



생활코딩 HTML 강좌

LEE GYURLN



#6. 태그

`` 굵게 ``

`<u>` 밝출 `</u>`

#7. 제목

`<h1>` heading ; 제목 ; 가장 큰 `</h1>`

↳ `h1 > h2 > ... > h6`

#9. 줄바꿈

`
` : 줄바꿈 태그 ; 닫지 않는다 (`
`)

`<p> ~ </p>` : 단락 태그 . 정보로써 `
` 보다 더 좋다

↳ `<p style="margin-top: 45px;"> ~ </p>` 처럼 CSS 적용 가능

#11. 속성

``

속성 attribute 순서 상관 X

#12. 목차

`` unordered list ⇒ ~

` ~ ` ↗ 자식
태그
` ~ `

`` ↗ 부모 태그

`` ordered list ⇒ 1. ~, 2. ~, ...

` ~ `
` ~ `

``

#13. 문서구조

`<!doctype html>` : 관용적으로 써준다

`<html>` : `<head>` 태그와 `<body>` 태그의 상위 태그

`<head>` : 그 문서에 대한 정보

`<title>` 제목 `</title>` : 웹페이지의 제목

`<meta charset="utf-8">` ⇒ 한글 프린트 가능

`</head>`

`<body> ~ </body>`

`</html>`

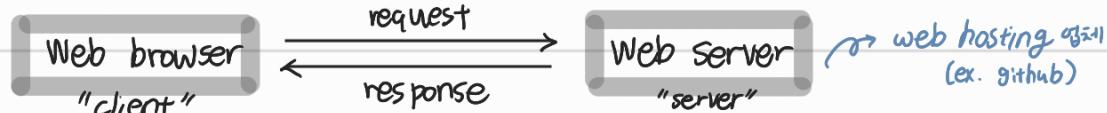
#14. 링크

` ~ `

↑ hyperlink reference
anchor

↑ 클릭 전 커서 제목

#17. 서버



#cf. 주석

`<!-- ~-->`

`/* ~ */`

#Entity name

`<` ⇒ < , `>` ⇒ > , ` ` ⇒ (공백) , `&` ⇒ & , `"` ⇒ "



생활코딩 CSS 강좌

LEEGYURIN



#3. Style 태그

<head>

<style>

declaration
선언자

a { color : red; }

</style> selector 선택자

모든 a 태그에 대해

</head>

→ 중복의 제거 가능,
유지·보수의 편의성↑

<h1> WEB </h1>

 HTML

#4. Style 속성

 HTML

↳ HTML에서 style이라는 속성은 CSS 문법에 따라 해석.

c.f. 선언자) text-decoration : none ; 글씨 효과 없음

text-decoration : underline ; 밑줄 효과

font-size : 46px ;

text-align : center ; 가운데 정렬

#7. 클래스

<head>

<style>

a { color: black; }

.saw { color: gray; }

</style>

</head>

<body>

 H
 C
 J

</body>

* head의 style 태그 안에서, 선택자 앞에 . 붙이면 class 의미, # 붙이면 id class 의미.

ex) .saw ex) #active

↳ id는 중복 X. 딱 한번만 나옴.

* body의 어떤 태그의 속성으로 class 속성 지정 가능.

↳ 여러 class 동시 지정 가능, 띄어쓰기로 구분. 보다 가까이 있는 것의 속성 따름.

★ 우선순위: # — > . — > — Ⓛ 가까운 것

ex) class = "saw" ex) class = "saw active" ex) class = "saw" id = "active"

* head의 style 태그 안에서, #grid ol li ~ } 처럼 써주면 id가 grid인 부모태그 안의 자식태그로 있는 <a> 만 그 속성 적용.

#8. 박스

* <n1> 태그는 화면 전체를 씁. vs. <car> 태그는 자기 자신의 content 만큼의 부피만을 차지함.

⇒ block level element ex) <div>

⇒ inline element

ex) <i> 등

↳ <head> 안 <style> 안 각 태그 selector에 대해 h1 { border-width: 5px;

border-color: red; border-style: solid; } 처럼 꼬여줘보면 알 수 있음.

↳ <head> 안 <style> 안 태그 selector에 대해 h1 { display: inline; },

a { display: block; } 처럼 해주면 기본(default) 걸 바꿀 수 있음.

c.f.) a { display: none; } 처럼 해주면 해당 태그 아예 안보임.

* 중복의 제거 : <head> 안 <style> 안에서, h1, a { border: 5px solid red; } 처럼 표현 가능

* CSS 박스모델 ☆



ex) <head> <style> h1 {

border: 5px solid red;
padding: 20px;
margin: 20px;
display: block;
width: 100px; }

</style> </head>

* 웹페이지 우클릭 → 검사 → 태그 클릭 → style 통해 확인 가능 ⇒ 웹개발 도구

* 박스 모델 응용 : declaration의 property를 border-bottom, padding-left 등으로 응용 가능.

