9/22 강의자료

1교시

2-01 배경색 바꾸기

▼ 이벤트 리스너와 이벤트 핸들러란?

이벤트 리스너는 특정 이벤트에 대해서 일어날 동작을 정의 할 수 있습니다.

형태는 target.addEventLister('이벤트', eventHandler); 와 같은데요

이때 이벤트 핸들러란 이벤트가 발생했을 때 처리를 담당하는 실행 함수를 가리킵니다.

이벤트 핸들러 함수에는 이벤트 객체(target)가 인자로 전달 되는데요. 전달 받은 이벤트 객체를 이용하셔 이벤트의 성질을 결정하거나 이벤트 기본동작을 막을 수있습니다.

▼ 이벤트 핸들러를 등록방법은?

이벤트 핸들러를 등록하는 방법은 크게 세가지가 있는데요

먼저 HTML 코드내에 속성으로서 등록을하는 인라인 방식과

DOM요소에 프로퍼티를 사용하여 바로 실행할 함수를 등록하는 방법

그리고 마지막으로 이벤트 리스너를 사용하는 방법이 있는데요

앞의 두가지 방법에는 여러가지 이벤트 핸들러를 등록하기 힘들다는 단점과 수정이 어렵다는 단점이 존재해 이벤트 리스너를 통한 이벤트 등록방법을 가장 권장드립니다.

이들의 가장 큰 차이점은

여러개의 이벤트 핸들러르 등록하는 방법과 이벤트를 제거하는 방법의 차이가 있는데요

코드로 예시를 들겠습니다 🙂

addEventListener

특정 DOM요소에 이벤트 리스너를 등록할 때는 addEventListener를 사용하는데요

> DOM객체. addEventListener(이벤트명, 실행할 함수, 옵션)

이벤트명 : Javascript에서 발생할 수 있는 이벤트 명을 입력

실행할 함수 : 해당 변수는 생략 가능하며, 해당 이벤트가 발생할 때 실행할 함수 명을 입력

옵션: 옵션은 생략이 가능하며, 자식과 부모 요소에서 발생하는 버블링을 제어하기 위한 옵션

사용 예시

```
blockOne.addEventListener("mouseenter", () => {
    blockOne.className = "red";
});

blockTwo.addEventListener("mouseenter", function () {
    blockTwo.className = "yellow";
});

function green() {
    blockThree.className = "green";
}

blockThree.addEventListener("mouseenter", green);
```

이벤트 삭제 예시

https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_element_addeventlistener_remove

tip) 이벤트 한번만 실행되도록하기

한번 실행된 후 이벤트를 제거해주는 방법도 있지만

addEventListener의 3번째 파라미터를 사용하면 쉽게 해결할 수 있습니다.

▼ 자주쓰는 이벤트 종류

마우스 이벤트

키보드 이벤트

포커스 이벤트

입력 이벤트

스크롤 이벤트

윈도우 창 이벤트

▼ class를 추가 변경하는 방법

크게 className과 classList를 사용하는 방식이 있습니다.

className은 class 전체를 가져오며 새로운 값을 가져왔을 때 이전의 모든 클래스가 사라지고 덮어쓰기 방식으로 할당됩니다.

className을 잘못 사용하면 이전에 설정한 class들이 모두 사라질 위험이 있기 때문에 classList를 사용하시길 권장드립니다.

classList의 메서드

add : 지정한 클래스 값 추가

remove : 지정한 클래스 값 제거

toggle : 지정한 클래스가 있으면 제거, 없으면 추가

2-02 화면의 상단으로 이동하기

▼ DOM요소 지정하는 방법은?

DOM요소을 지정하는 방법으로는

Id를 이용할 때 : getElementById(id) : 입력된 id를 가진 요소 (단수)

Class를 이용할 때 : getElementsByClassName(names) : 입력된 names 클래스를 가진 요소들 (복수)

Tag를 이용할 때 : getElementsByTagName(name) : 입력된 name 태그를 가진 요소들(복수)

CSS 선택자를 이용할 때 :

1. querySelector(selectors) : selectors로 선택한 요소중 첫번째 값 (단수)
2. querySelectoraAll(selectors) : selectors로 선택한 모든 요소 (복수)

여러분의 기호에 맞게 사용하시면 됩니다~!

참고 : https://developer.mozilla.org/ko/

▼ 스크롤을 이동시키는 방법은?

window의 메서드인 scrollTo를 이용해서 원하는 위치로 스크롤을 이동시킬 수 있습니다.

```
window.scrollTo({top, left, behavior})

// top : 세로위치, left: 가로위치, behavior: 스크롤 효과속성
// behavior 옵션
// - auto : 기본값, 바로 위치로 이동
// - smooth : 부드럽게 이동
```

▼ window객체란?

웹 브라우저의 창(window)을 나타내는 객체로 JS의 모든 객체, 전역변수, 전역함수들은 자동으로 window의 프로퍼티가 됩니다. 사실은 우리가 선언한 변수나 함수에도 앞에 window가 붙어있지만 모든 객체를 포함하고 있기때문에 생략해서 사용하는 것입니다.

2교시

2-03 출석하기

▼ querySelector 심화편

1주차에 열심히 배웠던 CSS Selector!! 여기서 한 번 다시 복습해 볼까요?

querySelector를 사용하면 우리가 배웠던 CSS 선택자를 사용해서 DOM요소를 택할 수 있습니다!!

2-04 FORM 다루기

▼ form의 기본 이벤트, 특성

form 태그는 웹 페이지에서의 입력 양식을 의미하는 태그로 로그인 창, 회원가입 폼등으로 사용되는데요

화면에 보이지 않는 추상적인 태그입니다.

