모던 자바스크립트로 배우는 리액트 입문

내용

- 모던 자바스크립트 기초
  - DOM, 가상 DOM
  - 패키지 관리자
  - 모듈핸들러, 트랜스파일러
  - SPA와 기존 웹시스템의 차이
- 모던 자바스크립트 기능(문법적 요소)
  - 변수 선언 키워드: const, let
  - 템플릿 문자열
  - 화살표 함수 ()=>{}
  - 분할 대입 {} []
  - 디폴트값 =
  - 스프레드 구문
  - 객체 생략 표기법
  - map, filter
  - 삼항연산자/논리연산자 && ||
- 자바스크립트 DOM 조작

자바스크립트로 DOM 조작

웹페이지 동적(Dynamic)으로 만들기-DOM API

- ## Finding HTML Elements

Method	Description
document.getElementById(*id*)	Find an element by element id
document.getElementsByTagName(*name*)	Find elements by tag name
document.getElementsByClassName(*name*)	Find elements by class name

Changing HTML Elements

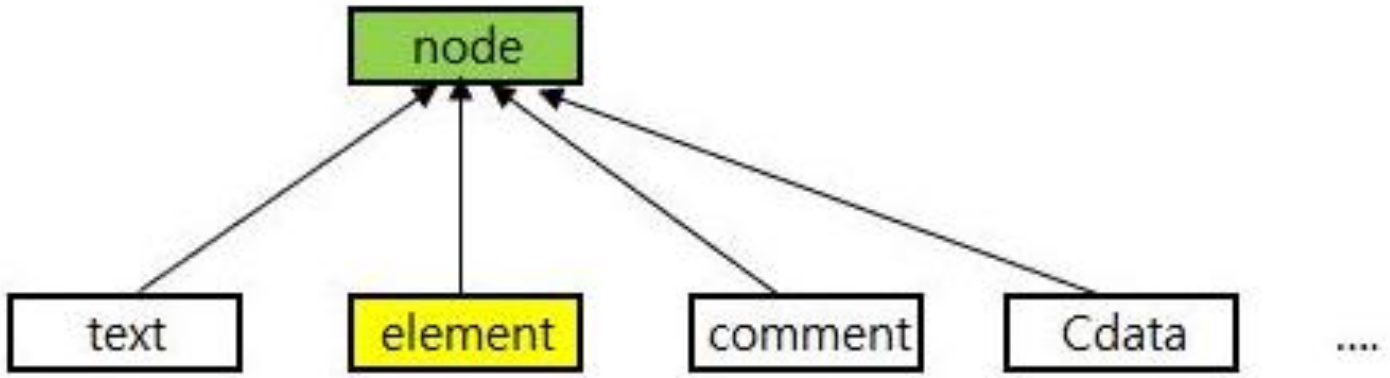
Property	Description
element.innerHTML = *new html content*	Change the inner HTML of an element
element.attribute = new value	Change the attribute value of an HTML element
element.style.*property = new style*	Change the style of an HTML element
Method	**Description**
element.setAttribute*(attribute, value)*	Change the attribute value of an HTML element

Adding and Deleting Elements

Method	Description
document.createElement(*element*)	Create an HTML element
document.removeChild(*element*)	Remove an HTML element
document.appendChild(*element*)	Add an HTML element
document.replaceChild(*new, old*)	Replace an HTML element
document.write(*text*)	Write into the HTML output stream

