



Seoul
Software
ACademy

웹 개발자 부트캠프 과정

SeSAC x CODINGOn

With. 팀 뽀빠

Props

props란?

- properties 를 줄인 표현으로 컴포넌트 속성을 설정할 때 사용하는 요소
- props는 컴포넌트끼리 값을 전달하는 수단
- 상위 컴포넌트에서 하위 컴포넌트로 전달 (단방향 데이터 흐름)

```

{/* 일반 사용법 */}
<ClassComponent></ClassComponent>

{/* props 사용법 */}
<ClassComponent title="제목" content="내용"></ClassComponent>
  
```

함수형 컴포넌트 props

- 부모 컴포넌트에서 전달한 props는 함수형 컴포넌트에서 함수의 파라미터로 전달받으며, JSX 내부에서 { } 기호로 감싸서 사용한다.

```
<FuncComponent name="코딩온"></FuncComponent>
```

* 부모 컴포넌트에서 name props 전달

```
const FuncComponent = (props) => {  
  return (  
    <>  
      <div>안녕? {props.name}</div>  
      <div>반가워!</div>  
    </>  
  );  
}
```

* 자식 컴포넌트에서 name props 받음

defaultProps

- 부모 컴포넌트에서 props가 전달되지 않았을 때 기본값으로 보여줄 props를 설정하는 것

```
<FuncComponent ></FuncComponent>
```

* 부모 컴포넌트

```
const FuncComponent = (props) => {
  return (
    <>
    <div>안녕 ? {props.name}</div>
    <div>반가워 !</div>
    </>
  );
}
```

```
FuncComponent.defaultProps = {
  name: '홍길동'
}
```

* 자식 컴포넌트

props.children

- 부모 컴포넌트에서 자식 컴포넌트를 호출할 때 태그 사이에 작성한 문자열

```
<FuncComponent name="코딩 온">자식 내용</FuncComponent>
```

* 부모 컴포넌트

```
const FuncComponent = (props) => {
  return (
    <>
    <div>안녕? {props.name}</div>
    <div>반가워!</div>
    <h4>{props.children}</h4>
    </>
  );
}
```

* 자식 컴포넌트

propTypes

- 컴포넌트의 필수 props를 지정하거나 props의 타입을 지정할 때 사용
- JavaScript의 “유연한 특성” 을 해결하기 위해 권장되는 기능
- 정해진 타입이 아닌 다른 타입으로 정보가 전달될 시, 제대로 동작은 하지만 console에 오류가 나온다.

```
import PropTypes from "prop-types";
```

```
FuncComponent.propTypes = {  
  |   name: PropTypes.string  
}
```

클래스형 컴포넌트 props

```
class ClassComponent extends Component {  
  render() {  
    return(  
      <h1>Class Component 입니다. 이름은 { this.props.name }</h1>  
    );  
  }  
}
```


클래스형 컴포넌트 props

```
class ClassComponent extends Component {
  render() {
    return(
      <h1>Class Component 입니다. 이름은 { this.props.name }</h1>
    );
  }

  static defaultProps = {
    name: "기본 이름"
  };

  static propTypes = {
    name: PropTypes.string
  };
}
```

or

```
ClassComponent.defaultProps = {
  name: '홍길동'
}

ClassComponent.propTypes = {
  name: PropTypes.string
}
```

