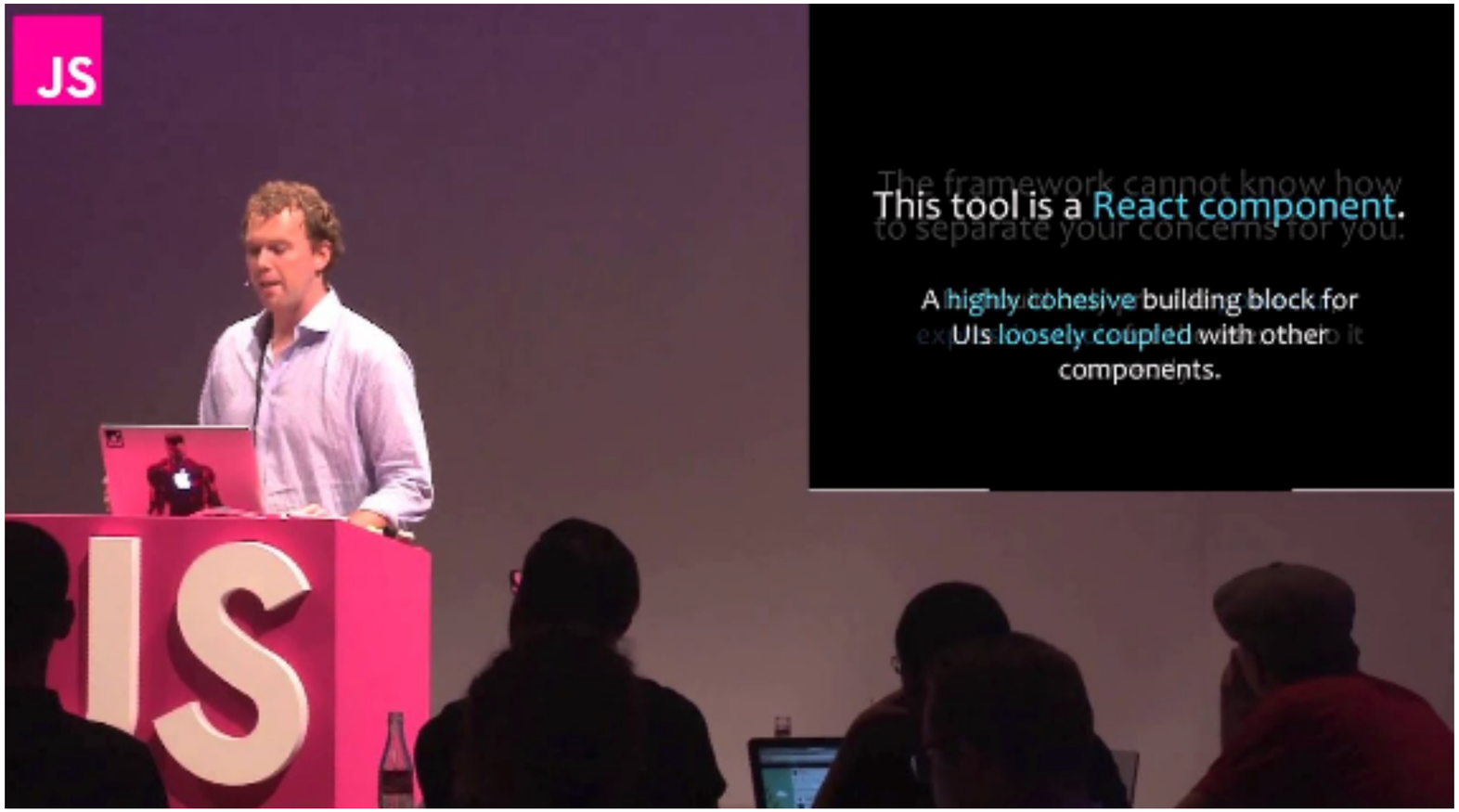
JSX를 사용하는 이유

왜 로직과 뷰는 분리되지 않아야 하는가

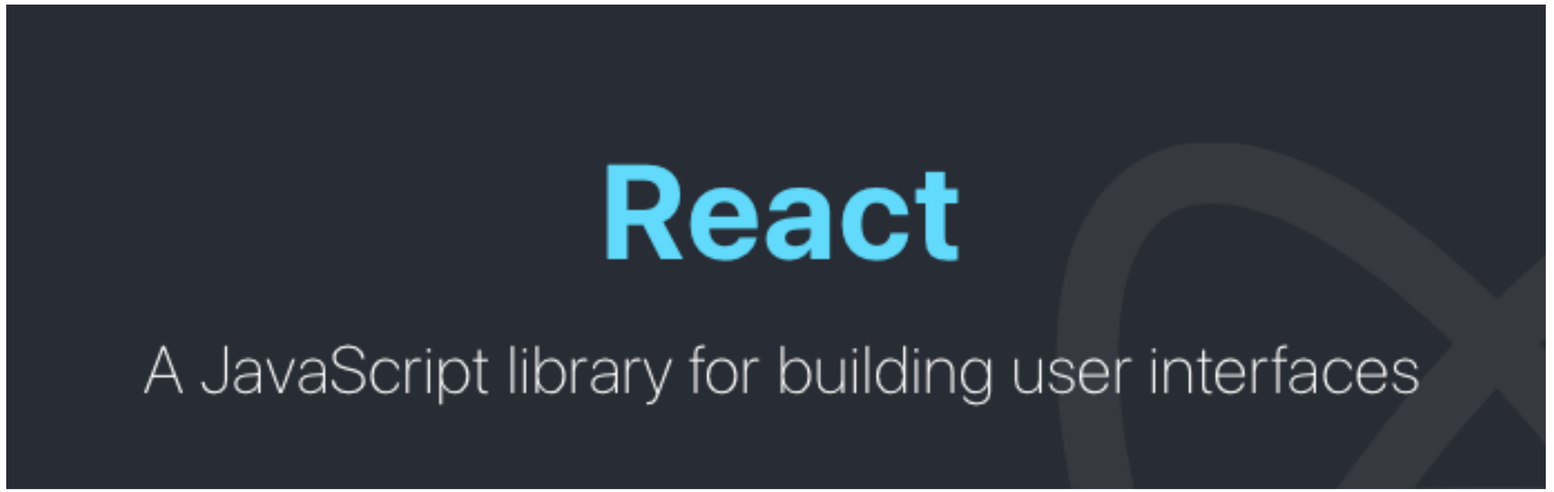
2021.06.16 최진우

Pete Hunt: React: Rethinking best practices - JSConf EU

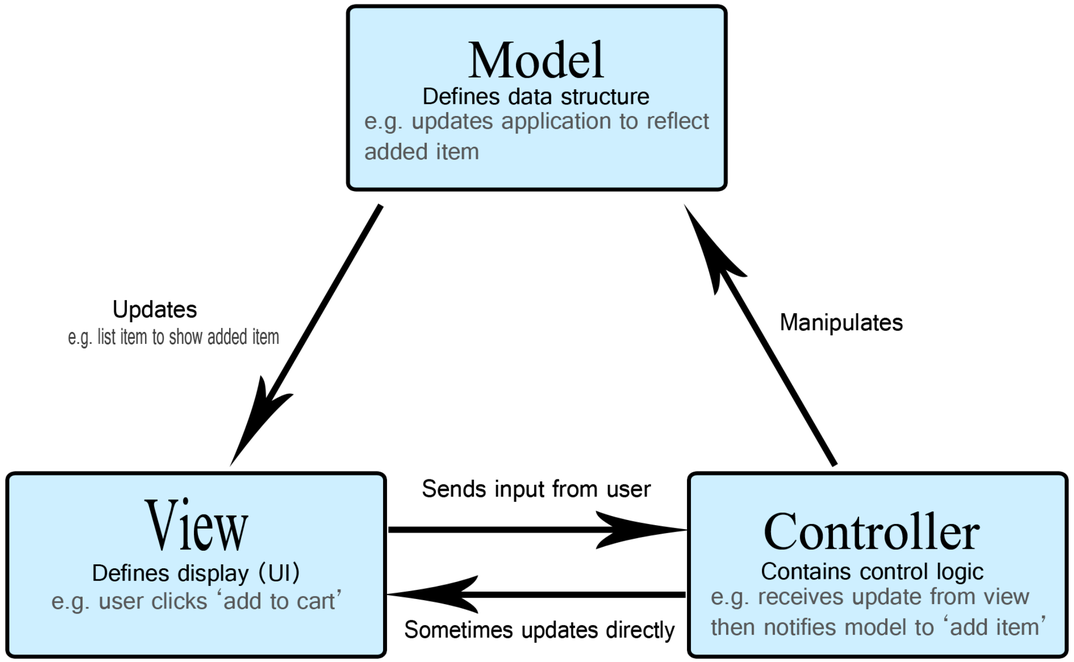


리액트

UI 라이브러리



MVC
리액트의 역할


Model
Defines data structure
e.g. updates application to reflect added item
Updates
e.g. list item to show added item
Manipulates
View
Defines display (UI)
e.g. user clicks 'add to cart'
Sends input from user
Sometimes updates directly
Controller
Contains control logic
e.g. receives update from view then notifies model to 'add item'

JSX란?

React에서는 이벤트가 처리되는 방식, 시간에 따라 state가 변하는 방식, 화면에 표시하기 위해 데이터가 준비되는 방식 등 렌더링 로직이 본질적으로 다른 UI 로직과 연결된다는 사실을 받아들입니다.

