4. 리액트의 작동 원리

Prof. **Seunghyun Park** (sp@hansung.ac.kr)

Division of Computer Engineering

학습 목표: 4장. 리액트의 작동 원리

- React element 생성
  - `React.createElement(` *`type`*`,` *`props`*`,` *`children`* `)`
- ReactDOM 렌더링
  - `ReactDOM.render(` *`element`*`,` *`container`* `)`
- React component

페이지 설정

- React: 뷰를 만들기 위한 라이브러리

https://unpkg.com/react@16.14.0/umd/react.development.js

https://unpkg.com/react-dom@16.14.0/umd/react-dom.development.js

- ReactDOM: UI를 브라우저에 렌더링 할 때 사용하는 라이브러리

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>순수 리액트 예제</title>
</head>
<body>

<div id="react-container"></div>

<script src="https://unpkg.com/react@16/umd/react.development.js"></script>
<script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
<script>

/* ch04-01-01-page-setup.html */
// 순수 리액트와 자바스크립트 코드

</script>

</body>
</html>
```

타겟 컨테이너

React와 ReactDOM 라이브러리

React element 생성과 ReactDOM 렌더링

https://ko.reactjs.org/docs/react-api.html#createelement
https://ko.reactjs.org/docs/react-dom.html#render

```
<!-- Target Container -->
<div id="react-container"></div>

/* ch04-02-01-elements.html */
const dish = React.createElement(
  "h1", { id: "recipe-0" }, "구운 연어"
)


ReactDOM.render(
    dish,
    document.getElementById('react-container')
)

console.log('dish', dish)
```

element 생성
- type: h1
- property: id="recipe-0"
- 자식노드: 텍스트 ("구운 연어")

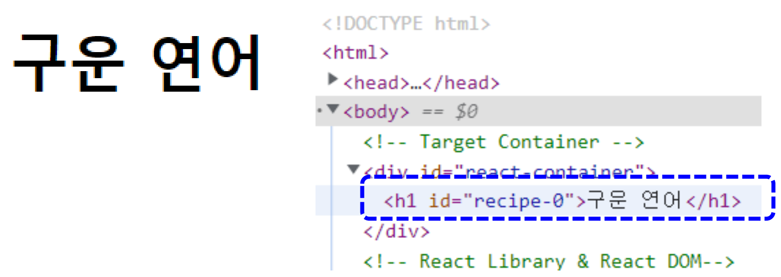
ReactDOM 렌더링
- element (dish: h1)
- 대상: 'react-container'

```
React.createElement(
  type, [props], [...children]
)
```

- 인자로 주어지는 타입에 따라 새로운 **리액트 엘리먼트를 생성하여 반환**

