Lifting State Up (передача даних вверх для батьківських елементів) <https://reactjs.org/docs/lifting-state-up.html>

Загальна схема двосторонньої передачі інформації

|  |  |
| --- | --- |
| **Передача даних вниз** (від батьківських елементів до дочірніх) | **Передача даних вверх** (від дочірніх до батьківських) |
| Батьківський елемент  Передаємо значення як властивості дочірнього  Дочірній читає значення з props |  |
| Приклад | Приклад. |

Приклад. Фіксації того, чи авторизувався клієнт.

|  |  |
| --- | --- |
|  |  |
| Контрольовані дочірні компоненти (як правило функціональні компоненти без підтримки стану, які одержують як властивості значення, так і властивості-методи від предка) | function LoginButton(props) {  return (  <button onClick={**props.onClick**}> // 🡨 Використання метода, який передали через **props**  Login  </button>  );  }  //--------------------------------------------  function LogoutButton(props) {  return (  <button onClick={**props.onClick**}> // 🡨Використання метода, який передали через **props**  Logout  </button>  );  } |
| Базовий компонент (у цьому компоненті описуються методи, які потім передаються у дочірні через props, у вигляді властивостей-методів і можуть викликатися дочірніми елементами) | class LoginControl extends React.Component {  constructor(props) {  super(props);  **// Прив’язка контекста до методів, які передаються у дочірні компоненти**  this.handleLoginClick = this.handleLoginClick.bind(this);  this.handleLogoutClick = this.handleLogoutClick.bind(this);  this.state = {isLoggedIn: false};  }  handleLoginClick() {  this.setState({isLoggedIn: true});  }  handleLogoutClick() {  this.setState({isLoggedIn: false});  }  render() {  const isLoggedIn = this.state.isLoggedIn;  let button;  if (isLoggedIn) {  button = <LogoutButton **onClick={this.handleLogoutClick}** />; //🡨Передача метода дочірньому компоненту  } else {  button = <LoginButton **onClick={this.handleLoginClick}** />; //🡨Передача метода дочірньому компоненту  }  return (  <div>  <Greeting isLoggedIn={isLoggedIn} />  {button}  </div>  );  }  } |
| Використання базового компонента | ReactDOM.render(  <LoginControl />,  document.getElementById('root')  ); |

Приклад. Конвертер валют. Синхронізувати між собою елементи вводу кількості грошових одиниць у гривнях, доларах, євро.



|  |  |
| --- | --- |
| Дочірній компонент | *//--------------------- Дочірній демонстраційний компонент (не містить state)* **const** *Currency*=(props)=>{  **return** (  <**label** >  {props.**title**}:  <**input type="text" value=**{props.gr/props.course} **onChange=**{props.**onChange**}/>  </**label**>  ) } |
| Базовий компонент | *//----------------------- Батьківський компонент (містить state)* **class** Converter **extends** Component{  constructor(props)  {  **super**(props);  **this**.onChange=**this**.onChange.bind(**this**);   **this**.**state**={ *// Задаємо початкове значення властивостей у this.state* **grivnas**:0  }  }  convert(value){  **let** val=parseFloat(value);  **if**(Number.isNaN(val))  val=0;  **return** val;  }  onChange(course,e)  {  **const** value= **this**.convert(e.**target**.**value**);  **this**.setState({  **grivnas**: value \* course *//⎜ Задаємо нові значення для властивостей у this.state* })  }  render(){  **return** (  <**React.Fragment**>  //Передаємо дочірньому компоненту метод **this**.onChange з фіксацією course=1  <**Currency title="Гривні" gr=**{**this**.**state**.**grivnas**} **course=**{1} **onChange=**{**this**.onChange.bind(**this**,1)}/>  //Передаємо дочірньому компоненту метод **this**.onChange з фіксацією course= **this**.**props**.dollar  <**Currency title="Долари" gr=**{**this**.**state**.**grivnas**} **course=**{**this**.**props**.dollar} **onChange=**{**this**.onChange.bind(**this**,**this**.**props**.dollar)}  //(альтернативний варіант)Передаємо дочірньому компоненту метод, у якому викликаємо **this**.onChange  <**Currency title="Євро" gr=**{**this**.**state**.**grivnas**} **course=**{**this**.**props**.euro} **onChange=**{(e)=>**this**.onChange(**this**.**props**.euro,e)}/>  </**React.Fragment**>  )  } } |
| Використання базового компонента | **lass** App **extends** Component {  render() {  **return** (  <**div className="App"**>    <**Converter dollar=**{27} **euro=**{31}/>  . . . . . . . . . . . .  )  } |