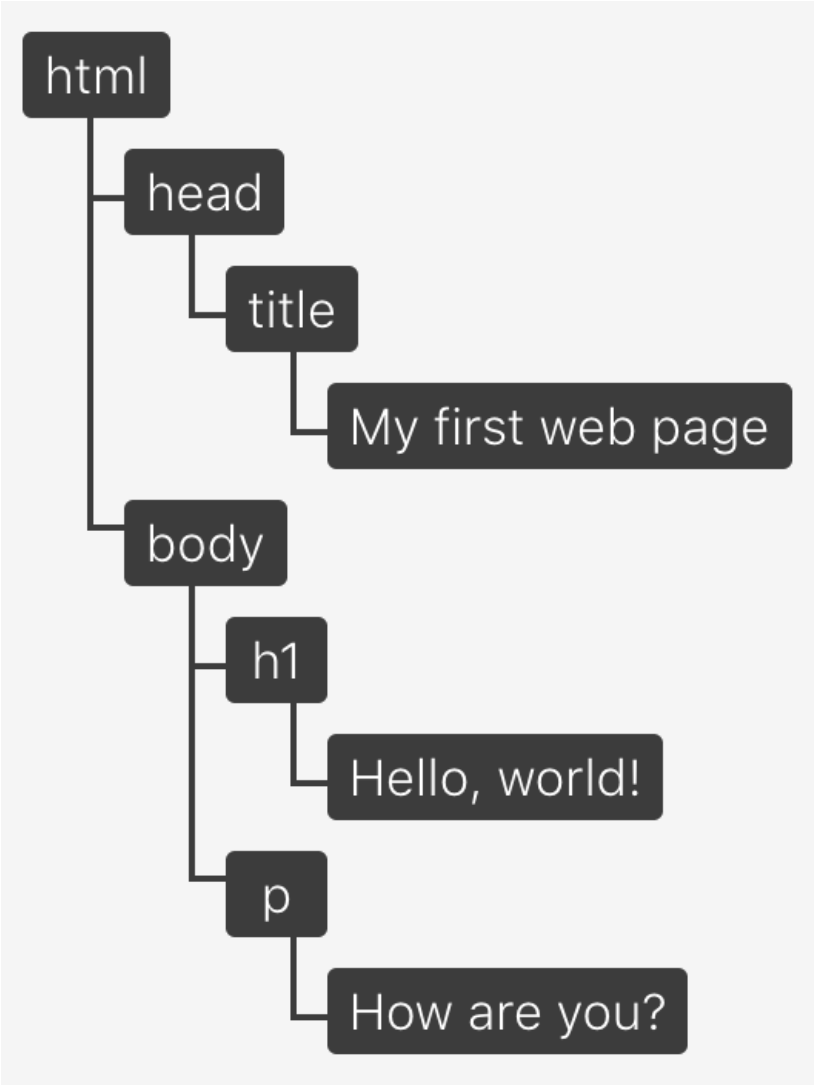
DOM 노드(node)와 요소(element)

- DOM 문서: **node의 계층구조**로 이루어짐. 다양한 유형으로 존재함.
  - element: 노드 중 하나의 유형으로, html 문서에서 , <p>, <title>과 같은 태그 사용해서 작성된 노드



DOM 노드(node)와 요소(element)

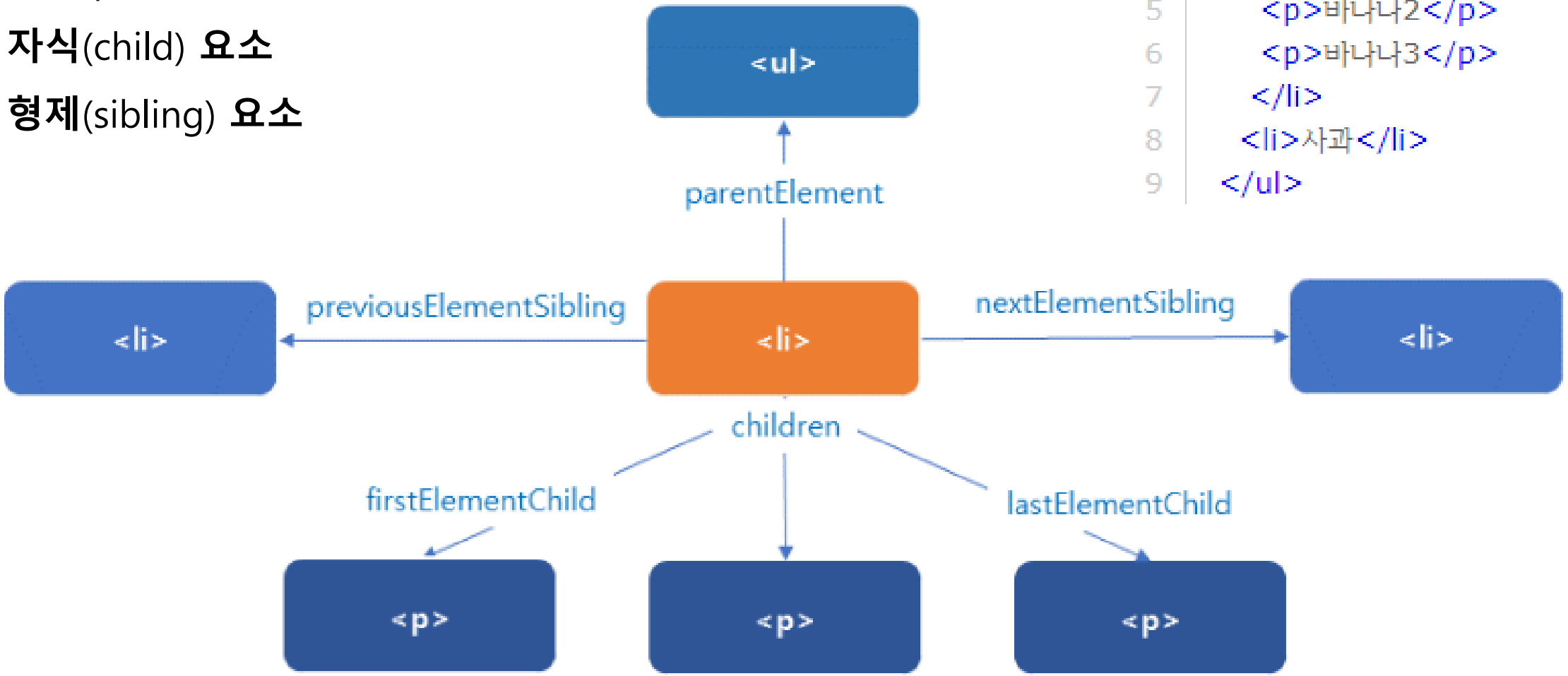
```html
<!doctype html>
<html land="ko">
    <head>
        <title>My first web page</title>
    </head>
    <body>
        <h1>Hello, world!</h1>
        <p>How are you?</p>
    </body>
</html>
```