```
ReactDOM.render(
  element, container[, ...callback]
)
```

- 인자로 주어지는 렌더링 할 **리액트 엘리먼트**를
  제공된 컨테이너의 **DOM** (렌더링이 일어날 대상 DOM)**에 렌더링,**
- 구성요소에 대한 참조를 반환

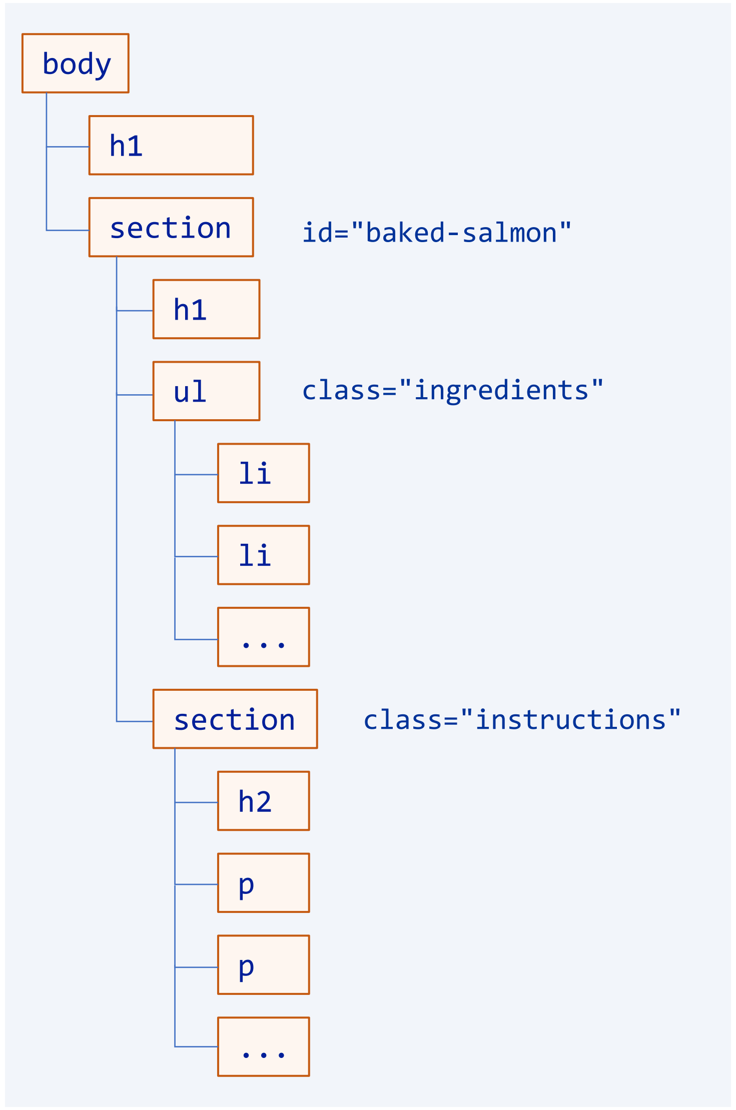


구운 연어

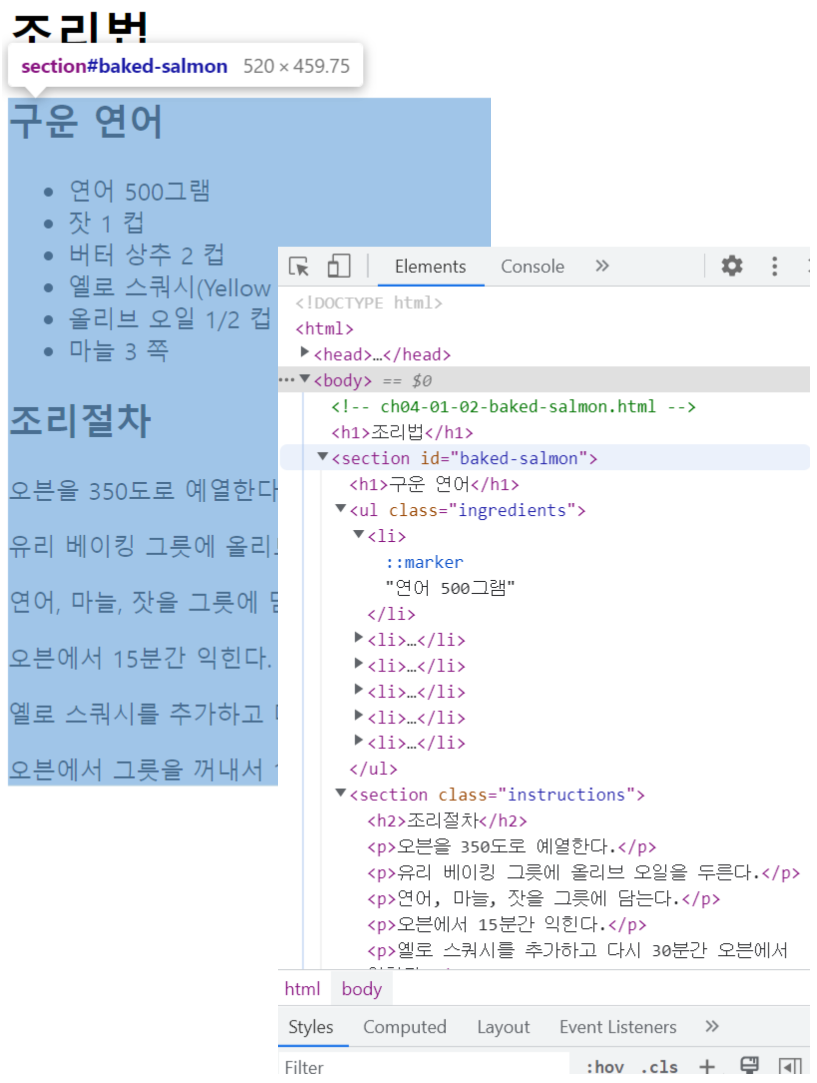
```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
▼<body> == $0
   <!-- Target Container -->
  ▼<div id="react-container">
     <h1 id="recipe-0">구운 연어</h1>
   </div>
   <!-- React Library & React DOM-->
```

