



AhnLab SEM

2022. 4. 13 (chapter. 05)

- 자바스크립트 기본 -

인선미

5. 문서 객체 모델과 이벤트

- 1) 이벤트
- 2) 문서 객체 모델
- 3) 이벤트 활용



이벤트



이벤트와 이벤트 핸들러



이벤트



- 웹 브라우저나 사용자가행하는 어떤 동작을 의미
- 키보드의 키를 누르는 것
- 웹 브라우저에서 새로운 페이지를 불러오는 것
- 폼 내용을 입력할 때



이벤트와 이벤트 핸들러



이벤트와 함수

- 대부분의 함수는 사용자가 화면에서 버튼을 클릭하거나 항목을 선택했을 때 실행됨
- 버튼을 클릭하거나 항목을 선택하는 것을 이벤트라고 함
- 이벤트가 발생했을 때 실행하는 함수를 이벤트 핸들러



이벤트와 이벤트 핸들러



마우스 클릭과 이벤트 핸들러

- HTML 태그 안에 on이벤트명에 이벤트를 처리할 함수를 작성



이벤트와 이벤트 핸들러



```
<ul>  
  <li><a href="#" onclick="alert('버튼을 클릭했습니다.')">Green</a></li>  
  <li><a href="#" onclick="alert('버튼을 클릭했습니다.')">Orange</a></li>  
  <li><a href="#" onclick="alert('버튼을 클릭했습니다.')">Purple</a></li>  
</ul>
```



이벤트와 이벤트 핸들러



```
<style>
  #result {
    width:500px;
    height:300px;
    margin:30px auto;
    border:2px solid #ccc;
    border-radius:15px;
  }
</style>
```



이벤트와 이벤트 핸들러



```
<ul>
  <li><a href="#" onclick="changeBg('green')">Green</a></li>
  <li><a href="#" onclick="changeBg('orange')">Orange</a></li>
  <li><a href="#" onclick="changeBg('purple')">Purple</a></li>
</ul>
<div id="result"></div>

<script>
  function changeBg(color) {
    var result = document.querySelector('#result');
    result.style.backgroundColor = color;
  }
</script>
```



문서 객체 모델



문서 객체 모델



문서 객체 document object

- HTML 요소(element): html, head, body, title, h1, div, span
- 자바스크립트에서는 문서 객체라고 부름
- 문서 객체 = HTML 요소



문서 객체 모델



문서 객체 모델 DOM, Document Objects Model

- 문서 객체를 조합해서 만든 전체적인 형태
- 웹 페이지 내의 모든 콘텐츠를 객체로 나타내줌



문서 객체 모델



DOMContentLoaded 이벤트

- 웹 브라우저가 문서 객체를 모두 읽고 나서 실행하는 이벤트
- 브라우저가 HTML을 전부 읽고 DOM 트리를 완성하는 즉시 발생
- 이미지 파일()이나 스타일시트 등의 기타 자원은 기다리지 않음



문서 객체 모델



```
document.addEventListener('DOMContentLoaded', () => { })
```



document라는 문서 객체의 DOMContentLoaded 이벤트가 발생했을 때,
매개변수로 지정한 콜백 함수를 실행해라



문서 객체 모델



문서 객체 가져오기

- document.body 코드를 사용해 문서의 body 요소를 읽어들이 수 있음

```
document.head  
document.body  
document.title
```



문서 객체 모델



문서 객체 가져오기

- body 요소 내부에 만든 다른 요소에 접근
- 메소드를 사용해 접근
- CSS 선택자



```
document.querySelector(선택자)           // 요소 하나만 추출  
document.querySelectorAll(선택자)       // 여러 개 추출
```



문서 객체 모델



CSS 선택자

이름	선택자 형태	설명
태그 선택자	태그	특정 태그를 가진 요소 추출
아이디 선택자	#아이디	특정 id를 가진 요소를 추출
클래스 선택자	.클래스	특정 class를 가진 요소 추출
속성 선택자	[속성=값]	특정 속성 값을 갖고 있는 요소 추출
후손 선택자	선택자_A 선택자_B	선택자_A아래에 있는 선택자_B 선택

문서 객체 모델

querySelector() 메소드

- 요소를 하나만 추출
- h1 태그를 추출하고 조작


```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>DOMContentLoaded</title>
6  <script>
7      document.addEventListener('DOMContentLoaded', () => {
8          const header = document.querySelector('h1')
9
10         header.textContent = 'HEADERS'
11         header.style.color = 'white'
12         header.style.backgroundColor = 'black'
13         header.style.padding = '10px'
14     })
15 </script>
16 </head>
17 <body>
18     <h1></h1>
19 </body>
20 </html>
```

HEADERS

문서 객체 모델



querySelectorAll() 메소드

- 문서 객체 여러 개를 배열로 읽어 들임
- 내부 요소에 접근하고 활용하기 위해 반복을 돌려야 함
- 일반적으로 forEach() 메소드 사용해서 반복




```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    const headers = document.querySelectorAll('h1')

    headers.forEach((header) => {
      header.textContent = 'HEADERS'
      header.style.color = 'white'
      header.style.backgroundColor = 'blue'
      header.style.padding = '10px'
    })
  })
</script>
```

```
<body>
  <h1></h1>
  <h1></h1>
  <h1></h1>
  <h1></h1>
</body>
```

HEADERS

HEADERS

HEADERS

HEADERS

문서 객체 모델



이벤트 설정하기

- addEventListener() 메소드
- **문서객체.addEventListener(이벤트 이름, 콜백함수)**
- 콜백 함수는 이벤트 리스너 또는 이벤트 핸들러라고 부름
- h1 태그를 클릭할 때 이벤트 리스너를 호출



