

사용준비

2022년 10월 8일 토요일 오후 6:10

1. 다운 로드 받은 파일 활용
- 코드를 저장하고 제이쿼리 사용 준비

JQuery.min.js란 파일이 있을 때

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jQuery Basic</title>
<script src = "jQuery.min.js"></script>
</script>
// 여기에 코드 입력
</script>
</head>
<body>
</body>
</html>
```

화면 캡처: 2022-10-16 오후 4:06

2. CDN 사용
- 사용자에게 간편하게 콘텐츠를 제공하는 방식

기타 CDN

다음 CDN은 또한 jQuery 릴리스의 압축 및 비압축 버전을 호스팅합니다. jQuery 1.9부터 [소스일 파일](#) 을 호스팅할 수도 있습니다 . 사이트의 문서를 확인하십시오.

jQuery 릴리스와 해당 릴리스 사이에 지연이 있을 수 있습니다. 블로그 게시물이 공개되는 동시에 파일을 받게 됩니다. 베타 및 릴리스 후보는 이러한 CDN에서 호스팅되지 않습니다.

- [구글 CDN](#)
- [마이크로소프트 CDN](#)
- [CDNJS CDN](#)
- [jsDelivr CDN](#)

제이쿼리

3.x 스니펫:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
```

2.x 스니펫:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js"></script>
```

1.x 스니펫:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js">
</script>
```

화면 캡처: 2022-10-16 오후 4:10

위 주소 활용해서 태그 만들어 넣으면 된다

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jQuery Basic</title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.1/jquery.min.js"></script>
</head>
<body>
<body>
</html>
```

화면 캡처: 2022-10-16 오후 4:12

JQuery 객체

2022년 10월 16일 일요일 오후 4:12

- 제이쿼리 라이브러리는 \$ 함수를 활용한다
- 보통 \$ 함수의 매개변수에는 문서 객체, css 형식, HTML형식의 문자열 삽입
- [\$(매개_변수).메소드(매개_변수, 매개_변수)]
- 위와 같은 기본 형태
- [window.jQuery = window.\$ = jQuery;]
- jQuery 이름의 함수를 너무 많이 사용하기 때문에 \$ 식별자로 간단하게 입력할 수 있게 만든 것

```
// 일반 문서 객체로 jQuery 객체를 생성한다
$(document)

// css 선택자로 jQuery 객체를 생성한다
$('h1')

// HTML 문자열로 jQuery 객체를 생성한다
$('<h1></h1>')
```

- Css 선택자를 사용하면 '기존의 문서 객체 선택'
- HTML 문자열을 사용하면 '새로운 문서 객체 생성'
- 문서 객체 사용하면 '기존의 문서 객체 선택' 과 '새로운 문서 객체 생성' 모두 가능

- 이전에는 문서 객체 모델을 다룰 때 window 객체의 load 이벤트를 사용해야 화면이 구성된 이후부터 문서 객체 조작이 가능 했지만
- jQuery에서는 document 객체의 ready 이벤트를 활용한다

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>jQuery Basic</title>
  <script src="https://code.jquery.com/jquery-3.6.1.min.js"></script>
  <script>
    $(document).ready(function () {

    });
  </script>
</head>
<body>
</body>
</html>
```

화면 캡처: 2022-10-16 오후 4:24

시작 코드

문서 객체 선택

2022년 10월 16일 일요일 오후 4:25

- Css 선택자를 사용해 문서 객체를 선택하는 경우
- `$()` 함수를 사용해 다음 형태로 문서 객체를 선택한다

```
// h1 태그를 선택
$('h1')

// h1 태그 중에서 class 속성으로 logo를 가진 태그를 선택
$('h1.logo')

//id 속성이 head인 태그를 선택한다
$('#head')

// input 태그 중에서 type 속성이 submit 인 태그를 선택
$('input[type=submit]')
```

