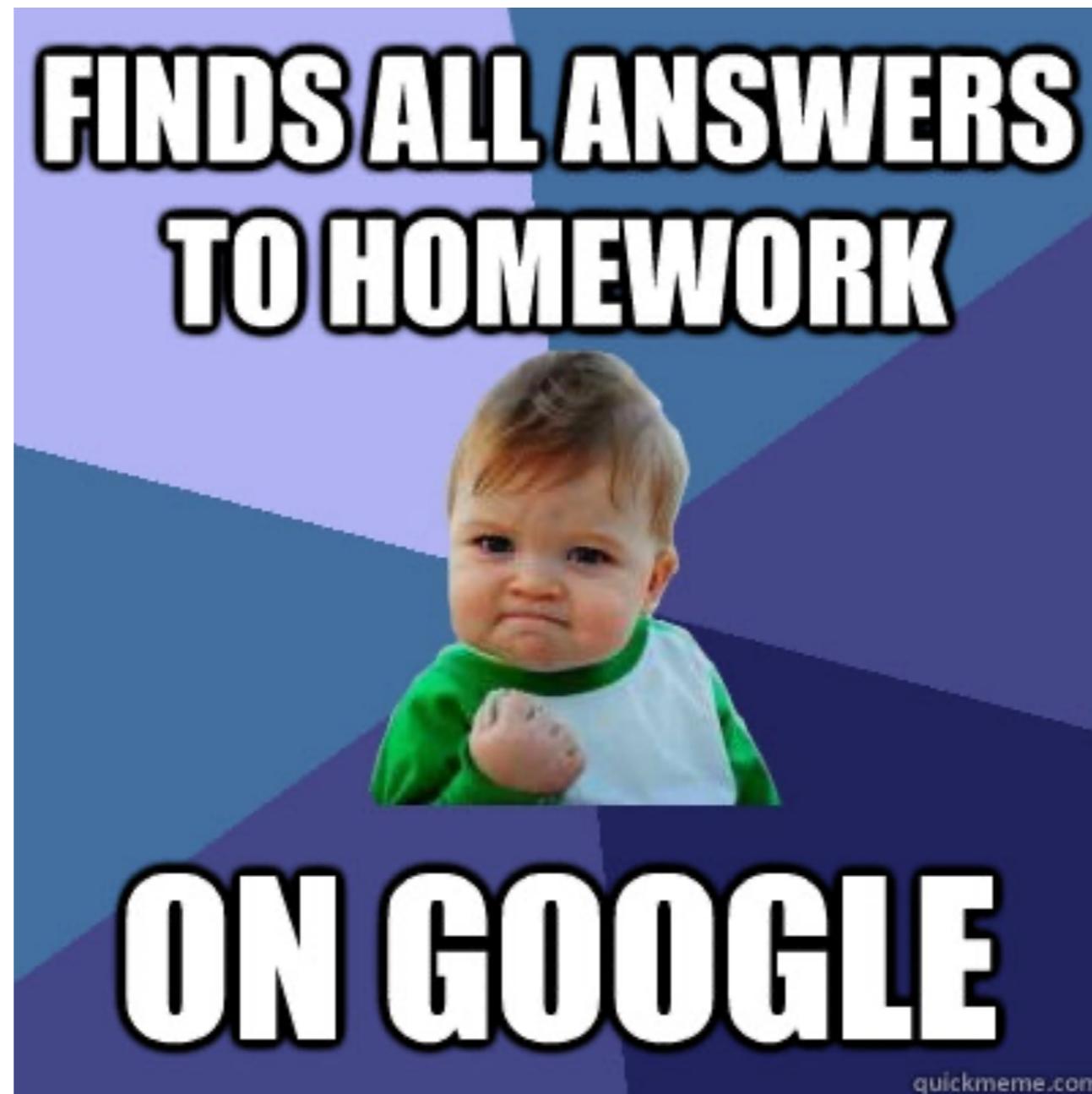


SESSION

몇 번째인지 모름

Javascript: DOM control

숙제 검사 !



과제해설1

```
// solution1

const getRandomHex = () => {
    // 16진수를 구성하기 위한 문자열
    const hexa = '1234567890abcdef'
    let randomColor = '#'
    // hexCode 6자리를 위한 반복문
    for (let i = 0; i < 6; i++) {
        // hex 문자열에서 무작위 index 선택한 뒤 반환하는 문자열에 추가
        randomColor += hexa[Math.floor(Math.random() * 16)]
    }
    return randomColor
}

// 0.1 초에 한번씩
setInterval(() => {
    // html Body DOM의 배경색을 바꿈
    document.querySelector('body').style.backgroundColor = getRandomColor()
}, 100)
```

과제해설1

```
// solution2
const getRandomRGB = () => {
  const getRandomNumber = () => Math.floor(Math.random() * 256);

  return `rgb(${getRandomNumber()}, ${getRandomNumber()}, ${getRandomNumber()})`
}

setInterval(()=>{
  document.querySelector('body').style.backgroundColor = getRandomRGB()
})
```