이벤트 설정하기



```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    let counter = 0;
    const h1 = document.querySelector('h1');

    h1.addEventListener('click', (event) => {
      counter++;
      h1.textContent = `클릭 횟수: ${counter}`;
    })
  })
</script>
```

클릭 횟수: 16

이벤트 설정하기



```
<style>
  h1{
    /* 클릭을 여러 번 했을 때
    |   글자가 선택되는 것을 막기 위한 스타일 */
    user-select: none;
  }
</style>
</head>
<body>
  <h1>클릭 횟수: 0</h1>
</body>
```

클릭 횟수: 16

문서 객체 모델



이벤트 설정하기

- removeEventListener() 메소드
- 문서객체.removeEventListener(이벤트 이름, 이벤트 핸들러)



이벤트 설정하기 제거하기



```
<script>
document.addEventListener('DOMContentLoaded', () => {
  let counter = 0;
  let isConnected = false;

  const h1 = document.querySelector('h1');
  const p = document.querySelector('p');
  const connectButton = document.querySelector('#connect');
  const disconnectButton = document.querySelector('#disconnect');

  const listener = (event) => {
    h1.textContent = `클릭 횟수: ${counter++}`;
  }
}
```

클릭 횟수: 15

이벤트 연결

이벤트 제거

이벤트 연결 상태: 해제


```
connectButton.addEventListener('click', () => {
  if (isConnect === false) {
    h1.addEventListener('click', listener);
    p.textContent = '이벤트 연결 상태: 연결';
    isConnect = true;
  }
})
disconnectButton.addEventListener('click', () => {
  if (isConnect === true) {
    h1.removeEventListener('click', listener);
    p.textContent = '이벤트 연결 상태: 해제';
    isConnect = false;
  }
})
})
</script>
```

클릭 횟수: 15

이벤트 연결

이벤트 제거

이벤트 연결 상태: 해제

이벤트 설정하기 제거하기



```
<body>
  <h1>클릭 횟수: 0</h1>
  <button id="connect">이벤트 연결</button>
  <button id="disconnect">이벤트 제거</button>
  <p>이벤트 연결 상태: 해제</p>
</body>
```

AhnLab SEM

클릭 횟수: 15



이벤트 연결

이벤트 제거

이벤트 연결 상태: 해제



이벤트 활용



이벤트 활용



이벤트 모델

- 이벤트 모델: 이벤트를 연결하는 방법
- 표준 이벤트 모델: `addEventListener()`

```
document.body.addEventListener('keyup', () => {  
  
}))
```

이벤트 활용



고전적인 이벤트 모델

- 문서 객체가 가지고 있는 onXX으로 시작하는 속성에 함수를 할당해서 이벤트 연결

```
document.body.onkeyup = (event) => {
```



속성

```
}
```


이벤트 활용



인라인 이벤트 모델

- HTML요소에 직접 onXX으로 시작하는 속성에 함수를 할당해서 이벤트 연결

```
<script>
  const listener = (event) => {

  }
</script>
<body onkeyup="listener(event)">

</body>
```


이벤트 활용

키보드 이벤트

- keydown
- keypress
- keyup

키보드 이벤트



```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    const textarea = document.querySelector('textarea');
    const h1 = document.querySelector('h1');

    textarea.addEventListener('keyup', (event) => {
      const length = textarea.value.length;
      h1.textContent = `글자 수: ${length}`;
    })
  })
</script>
```

```
<body>
  <h1></h1>
  <textarea></textarea>
</body>
```

글자 수: 12

안녕하세요 반갑습니다.

이벤트 활용

글자 입력 양식 이벤트

- form 입력양식
- 로그인이나 회원가입 양식



```
<script>
```

```
document.addEventListener('DOMContentLoaded', () => {
  const input = document.querySelector('input')
  const button = document.querySelector('button')
  const p = document.querySelector('p')
```

```
  button.addEventListener('click', () => {
    const inch = Number(input.value)
    if (isNaN(inch)) {
      p.textContent = '숫자를 입력해주세요'
      return
    }
    const cm = inch * 2.54
    p.textContent = `${cm} cm`
  })
})
```

```
</script>
```

```
<body>
  <input type="text" value="17" /> inch<br>
  <button>계산</button>
  <p></p>
</body>
```

inch

43.18 cm

이벤트 활용



글자 입력 양식 이벤트

- 인터넷에서 특정 사이트에 가입할 때 이메일과 전화번호 유효성 등을 검사
- 일반적으로 이런 유효성 검사를 할 때에는 정규 표현식regular expression을 사용



```
<script>
  document.addEventListener('DOMContentLoaded', () => {
    const input = document.querySelector('input');
    const p = document.querySelector('p');
    const isEmail = (value) => {
      // 골뱅이를 갖고 있고 && 골뱅이 뒤에 점이 있다면
      return (value.indexOf('@') > 1)
        && (value.split('@')[1].indexOf('.') > 1);
    }

    input.addEventListener('keyup', (event) => {
      const value = event.currentTarget.value;
      if (isEmail(value)) {
        p.style.color = 'green';
        p.textContent = `이메일 형식입니다: ${value}`;
      } else {
        p.style.color = 'red';
        p.textContent = `이메일 형식이 아닙니다: ${value}`;
      }
    })
  })
</script>
```

```
<body>
  <input type="text">
  <p></p>
</body>
```

이메일 형식입니다: abc@gmail.com

미션> ToDoList 만들기



할 일 목록

추가하기

☐ 과제 제출하기

제거하기

☐ 마트 장보기

제거하기

☐ 영화 보기

제거하기

☒ ~~안경 맞추기~~

제거하기



감사합니다.



AhnLab SEM