

🖥️ 시간표 기능 넣기 🖥️



이제 시간표에 **일정 추가 기능**을 넣어볼 것인데요!

저는 부트스트랩과 CSS를 활용해서
시간표를 다듬어봤어요 😊

간단한 자기소개

이름❤️ 박민선
mbti❤️ ISTP
학교❤️ 한양대학교 ERICA캠퍼스
학과❤️ 소프트웨어학부 컴퓨터전공
취미❤️ 나들이 가기
좋아하는 음식❤️ 일식, 고기 등

나의 시간표

시간	일	월	화	수	목	금	토
9:00-10:00							
10:00-11:00							
11:00-12:00							
12:00-13:00							
13:00-14:00							
14:00-15:00							
15:00-16:00							
16:00-17:00							
17:00-18:00							

멋쟁이 사자처럼 10기

박민선

먼저 기능까지 추가된 완성본을 한번 볼까요?

칸을 클릭하면, 아래처럼 input 박스가 등장해요

나의 시간표				
시간	일	월	화	수
9:00-10:00	<input type="text"/>			
10:00-11:00				

input 박스에 내용을 입력하고 엔터를 누르면

나의 시간표				
시간	일	월	화	
9:00-10:00	운동			
10:00-11:00				

일정이 추가돼요.
그리고 수정 상태에서 박스를 클릭하면 색이 무작위로 변해요.

나의 시간표

시간	일	월	화	수
9:00-10:00	운동하기			
10:00-11:00				
11:00-				

다시 클릭하면 내용을 수정할 수도 있습니다

나의 시간표

시간	일	월	화	수
9:00-10:00	밥먹기			
10:00-11:00				
11:00-				

나의 시간표

시간	일	월	화	수
9:00-10:00	밥먹기			
10:00-11:00				
11:00-12:00				

만약 일정을 삭제하고 싶다면,
칸을 클릭해서 내용을 지워주면 돼요

나의 시간표

시간	일	월	화	수
9:00-10:00	<input type="text"/>			
10:00-11:00				
11:00-				

나의 시간표

시간	일	월	화
9:00- 10:00			
10:00- 11:00			
11:00-			

이 기능을 **JavaScript**를 사용해서 만들어볼 거예요

JavaScript



시간표의 기능을 구현하기에 앞서,
구현에 필요한 몇 가지 JavaScript 지식을 학습하고자 합니다.

크롬이나 Edge 등의 웹 브라우저에서
개발자 도구를 켜고(단축키 F12) 이후에 나오는 코드들을
따라 쳐보시면 이해가 잘 될 겁니다 !

JavaScript의 변수 선언 방식에는 크게 'const'와 'let'이 있습니다.

const로 선언한 변수는 수정이 불가능합니다.

```
> const a = 1
< undefined
> a = 2
✖ ▶ Uncaught TypeError: Assignment to constant variable.
   at <anonymous>:1:3
```

하지만 let으로 선언한 변수는 수정이 가능합니다.

```
> let b = 1
< undefined
> b = 2
< 2
```


때문에

`const`는 주로 함수를 선언하거나
HTML태그를 선택하여 변수에 넣어줄 때 사용하고,

`let`은 주로 변수나 수정해야할 값을 선언할 때 사용합니다.

함수 선언

JavaScript에선 아래의 3가지 방식으로 함수를 선언합니다.

```

1  function fn1(a, b) {
2      console.log('함수');
3
4      let c = a + b;
5
6      return c;
7  };
8
9  const fn2 = function fn(a, b) {
10     console.log('함수를 변수에 넣기');
11
12     let c = a - b;
13
14     return c;
15 };
16
17 const fn3 = (a, b) => {
18     console.log('화살표 함수');
19
20     let c = a * b;
21
22     return c;
23 }
    
```

일반적인 함수

함수를 변수에 넣기

화살표 함수

세 함수는 모두 다음과 같이 호출할 수 있습니다.

```
console.log(fn1(1, 2));
함수
3
undefined
console.log(fn2(1, 2));
함수를 변수에 넣기
-1
undefined
console.log(fn3(1, 2));
화살표 함수
2
```

Console.log(함수명)으로 함수를 호출하면서
그 함수의 리턴값을 화면에 출력해주고 있습니다.