오답노트

```
// error 1
const getRandomColor = () => {
    var ran1 = Math.floor(Math.random() * 226);
    var ran2 = Math.floor(Math.random() * 226);
    var ran3 = Math.floor(Math.random() * 226);
    const colorR = ran1.toString(16);
    const colorG = ran2.toString(16);
    const colorB = ran3.toString(16);
    const colorName = `#${colorR}${colorG}${colorB}`;

    return colorName;
};

// error 2
const getRandomColor = () => {
    const hexa = '#' + Math.round(Math.random() * 0xFFFFFF).toString(16);

    return hexa
};
```

오답노트

```
const getRandomColor = () => {
    var ran1 = Math.floor(Math.random() * 226);
    var ran2 = Math.floor(Math.random() * 226);
    var ran3 = Math.floor(Math.random() * 226);
    // 1. 다음부터는 var 쓰시면 진짜 수업하다 울면서 나갑니다. 수업 들어주세요...
    | // var -> const or let
    // 2. 두자리 16진수를 얻기 위해서는 255까지의 정수를 변환해야 합니다.
    | // 226 -> 256 or 0x100
    const colorR = ran1.toString(16);
    const colorG = ran2.toString(16);
    const colorB = ran3.toString(16);
    // 3. 문제는 ran0/ 16보다 작은 경우일 때입니다.
    // ran1 => 6, ran1.toString(16) => "6"
    // ran2 => 15, ran2.toString(16) => "f"
    // ran3 => 32, ran3.toString(16) => "20"
    // 이렇게 hex code를 만들면 #6f20 : 색깔 안나옵니다.
    const colorName = `#${colorR}${colorG}${colorB}`;
    return colorName;
};
```

오답노트

```
// error 2
const getRandomColor = () => {
  const hexa = '#' + Math.round(Math.random() * 0xFFFFFF).toString(16);
  // 이 코드도 동일한 문제를 가지고 있네요.
  // 0x100000 미만 숫자들은 자릿수가 맞지 않아요 + 곱하는 수도 0x1000000 7자리로 잡아야합니다.
  // 그렇지만 씽크빅입니다! 짤어! 짤어! ❶ 너무 좋아용~~
  return hexa
};
```

오답노트

```
const getRandomColor = () => {
  const generateRandomNum = () => Math.floor(Math.random() * 256);
  const getTwoDigitHex = (number) => number < 16 ? "0" + number.toString(16) : number.toString(16)

  return `#${getTwoDigitHex(generateRandomNum())}${getTwoDigitHex(generateRandomNum())}${getTwoDigitHex(generateRandomNum())}`
};


```

과제해설2

```
const clockContent = document.querySelector('.now')

const getTime = () => {
    const date = new Date()
    // 시간을 계산해서 clockContent에 표시하는 로직
    const hours = date.getHours()
    const minutes = date.getMinutes()
    const seconds = date.getSeconds()
    clockContent.innerText = `${hours < 10 ? `0${hours}` : hours}시 ${minutes < 10 ? `0${minutes}` : minutes}분 ${seconds < 10 ? `0${seconds}` : seconds}초`
}

const initTime = () => {
    getTime()
    setInterval(getTime, 1000)
}

initTime()
```

오늘의 강의!