React는 별도의 파일에 마크업과 로직을 넣어 *기술*을 인위적으로 분리하는 대신, 둘 다 포함하는 "컴포넌트"라고 부르는 느슨하게 연결된 유닛으로 *관심사*를 분리합니다. 이후 섹션에서 다시 컴포넌트로 돌아오겠지만, JS에 마크업을 넣는 게 익숙해지지 않는다면 이 이야기가 확신을 줄 것입니다.

템플릿이 아닌,

컴포넌트를 만들자

- 관심사의 분리
  - 낮은 결합도
  - 높은 응집도
- 기존의 MVVM 구조
  - 색상 관리
  - 높은 결합도, 높은 응집도

Keep your components small

작은 컴포넌트를 만들자

Only put display logic in your components

Display 로직만을 넣자

XML-like

(Babel)

Native Javascript

```jsx
const element = (
  <h1 className="greeting">
    Hello, world!
  </h1>
);
```

```jsx
const element = React.createElement(
  'h1',
  {className: 'greeting'},
  'Hello, world!'
);
```

React.createElement()

특정한 구조의 객체 생성

```
// 주의: 다음 구조는 단순화되었습니다
const element = {
  type: 'h1',
  props: {
    className: 'greeting',
    children: 'Hello, world!'
  }
};
```