화면 캡처: 2022-10-16 오후 4:29

- 객체 탐색 메소드

메소드	설명
Parent()	부모 태그 선택
Find()	후손 태그 찾기

- 예를 들어 h1 태그에 parent() 메소드와 find()메소드를 사용하는 경우

```
// h1 태그의 부모 태그 선택
$('h1').parent()

// h1 태그 내부에 있는 i 태그 선택
$('h1').find(i)
```

화면 캡처: 2022-10-16 오후 4:33

- H1 태그가 여러 개 선택 된 상태로 parent() 메소드와 find() 메소드를 사용하면
- 첫 번째 h1 태그를 기준으로 부모와 후손을 찾는다

문서 객체 개별 조작

2022년 10월 16일 일요일 오후 4:34

- \$() 함수를 사용하면 여러 개의 문서 객체를 선택 할 수 있다
- 이러한 문서 객체가 몇 개 선택 되었는지 확인 할 때 length 속성 사용
- 선택된 문서 객체의 수

속성	설명
length	선택된 문서 객체의 수 구한다

- 배열처럼 대괄호를 사용해서 요소 하나를 꺼낼 수 도 있고 get() 메소드 활용 가능
- 선택된 문서 객체 추출

메소드	설명
Get()	선택한 문서 객체 중 하나를 선택

- 단, get() 메소드를 사용해 추출한 요소는 jQuery 문서 객체가 아니라 일반 문서 객체이다
- jQuery 객체로 다시 만들고 싶다면 \$() 함수의 매개 변수로 넣어야 한다

```
<head>
<meta charset="UTF-8">
<title>jQuery Basic</title>
<script src="jquery-3.6.1.min.js"></script>
<script>
// 웹 페이지를 모두 불러오려면
$(document).ready(function () {
    //h1태그를 모두 추출하고
    const $headers = $('h1');
    //h1 태그의 개수만큼 반복을 돌려서
    for(let i = 0; i <$headers.length; i++){
        //특정 위치에서
        if(i % 2 == 0){
            //1 번째 요소 추출
            const domElement = $headers.get(i);
            // 배경 색상 적용
            $(domElement).css('backgroundColor', 'red');
        }
    }
});
</script>
</head>
<body>
<h1>HeaderA</h1>
<h1>HeaderB</h1>
<h1>HeaderC</h1>
<h1>HeaderD</h1>
```

- 변수 이름 const\$headers= \$('h1'); 자바스크립트에서는 변수 이름으로 _와 \$ 사용 가능
- jQuery를 이러한 형태로 활용 하다보면 jQuery 문서 객체와 일반 문서 객체가 섞인다
- 이때 두 가지를 구분하려고 jQuery 문서 객체가 들어있는 변수는 앞에 \$를 붙여 구분한다
- 개발자 관례임

- 선택된 문서 객체 반복 적용

메소드	설명
Each()	선택한 문서 객체에 반복을 적용

- Each() 메소드를 사용 할 때 기본적인 자바스크립트에서 제공하는 Array 객체의 forEach() 메소드와 인덱스 요소 순서가 다르다
- Each() 메소드의 콜백함수

Array 객체의 forEach() 메소드	jQuery의 each() 메소드
[].forEach(function (item, index) {	\$(h1).each(function (index, item) {
});	});