Javascript

Event &

DOM Control



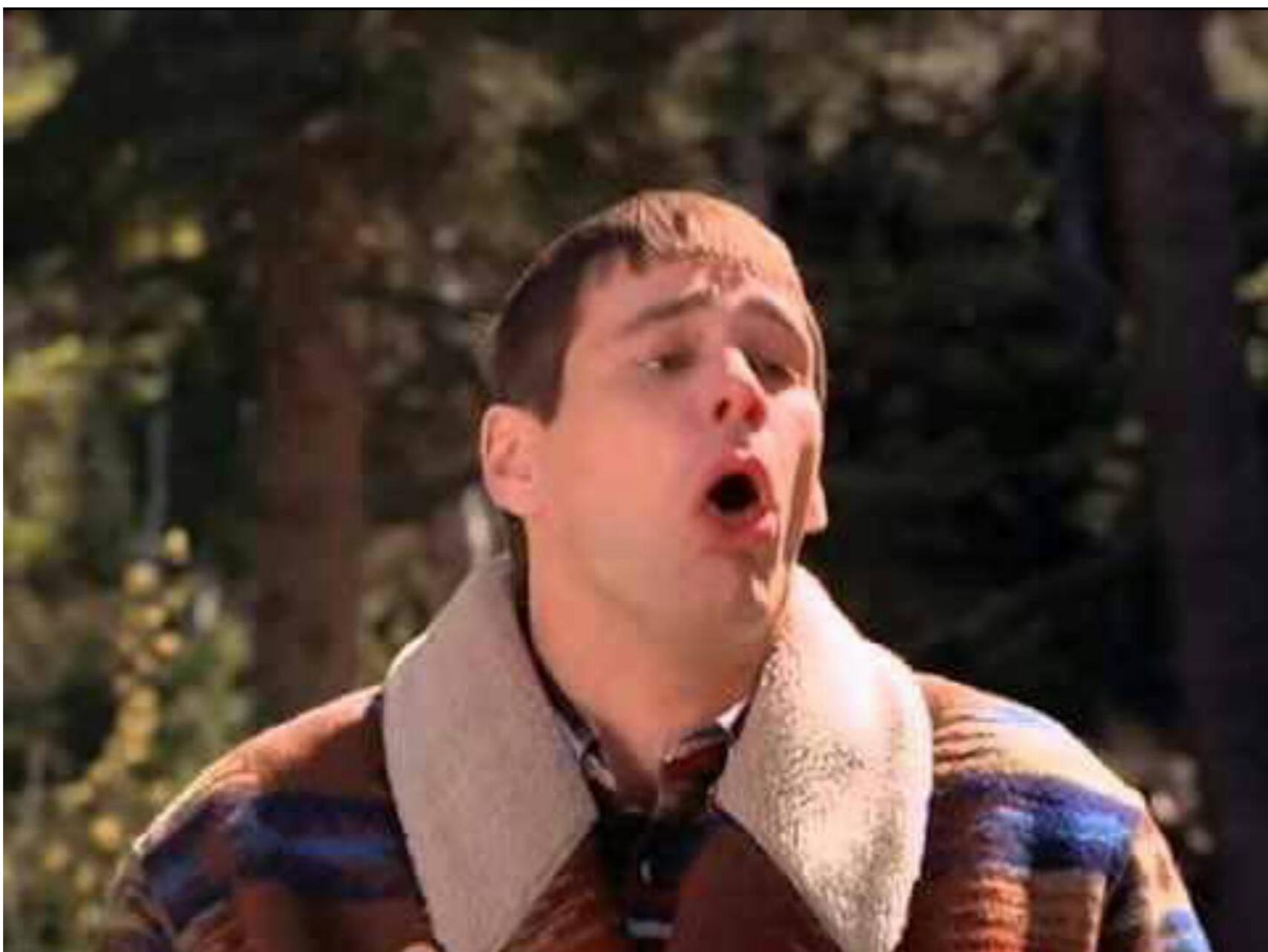
DOM

DOM이란?

DOM

Document Object Model

문서 객체 모델(The Document Object Model, 이하 DOM)은 HTML, XML 문서의 프로그래밍 interface이다. DOM은 문서의 구조화된 표현(structured representation)을 제공하며 프로그래밍 언어가 DOM 구조에 접근할 수 있는 방법을 제공하여 그들이 문서 구조, 스타일, 내용 등을 변경할 수 있게 돋는다. DOM은 구조화된 nodes와 property와 method를 갖고 있는 objects로 문서를 표현한다.



DOM이란?

DOM

Document Object Model

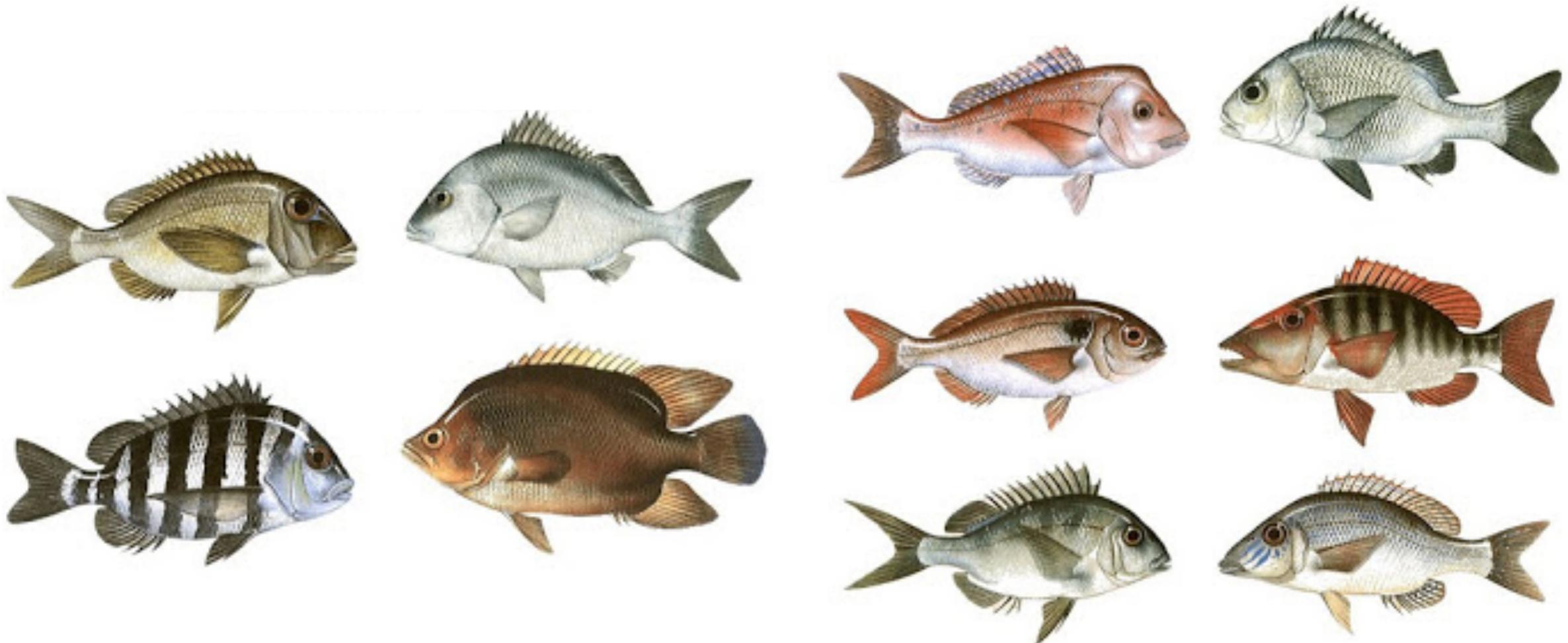
문서 객체 모델(The Document Object Model, 이하 DOM)은 HTML, XML 문서의 프로그래밍 **interface**이다. DOM은 문서의 **구조화된 표현**(structured representation)을 제공하며 **프로그래밍 언어가 DOM 구조에 접근할 수 있는 방법을 제공**하여 그들이 문서 구조, 스타일, 내용 등을 변경할 수 있게 돕는다. DOM은 구조화된 nodes와 property와 method를 갖고 있는 **objects로 문서를 표현한다.**