```
dish Object
  $$typeof: Symbol(react.element)          _owner: null
  key: null                                  _store: {validated: false}
  props: {id:'recipe-0', children: '구운 연어'}  _self: null
  ref: null                                  _source: null
  type: "h1"                                 [[Prototype]]: Object
```

예제 1-2. baked-salmon (html)



```html
<!-- ch04-01-02-baked-salmon.html -->
<h1>조리법</h1>

<section id="baked-salmon">
  <h1>구운 연어</h1>
  <ul class="ingredients">
    <li>연어 500그램</li>
    <li>잣 1 컵</li>
    <li>...</li>
    <li>...</li>
    <li>...</li>
    <li>...</li>
  </ul>
  <section class="instructions">
    <h2>조리절차</h2>
    <p>오븐을 350도로 예열한다.</p>
    <p>...</p>
    <p>...</p>
    <p>...</p>
    <p>...</p>
    <p>...</p>
  </section>
</section>
```



- 연어 500그램
- 잣 1 컵
- 버터 상추 2 컵
- 옐로 스쿼시(Yellow
- 올리브 오일 1/2 컵
- 마늘 3 쪽

오븐을 350도로 예열한다

유리 베이킹 그릇에 올리

연어, 마늘, 잣을 그릇에

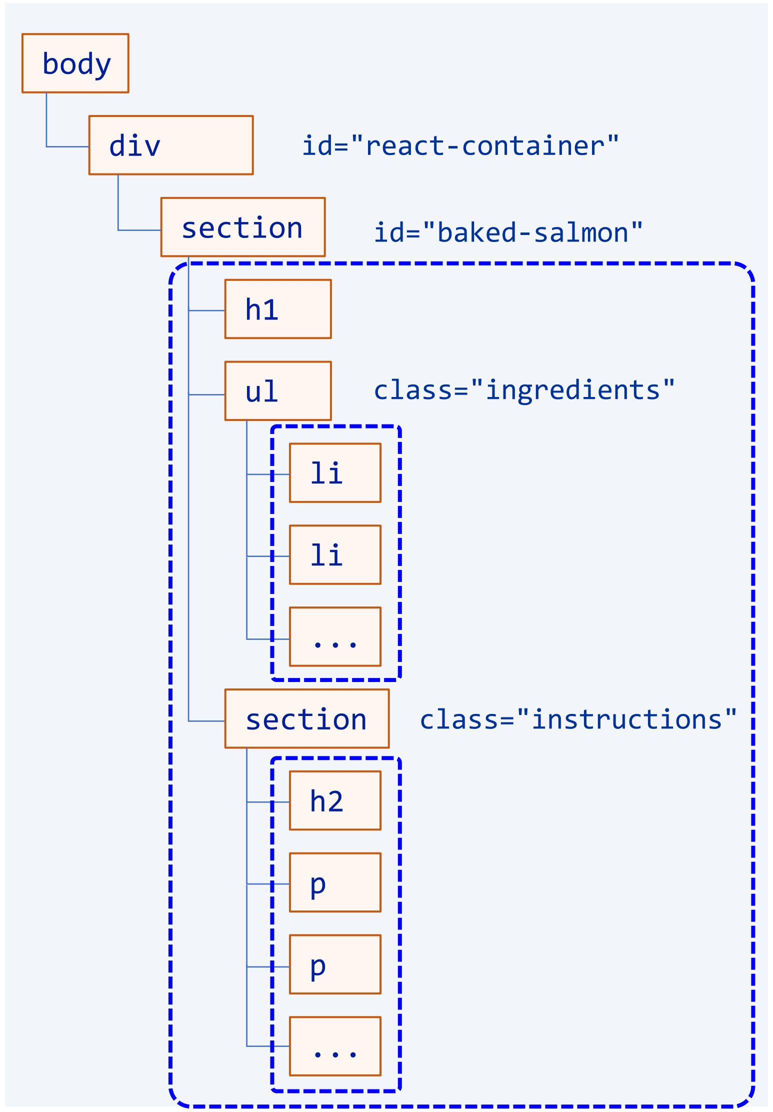
오븐에서 15분간 익힌다.

옐로 스쿼시를 추가하고

오븐에서 그릇을 꺼내서