- Each() 메소드의 콜백 함수 내부에서 this 는 문서객체를 나타낸다
- Each()메소드를 사용해 문서 객체 반복

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jQuery Event</title>
<script src="jquery-3.6.1.min.js"></script>
<script>
// 웹 페이지를 모두 불러오려면
$(document).ready(function () {
    //h1태그를 모두 추출하고 반복을 돌린다
    $('h1').each(function (index, element) {
        //특정 위치에서
        if(index % 2 == 0){
            // 배경 색상 적용
            $(this).css('backgroundColor', 'red');
        }
    });
});
</script>
</head>
<body>
<h1>HeaderA</h1>
<h1>HeaderB</h1>
<h1>HeaderC</h1>
<h1>HeaderD</h1>
</body>
</html>
```

화면 캡처: 2022-10-16 오후 4:58

HeaderA

HeaderB

HeaderC

HeaderD

화면 캡처: 2022-10-16 오후 4:58

[odd 선택자와 even 선택자]

- 홀수 번째 또는 짝수 번째 요소를 선택해서 무언가를 적용하는 코드는 자주 사용한다
- 따라서 쉽게 사용할 수 있는 형태를 제공해준다
- Odd선택자(홀수)와 even 선택자(짝수)
- \$() 함수 의 매개변수로 넣은 선택자 뒤에 :odd 또는 :even 을 넣어 사용할 수 있다

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jQuery Event</title>
<script src="jquery-3.6.1.min.js"></script>
<script>
// 웹 페이지를 모두 불러오려면
$(document).ready(function () {
    // 짝수 위치에 h1 태그에 배경 색상을 적용
    $('h1:even').css('backgroundColor', 'red');
});
</script>
```

화면 캡처: 2022-10-16 오후 5:07

문서 객체 조작

2022년 10월 16일 일요일 오후 5:07

1. 글자 조작

- 글자 조작 메소드

메소드	설명
text()	html태그 내부의 문자를 조작
html()	html태그 내부의 문자를 조작(html태그 외식

- jQuery 객체가 가진 대부분의 메소드는 하나의 이름으로 get(추출, 가져오기)와 Set(설정, 할당)을 수행한다

```
// h1 태그 내부의 문자를 가져온다
$('#h1').text()
```

```

//n1 값의 네번째 요소를 가져온다
$(n1).get(4)

```

- 선택자로 여러 개의 문서 객체를 선택 할 때 `text()` 메소드는 모든 문서 객체 내부의 문자를 출력
- `Html()` 메소드는 첫 번째 문서 객체 내부의 문자를 출력

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>jQuery Test</title>
<script src="jquery-3.6.1.min.js"></script>
</script>
<script>
$(document).ready(function () {
    // jQuery의 ready() 메소드를 사용하면 문서 로딩이 끝난 후에 실행
    alert("jQuery Test");
    alert($("#test").html());
});
</script>
</head>
<body>
<p>test</p>
<p>test</p>
<p>test</p>
</body>
```

- `Text()` 메소드는 모든 `p` 태그 내부의 문자를
- `Html()` 메소드는 첫번째로 선택된 `p` 태그 문자를 출력한다
- 실제로 `get` 형태도 사용할 때는 하나의 객체만 선택한 상태로 사용할 때가 많다
- 문서 객체 내부의 문자를 변경 할 때

```

//std::cout << "테두리 문자는 공백입니다." << endl;
//std::cout << "테두리 문자는 공백입니다." << endl;

```

화면 캡처: 2022-10-16 오후 5:28

- `Text()` 메소드와 `html()` 메소드 모두 여러 개의 문서 객체를 선택하고 `set` 형태의 메소드를 사용하면 모든 객체 내부에 문자가 들어간다
- `기능(메소드)` 이름이 다르므로 당연히 차이가 있다

[illegible]

화면 캡처: 2022-10-16 오후 5:32

Get 형태에서 text() 메소드는 모두 가져오고
Html() 메소드는 하나만 가져오므로 set 형태도 동일 할 것이라 생각하지만
Set 형태는 두 메소드 모두 선택한 문서 객체에 적용한다

2. 스타일 조작

- 스타일 조작 메소드

메소드	설명
Css()	스타일 조작 한다

- 스타일도 GET 형태와 SET 형태를 나누어 사용한다
- GET 형태는 선택자로 선택된 문서 객체 중 첫 번째 문서 객체의 스타일을 가져온다

```
// #1. 제3의 color 스타일 속성을 지정한다
let('1').css('color')

// #1. 제3의 color 스타일 속성을 rgb로 정할
let('1').css('color', 'red')

// #1. 제3의 color 스타일 속성과 backgroundColor 스타일 속성을 동시 변경 할경우이다
let('1').css({
  color: 'red',
  backgroundColor: 'orange'
})
```

화면 캡처: 2022-10-17 오후 7:28

스타일 조직

[illegible]

그라데이션 효과

3. 속성 조직

- 속성 조직 메소드

메소드	서형○
Attr()	속성을 조작

- 속성도 GET과 SET 형태 나 | 뉘서 사용
- Get 형태는 선택자로 선택된 문서 객체중 첫 번째 문서 객체의 속성을 가져온다