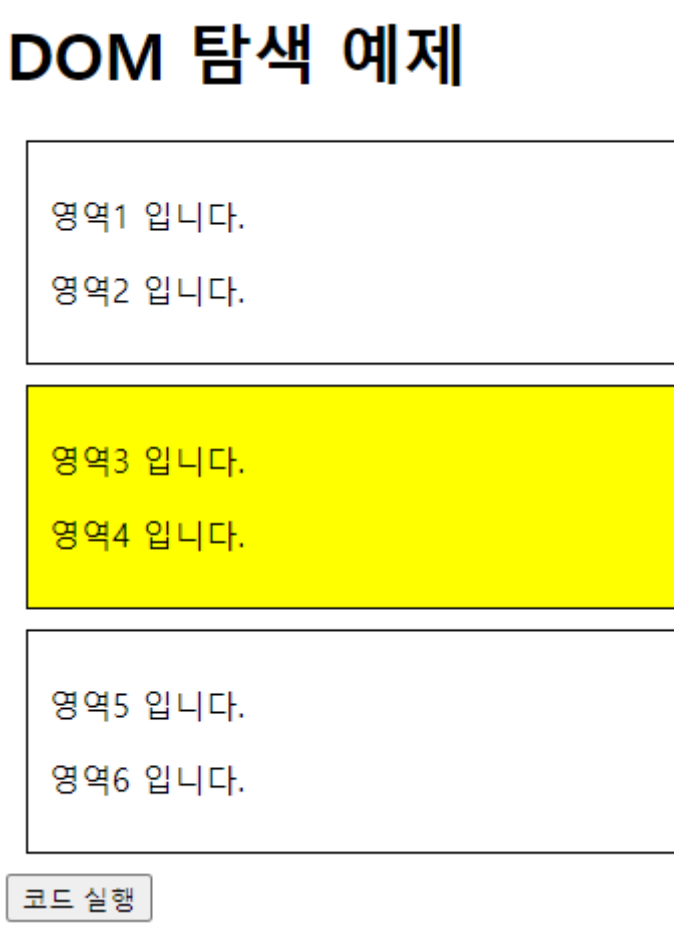
DOM 요소(element) 탐색

- **Element 탐색**
  - **부모**(parent) **요소**
  - **자식**(child) **요소**
  - **형제**(sibling) **요소**

```
1  <ul>
2    <li>딸기</li>
3    <li id='my_li'>
4      <p>바나나1</p>
5      <p>바나나2</p>
6      <p>바나나3</p>
7     </li>
8    <li>사과</li>
9  </ul>
```