#10. 그리드

<div> ~ </div> ↴ : non- 의미의 태그. <div> 는 block level element임.

 ~ ↴ : only 디자인 위한. 은 inline element임.

* 그리드 쓰기 위해서는 부모 태그가 꼭 필요.

<head> <style>

#grid { border: 5px solid pink;
display: grid; ↪ display: block;
grid-template-columns: 150px 1fr; }

div { border: 5px solid gray; }

</style> </head>

↗ <body>

<div id="grid">

<div> NAVI </div>

<div> GATION </div>

</div>

</body>

* fr은 비율 나타냄. 2fr 1fr; 로 써주면 2:1 비율로 자동 분배.

* column: 열, row: 행.

#12. 반응형

* 미디어 쿼리

<head> <style>

↗ 조건문

@media (max-width: 800px) { div { display: none; } }

</style> </head>

#14. 재사용

* CSS 코드 재사용 막기 : CSS 코드만 저장해놓은 style.css 파일 만들기 + 기존 html 파일 안에 불러오기

<head>

<link rel="stylesheet" href="style.css" >

</head>



생활코딩 JavaScript 강좌



#3. script 태그

```
<body>
```

(script) 태그 : JavaScript의 문법을 따름을 명시

```
<script> document.write(1+1); </script> </body>
```

if
1+1

2로 출력됨.

#4. 이벤트

```
<input type="button" value="hi" onclick="alert('hi')"> ⇒ [hi]
```

```
<input type="text" onchange="alert('changed')"> ⇒ [ ]
```

```
<input type="text" onkeydown="alert('key down!')"> ⇒ [ ]
```

↳ 이벤트.

#5. 콘솔

* 웹페이지에서 우클릭 → 검사 → 콘솔 (Console) → JS 실행 가능.

↳ esc 버튼 → element 창 console 창 같이 품.

ex) alert('Java' + 'n').length ⇒ 그 문자열의 길이 알려줌.

#6. String

String의 Method 몇 예 : 'Hello world'.toUpperCase ⇒ "HELLO WORLD"

'Hello world'.indexOf('o') ⇒ 4

'hello'.trim() ⇒ "hello"

#7. 변수

콘솔 창에서 var name = 'gyurin'; 과 같이 선언 가능.

#12. 태그선택

```
<input type="button" value="night" onclick = "document.querySelector('body').style.backgroundColor = 'black';  
document.querySelector('body').style.color = 'white';">
```

#15. Boolean

JS에서의 비교연산자: ===, ==, !=, !=, >, >=, <, <=

일치 등등

ex) 1 === '1'은 false, 1 == '1'은 true

#17. 조건문

```
<input id="night-day" type="button" value="night" onclick = "if (document.querySelector('#night-day').value === 'night') {  
    document.querySelector('body').style.backgroundColor = 'black';  
    document.querySelector('body').style.color = 'white';  
    document.querySelector('#night-day').value = 'day'; }  
else { document.querySelector('body').style.backgroundColor = 'white';  
    document.querySelector('body').style.color = 'black';  
    document.querySelector('#night-day').value = 'night'; } }>
```

* 위 코드를 한 파일에 두 번 쓸 경우, 두 번째 것은 id를 night-day 2 등과 같이 다르게

바꿔서 해야 함. or 리팩토링 (this)

#18. 리팩토링

* 조건문에서 바뀌는 게 자기자신일 경우, id로 지정 안하고 this로 간결히 표현 가능.

* 너무 긴 반복되는 구문은 variable로 지정해서 대체 가능.

```
<input type="button" value="night" onclick="
    var target = document.querySelector('body');
    if (this.value === 'night') {
        target.style.backgroundColor = 'black';
        target.style.color = 'white';
        this.value = 'day';
    } else {
        target.style.backgroundColor = 'white';
        this.value = 'night';
    }
">>
```

#20. 배열

```
var fruits = ["apple", "kiwi"];      ⇒ 배열 선언, 구성 요소들은 element(원소)라 함.
document.write(fruits[0]);
document.write(fruits[1]);           | ⇒ get
fruits.push('banana');            ⇒ add (array의 다양한 Method)
document.write(fruits.length);     ⇒ count
```

#21. 반복문

* JS의 Loop문 종류: for문, do...while문, while문, 레이블, break, continue, for...in, for...of

ex) var i=0; while(i<3){ document.write('2'); i+=1; }

#23. ↳ 활용

```
<input type="button" value="night" onclick="
    var target = document.querySelector('body');
    if (this.value === 'night') {
        var alist = document.querySelectorAll('a');
        var i = 0;
        while (i < alist.length) {
            alist[i].style.color = 'powderblue';
            i += 1;
        }
    } else {
        ...
    }
">>
```