```
<!DOCTYPE html>
<html>
 ▶<head>…</head>
···▼<body> == $0
    <!-- ch04-01-02-baked-salmon.html -->
    <h1>조리법</h1>
   ▼<section id="baked-salmon">
      <h1>구운 연어</h1>
     ▼<ul class="ingredients">
       ▼<li>
          ::marker
          "연어 500그램"
        </li>
       ▶<li>…</li>
       ▶<li>…</li>
       ▶<li>…</li>
       ▶<li>…</li>
       ▶<li>…</li>
      </ul>
     ▼<section class="instructions">
        <h2>조리절차</h2>
        <p>오븐을 350도로 예열한다.</p>
        <p>유리 베이킹 그릇에 올리브 오일을 두른다.</p>
        <p>연어, 마늘, 잣을 그릇에 담는다.</p>
        <p>오븐에서 15분간 익힌다.</p>
        <p>옐로 스쿼시를 추가하고 다시 30분간 오븐에서
```

예제 1-2. baked-salmon (react)



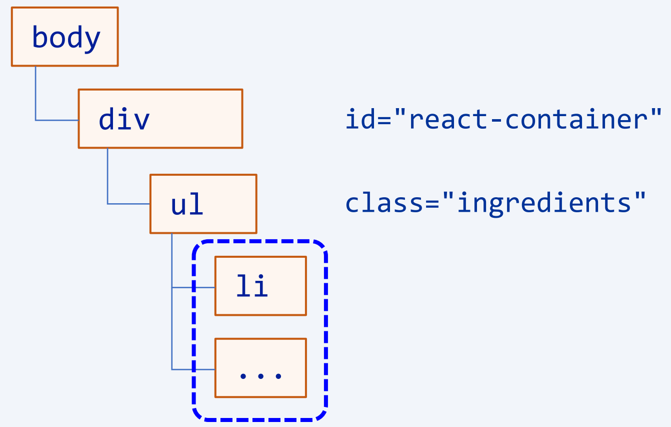
```
<!-- Target Container -->
<div id="react-container"></div>

/* ch04-02-03-elements.html */
const dish = React.createElement(
  "section", {id: "baked-salmon"},
    React.createElement("h1", null, "구운 연어"),
    React.createElement(
      "ul",  {"className": "ingredients"},
        React.createElement("li", null, "연어 500그램"),
        React.createElement("li", null, "잣 1 컵"),
        React.createElement("li", null, "..."),
        React.createElement("li", null, "...")
      ),
    React.createElement(
      "section", {"className": "instructions"},
        React.createElement("h2", null, "조리절차"),
        React.createElement("p", null, "오븐을..."),
        React.createElement("p", null, "유리..."),
        React.createElement("p", null, "..."),
        React.createElement("p", null, "...")
    )
)

ReactDOM.render(dish, document.getElementById('react-container'))
console.log('dish element', dish)
```

section은 자식 엘리먼트가 3개: h1, ul, section

ul은 자식 엘리먼트가 6개: li, …

section은 자식 엘리먼트가 7개: h2, p, …

예제 1-2. baked-salmon (react: *props.children*, 계속)



```
/* ch04-02-04-elements.html */
var items = [
  "연어 500그램",
  "잣 1 컵",
  "버터 상추 2 컵",
  "옐로 스쿼시(Yellow Squash, 호박의 한 종류) 1개",
  "올리브 오일 1/2 컵",
  "마늘 3 쪽"
]
```

```
/* ch04-02-04-1-elements.html */
React.createElement(
  "ul", { className: "ingredients" },
    React.createElement("li", null, "...")
    React.createElement("li", null, "...")
    ...
);
```

props.children 을 array로 생성

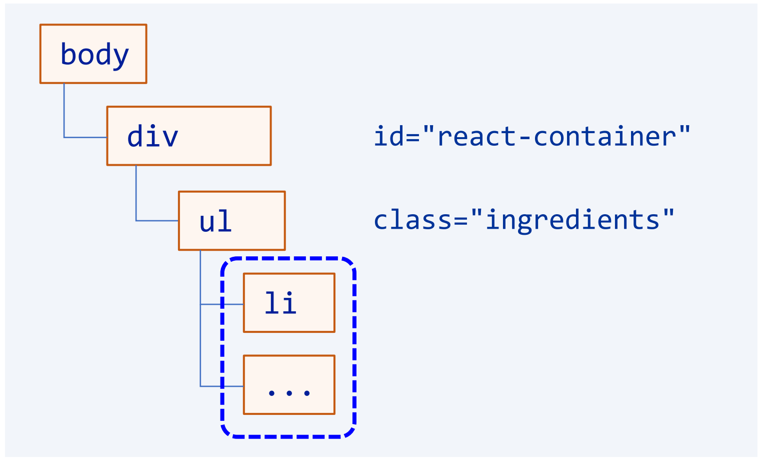
```
const ingredients = React.createElement(
  "ul", { className: "ingredients" },
    items.map( (ingredient) =>
      React.createElement("li", null, ingredient)
    )
)
```

반복 작업 단순화를 위해 array.map() 활용

items 배열의 모든 요소를 활용해 React.createElement() 호출