DOM 요소(element) 탐색 기법




```
<body>
    <h1 id="header">DOM 탐색 예제</h1>

    <div class="container">
      <p>영역1 입니다.</p>
      <p>영역2 입니다.</p>
    </div>

    <div class="container highlight">
      <p>영역3 입니다.</p>
      <p>영역4 입니다.</p>
    </div>

    <div class="container2">
      <p>영역5 입니다.</p>
      <p>영역6 입니다.</p>
    </div>

    <button>코드 실행</button>
</body>
```

DOM 요소(element) 탐색 기법

```
<style>
  .container {
    margin: 10px;
    padding: 10px;
    border: 1px solid #000;
  }
  .highlight {
    background-color: yellow;
  }
</style>
```

DOM 요소(element) 탐색 기법

1. ID로 요소 찾기 getElementById()

- h1 태그의 ID를 기준으로 텍스트 변경하기

```
const headerElement = document.getElementById('header');
headerElement.textContent = '새로운 제목입니다.';
```

2. 클래스 이름으로 요소 찾기 getElementsByClassName()

- 클래스가 container인 첫 번째 요소의 텍스트를 변경

```
const containers = document.getElementsByClassName('container');
containers[0].textContent = '첫 번째 컨테이너입니다.';
```

DOM 요소(element) 탐색 기법

3. CSS 선택자로 단일 요소 찾기 querySelector()

- 첫 번째 container 요소의 배경색을 변경

```
const firstContainer = document.querySelector('.container');
firstContainer.style.backgroundColor = 'lightblue';
```

4. CSS 선택자로 여러 요소 찾기 querySelectorAll:

- 모든 p 요소의 텍스트를 반복적으로 변경.

```
const paragraphs = document.querySelectorAll('p');

paragraphs.forEach((p, index) => { p.textContent = `문단 ${index + 1}`; });
```

DOM 요소(element) 탐색 기법

5. 부모 요소 찾기 parentElement

- 특정 요소의 부모 요소에 테두리를 추가

```
const paragraph = document.querySelector('.container p');
const parentDiv = paragraph.parentElement;
parentDiv.style.border = '2px solid red';
```

6. 자식 요소 찾기 children

- container 요소의 첫 번째 자식 요소의 텍스트를 변경

```
const firstDiv = document.querySelector('.container2');

const childElements = firstDiv.children;

childElements[0].textContent = '첫 번째 자식 요소입니다.';
```

DOM 요소(element) 탐색 기법

7. 태그 이름으로 요소 찾기 **getElementsByTagName()**

- 모든 div 요소 중 마지막 요소의 배경색을 변경

```
const divs = document.getElementsByTagName('div');
const = divs.length;
divs[length-1].style.backgroundColor = 'lightgreen';
```

DOM 탐색 실습

DOM 탐색 예제

첫 번째 컨테이너의 첫 번째 문단입니다.

첫 번째 컨테이너의 두 번째 문단입니다.

두 번째 컨테이너의 첫 번째 문단입니다.

두 번째 컨테이너의 두 번째 문단입니다.

새로운 DOM 탐색 제목입니다.

변경된 첫 번째 문단입니다.

이것은 첫 번째 컨테이너의 두 번째 자식입니다.

짝수 문단 초록색, 홀수 문단 파란색

두 번째 컨테이너의 첫 번째 문단입니다.

두 번째 컨테이너의 두 번째 문단입니다.

DOM 탐색 실습

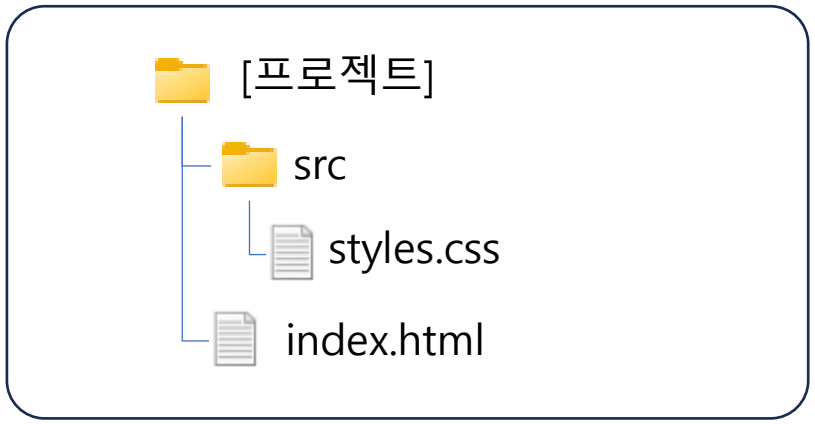
```
<body>
    <h1 id="main-title">DOM 탐색 예제</h1>

    <div class="container">
        <p>첫 번째 컨테이너의 첫 번째 문단입니다.</p>
        <p>첫 번째 컨테이너의 두 번째 문단입니다.</p>
    </div>

    <div class="container">
        <p>두 번째 컨테이너의 첫 번째 문단입니다.</p>
        <p>두 번째 컨테이너의 두 번째 문단입니다.</p>
    </div>
<body>
```

```
<style>
        .container {
            margin: 10px;
            padding: 10px;
            border: 1px solid #000;
        }
        .highlight {
            background-color: yellow;
        }
        p {
            color: #333;
        }
</style>
```