세 함수 모두 함수 내에서 console.log 코드가 있기에
각 '함수', '함수를 변수에 넣기', '화살표 함수'가 출력되고,
그 리턴값이 화면에 출력된 것을 볼 수 있습니다.

화살표 함수의 경우 함수를 간결하게 표현할 수 있기에 편리하고 많이 사용되는 형태입니다.

또한 JavaScript에서는 함수를 호출할 때, 인자로 다른 함수를 넘겨줄 수 있습니다.

이를 CallBack 함수라고 부릅니다.

예를 들어 (다음 페이지)

callbackFunction은 "4입니다."를 출력해주는 함수이고,
fn은 콜백함수와 숫자를 입력 받고 입력 받은 숫자가 4일 경우
입력 받은 콜백함수를 호출해주는 함수입니다.

```
> const callbackFunction = () => {
  console.log('4입니다.');
```

```
}

const fn = (fn, num) => {
  if (num == 4) {
    fn();
  }
  else {
    console.log('4가 아닙니다.');
```

```
}

< undefined
> fn(callbackFunction, 4)
4입니다.
< undefined
> fn(callbackFunction, 3)
4가 아닙니다.
```

자바스크립트 이벤트

자바스크립트 이벤트란 사용자가 어떤 동작을 했을 때
어떠한 사건을 발생시키는 것을 말합니다.

이벤트의 종류에는 click, keydown 등이 있고,

이벤트가 발생할 때에는 event객체가 생성됩니다.

이 event객체에는 사용자가 건드린 html요소가 들어있습니다.

```
const onClick = (e) => {
  console.log(e);
}
```

```
</table>
<button onClick=onClick(this)>HTML내용</button>
```

예시로 event객체를 인자로 받아 콘솔에 출력해주는 함수가 있고,
버튼을 클릭하면 이 함수가 호출되도록 했습니다.
(this는 event객체를 인자로 넘겨주는 것입니다.)


```
<button onclick="onClick(this)">HTML내용</button> scripts.js:27
```

버튼을 클릭할 시 클릭한 버튼의 HTML요소가 출력되는 것을 볼 수 있습니다.

저희는 이것을 활용하여 시간표의 기능을 완성할 것입니다.

이제 직접 시간표의 기능을 구현해보면서 학습해보겠습니다 !

먼저 시간표 html 파일과 동일한 경로에 js파일을 생성해줍니다.

 시간표.html

 scripts.js

```

const box = document.querySelectorAll('td');

const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className !== "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};

const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};

box.forEach((target) => target.addEventListener("click", onBoxClick));

```

전체 코드는 다음과 같습니다.

하나씩 설명해드리겠습니다 !

```
const box = document.querySelectorAll('td');
```

```
const onBoxClick = (e) => {  
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {  
    e.target.className = "box-edit";  
    e.target.innerHTML = `value="${e.target.innerHTML}"/>`;  
  }  
  else if(e.target.className !== "input-text"){  
    let num1 = Math.floor(Math.random() * 256);  
    let num2 = Math.floor(Math.random() * 256);  
    let num3 = Math.floor(Math.random() * 256);  
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";  
  }  
};
```

```
const Enter = (e) => {  
  if (event.keyCode == 13) {  
    e.parentNode.className = "box";  
    e.parentNode.innerHTML = e.value;  
  }  
};
```

```
box.forEach((target) => target.addEventListener("click", onBoxClick));
```

```
const box = document.querySelectorAll('td');
```