그래서 요약하면?

HTML 문서 (Document)를 모델링(Model)하고 있는 객체의 집합(Object)이다.
브라우저가 만든 DOM은 node, property, method로 구성되어 있고 이를 통해
우리는 문서를 컨트롤 할 수 있다 => API! DOM API! 와!

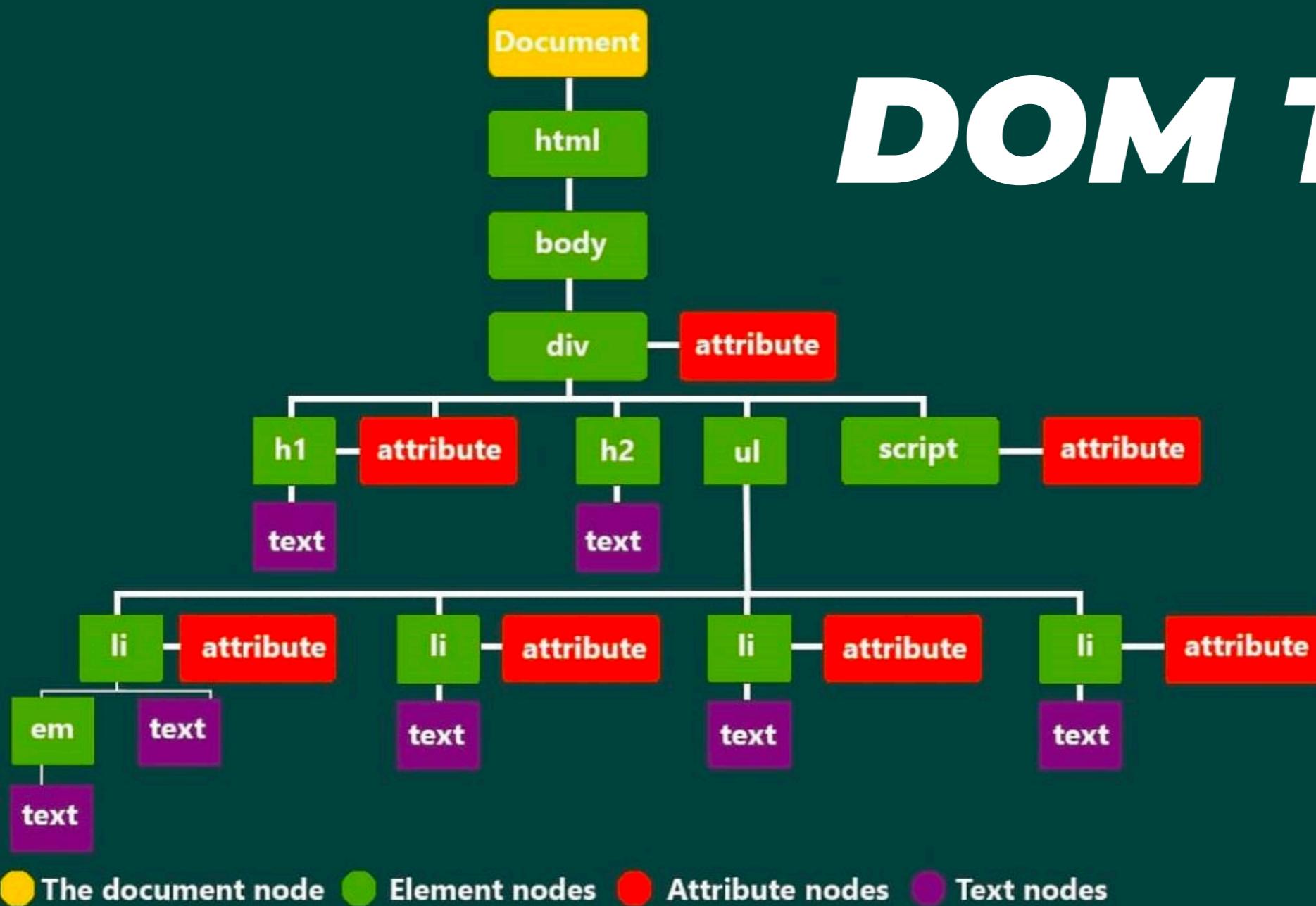
잘 모르겠죠?