템플릿 만들기



Hello World!

영역1 입니다.

영역2 입니다.

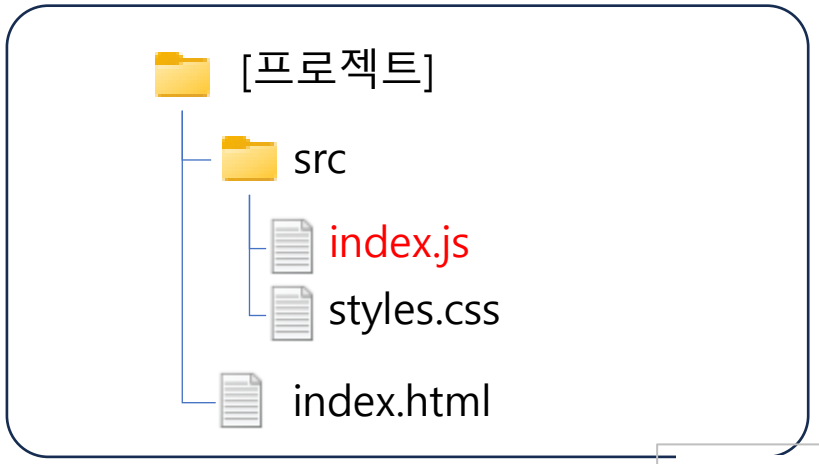
index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>JavaScript에서의 DOM 조작</title>
    <meta charset="UTF-8" />
    <link rel="stylesheet" href="src/styles.css" />
  </head>
  <body>
     <div id="root">
        <h1 id="title">Hello World!</h1>
        <div class="container">
          <p>영역1 입니다.</p>
        </div>
        <div class="container">
          <p>영역2 입니다.</p>
        </div>
     </div>
   </body>
</html>
```

styles.css

```
.container {
  border: solid 1px #ccc;
  padding: 16px;
  margin: 8px;
}
```

자바스크립트를 이용한 DOM 요소 탐색




```
console.log("test");
```

1. index.js 파일 생성

```
<script src="src/index.js"></script>
```

2. index.html 코드 추가

3. index.js 코드 추가

```
const title1 = document.getElementById("title");
console.log(title1);
```

```
<h1 id="title">Hello World!</h1>
```

```
const containers = document.getElementsByClassName("container");
console.log(containers);
```

```
▼ HTMLCollection(2) [div.container, div.container]
  ▶ 0: div.container
  ▶ 1: div.container
    length: 2
  ▶ [[Prototype]]: HTMLCollection
```

자바스크립트를 이용한 DOM 요소 추가

- 자식 요소로 DOM 추가

index.js

에 <h1>타이틀 추가</h1> 요소추가하기

1. 추가할 DOM 요소 생성

```
const h1element = document.createElement("h1")
```

2. 생성한 요소에 텍스트 추가하기

```
h1element.textContent = " 타이틀이 추가되었습니다"
```

3. 생성한 요소를 기존 요소(id='root')에 추가

   1) 상위 요소 가져오기

```
const old = document.getElementById("root")
```

   2) 상위 요소에 붙이기

```
old.appendChild(h1element);
```

자바스크립트를 이용한 DOM 요소 추가

- 형제 요소로 DOM 추가

 와 형제 요소로 아래 내용 추가하기

```
<div class="container">
    <p>영역3 입니다.</p>
</div>
```

Hello World!

영역1 입니다.

영역2 입니다.

영역3 입니다

타이틀이 추가되었습니다

1. 추가할 DOM 요소 div 생성하고 클래스 이름 붙이기

```
const divEl1 = document.createElement("div");
divEl1.className = "container";
```

2. 추가할 DOM 요소 p 생성하고 내용 붙이기

```
const pEl = document.createElement("p");
pEl.textContent = "영역3 입니다";
```

3. Div 요소에 p 요소 붙이기

```
divEl1.appendChild(pEl);
```

4. 상위 요소 찾기

'container' 클래스를 가진 마지막 요소를 찾습니다.