```
// tag 태그의 src 속성을 가져온다
let(tag) = attr('src')

// tag 태그의 src 속성을 "https://via.placeholder.com/200x200"으로 설정한다
let(tag) = attr('src', "https://via.placeholder.com/200x200")

// tag 태그의 src 속성을 alt 속성을 할당해 지정한다
let(tag) = attr({
  src: "https://via.placeholder.com/200x200",
  alt: "placeholder.it"
})
```

화면 캡처: 2022-10-17 오후 7:37

```
function main()
{
  //1. 变量
  var a = 10;
  var b = 20;
  var c = 30;
  var d = 40;
  var e = 50;
  var f = 60;
  var g = 70;
  var h = 80;
  var i = 90;
  var j = 100;
  var k = 110;
  var l = 120;
  var m = 130;
  var n = 140;
  var o = 150;
  var p = 160;
  var q = 170;
  var r = 180;
  var s = 190;
  var t = 200;
  var u = 210;
  var v = 220;
  var w = 230;
  var x = 240;
  var y = 250;
  var z = 260;
  var aa = 270;
  var bb = 280;
  var cc = 290;
  var dd = 300;
  var ee = 310;
  var ff = 320;
  var gg = 330;
  var hh = 340;
  var ii = 350;
  var jj = 360;
  var kk = 370;
  var ll = 380;
  var mm = 390;
  var nn = 400;
  var oo = 410;
  var pp = 420;
  var qq = 430;
  var rr = 440;
  var ss = 450;
  var tt = 460;
  var uu = 470;
  var vv = 480;
  var ww = 490;
  var xx = 500;
  var yy = 510;
  var zz = 520;
  var aaa = 530;
  var bbb = 540;
  var ccc = 550;
  var ddd = 560;
  var eee = 570;
  var fff = 580;
  var ggg = 590;
  var hhh = 600;
  var iii = 610;
  var jjj = 620;
  var kkk = 630;
  var lll = 640;
  var mmm = 650;
  var nnn = 660;
  var ooo = 670;
  var ppp = 680;
  var qqq = 690;
  var rrr = 700;
  var sss = 710;
  var ttt = 720;
  var uuu = 730;
  var vvv = 740;
  var www = 750;
  var xxx = 760;
  var yyy = 770;
  var zzz = 780;
  var aaaa = 790;
  var bbbb = 800;
  var cccc = 810;
  var dddd = 820;
  var eeee = 830;
  var ffff = 840;
  var gggg = 850;
  var hhhh = 860;
  var iiiii = 870;
  var jjjjj = 880;
  var kkkkk = 890;
  var lllll = 900;
  var mmmmm = 910;
  var nnnnn = 920;
  var ooooo = 930;
  var ppppp = 940;
  var qqqqq = 950;
  var rrrrr = 960;
  var sssss = 970;
  var ttttt = 980;
  var uuuuu = 990;
  var vvvvv = 1000;
  var wwww = 1010;
  var xxxxx = 1020;
  var yyyyy = 1030;
  var zzzzz = 1040;
  var aaaaa = 1050;
  var bbbbb = 1060;
  var ccccc = 1070;
  var ddddd = 1080;
  var eeeee = 1090;
  var fffff = 1100;
  var ggggg = 1110;
  var hhhhh = 1120;
  var iiiii = 1130;
  var jjjjj = 1140;
  var kkkkk = 1150;