이 코드는 td라는 태그를 모두 골라 배열의 형태로 box에 넣어주는 코드입니다.

```
> console.log(box) VM289:1
```

NodeList(63) [td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td, td]

- ▶ 0: td
- ▶ 1: td
- ▶ 2: td
- ▶ 3: td
- ▶ 4: td
- ▶ 5: td
- ▶ 6: td
- ▶ 7: td
- ▶ 8: td
- ▶ 9: td
- ▶ 10: td
- ▶ 11: td
- ▶ 12: td
- ▶ 13: td
- ▶ 14: td
- ▶ 15: td
- ▶ 16: td
- ▶ 17: td
- ▶ 18: td
- ▶ 19: td
- ▶ 20: td

**box를 출력해보면 각 td태그가
배열의 형태로 box 안에 들어간 것을
알 수 있습니다.**

```
document.querySelector('tag') // 태그 이름으로 찾기
document.querySelector('.class') // 클래스 이름으로 찾기
document.querySelector('#id') // 아이디로 찾기
document.getElementById('id') // 아이디로 찾기
```

앞서 말한 `querySelectorAll` 외에도 자주 쓰이는 것들을 가져왔습니다.

신경써야 하는 부분은 `querySelector`에서

클래스 이름 앞에는 .
아이디 앞에는 #

을 붙여야 합니다.


```
const box = document.querySelectorAll('td');

const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className !== "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};

const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};

box.forEach((target) => target.addEventListener("click", onBoxClick));
```

```
box.forEach((target) => target.addEventListener("click", onBoxClick));
```

forEach는 주어진 함수를 배열 요소 각각에 실행합니다.

addEventListener는 첫번째 인자로 주어진 행위가 감지되면
두번째 인자로 주어진 함수를 실행합니다.

위 코드에서 box안에는 시간표의 각 네모 칸(td태그들)이 들어있고
target에는 각 td태그 하나씩 들어가게 되며 각 태그에 대해
addEventListener가 적용되므로,

그 칸들 각각을 클릭할 시 onBoxClick 이라는 함수가 실행됩니다.

```
const box = document.querySelectorAll('td');
```

```
const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className !== "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};
```

```
const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};
```

```
box.forEach((target) => target.addEventListener("click", onBoxClick));
```

이 코드는 시간표를 클릭 했을 때 발동되는 함수입니다.

복잡해 보이지만 매우 간단합니다.

```

const box = document.querySelectorAll('td');

const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className !== "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};

```

**if문 부분입니다. 앞서 설명했듯이 e.target에는
유저가 클릭한 부분의 html요소가 들어있습니다.**

```

const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML
  }
};

```

**따라서 해당 태그의 class가 "box-edit"이 아니면서
동시에 "input-text"가 아닐 때 if문 안의 코드를 발동시킨다는 뜻입니다.**

```

box.forEach((target) => target.addEventListener("click", onBoxClick));

```

```
e.target.className = "box-edit";
e.target.innerHTML = `

```

class이름을 "box-edit"으로 바꾸어 주고,

innerHTML을 사용하면 해당 태그 내부에 HTML코드를 넣을 수 있습니다.

박스 내부에 input 태그를 추가하여 시간표를 수정할 수 있도록 해주었습니다.

여기서 문자열을 입력하는 방법으로 `(1번 키보드 왼쪽)`을 사용하는데,
 ``로 감싼 문자열 내에서 `\${변수이름}`을 사용하면 변수 값을 문자열 내에
 삽입해줄 수 있습니다.



앞으로 웹 개발을 하면서 많이 쓰일 요소입니다.

```

const box = document.querySelectorAll('td');

const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className != "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};

```

이 코드는 수정 중인 상태에서 박스를 클릭할 시 박스의 색을 랜덤으로 바꾸어 주는 코드입니다.

```

const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML =
  }
};

```

만약 클릭한 곳이 input태그 내부이면 안되기 때문에, if문에 className != "input-text"를 넣어주었습니다.

```

box.forEach((target) => target.addEventListener("click", onBoxClick));

```

```
let num1 = Math.floor(Math.random() * 256);
let num2 = Math.floor(Math.random() * 256);
let num3 = Math.floor(Math.random() * 256);
e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
```

Math.random()은 0~1 사이의 랜덤한 **난수**를 생성합니다.