```
const containers =
document.getElementsByClassName('container');
const lastContainer = containers[containers.length - 1];
```

5. 마지막 'container' 요소 아래에 새로 만든 요소 추가

```
lastContainer.after(divEl1);
```

자바스크립트를 이용한 DOM 요소 추가

- DOM 요소 추가 함수: prepend()는 앞쪽에 추가된다.

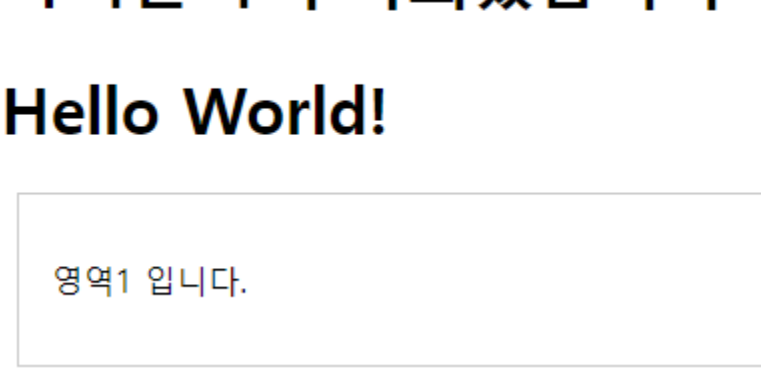
index.js

1. 추가할 DOM 요소 생성
   const h1element = document.createElement("h1")

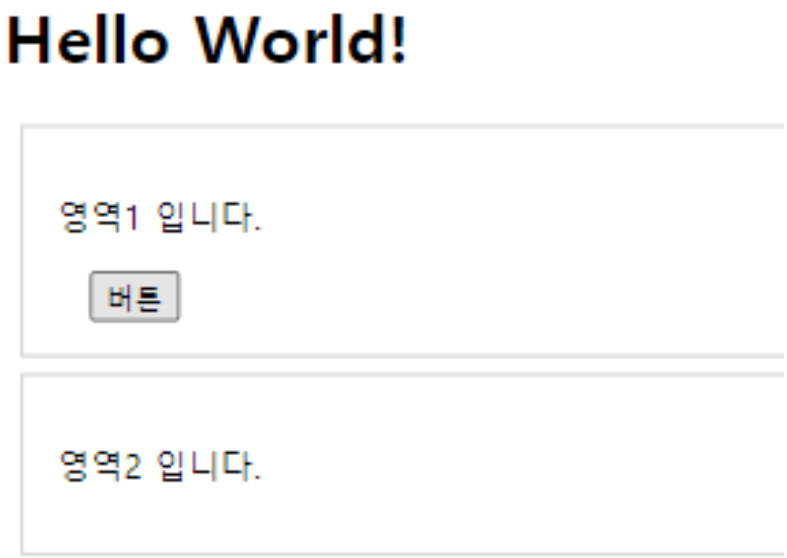
2. 생성한 요소에 텍스트 추가하기
   h1element.textContent = " 타이틀이 추가되었습니다"

3. 생성한 요소를 기존 요소(id='root')에 추가
   1) 상위 요소 가져오기
   const old = document.getElementById("root")

   2) 상위 요소에 붙이기
   old.prepend(h1element);



자바스크립트를 이용한 DOM 요소 추가




```
<div id="root">
   <h1 id="title">Hello World!</h1>
   <div class="container">
     <p>영역1 입니다.</p>
   </div>
   <div class="container">
     <p>영역2 입니다.</p>
   </div>
</div>
```

1. 버튼 요소 만들기
2. 버튼 요소 텍스트 붙이기
3. 버튼 요소 상위 요소 탐색
4. 상위 요소에 append

자바스크립트를 이용한 DOM 요소 추가