  var lllll = 1160;
  var mmmmm = 1170;
  var nnnnn = 1180;
  var ooooo = 1190;
  var ppppp = 1200;
  var qqqqq = 1210;
  var rrrrr = 1220;
  var sssss = 1230;
  var ttttt = 1240;
  var uuuuu = 1250;
  var vvvvv = 1260;
  var wwww = 1270;
  var xxxxx = 1280;
  var yyyyy = 1290;
  var zzzzz = 1300;
  var aaaaa = 1310;
  var bbbbb = 1320;
  var ccccc = 1330;
  var ddddd = 1340;
  var eeeee = 1350;
  var fffff = 1360;
  var ggggg = 1370;
  var hhhhh = 1380;
  var iiiii = 1390;
  var jjjjj = 1400;
  var kkkkk = 1410;
  var lllll = 1420;
  var mmmmm = 1430;
  var nnnnn = 1440;
  var ooooo = 1450;
  var ppppp = 1460;
  var qqqqq = 1470;
  var rrrrr = 1480;
  var sssss = 1490;
  var ttttt = 1500;
  var uuuuu = 1510;
  var vvvvv = 1520;
  var wwww = 1530;
  var xxxxx = 1540;
  var yyyyy = 1550;
  var zzzzz = 1560;
  var aaaaa = 1570;
  var bbbbb = 1580;
  var ccccc = 1590;
  var ddddd = 1600;
  var eeeee = 1610;
  var fffff = 1620;
  var ggggg = 1630;
  var hhhhh = 1640;
  var iiiii = 1650;
  var jjjjj = 1660;
  var kkkkk = 1670;
  var lllll = 1680;
  var mmmmm = 1690;
  var nnnnn = 1700;
  var ooooo = 1710;
  var ppppp = 1720;
  var qqqqq = 1730;
  var rrrrr = 1740;
  var sssss = 1750;
  var ttttt = 1760;
  var uuuuu = 1770;
  var vvvvv = 1780;
  var wwww = 1790;
  var xxxxx = 1800;
  var yyyyy = 1810;
  var zzzzz = 1820;
  var aaaaa = 1830;
  var bbbbb = 1840;
  var ccccc = 1850;
  var ddddd = 1860;
  var eeeee = 1870;
  var fffff = 1880;
  var ggggg = 1890;
  var hhhhh = 1900;
  var iiiii = 1910;
  var jjjjj = 1920;
  var kkkkk = 1930;
  var lllll = 1940;
  var mmmmm = 1950;
  var nnnnn = 1960;
  var ooooo = 1970;
  var ppppp = 1980;
  var qqqqq = 1990;
  var rrrrr = 2000;
  var sssss = 2010;
  var ttttt = 2020;
  var uuuuu = 2030;
  var vvvvv = 2040;
  var wwww = 2050;
  var xxxxx = 2060;
  var yyyyy = 2070;
  var zzzzz = 2080;
  var aaaaa = 2090;
  var bbbbb = 2100;
  var ccccc = 2110;
  var ddddd = 2120;
  var eeeee = 2130;
  var fffff = 2140;
  var ggggg = 2150;
  var hhhhh = 2160;
  var iiiii = 2170;
  var jjjjj = 2180;
  var kkkkk = 2190;
  var lllll = 2200;
  var mmmmm = 2210;
  var nnnnn = 2220;
  var ooooo = 2230;
  var ppppp = 2240;
  var qqqqq = 2250;
  var rrrrr = 2260;
  var sssss = 2270;
  var ttttt = 2280;
  var uuuuu = 2290;
  var vvvvv = 2300;
  var wwww = 2310;
  var xxxxx = 2320;