다음 중 감성돔을 고르시오 (4점)



괜찮아요 천천히 눈으로 보면서, 쓰면서 공부해봅시다

DOM Tree



까서 눈으로 봅시다.

Index.html

5 index.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <div class="parent">
10         <p id="son" class="child"> 자식입니다.</p>
11         <p id="daughter" class="child"> 자식입니다.</p>
12     </div>
13     <script src="./dom.js"></script>
14  </body>
15  </html>
```

부모 요소만 잡아줍시다.

dom.js

```
const parent =             넌 누구니?  
_____  
document.querySelector('.parent')
```

어디서 왔니?

이런 거 어떻게 알고 찾아 쓰나요?

console.log / console.dir

DOM Tree 까기

Console.**log**(document)

Console.**dir**(document)

오,,,



F12의 지옥

응 아니야



우욱..

```
▶ body: body
  characterSet: "UTF-8"
  charset: "UTF-8"
  childElementCount: 1
▶ childNodes: NodeList(2) [html, html]
▶ children: HTMLCollection [html]
  compatMode: "CSS1Compat"
  contentType: "text/html"
  cookie: "csrftoken=rKgb1bMt9ojKIBdssSFynSGx7giylWZFfj28dASGUGwWnpPDHBxUxIkGg1pVY87A"
  currentScript: null
▶ defaultView: Window {parent: Window, opener: null, top: Window, length: 0, frames: Window, ...}
  designMode: "off"
  dir: ""
▶ doctype: html
▶ documentElement: html
  documentURI: "http://127.0.0.1:5500/index.html"
  domain: "127.0.0.1"
▶ embeds: HTMLCollection []
▶ featurePolicy: FeaturePolicy {}
  fgColor: ""
▶ firstChild: html
▶ firstElementChild: html
▶ fonts: FontFaceSet {onloading: null, onloadingdone: null, onloadingerror: null, ready: Promise, status: "loaded", ...}
▶ forms: HTMLCollection []
  fullscreen: false
  fullscreenElement: null
  fullscreenEnabled: true
  ...
```



본인 나루토 안봄

모든 것은 MDN에



MDN 검색

로그인

기술 ▾

참조문서 및 가이드 ▾

사용자 의견 ▾

문서 객체 모델(DOM)

개발자를 위한 웹 기술 > Web API > 문서 객체 모델(DOM)

한국어 ▾

On this Page

[DOM 인터페이스](#)

[HTML DOM](#)

[SVG 인터페이스](#)

[명세](#)

[같이 보기](#)

문서 객체 모델(DOM)은 메모리의 웹 페이지 문서 구조를 표현함으로써 스크립트 및 프로그래밍 언어와 페이지를 연결합니다. 이때 스크립트는 주로 JavaScript를 의미하나 HTML, SVG, XML 객체를 문서로 모델링하는 것은 JavaScript 언어의 일부가 아닙니다.

DOM은 문서를 논리 트리로 표현합니다. 트리의 각 브랜치는 노드에서 끝나며, 각 노드는 객체를 갖습니다. DOM 메서드를 사용하면 프로그래밍적으로 트리에 접근할 수 있습니다. 이를 통해 문서의 구조, 스타일, 콘텐츠를 변경할 수 있습니다.

노드는 이벤트 처리기도 포함할 수 있습니다. 이벤트가 발생한 순간, 해당 이벤트와 연결한 처리기가 발동합니다.

▶ 더 알아보려면: [DOM 소개](#) 문서를 방문해 보세요.

관련 주제

[Document Object Model](#)

▼ Guides

[DOM 소개](#)

■ CSS 파일 만들고 연결하자! 쉽죠?

