학습결과서

이름 : 박병찬

날짜 : 2020.07.06(월)

**오늘 학습 한 내용**

- Refs와 함수형 컴포넌트

-

**오늘 알게된 사실**

- Refs는 render 메소드에서 생성된 DOM 노드나 React 엘리먼트를 액세스하는 방법을 제공한다.

일반적인 React 데이터 흐름에서, 부모 컴포넌트가 자식 컴포넌트와 소통하는 방법은 props를 이용하는 것뿐이다. 그러나 일부 케이스에서 일반적인 데이터 흐름을 벗어나 자식 컴포넌트를 수정해야 할 경우가 있다. 수정해야 하는 자식 요소는 React의 컴포넌트거나 DOM 엘리먼트일 수 있다. 이런 경우를 위해 React는 Refs를 사용한다.

**Refs를 써야 할 때**

\* 포커스, 문장선택 혹은 미디어 재생을 관리해야 할 경우

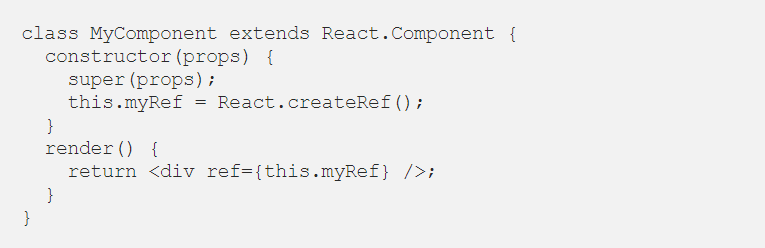
\* 필수 애니메이션을 트리거링할 때

\* 써드 파티 DOM라이브러리와 통합할 때

선언적으로 수행할 수 있는 모든 것에 refs를 사용하지 마세요.

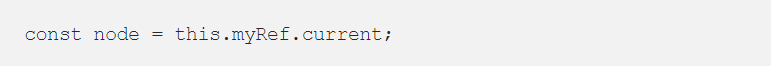
**Refs 만들기**

Refs는 React.createRef를 사용하고 ref 어트리뷰트를 활용하여 React 엘리먼트에 추가한다. Refs는 일반적으로 컴포넌트가 구성될 때 컴포넌트 전체에서 참조될 수 있도록 인스턴스 속성에 할당된다.



**Refs 액세스**

렌더링 시 엘리먼트에 refs를 전달하면, 노드에 대한 참조에 ref의 “current” 어트리뷰트로 접근할 수 있다.

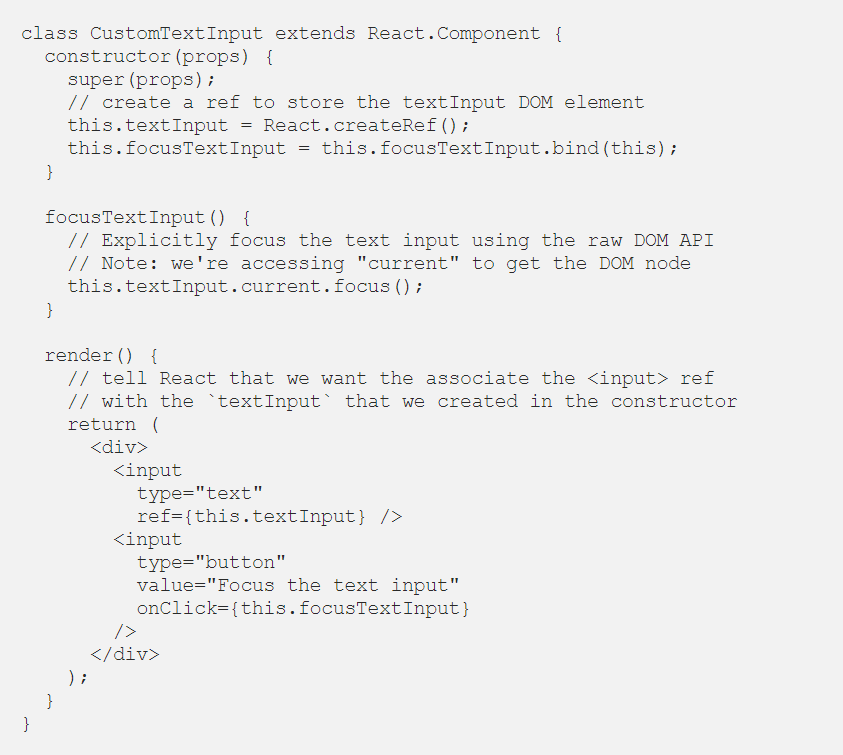


ref의 값은 노드의 타입에 따라 다르다.

* ref 어트리뷰트가 HTML 엘리먼트로 사용되면 React.createRef를 사용하여 생성자에서 생성된 ref는 기본 DOM 엘리먼트를 current property로 받는다.
* 커스텀 클래스 컴포넌트에서 ref 어트리뷰트가 사용되면, ref 오브젝트는 current로 마운트된 컴포넌트의 인스턴스를 받는다.
* 함수형 컴포넌트에서 ref 어트리뷰트를 사용하지 마세요, 왜냐하면 인스턴스를 가지고 있지 않기 때문입니다.

**Ref를 DOM 엘리먼트에 추가하기**

아래 코드는 DOM노드에 대한 참조를 저장하기 위해 ref를 사용한다.



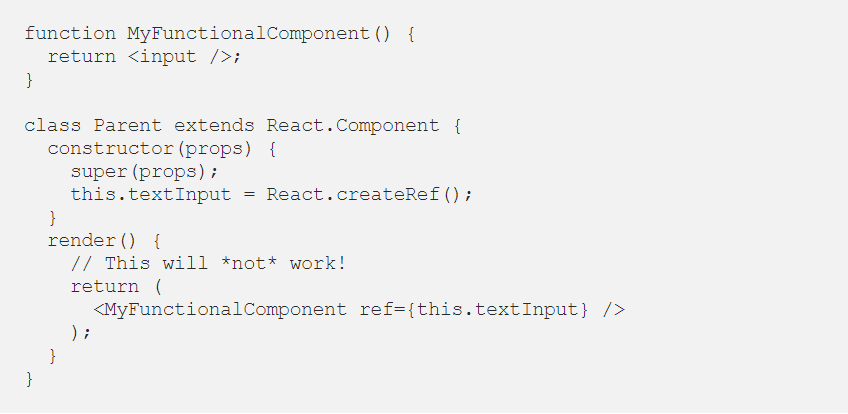
React는 컴포넌트가 마운트될 때 DOM엘리먼트와 같이 current 프로퍼티에 할당하고 언마운트될 때 null을 할당한다.

**클래스 컴포넌트에 Refs 추가하기**

마운트 직후에 클릭되는 것을 테스트하기 위해 CustomTextInput을 래핑하려면 ref를 사용하여 custom input에 액세스하고 focusTextInput 메소드를 수동으로 호출할 수 있다.

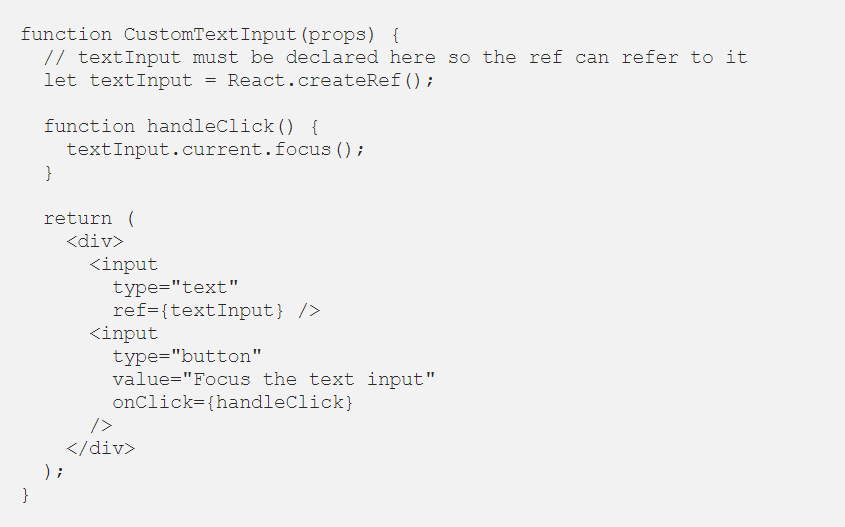
- Refs와 함수형 컴포넌트

인스턴스를 갖고 있지 않기 때문에 ref 어트리뷰트를 함수형 컴포넌트에 사용하지 마세요.



ref를 사용하기 원한다면 컴포넌트를 클래스로 변경해야 한다.

그러나 DOM 엘리먼트 또는 클래스 컴포넌트를 참조한다면 함수형 컴포넌트 안에서 ref를 사용할 수 있다.

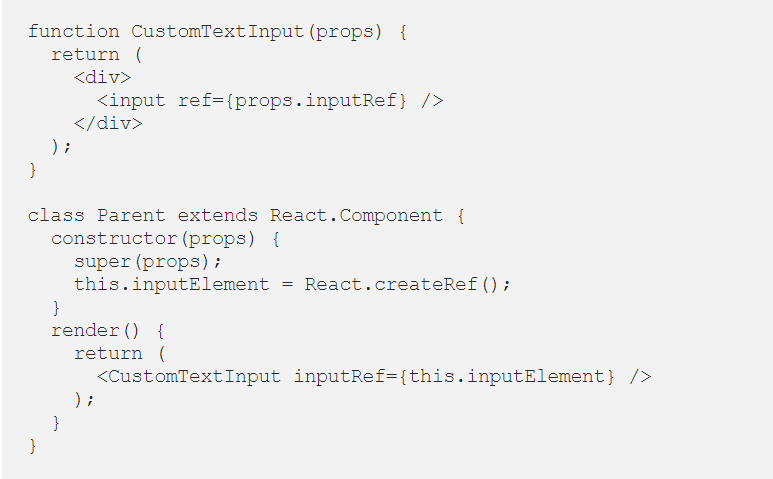


DOM Refs를 부모 컴포넌트에 노출하기

드문 케이스로 부모 컴포넌트에서 자식 노드의 DOM에 접근해야 할 경우가 있을 것이다. 이런 방식은 컴포넌트의 캡슐화를 방해하기 때문에 추천하지 않는다. 하지만 포커스를 트리거링하거나 자식 DOM 노드의 사이즈를 측정하기 위해 때때로 필요할 수 있다.

우리는 ref를 자식의 특수한 props로 노출하는 것을 추천한다. 이 prop은 ref와는 달리 어떤 것으로도 네이밍될 수 있다. 자식 컴포넌트는 prop를 ref 속성으로 DOM 노드에 전달할 수 있다. 이로 인해 부모는 ref를 중간의 컴포넌트를 통해 하위 DOM 노드로 전달할 수 있다.

이 예제는 클래스, 함수형 컴포넌트 모두 작동한다.

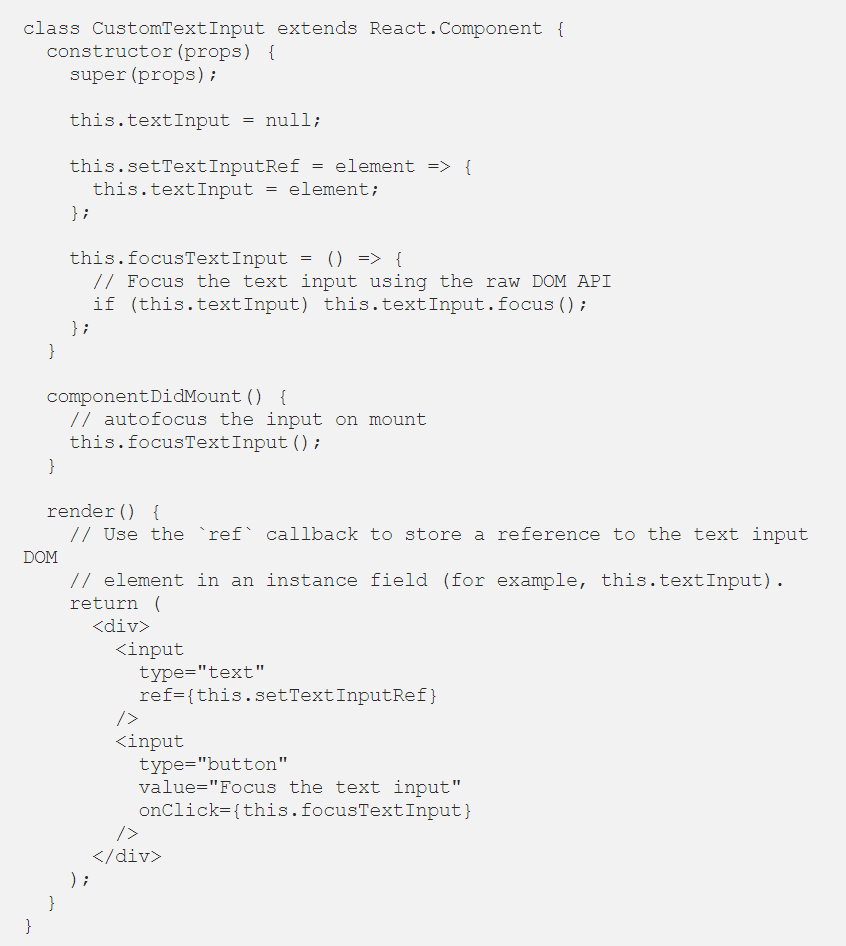


위의 예제에서 Parent는 클래스 프로퍼티인 this.inputElement를 inputRef prop으로 CustomTextInput에 전달하고, CustomTextInput은 특수 ref 어트리뷰트와 동일한 ref를 input에 전달한다. 그 결과 부모의 this.inputElement.current에 CustomTextInput의 input 엘리먼트에 대항하는 DOM노드로 설정된다.

**Callback Refs**

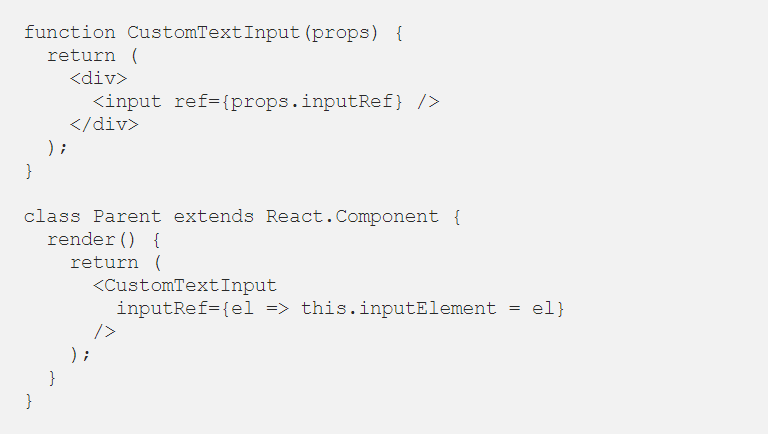
createRef( )를 통해 생성된 ref 어트리뷰트를 넘기는 것 대신, 함수를 넘긴다. 이 함수는 argument로 HTML DOM 엘리먼트 또는 React 컴포넌트 인스턴스를 받고, 다른 곳에 저장하고 접근할 수 있다.

아래 예제는 일반적인 패턴(ref callback을 사용하여 인스턴스 속성에 DOM노드에 대한 참조를 저장하는)을 구현한 것이다.



React는 컴포넌트가 마운트될 때 DOM 엘리먼트와 ref callback을 언마운트될 때는 null과 함께 호출한다. ref callback은 componentDidMount 또는 componentDidUpdate 라이프 사이클 훅 전에 호출된다.

React.createRef( )로 생성된 객체 ref를 사용하여 callback refs를 컴포넌트 간에 전달할 수 있다.



위의 예제에서 Parent는 자신의 ref callback을 inputRef prop로 CustomTextInput에 전달한다. CustomTextInput는 특수 ref 어트리뷰트인 함수를 input에 전달한다. 그 결과로 Parent 안의 this.inputElement는 CustomTextInput의 input 엘리먼트와 상응하는 DOM 노드로 셋팅된다.

출처 : <https://medium.com/@enro2414_40667/react-refs-%EA%B7%B8%EB%A6%AC%EA%B3%A0-dom-8900c72e7ab1>

-자바스크립트 배열 객체의 내장 함수인 map 함수는 기존 배열로 새로운 배열을 만드는 역할을 한다. 이를 이용하여 반복되는 컴포넌트를 렌더링할 수 있다.

example)

const names = [‘눈사람’, ‘얼음’, ‘눈’, ‘바람’];

const nameList = names.map(name => <li>{name}</li>);

<ul>

{nameList}

</ul>

* 리액트에서 key는 컴포넌트 배열을 렌더링했을 때 어떤 원소에 변동이 있었는지 알아내려고 사용한다.

예를 들어 유동적인 데이터를 다룰 때는 원소를 새로 생성할 수도, 제거할 수도, 수정할 수도 있다. key가 없을 때는 가상 DOM을 비교하는 과정에서 리스트를 순차적으로 비교하면서 변화를 감지한다. 하지만 key가 있다면 이 값을 사용하여 어떤 변화가 일어났는지 더욱 빠르게 알아낼 수 있다.

**내가 부족하다고 느낀 부분**

- 리액트 기초 지식

**수업에 바라는 점**

-

-

-