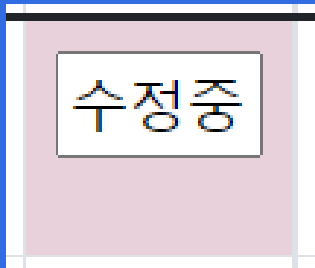
Math.floor()는 주어진 숫자와 같거나 더 작은 수 중 가장 큰 **정수**를 반환합니다.

즉 num1, 2, 3은 각각 0~256 사이의 숫자를 랜덤으로 생성하는 것입니다.

style.backgroundColor는 배경색을 지정해주는 코드입니다.

색으로는 rgb(red, green, blue, 투명도(0~1))가 들어가며,

red, green, blue은 각각 랜덤한 숫자로 넣어주었습니다.



```

const box = document.querySelectorAll('td');

const onBoxClick = (e) => {
  if ((e.target.className !== "box-edit") && e.target.className !== "input-text") {
    e.target.className = "box-edit";
    e.target.innerHTML = `<input class="input-text" type="text" onkeydown="Enter(this)"
value="${e.target.innerHTML}"/>`;
  }
  else if(e.target.className != "input-text"){
    let num1 = Math.floor(Math.random() * 256);
    let num2 = Math.floor(Math.random() * 256);
    let num3 = Math.floor(Math.random() * 256);
    e.target.style.backgroundColor = "rgb(" + num1 + "," + num2 + "," + num3 + ", 0.3)";
  }
};

```

```

const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};

```

**이 코드는 input태그 안에서 키보드를 누를 시
발동되는 함수입니다.
if문 내의 코드는 누른 키가 엔터일 경우 실행됩니다.**

```

box.forEach((target) => target.addEventListener("click", onBoxClick));

```



```
const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};
```

parentNode는 현재 html태그의 부모 태그를 가리키는 코드입니다.

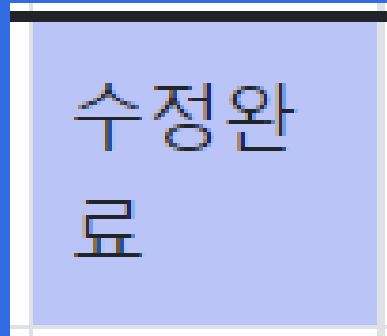
```
▼ <td class="box-edit"> == $0
  <input class="input-text" type="text" onkeydown="Enter(this)" value>
</td>
```

수정 중일 때의 html구조를 보면
td태그 내에 자식 태그로 input 태그가 있습니다.

input태그 내에서 엔터키를 눌렀으니 e에는 input태그가 들어있을 것이고,
e.parentNode는 td태그를 가리키게 됩니다.

```
const Enter = (e) => {
  if (event.keyCode == 13) {
    e.parentNode.className = "box";
    e.parentNode.innerHTML = e.value;
  }
};
```

따라서, 박스(td태그)의 class는 "box"가 되고,
내부 HTML이 e.value로 바뀌었으니
input태그는 사라지고 input태그에 있던 값이 그 자리를 대신할 것입니다.



나의 시간표

시간	일	월	화	수	목	금	토
9:00-10:00	기상	기상	기상	기상	기상	기상	
10:00-11:00							
11:00-12:00							
12:00-13:00	점심			점심			
13:00-14:00							
14:00-							
15:00-16:00							
16:00-							

이렇게 시간표를 완성해보았습니다.

추가로 궁금하신 부분이 있으시면 "구글링" 혹은 "운영진에게 질문" 해주세요 !