```
. red{  
    color: red;  
}
```

CSS 파일 하나 만들어서 연결해주고,
간단하게 텍스트 색깔만 지정한
클래스를 하나 만들어줍시다

querySelector

```
const parent = document.querySelector('.parent');
const son = document.querySelector('#son');
const daughter = parent.querySelector('#daughter');

console.log(parent, son, daughter)
```

자식을 잡아줍니다.



classList

```
const parent = document.querySelector('.parent');
const son = document.querySelector('#son');
const daughter = parent.querySelector('#daughter');

son.classList.add('red');
// son.classList.remove('red');
// son.classList.toggle('red');
```

classList.add('클래스명')을 통해 클래스를 추가해 줄 수 있습니다. 당연히 삭제도 되겠죠? MDN에서 확인합니다.

classList : 미니 실습

나머지 자식도 바꿔볼까요?

마크업 짤 때 하는게 편합니다.

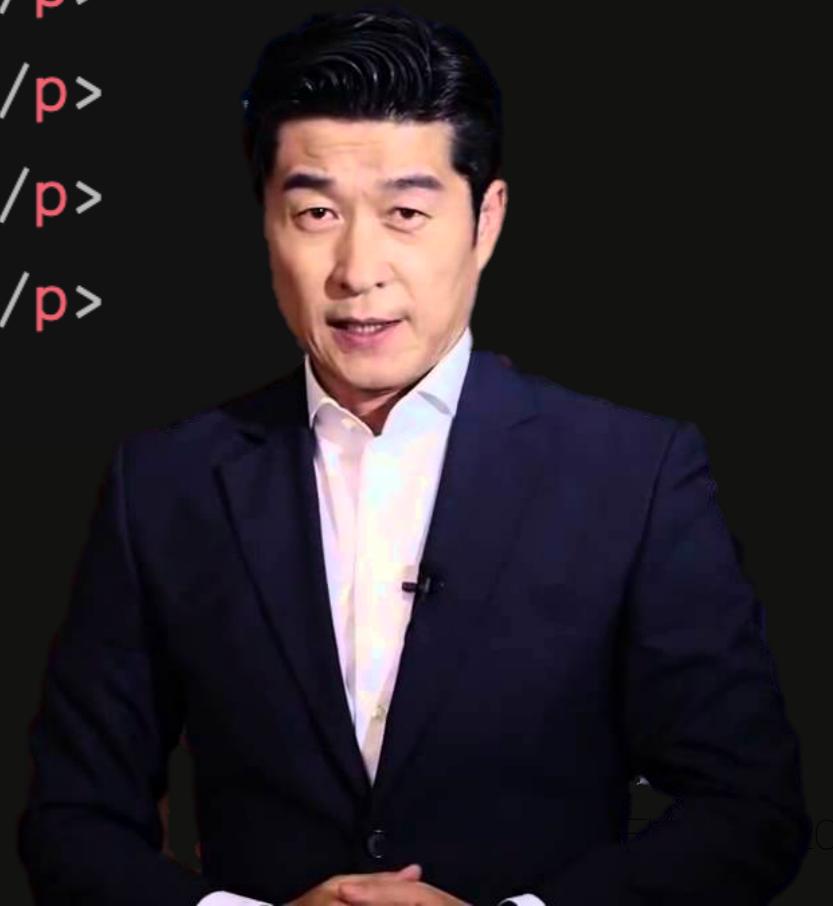
```
const parent = document.querySelector('.parent');
const son = document.querySelector('#son');
const daughter = parent.querySelector('#daughter');

son.classList.add('red');
daughter.classList.add('red');
```

어느 세월에..
그냥 html에다 클래스 추가 합시다.

그런데 말입니다.

```
<div class="parent">  
  <p id="son" class="child"> 자식입니다.</p>  
  <p id="daughter" class="child"> 자식입니다.</p>  
  <p id='dog1' class="child">저도 자식입니다.</p>  
  <p id='dog2' class="child">저도 자식입니다.</p>  
  <p id='dog3' class="child">저도 자식입니다.</p>  
  <p id='cat1' class="child">저도 자식입니다.</p>  
  <p id='cat2' class="child">저도 자식입니다.</p>  
  <p id='cat3' class="child">저도 자식입니다.</p>  
  <p id='cat4' class="child">저도 자식입니다.</p>  
</div>
```



querySelectorAll()

```
const children = parent.querySelectorAll('.child');
console.log(children);
```

nodeList

```
 NodeList(9) [p#son.child, p#daughter.child, p#dog1.child,  
▼ p#dog2.child, p#dog3.child, p#cat1.child, p#cat2.child,  
p#cat3.child, p#cat4.child] i  
▶ 0: p#son.child  
▶ 1: p#daughter.child  
▶ 2: p#dog1.child  
▶ 3: p#dog2.child  
▶ 4: p#dog3.child  
▶ 5: p#cat1.child  
▶ 6: p#cat2.child  
▶ 7: p#cat3.child  
▶ 8: p#cat4.child  
length: 9  
▶ __proto__: NodeList
```

NodeList? 아무튼
배열 비슷한 게 나옵니다

실습1.



저희집 개자슥덜 입니다.
커엽죠...?

나머지 자식들의
글자색도 바꿔주세요.

실습1. 해설 (ver.1)

```
const children = parent.querySelectorAll('.child');
// children 배열 순회하면서 자식요소 하나하나 클래스 추가!
for (let i = 0; i < children.length ; i++){
  children[i].classList.add('red')
}
```

실습1. 해설 (ver.2)

```
//solution1
children.forEach(child => {
  child.classList.add('red')
});

//solution2
children.forEach(function(child){
  child.classList.add('red');
})

//solution3
const setClass = (child) => {
  child.classList.add('red')
}

children.forEach(setClass)
```

실습1. 해설 (ver.2)

```
arr.forEach(callback(currentValue[, index[, array]])[, thisArg])
```

매개 변수

callback

각 요소에 대해 실행할 함수. 다음 세 가지 매개변수를 받습니다.

currentValue

처리할 현재 요소.

index | Optional

처리할 현재 요소의 인덱스.

array | Optional

forEach()를 호출한 배열.

thisArg | Optional

callback을 실행할 때 this로 사용할 값.

실습1. 해설 (ver.2 심화)

Array.prototype.forEach()

: Array에서만 쓸 수 있는 문법이다!