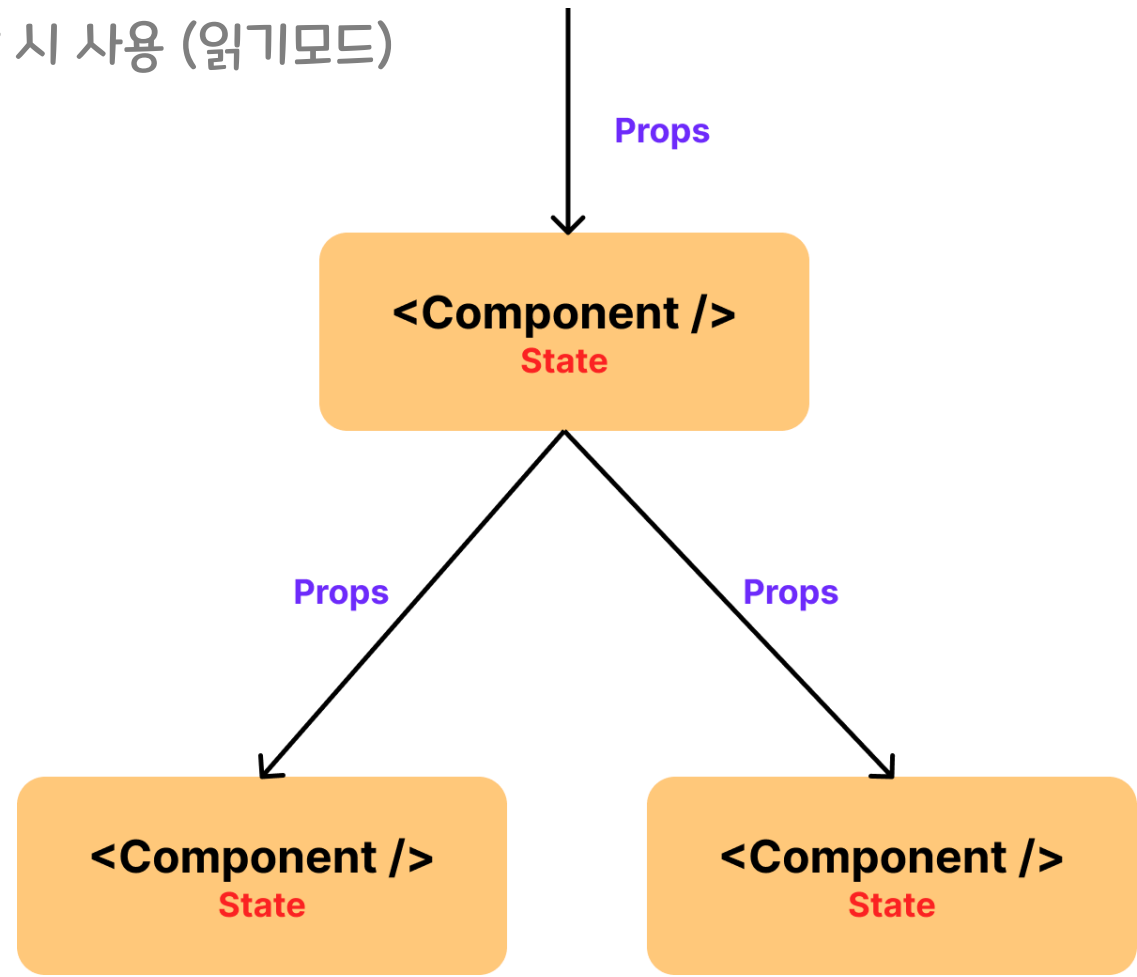
State

state란?

- React에서 앱의 유동적인 데이터를 다루기 위한 개체
- 계속해서 변하는 특정 상태
- 상태에 따라 다른 동작을 함
- 왜 사용할까?
 - State가 변경될시 자동으로 재랜더링 되기 때문
 - 이 점이 변수와 다른 점

state vs. props

- **props**: 부모 컴포넌트에서 자식 컴포넌트에 데이터 전달 시 사용 (읽기모드)
 - React는 **단방향 데이터 흐름**! 기억하시죠? 😊
- **state**: 특정 컴포넌트가 갖는 상태 (값)
 - 컴포넌트 **내부에서** 선언되고 **내부에서** 값을 변경함!



클래스형 컴포넌트의 state

- 기존 형태

```
import React, { Component } from "react";

class ClassState extends Component {
  constructor(props) {
    super(props);

    this.state = {
      number: 0,
    };
  }
  render() {
    const { number } = this.state;
    return (
      <div> <h3>Number : {number}</h3> </div>
    );
  }
}
export default ClassState;
```

클래스형 컴포넌트의 state

- 현재 버전

```
import React, { Component } from "react";

class ClassState extends Component {
  state = {
    number: 0,
  };
  render() {
    const { number } = this.state;
    return (
      <div> <h3>Number : {number}</h3> </div>
    );
  }
}

export default ClassState;
```

함수형 컴포넌트의 state

- 함수형 컴포넌트는 state 기능이 원래 없었다.
- React 16.8 버전 이후부터 `useState`라는 함수가 생겼고, 이를 통해 함수형 컴포넌트에서도 상태를 관리할 수 있음!

함수형 컴포넌트의 useState()

```
import React, { useState } from 'react';

const SayFunction = () => {
  const [ message, setMessage ] = useState("");
  const onClickEnter = () => { setMessage("안녕하세요~"); };
  const onClickLeave = () => { setMessage("안녕히가세요."); };

  return (
    <div>
      <button onClick={onClickEnter}>입장</button>
      <button onClick={onClickLeave}>퇴장</button>
      <h1>{message}</h1>
    </div>
  );
};

export default SayFunction;
```


함수형 컴포넌트의 useState()

```
import React, { useState } from 'react';

const SayFunction = () => {
  const [ message, setMessage ] = useState("");
  const onClickEnter = () => { setMessage("안녕하세요~"); };
  const onClickLeave = () => { setMessage("안녕히가세요."); };

  return (
    <div>
      <button onClick={onClickEnter}>입장</button>
      <button onClick={onClickLeave}>퇴장</button>
      <h1>{message}</h1>
    </div>
  );
};

export default SayFunction;
```

- useState 함수의 인자에는 **상태의 초기값**
 - useState의 초기값은 숫자일 수도, 문자일 수도, 배열일 수도 있다. 즉, **값의 형태가 자유로움**
- useState 함수는 배열을 반환
 - 첫번째 원소: **현재 상태**
 - 두번째 원소: **상태를 바꿔주는 setter 함수**

```
const [ message, setMessage ] = useState("");
```

[현재상태 , setter함수]

: useState는 배열 반환

함수형 컴포넌트의 useState()

```
import React, { useState } from 'react';
function Counter() {
  const [ number, setNumber ] = useState(0);
  const onClickEnter = () => setNumber(number+1);

  return (
    <div>
      <h1>{number}</h1>
      <button onClick={onClickEnter}>+1</button>
    </div>
  );
}

export default Counter;
```