실제로 사용자가 양식을 입력하기 위해서는 input태그를,Button 태그 등을 form태그 안에 넣어 사용합니다.

input태그의 타입으로는 text, password, button, submit 등이 있으며

submit의 기본이벤트로 새로고침이 있습니다. 하지만 우리가 풀어야 할 문제에서는 새로고침이 되면서 text값이 사라지므로 이를 없애줘야하는데

기본 이벤트를 삭제하는 메서드인 preventDefault()를 사용하시면 새로고침을 막을 수 있습니다.

참고 : https://developer.mozilla.org/ko/docs/Web/API/Event/preventDefault

▼ JS로 태그안에 텍스트를 추가하는 방법은?

JS로 태그안에 텍스트를 추가하는 여러가지 방법이 있는데요 innerHTML, outerHTML, textContent 등이 있습니다. 이를 비교해보겠습니다.

innerHTML : 요소 노드 내부의 HTML 코드를 문자열로 리턴해주며 새로운 값이 기존의 값을 덮어씁니다. (내부의 줄바꿈, 들여쓰기 모두 포함)

outerHTML : 요소 노드 자체의 전체적인 HTML 코드를 문자열로 리턴해주며 새로운 값이 기존의 값을 겊어씁니다. (내부의 줄바꿈, 들여쓰기 모두 포함)

textContent : 요소 안의 내용들 중에서 HTML 태그 부분은 제외하고 텍스트만 가져오며 새로운 값을 할당하면 innerHTML과 마찬가지로 내부의 값을 완전히 새로운 값으로 교체합니다. (내부의 줄바꿈, 들여쓰기 모두 포함)

하지만 다른점은 innerHTML처럼 내부의 태그도 가져오는것이아니라 단어그대로 text만가져옵니다. 이때 html코드를 적어도 그저 text로 인식합니다.

```
console.log("innerHTML : ", form.innerHTML);
console.log("outerHTML : ", form.outerHTML);
console.log("textContent : ", form.textContent);
```

```
innerHTML :
            <label>입력</label>
            <input type="text" id="input">
            <button type="submit" id="btn">제출</button>

outerHTML :  <form id="form">
            <label>입력</label>
            <input type="text" id="input">
            <button type="submit" id="btn">제출</button>
        </form>
textContent :
            입력

            제출
```

2-05 N번째 요소 만들기

▼ JS로 HTML에 태그를 추가하는 방법은?

우리가 1주차에 배웠던 HTML을 생각해보면

HTML을 추가하려면 무엇이 필요할까요?

먼저 어떤 태그를 사용할 것인지, 그리고 태그에 어떤 속성을 넣을 것인지, 어디에 추가할 것인지 정도가 있겠죠?

첫번째로 어떤 태그를 사용할 것인지는 **createElement( )** 라는 메서드를 사용해서 원하는 태그를 생성할 수 있습니다.

```
const addTag = document.createElement("li");
```

추가 해줄 속성으로는 **setAttribute( )** 라는 메서드를 사용할 수 있는데요

```
addTag.createElementsetAttribute("속성" : "값")
```

위의 방법으로 class를 추가해줄 수 있습니다. (class를 추가하는 다른 방법은 밑에서 배워보겠습니다.)

마지막으로 어느 위치에 넣어줄지를 정해야하는데 **appendChild( )**를 사용할 수 있습니다.

appendChild()는 지정해놓은 태그의 자식태그중 가장 마지막에 새로 생성한 태그를 넣어줍니다.

```
myUl.appendChild(addTag);
```

3교시

2-06 태그의 속성에 따라 하이라이트하기

▼ JS로 CSS 속성 바꾸는 방법은?

DOM요소의 style을 바꾸는 방법은 style을 인라인 방식으로 넣는 방법과 내부스타일시트, 외부 스타일 시트을 통해서 넣는 방식이 있습니다.

먼저 인라인 방식은 DOM요소.style 뒤에 넣고 싶은 CSS값을 카멜케이스 방식을 이용해서 넣어주면 됩니다.

```
str.style.color = "blue"
```

내부 스타일 시트를 통해 넣는 방식은 먼저 document.createElement('style');을 통해 내부 스타일 시트를 만들어주고 그 안에 style값을 넣어주는 방식입니다.

```
let styleTest = document.createElement('style');
styleTest.innerHTML="div{height:30px;} a{color:#fff;}";
document.head.appendChild(styleTest);
```

외부 스타일 시트를 통해 넣는 방식은 CSS파일에 따로 style을 작성한 후 HTML에 class를 추가해서 넣는 방식입니다. 1번문제에서 배웠던 classList.add를 통해 작성한 css style을 추가해주는방식입니다.

```
boxs.classList.add("on"); // DOM요소인 boxs에 on이라는 className을 추가함
```

2-07 불 좀 꺼줄래?

▼ 이벤트리스너의 콜백함수 응용

1번 문제에서 배웠던 이벤트리스너에서 이벤트 핸들러를 등록하는 방법을 다양하게 실습해 봐요!

(선택)

▼ this란..?

이를 알아야 화요일 자료의 function_arrow function2.js의 코드를 제대로 이해 할 수 있습니다.

this란 자신이 속한 객체 또는 자신이 생성할 인스턴스를 가리키는 자기 참조 변수로서 this의 값은 어느 위치인지 어디서 호출하는지 어떤 함수에 있는지에 따라 달라지는 특성이 있습니다.

예제코드를 통해 알아보겠습니다.

일반함수 ⇒ this : this를 호출한 객체

화살표함수 ⇒ this : this를 호출한 객체의 부모