```
//'영역1입니다. ' 문장 아래에 버튼 추가하기
const buttonEl = document.createElement("button");
buttonEl.textContent = "버튼";

const containers =
document.getElementsByClassName("container");
console.log(containers);

const firstContainer = containers[0];
firstContainer.appendChild(buttonEl);
```

자바스크립트를 이용한 DOM **삭제**

- removeChild(“엘리먼트”): 지정된 엘리먼트를 삭제
- 특정 요소의 자녀 요소를 모두 삭제: 요소명.textContext = null

  예시: body 태그 아래를 모두 삭제 bodyEl[0].textContent = null;

- 실습: `<h1 id="title">Hello World!</h1>` 삭제하기

Hello World!

문단 영역입니다.

이벤트 처리

이벤트 처리

- 요소에 대한 이벤트 수신기 추가
- addEventListener(type, listener, false)
  - Type: 이벤트 유형
  - Listener: 지정한 이벤트를 수신할 함수(이벤트 처리 함수)

이벤트 처리 예

```html
<table id="outside">
  <tr>
    <td id="t1">one</td>
  </tr>
  <tr>
    <td id="t2">two</td>
  </tr>
</table>
```

```html
<script>
  // 표에 이벤트 수신기 추가
  //1. 표 요소 가져오기
  const el = document.getElementById("outside");

  //2. 이벤트 처리기 정의
  function modifyText() {
    const t2 = document.getElementById("t2");
    if (t2.firstChild.nodeValue == "two") {
      t2.firstChild.nodeValue = "둘";
    } else {
      t2.firstChild.nodeValue = "two";
    }
  }

  //3. 표 요소에 'click' 이벤트 수신기 추가하기
  el.addEventListener("click", modifyText, false);
</script>
```

이벤트 처리

- 익명 함수/화살표 함수로 이벤트 처리기 호출 가능

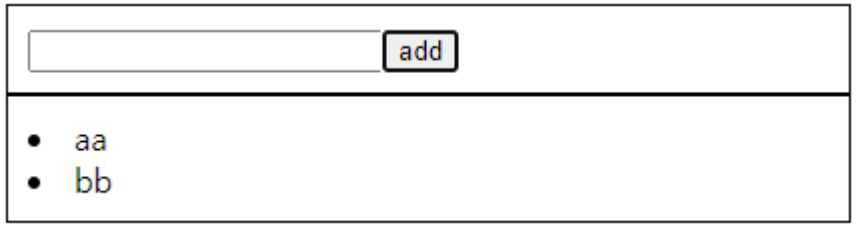
```
<script>
  //1. 표 요소 가져오기
  const el = document.getElementById("outside");

  //2. 이벤트 처리기 정의
  function modifyText(new_text) {
    const t2 = document.getElementById("t2");
    if (t2.firstChild.nodeValue == "two") {
      t2.firstChild.nodeValue = new_text;
    } else {
      t2.firstChild.nodeValue = "two";
    }
  }

  //3. 표 요소에 'click' 이벤트 수신기 추가하기
  el.addEventListener("click", function () {
    modifyText("newText");
  });
</script>
```

이벤트 처리 예

- 입력창의 텍스트를 추가하는 이벤트처리기 만들기




```
<div class="container">
  <input id="in1" />
  <button id="button">add</button>
</div>
<div id="list" class="container"></div>
```

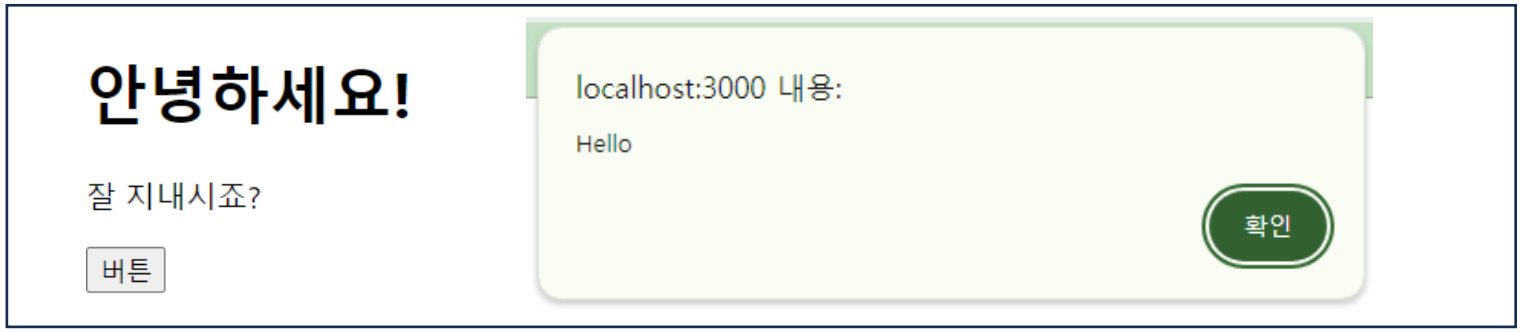
```
<script>
  //1. 버튼 요소 가져오기
  const btnel = document.getElementById("button");