  var yyyyy = 2330;
  var zzzzz = 2340;
  var aaaaa = 2350;
  var bbbbb = 2360;
  var ccccc = 2370;
  var ddddd = 2380;
  var eeeee = 2390;
  var fffff = 2400;
  var ggggg = 2410;
  var hhhhh = 2420;
  var iiiii = 2430;
  var jjjjj = 2440;
  var kkkkk = 2450;
  var lllll = 2460;
  var mmmmm = 2470;
  var nnnnn = 2480;
  var ooooo = 2490;
  var ppppp = 2500;
  var qqqqq = 2510;
  var rrrrr = 2520;
  var sssss = 2530;
  var ttttt = 2540;
  var uuuuu = 2550;
  var vvvvv = 2560;
  var wwww = 2570;
  var xxxxx = 2580;
  var yyyyy = 2590;
  var zzzzz = 2600;
  var aaaaa = 2610;
  var bbbbb = 2620;
  var ccccc = 2630;
  var ddddd = 2640;
  var eeeee = 2650;
  var fffff = 2660;
  var ggggg = 2670;
  var hhhhh = 2680;
  var iiiii = 2690;
  var jjjjj = 2700;
  var kkkkk = 2710;
  var lllll = 2720;
  var mmmmm = 2730;
  var nnnnn = 2740;
  var ooooo = 2750;
  var ppppp = 2760;
  var qqqqq = 2770;
  var rrrrr = 2780;
  var sssss = 2790;
  var ttttt = 2800;
  var uuuuu = 2810;
  var vvvvv = 2820;
  var wwww = 2830;
  var xxxxx = 2840;
  var yyyyy = 2850;
  var zzzzz = 2860;
  var aaaaa = 2870;
  var bbbbb = 2880;
  var ccccc = 2890;
  var ddddd = 2900;
  var eeeee = 2910;
  var fffff = 2920;
  var ggggg = 2930;
  var hhhhh = 2940;
  var iiiii = 2950;
  var jjjjj = 2960;
  var kkkkk = 2970;
  var lllll = 2980;
  var mmmmm = 2990;
  var nnnnn = 3000;
  var ooooo = 3010;
  var ppppp = 3020;
  var qqqqq = 3030;
  var rrrrr = 3040;
  var sssss = 3050;
  var ttttt = 3060;
  var uuuuu = 3070;
  var vvvvv = 3080;
  var wwww = 3090;
  var xxxxx = 3100;
  var yyyyy = 3110;
  var zzzzz = 3120;
  var aaaaa = 3130;
  var bbbbb = 3140;
  var ccccc = 3150;
  var ddddd = 3160;
  var eeeee = 3170;
  var fffff = 3180;
  var ggggg = 3190;
  var hhhhh = 3200;
  var iiiii = 3210;
  var jjjjj = 3220;
  var kkkkk = 3230;
  var lllll = 3240;
  var mmmmm = 3250;
  var nnnnn = 3260;
  var ooooo = 3270;
  var ppppp = 3280;
  var qqqqq = 3290;
  var rrrrr = 3300;
  var sssss = 3310;
  var ttttt = 3320;
  var uuuuu = 3330;
  var vvvvv = 3340;
  var wwww = 3350;
  var xxxxx = 3360;
  var yyyyy = 3370;
  var zzzzz = 3380;
  var aaaaa = 3390;
  var bbbbb = 3400;
  var ccccc = 3410;
  var ddddd
```

화면 캡처: 2022-10-17 오후 7:43

문서 객체 생성

2022년 10월 17일 월요일 오후 7:44

- 문서 객체 생성할 때 `$()` 함수의 매개 변수에 'HTML' 형식의 문자열을 입력한다
- `[$('<h1></h1>')]`