```
ReactDOM.render(ingredients, document.getElementById('react-container'))
console.log('ingredients', ingredients)
```

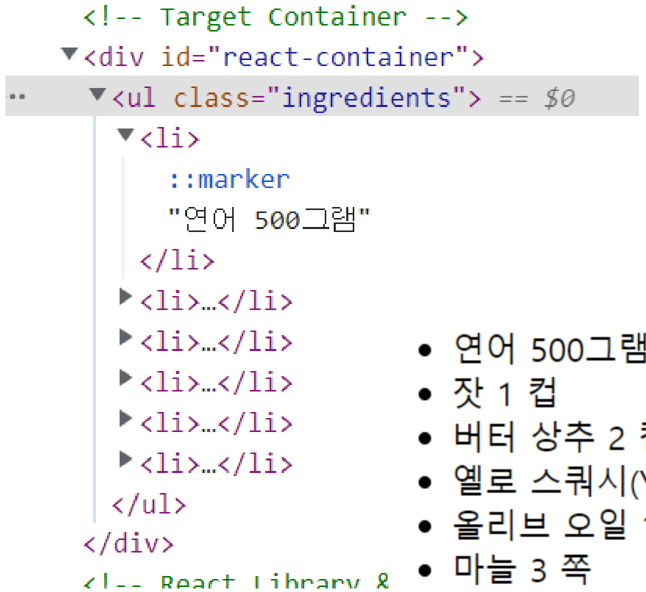
예제 1-2. baked-salmon (react: *props.children*)



```
/* ch04-02-04-elements.html */
var items = [ ... ]
const ingredients = React.createElement( "ul", { className: "ingredients" },
  items.map( (ingredient, i) => React.createElement("li", null, ingredient) ) )

ReactDOM.render(ingredients, document.getElementById('react-container'))
console.log('ingredients', ingredients)
```

props.children

```
...
props:{className: 'ingredients', children: Array(6)}
 children:(6) [{…}, {…}, {…}, {…}, {…}, {…}]
  0:{$$typeof: Symbol(react.element), type: 'li', key: null, ref: null, props: {…}, …}
  1:{$$typeof: Symbol(react.element), type: 'li', key: null, ref: null, props: {…}, …}
  ...
 length:6
[[Prototype]]:Array(0)
```



```
  <!-- Target Container -->
 ▼<div id="react-container">
   ▼<ul class="ingredients"> == $0
     ▼<li>
        ::marker
        "연어 500그램"
      </li>
     ▸<li>…</li>
     ▸<li>…</li>
     ▸<li>…</li>
     ▸<li>…</li>
     ▸<li>…</li>
    </ul>
  </div>
  <!-- React Library &
```

- 연어 500그램
- 잣 1 컵
- 버터 상추 2 컵
- 옐로 스쿼시(Yellow Squash, 호박의 한 종류) 1개
- 올리브 오일 1/2 컵
- 마늘 3 쪽

```
Warning: Each child in a list should have a unique "key" prop.
Check the top-level render call using <ul>. See
https://fb.me/react-warning-keys for more information.
  in li
```

예제 1-2. baked-salmon (react: *props.children*)

```
/* ch04-02-05-elements.html */
var items = [ ... ]
const ingredients = React.createElement( "ul", { className: "ingredients" },
   items.map( (ingredient, i) => React.createElement("li", { key: i }, ingredient) )
)
ReactDOM.render(ingredients, document.getElementById('react-container'))
console.log('ingredients', ingredients)
```

```
...
props:{className: 'ingredients', children: Array(6)}
 children:(6) [{…}, {…}, {…}, {…}, {…}, {…}]
  0:{$$typeof: Symbol(react.element), type: 'li', key: '0', ref: null, props: {…}, …}
  1:{$$typeof: Symbol(react.element), type: 'li', key: '1', ref: null, props: {…}, …}
  ...
 length:6
[[Prototype]]:Array(0)
```

```
/* ch04-02-04-elements.html */
const ingredients = React.createElement( "ul", { className: "ingredients" },
   items.map( (ingredient) => React.createElement("li", null, ingredient) ) )
```