```
> children
< NodeList(9) [p#son.child, p#daughter.child, p#dog1.child,
  ▶ p#dog2.child, p#dog3.child, p#cat1.child, p#cat2.child,
  p#cat3.child, p#cat4.child]
> Array.isArray(children)
< false
```

1. 유사배열이란?
2. NodeList란?

실습1. 해설 (ver.3)

```
parent.classList.add('red');
```

스크립트

textContent/ innerText/ innerHTML

```
<p id="son" class="child">저는 자식입니다.</p>
```

```
console.log(son.textContent)
```

```
console.log(son.innerText)
```

```
console.log(son.innerHTML)
```

textContent/ innerText/ innerHTML

```
<p id="son" class="child">저는<b style="display:none;">자식입니다.</b></p>
```

textContent

가급적 `textContent`를 사용하는 것이 좋습니다. 성능과 보안에 강점이 있고, 결과적으로 해당 노드의 `raw text`를 얻게 됨으로써 이후 의도한 대로 가공할 수 있기 때문입니다.

innerText

특정 노드에 렌더링 된(화면에 보이는 그대로의) 텍스트를 읽어올 때 유용합니다. 하지만 내부에 특별히 스타일 적용이 없는 문자열을 할당할 때는 성능상 적합하지 않습니다. 단, IE(IE8 이하) 환경을 중점으로 고려한 프로젝트라면 `textContent` 대신 사용하도록 합니다. 😊

innerHTML

HTML Parsing이 필요한 문자열에만 사용하도록 합니다. 그게 아니라면, 성능상 좋지 않고 XSS 공격에도 취약하므로 `innerHTML`은 사용하지 않는 것이 좋습니다.

textContent/ innerText/ innerHTML

내용을 바꿔보자!

```
son.textContent = '저는 아들입니다.'
```

```
daughter.textContent = '저는 딸입니다.'
```

```
son.innerHTML = '<b>저는</b> 아들입니다.'
```

createElement() / appendChild()



자식이 또 생겼네요

createElement() / appendChild()

```
const father = document.createElement('p');
father.textContent = '저도 자식입니다.'  
  
parent.appendChild(father)
// parent.insertBefore(father, son)
```

setAttribute

애비다



console.log(father)

setAttribute

```
father.setAttribute('id','father');
father.setAttribute('class','parent');

console.log(father)
```

실습2. 속성 변경

attribute.html > ...

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4  |  <meta charset="UTF-8">
5  |  <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  |  <title>Document</title>
7  </head>
8  <body>
9  |  <a href="https://www.naver.com">네이버로 가기</a>
10 |  
11
12 <script>
13
14 </script>
15 </body>
16 </html>
```

- attribute.html을 새로 만들고
아래와 같이 네이버 링크와 이미지를 첨부해주세요
1. 스크립트 태그 내에서 링크를 ‘구글’로
 2. 사진은 다른 사진으로 변경해주세요

실습2.

```
<script>

  const link = document.querySelector('a');
  const image = document.querySelector('img');

  const originalLink = link.getAttribute('href');
  const originalImageSrc = link.getAttribute('src');

  link.setAttribute('href', 'https://google.com');
  link.textContent = '구글로 가기';
  image.setAttribute('src', 'https://picsum.photos/300/300');

</script>
```

addEventListener



addEventListener

마우스 액션을 잡아봅시다

대충 새로운 html과
Js 파일 혹은 inner script 만들어주세요

addEventListener: callback()

```
const title = document.querySelector('.title')

title.addEventListener('mouseover', function(event){
  console.log(event)
})

title.addEventListener('mouseout', function(event){
  consol.log(event)
})
```

EventTarget.addEventListener ('action', callback(event))

*callback parameter **event** ?

- 크롬이나 IE는 이벤트 객체 전달 안해도 동작함, 파이어폭스는 전달해야함.
그러니까 그냥 쓰자

Event.target

```
const title = document.querySelector('.title')
const GOLD = '#f1c40f'
const GRAY = '#34495e'

const onMouseOver = (event) => event.target.style.color = GOLD
title.addEventListener('mouseover', onMouseOver)
const onMouseOut = (event) => event.target.style.color = GRAY
title.addEventListener('mouseout', onMouseOut)
```

addEventListener

```
.title{  
    color: #34495e;  
}  
.title:hover{  
    color : #f1c40f;  
}
```

CSS로 합니다.

addEventListener

마우스 클릭을 잡아봅시다

<h1>자바스크립트 참쉽죠?</h1>

<button>진짜?</button>

마우스 클릭을 잡아봅시다.

