**img src=**{**this**.**props**.**data**.**src**} **alt="Продукт" style=**{{**width**:**"100px"**}}/>  <**p**>{**this**.**props**.**data**.**description**}</**p**>  //--------- Зчитування даних з контексту **this**.**context (знижка)**  <**p**>Price={**this**.**props**.**data**.**price**\*(1-**this**.**context**/100)}</**p**>  <**p**>Знижка: {**this**.**context**}</**p**>   </**div**>  )  } }  //--------------------------------------------------- **class** ProductTable **extends** Component{ //🡨 Список товарів  constructor(props) {  **super**(props);  }   render(){  **return** (  **this**.**props**.products.map((prod,i)=>({...prod,**id**:i})).map((prod)=> <**Product key=**{prod.**id**} **data=**{prod}/>)  )  } } //--------------------------------------------------- **class** Shop **extends** Component{ //🡨 Магазин   constructor(props) {  **super**(props);  **this**.**state**={  **discount**:0  }  **this**.**inp**=React.createRef();  }  onClick()  {  **this**.setState({  **discount**: +**this**.**inp**.**current**.**value** })  }  render()  {  **return** (  <**div**>  <**DiscountContext.Provider value=**{**this**.**state**.**discount**}> //🡨 Підписування на контекст  <**div**>  <**label**>  Знижка: <**input ref=**{**this**.**inp**} **type="number"**/>  </**label**>  <**button onClick=**{**this**.onClick.bind(**this**)}>  Застосувати знижку  </**button**>  </**div**>  <**div**>  <**ProductTable products=**{products}/>  </**div**>  </**DiscountContext.Provider**>  </**div**>  )  } } |

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |