함수 컴포넌트에서 state 사용

useState

<배열 비구조화 할당>

*const* array = [1,2];

*const* one = array[0];

*const* two = array[1];

*const* [one, two] = array;

원래 위와 같이 하는게 일반적이지만

비구조화 할당에서는 아래와 같이 변수를 요소 안에 구성하여 바로 배열의 값을 연결할 수 있다.

함수에서 usetState는 여러번 선언해서 사용할 수 있다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

*const* Say = () *=>* {

*const* [message, setMessage] = useState('');

*const* onClickEnter = () *=>* {setMessage('Hello')};

*const* onClickLeave = () *=>* {setMessage('Bye')};

*const* [color, setColor] = useState('black');

    return (

        <div>

            <button onClick={onClickEnter}>Entered</button>

            <button onClick={onClickLeave}>Leaved</button>

            <h1 style={{color}}>{message}</h1>

            <br></br>

            <button style={{color: 'red'}} onClick={()*=>*{setColor('red')}}>Red</button>

            <button style={{color: 'green'}} onClick={()*=>*{setColor('green')}}>Red</button>

            <button style={{color: 'blue'}} onClick={()*=>*{setColor('blue')}}>Red</button>

        </div>

    );

};

이러한 State 에서 배열이나 객체를 업데이트하기 위해서는 사본을 만들고 다시 useState로 복사하는 과정을 거쳐야한다.

4. 이벤트 핸들링

4-1. 리액트의 이벤트 시스템

원래 html에서 onClick을 alert를 했던 과정을 react에선 어떻게 하는지를 배우는 타임

<이벤트 사용시 주의사항>

1. 이벤트 이름은 카멜 표기법으로 작성합니다
2. 이벤트에 실행할 자바스크립트 코드를 전달하는 것이 아니라, 함수 형태의 값을 전달합니다.
3. DOM요소에만 이벤트를 설정할 수 있습니다.  
   (직접 만든 컴포넌트에 onClick을 넘기는 것은 props을 넘긴 것이지 이벤트가 발생하지 않음, 태그 요소에만 적용할 수 있음)

4.1.2 이벤트 종류

clipboard

composition

keyboard

focus

form

mouse

selection

touch

ui

wheel

media

image

animation

transition

4-2. 예제로 이벤트 핸들링 익히기

input 태그에 대해서 onChange시 e(이벤트 객체)의 구성은

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

*class* EventPractice extends Component {

    render() {

        return (

            <div>

                <h1>Event Practice</h1>

                <input

                    type="text"

                    name="message"

                    placeholder="Say anything"

                    onChange={(*e*) *=>* {

                        console.log(*e*);

                    }}

                />

            </div>

        );

    }

}

e.target.value를 log하면

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명 값이 바뀔 때 마다 그 때의 value가 기록됨

onChage할때 state의 값을 update하고 button을 클릭할 때 state.message를 ‘’로 초기화 하면서 alert로 현재 message를 알려줌

*class* EventPractice extends Component {

    state = {

        message: ''

    }

    render() {

        return (

            <div>

                <h1>Event Practice</h1>

                <input

                    type="text"

                    name="message"

                    placeholder="Say anything"

                    value={this.state.message}

                    onChange={(*e*) *=>* {

                        this.setState({

                            message: *e*.target.value

                        })

                    }}

                />

                <button onClick={()*=>*{

                    alert(this.state.message);

                    this.setState({

                        message: ''

                    })

                }}>

                    confirmed

                </button>

            </div>

        );

    }

}

이때 임의 메서드를 생성해서 onChange나 onClick 에 해당하는 함수를 뽑아서 작성 가능하다.

*constructor*(*props*) {

        super(*props*);

        this.handleChange = this.handleChange.bind(this);

        this.handleClick = this.handleClick.bind(this);

    }

    handleChange(*e*) {

        this.setState({

            message: *e*.target.value,

        });

    }

    handleClick() {

        alert(this.state.message);

        this.setState({

            message: '',

        });

    }

함수 설정하고 event callback에 걸면됨

이때, constructor에서 this.함수를 this에 binding해줘야 해당 태그에서 this에 정확히 접근가능

property initializer Syntax를 사용한 메서드 작성

    handleChange = (*e*) *=>*{

        this.setState({

            message: *e*.target.value,

        });

    }

    //호출시

    this.handleChange

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

state에 userName을 추가하고

input을 하나 더 추가하면

<input

                    type='text'

                    name='userName'

                    placeholder='User Name'

                    value={this.state.userName}

                    onChange={this.handleChange}

                />

handleChange를 아래와 같이 작성하여 한번에 해결 할 수 있다.

handleChange(*e*) {

        this.setState({

            [*e*.target.name]: *e*.target.value

            //input name에 설정한 state명을 불러와서 그 값의 변화일때 그 key에 value를 저장

        });

    }

enter입력시 event 핸들하는 방법은 input에

onKeyPress={this.handleKeyPressed}

달고,

keypresshandler를 선언하면 됨

handleKeyPressed = (*e*) *=>* {

        if(*e*.key === 'Enter'){

            this.handleClick();

        }

    }

4-3.