React

사용자 인터페이스를 만들기 위한 JavaScript 라이브러리

시작하기 자습서 읽어보기 >

선언형

React는 상호작용이 많은 UI를 만들 때 생기는 어려움을 줄여줍니다. 애플리케이션의 각 상태에 대한 간단한 뷰만 설계하세요. 그럼 React는 데이터가 변경됨에 따라 적절한 컴포넌트만 효율적으로 갱신하고 렌더링합니다.

선언형 뷰는 코드를 예측 가능하고 디버그하기 쉽게 만들어 줍니다.

컴포넌트 기반

스스로 상태를 관리하는
잡한 UI를 만들어보세요

컴포넌트 로직은 템플릿
데이터를 앱 안에서 손
있습니다.

간단한 컴포넌트

React 컴포넌트는 `render()` 라는 메서드를 구현하는데, 이것은 데이터를 입력받아 화면에 표시할 내용을 반환하는 역할을 합니다. 이 예제에서는 XML과 유사한 문법인 JSX를 사용합니다. 컴포넌트로 전달된 데이터는 `render()` 안에서 `this.props` 를 통해 접근할 수 있습니다.

React를 사용하기 위해서 JSX가 꼭 필요한 것은 아닙니다. JSX를 컴파일한 JavaScript 코드를 확인하려면 Babel REPL을 이용해보세요.

LIVE JSX EDITOR ☑ **JSX?**

```
class HelloMessage extends React.Component {
  render() {
    return (
      <div>
        Hello {this.props.name}
      </div>
    );
```

```
  }

}

ReactDOM.render(
  <HelloMessage name="Taylor" />,
  document.getElementById('hello-example')
);
```

RESULT

Hello Taylor

상태를 가지는 컴포넌트

컴포넌트는 `this.props` 를 이용해 입력 데이터를 다루는 것 외에도 내부적인 상태 데이터를 가질 수 있습니다. 이는 `this.state` 로 접근할 수 있습니다. 컴포넌트의 상태 데이터가 바뀌면 `render()` 가 다시 호출되어 마크업이 갱신됩니다.

LIVE JSX EDITOR ☑ **JSX?**

```
class Timer extends React.Component {
  constructor(props) {
    super(props);
    this.state = { seconds: 0 };
  }

  tick() {
    this.setState(state => ({
      seconds: state.seconds + 1
    }));
  }

  componentDidMount() {
    this.interval = setInterval(() => this.tick(), 1000);
  }
```

RESULT

Seconds: 0

애플리케이션

`props` 와 `state` 를 사용해서 간단한 Todo 애플리케이션을 만들 수 있습니다. 이 예제에서는

state를 사용해 사용자가 입력한 텍스트와 할 일 목록을 관리합니다. 이벤트 핸들러들이 인라인으로 각각 존재하는 것처럼 보이지만, 실제로는 이벤트 위임을 통해 하나로 구현됩니다.

LIVE JSX EDITOR ☑ **JSX?**

```
class TodoApp extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [], text: '' };
    this.handleChange = this.handleChange.bind(this);
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  render() {
    return (
      <div>
        <h3>TODO</h3>
        <TodoList items={this.state.items} />
        <form onSubmit={this.handleSubmit}>
          <label htmlFor="new-todo">
            What needs to be done?
```

RESULT

TODO

What needs to be done?

Add #1

외부 플러그인을 사용하는 컴포넌트

React는 유연하며 다른 라이브러리나 프레임워크를 함께 활용할 수 있습니다. 이 예제에서는 외부 마크다운 라이브러리인 **remarkable**을 사용해 `<textarea>`의 값을 실시간으로 변환합니다.

LIVE JSX EDITOR ☑ **JSX?**

```
class MarkdownEditor extends React.Component {
  constructor(props) {
    super(props);
    this.md = new Remarkable();
    this.handleChange = this.handleChange.bind(this);
    this.state = { value: 'Hello, **world**!' };
```

```
  }


  handleChange(e) {
    this.setState({ value: e.target.value });
  }

  getRawMarkup() {
    return { __html: this.md.render(this.state.value) };
  }
```

RESULT

Input

Enter some markdown

Hello, **world**!

Output

Hello, **world**!

시작하기 자습서 읽어보기 >