#28. 함수

<head> <script>

```
function nightDayHandler(self){          함수선언
    var target = document.querySelector('body');
    if(self.value === 'night'){
        target.style.backgroundColor = 'black';
        target.style.color = 'white';
        self.value = 'day';
    }
    var alist = document.querySelectorAll('a');
    var i = 0;
    while(i < alist.length){
        alist[i].style.color = 'powderblue';
        i = i + 1;
    }
} else {
    target.style.backgroundColor = 'white';
    target.style.color = 'black';
    self.value = 'night';
}
var alist = document.querySelectorAll('a');
var i = 0;
while(i < alist.length){
    alist[i].style.color = 'blue';
    i = i + 1;
}
```

</script> </head>

<body>

```

<h1><a href="index.html">WEB</a></h1>
<input id="night_day" type="button" value="night" onclick="
    nightDayHandler(this);
">
<input id="night_day" type="button" value="night" onclick="
    nightDayHandler(this);
">
<ol>
    <li><a href="1.html">HTML</a></li>
    <li><a href="2.html">CSS</a></li>
    <li><a href="3.html">JavaScript</a></li>
</ol>
<h2>JavaScript</h2>
<p>
    JavaScript (/dʒɑːvə,skript/[6]), often abbreviated as JS, is a high-le
</p>

```

</body>

#30. 객체

쓰기와 읽기

object name
↑
var fruits = { "red": "apple", "yellow": "banana" } ; → 객체 선언

member name / key
↑
↑ 배열에는 index!!

member value
↑
↑ 객체의 property 끝에는 괄호! comma로 구분

member name
↑
↑

member value
↑
↑

document.write ("red fruit is :" + fruits.red + "
"); → 객체의 멤버 value 읽기

fruits.violet = "grape"; → 객체에 멤버 추가

fruits["light green"] = "melon"; → 객체에 띠어쓰기 있는 멤버 name의 멤버 추가

document.write ("light green fruit is :" + fruits["light green"] + "
");

for (var key in fruits) { → 'for ... in ...' loop문으로 멤버 꺼내기 가능.

document.write (key + ':' fruits[key] + '
'); ?

⇒ red: apple ⌈ yellow: banana ⌈ light green : melon 출력.

→ 객체
순서X. VS.
key. 배열
순서O.
index.

#31. 객체와 반복문

#32. property 와 method

반복문

* 객체에 소속된 변수 : property ⌈ 객체에 소속된 함수 : Method

fruits.showAll = function() { → 객체 안에 method 넣기

for (var key in this) {

document.write (key + this[key] + '
'); ??

fruits.showAll(); → 객체의 method 실행

#33. ↳ 활용

```

<!doctype html>
<html>

    <head>
        <script>

            var Links = {
                setColor: function (color){
                    var alist = document.querySelectorAll('a');
                    var i = 0;
                    while(i < alist.length){
                        alist[i].style.color = color;
                        i = i + 1;
                    }
                }
            };
            var Body = {
                setColor: function (color){
                    document.querySelector('body').style.color = color;
                },
                setBackgroundColor: function (color){
                    document.querySelector('body').style.backgroundColor = color;
                }
            };
        </script>
    </head>

```

```

function nightDayHandler(self){
    var target = document.querySelector('body');
    if(self.value === 'night'){
        Body.setBackgroundColor('black');
        Body.setColor('white');
        self.value = 'day';
        Links.setColor('powderblue');
    } else {
        Body.setBackgroundColor('white');
        Body.setColor('black');
        self.value = 'night';
        Links.setColor('blue');
    }
}

```

```

<body>
  <h1><a href="index.html">WEB</a></h1>
  <input id="night_day" type="button" value="night" onclick="
    nightDayHandler(this);
  ">
  <ol>
    <li><a href="1.html">HTML</a></li>
    <li><a href="2.html">CSS</a></li>
    <li><a href="3.html">JavaScript</a></li>
  </ol>
  <h2>JavaScript</h2>
  <p>
    JavaScript (/dʒɑːvə skrɪpt/[6]), often abbreviated as JS, is a high-level
  </p>
</body>
</html>

```

#34. 파일포개기

* ~~.js 파일 생성 후 <head> 안에 추가. <script src="~.js"> </script>

#35. library 와

framework

* library는 필요한 코드, 기능만 딱딱 가져오는 것. ex) jQuery

* framework는 이미 거의 만들어진 코드에 내가 조금 더 개선시키는 것.

* jQuery 활용 예 : CDN (Content Delivery Network) 통해 반복문 사용을 없앨 수 있음.

```

var Color = {
  setColor : function(color1,color2,color3,color4){
    $('body').css('color', color1).css('background',color2);
    $('a').css('color',color3);
    $('ul').css('color',color4);
  }
  function nightDayHandler(self){
    if(self.value === 'night'){
      Color.setColor('white','black','powderblue','powderblue');
      self.value = 'day';
    } else {
      Color.setColor('black','white','blue','blue');
      self.value = 'night';
    }
  }
}

```

#36. UI & API

* UI (User Interface) 는 유저가 특정 조작 장치를 사용해 시스템에 영향하는 것.

* API (Application Programming Interface) 는 프로그래밍 언어가 제공하는 기능 사용하기 힘.