**Використання посилань. React.createRef()**

[**https://reactjs.org/docs/refs-and-the-dom.html**](https://reactjs.org/docs/refs-and-the-dom.html)

Може бути використано тільки для елементів DOM і елементів, які створені за допомогою класів. Посилання буде містити або елемент DOM або «змонтований» екземпляр компонента (тому не можна використовувати з функціональними компонентами ).

|  |  |
| --- | --- |
| Загальний форма | Приклад |
| Створення посилання  посилання = **React.createRef();** | class CustomTextInput extends React.Component {  constructor(props) {  super(props);  // створюємо нове писилання  this.textInput = **React.createRef(); //🡨 Створення посилання** textInput  } |
| Присвоєння посилання елементу  **ref = {**посилання**}** | render() {  // tell React that we want to associate the <input> ref  // with the `textInput` that we created in the constructor  return (  <div>  <input type="text" **ref={this.textInput}** /> //🡨 **Присвоєння посилання елементу** input  </div>  );  } |
| Звертання до елемента через посилання  **this.**посилання**.current** | focusTextInput() {  **this.textInput.current.focus**(); //🡨 **Звертання до елемента за посиланням**  }  . . . . . . . . . . .  } |

**Посилання Callback Refs**

Для встановлення посилання необхідно написати функцію. Ця функція приймає як аргумент елемент DOM і зберігає її у відповідному полі-посиланні. Функція автоматично викликається коли компонент монтується (у функцію передається відповідний DOM елемент) і викликається зі значенням null коли елемент демонтується.

|  |  |
| --- | --- |
| Загальний форма | Приклад |
| Посилання спочатку дорівнює null | **class** MyRefExample **extends** Component{  constructor(props){  **super**(props);  **this**.**myRef**=**null**;  } |
| Функція callback, яка дозволяє встановити значення для посилання  <**елемент ref=**{  DOM\_елемент **=> this.посилання =** DOM\_елемент  }/> | render(){  **return** (  <**div**>  <**input type="text" ref=**{  **DOM\_element => this.myRef = DOM\_element**  }/>  <**button onClick=**{**this**.myClick.bind(**this**)}>Say OK</**button**>  </**div**>)  } |
| Використання посилання  **this.посилання** | myClick(){  **this.myRef**.**value**=**"ok"**;  }  } |

Приклад 2.

|  |  |
| --- | --- |
|  |  |
|  | class CustomTextInput extends React.Component {  constructor(props) {  super(props);  **//Поле посилання спочатку дорівнює null**  **this.textInput = null;**  **//Функція колбек (її передаємо у дочірній для одержання посилання на його DOM елемент)**  **this.setTextInputRef = element => {**  **this.textInput = element;**  **};**  this.focusTextInput = () => {  // Focus the text input using the raw DOM API  if (this.textInput)  **this.textInput**.focus(); **//Використання посилання**  };  }  componentDidMount() {  // autofocus the input on mount  this.focusTextInput();  }  render() {  // Use the `ref` callback to store a reference to the text input DOM  // element in an instance field (for example, this.textInput).  return (  <div>  <input  type="text"  ref={**this.setTextInputRef**} //Передаємо у дочірній елемент функцію-callback (буде викликатися автоматично)  />  <input  type="button"  value="Focus the text input"  onClick={this.focusTextInput}  />  </div>  );  } |

**Передача посилання у компонент для його дочірнього елемента.**

**Використання Forwarding refs to DOM components**

<https://reactjs.org/docs/forwarding-refs.html>

|  |  |
| --- | --- |
|  |  |
| //---- Опис компонента з можливістю прийому посилання ----  const компонент = **React.forwardRef**((props, **ref**) => (  . . . використання переданого посилання **ref** . . .  ));  // ---- Створення посилання ----  const **ref = React.createRef();**  Створення компонента і передача створеного посилання  <компонент **ref={ref}**> . . . </компонент>; | const FancyButton = **React.forwardRef**((props, **ref**) => (    <button **ref={ref}** className="FancyButton">  {props.children}  </button>  ));  // Створення посилання  const **myRef = React.createRef();**  // Передача посилання у компонент  <FancyButton **ref={ myRef }**>Click me!</FancyButton>; |
| //--- Використання посилання -----  посилання .current | **myRef.current.focus()** |

|  |  |  |
| --- | --- | --- |
| З використанням рядкових посилань (небажано) | Використання  Callback refs | З використанням  React.createRef() |
|  |  |  |
|  |  |  |

Приклад. Натискаючи на кнопку передаємо фокус у наступний input (використовуючи createRef створюємо посилання для кожного input)



|  |  |
| --- | --- |
|  |  |
|  | **// ----- Компонент для генерування одного input ------**  **const** *MyInput*=({myRef,text})=>{ //🡨 компонент одержує посилання myRef  **return** (  <**input ref=**{myRef} **value=**{text}/> //🡨 приєднуємо посилання до input  ) }  **// ----- Компонент, який генерує список input ------** **const** *InputsList*=({myRefs,texts})=>{   **return**(  texts.map((text,i)=>  <**MyInput key=**{i}  **myRef=**{myRefs[i]} //🡨 У **MyInput** передаємо посилання myRefs[i]  **text=**{text}/>)  ) }  **// ----- Компонент, який генерує створює список input і організовує передачу фокуса ------** **class** Switcher **extends** Component{  constructor(props){  **super**(props);  **let** btnCount=props.**count**;  **this**.**myRefs**=[]; //🡨 Створюємо масив посилань **this**.**myRefs**  **for**(**let** i=0;i<btnCount;i++)  **this**.**myRefs**.push(React.createRef());  **this**.**texts**=[];  **for**(**let** i=0;i<btnCount;i++)  **this**.**texts**.push(**`Inp -** ${i}**`**);  **this**.**num**=0;  }  myClick()  {  **this**.**num**++;  **if**(**this**.**num**>=**this**.**texts**.**length**)  **this**.**num**=0;  **this**.**myRefs**[**this**.**num**].**current**.focus(); //🡨 Передаємо фокус до наступного input  }  render()  {  **return**(  <**div**>  <**button onClick=**{**this**.myClick.bind(**this**)}>Focus</**button**>  <**InputsList**  **myRefs=**{**this**.**myRefs**} //🡨У список передаємо масив посилань  **texts=**{**this**.**texts**}/>  </**div**>  )   } } |
|  | *//==========================* **class** App **extends** Component {  render() {  **return** (  <**div className="App"**>  <**Switcher count=**{5}/>  )} |