축하합니다! 운영진에 당첨되었습니다

진짜?

```
const title = document.querySelector('h1')
const button = document.querySelector('button')

button.addEventListener('click', function(){
  title.textContent = '축하합니다! 운영진에 당첨되었습니다'
})
```

실습3. 누른 버튼 찾기

```
<h1>버튼을 바꿔보자</h1>  
<button>눌러주세요</button>  
<button>눌러주세요</button>  
<button>눌러주세요</button>
```

1. 위와 같은 HTML을 만들어주세요.

2. 버튼을 누르면 해당 버튼의 텍스트가 ‘짜잔’으로 바뀌게 해주세요

실습3.해설

```
buttons[0].addEventListener('click',function(event){  
    event.target.textContent = "짜잔"  
})
```

```
buttons[1].addEventListener('click',function(event){  
    event.target.textContent = "짜잔"  
})
```

```
buttons[2].addEventListener('click',function(event){  
    event.target.textContent = "짜잔"  
})
```

당신은 부지런한 사람입니다.

실습3.해설

```
for(let i = 0; i < buttons.length; i++){
  buttons[i].addEventListener('click',function(event){
    event.target.textContent = "짜잔"
  })
}
```

훌륭합니다

실습3. 해설

```
buttons.forEach(button => {
  button.addEventListener('click', function(event){
    event.target.textContent = '짜잔'
  })
})
```

크 ...

실습3.해설_심화

```
function onClick(){  
  ...  
  this.textContent = '짜잔'  
}  
  
buttons.forEach(button => {  
  button.addEventListener('click',onClick)  
})
```

this

키보드 이벤트를 잡아봅시다.

```
<h1>아래가 바뀝니다</h1>
<h2 id="result">_____</h2>
<input type="text" id="input"/>

const result = document.querySelector('#result')
const input = document.querySelector('#input')

input.addEventListener('keydown', function(event){
  console.log(event)
})
```

keyboardEvent

```
KeyboardEvent {isTrusted: true, key: "Enter", code: "Enter", location: 0, ctrlKey: false, ...} ⓘ
  altKey: false
  bubbles: true
  cancelBubble: false
  cancelable: true
  charCode: 0
  code: "Enter"
  composed: true
  ctrlKey: false
  currentTarget: null
  defaultPrevented: false
  detail: 0
  eventPhase: 0
  isComposing: false
  isTrusted: true
  key: "Enter"
  keyCode: 13
  location: 0
  metaKey: false
  ▶ path: (7) [input#input, div.header, div.container, body, html, ...]
  repeat: false
  returnValue: true
  shiftKey: false
  ▶ sourceCapabilities: InputDeviceCapabilities {firesTouchEvent: false}
  ▶ srcElement: input#input
  ▶ target: input#input
  timeStamp: 158069.35000000522
  type: "keydown"
  ▶ view: Window {parent: Window, opener: null, top: Window, length: ...}
  which: 13
  ▶ __proto__: KeyboardEvent
```

keyboardEvent

keydown	키보드를 누르는 순간 실행	키를 누르고 있을 시, 한 번 실행
keyup	키보드가 눌러진 상태에서 손을 떼는 순간 실행	
keypress	키보드를 누르는 순간 실행	키를 누르고 있을 시, 지속 실행 보조키(Ctrl, Alt, Shift) 등 보조키 동작 안함 대소문자 값을 다르게 가짐

Keypress는 한글이랑 안 맞아요ㅠ

엔터치면 입력!

```
function onKeyDown(event){  
    let inputValue = ''  
  
    inputValue = event.target.value;  
  
    if (event.key === 'Enter') {  
        result.textContent = inputValue;  
    }  
}  
  
input.addEventListener('keydown',onKeyDown)
```

실습4. 무엇을 눌렀나요

Enter / Alt / Control / Shift 키를 눌렀을 때
‘_____ 키를 눌렀네요’ 표시하기

실습4. 해설

```
const ENTER_KEY = 'Enter'
const ALT_KEY = 'Alt'
const CONTROL_KEY = 'Control'
const SHIFT_KEY = 'Shift'

function onKeyDown(event){
    let message = '키를 눌렀네요'

    switch(event.key){
        case ENTER_KEY:
            result.textContent = `${ENTER_KEY}${message}`
            break
        case ALT_KEY:
            result.textContent = `${ALT_KEY}${message}`
            break
        case CONTROL_KEY:
            result.textContent = `${CONTROL_KEY}${message}`
            break
        case SHIFT_KEY:
            result.textContent = `${SHIFT_KEY}${message}`
            break
    }
}

input.addEventListener('keydown',onKeyDown)
```

과제. 글자수 세기

글자 수 세기

안녕하세요 저는 상하입니다.

공백 포함 글자 수 : 15

공백 제외 글자 수 : 13