  //2. 이벤트 처리기 정의
  function onClickAdd() {
        //이벤트 처리 코드
  }

  //3. button 요소에 'click' 이벤트 수신기 추가하기
  btnel.addEventListener("click", function () {
    onClickAdd();
  });
</script>
```

이벤트 처리

- on이벤트로 처리기 직접 호출
  - **onClick 했을 때 “Hello” 출력하는 창 띄우기**

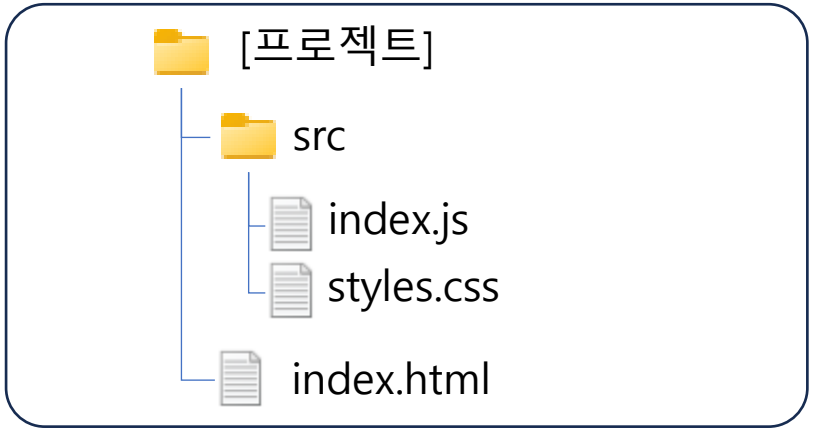



```
const onClickButton = () => {
  alert("Hello");
};
return (
  <div>
    <h1>안녕하세요!</h1>
    <p>잘 지내시죠?</p>
    <button onClick={onClickButton()}>버튼</button>
  </div>
);
```

과제

➢ 다음과 같은 기본 코드가 주어질 때, 버튼 클릭시 메모 추가 또는 삭제가 가능하도록 코드를 추가하시오.

- 폴더 구조



간단 메모 애플리케이션

index.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>간단 메모 애플리케이션</title>
    <meta charset="UTF-8" />
    <link rel="stylesheet" href="src/styles.css" />
  </head>
  <body>
    <div id="root">
      <h1 id="title">간단 메모 애플리케이션</h1>
    </div>

    <script src="src/index.js"></script>
  </body>
</html>
```

과제

styles.css

```css
.container {
  border: solid 1px #ccc;
  padding: 16px;
  margin: 8px;
}

li > div {
  display: flex;
  align-items: center;
}

button {
  margin-left: 16px;
}
```

과제

1. 입력 양식과 '추가' 버튼, 메모 목록 요소를 자바스크립트로 추가하기

  안에 다음 추가

```
<input id="add-text" />
<button id="add-button">추가</button>
<div class="container">
  <p>메모 목록</p>
  <ul id="memo-list"></ul>
</div>
```

과제

2. '추가' 버튼 이벤트 발생시 사용자 입력내용을 메모 목록에 추가하는 스크립트 작성하기

```
// [추가] 버튼 클릭 시, onClickAdd 함수를 실행하도록 addEventListener 등록
document
  .getElementById("add-button")
  .addEventListener("click", () => onClickAdd());
```


```
<p>메모 목록</p>
    <ul id="memo-list">
        <li>
          <div>
            <p>텍스트박스 입력 내용</p>
            <button>삭제</button>
          </div>
        <li>
    </ul>
```

과제

3. '삭제' 버튼 이벤트 발생시 해당 목록 삭제하는 스크립트 작성하기


```
// [삭제]버튼 클릭 시, onClickDel() 함수 실행하도록 addEventListener 등록
button.addEventListener("click", () => onClickDel());
```