```
$('#<h1></h1>')  
  .text('안녕')  
  .attr('data-test', 'test')  
  .css({  
    backgroundColor: 'red',  
    color: 'white'  
  });
```

- 문서 객체 추가 메소드

메소드	설명
<code>\$(A).prependTo(B)</code>	A 를 B 안쪽 앞에 추가한다
<code>\$(A).appendTo(B)</code>	A 를 B 안쪽 뒤에 추가한다
<code>\$(A).insertBefore(B)</code>	A 를 B 앞에 추가한다
<code>\$(A).insertAfter(B)</code>	A 를 B 뒤에 추가한다

```
<h1> insertBefore</h1>  
  <div class = "target">  
    <h1>prependTo</h1>  
    <h1>prependTo</h1>  
  </div>  
<h1>insertAfter</h1>
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>jQuery Basic</title>
    <script src="jquery-3.6.1.min.js"></script>
    <script>
      // 웹 페이지를 모두 불러오려면
      $(document).ready(function () {
        $('<h1></h1>')
          .text('안녕')
          .attr('data-test', 'test')
          .css({
            backgroundColor: 'red',
            color: 'white'
          })
          .appendTo('body');
      })
    </script>
  </head>
  <body>
  </body>
</html>
```

화면 캡처: 2022-10-18 오전 9:37

이벤트

2022년 10월 18일 화요일 오전 9:37

- jQuery의 이벤트 메소드

메소드	설명
On()	이벤트 연결
Off()	이벤트 제거

1. 이벤트 직접 연결

- 특정 쿼리도 이벤트를 연결하고, 특정 데이터를 넣을 때 이벤트가 발생 하는 것을 의미함
- jQuery로 이벤트를 직접 연결 시켜도 `on()` 메소드를 사용하여 가능하게 함
- {`selector`:"jQuery" 이벤트명, 콜백함수}
- 이벤트를 연결 할 때 `on()` 메소드의 3번째 인자는 이벤트 객체를 전달함
- 이벤트 객체를 사용하면 이벤트 정보를 알아 볼 수 있다.
- 경로 확인
- {`jQuery` - `event object`}
- <https://api.jquery.com/category/event/event-object/>
- 콜백함수 내부에서 this 키워드는 이벤트를 발생 시키는 자체를 가리킴
- This 키워드는 이벤트를 발생 시키는 객체임
- This 키워드를 활용해서 이벤트를 발생시킨 객체의 스타일을 변경할 수 있다.

```

5 // 컴파일러 옵션
6 #define STRIP
7
8 // 헤더 파일
9 #include <stdio.h>
10 #include <stdlib.h>
11 #include <string.h>
12 #include <math.h>
13
14 // 랜덤 숫자 생성 함수
15 int random() {
16     static int seed = 1;
17     return (seed * 1103515245 + 1) >> 16;
18 }
19
20 // 랜덤 숫자 생성 함수
21 int random() {
22     static int seed = 1;
23     return (seed * 1103515245 + 1) >> 16;
24 }
25
26 // 랜덤 숫자 생성 함수
27 int random() {
28     static int seed = 1;
29     return (seed * 1103515245 + 1) >> 16;
30 }
31
32 // 랜덤 숫자 생성 함수
33 int random() {
34     static int seed = 1;
35     return (seed * 1103515245 + 1) >> 16;
36 }
37
38 // 랜덤 숫자 생성 함수
39 int random() {
40     static int seed = 1;
41     return (seed * 1103515245 + 1) >> 16;
42 }
43
44 // 랜덤 숫자 생성 함수
45 int random() {
46     static int seed = 1;
47     return (seed * 1103515245 + 1) >> 16;
48 }
49
50 // 랜덤 숫자 생성 함수
51 int random() {
52     static int seed = 1;
53     return (seed * 1103515245 + 1) >> 16;
54 }
55
56 // 랜덤 숫자 생성 함수
57 int random() {
58     static int seed = 1;
59     return (seed * 1103515245 + 1) >> 16;
60 }
61
62 // 랜덤 숫자 생성 함수
63 int random() {
64     static int seed = 1;
65     return (seed * 1103515245 + 1) >> 16;
66 }
67
68 // 랜덤 숫자 생성 함수
69 int random() {
70     static int seed = 1;
71     return (seed * 1103515245 + 1) >> 16;
72 }
73
74 // 랜덤 숫자 생성 함수
75 int random() {
76     static int seed = 1;
77     return (seed * 1103515245 + 1) >> 16;
78 }
79
80 // 랜덤 숫자 생성 함수
81 int random() {
82     static int seed = 1;
83     return (seed * 1103515245 + 1) >> 16;
84 }
85
86 // 랜덤 숫자 생성 함수
87 int random() {
88     static int seed = 1;
89     return (seed * 1103515245 + 1) >> 16;
90 }
91
92 // 랜덤 숫자 생성 함수
93 int random() {
94     static int seed = 1;
95     return (seed * 1103515245 + 1) >> 16;
96 }
97
98 // 랜덤 숫자 생성 함수
99 int random() {
100     static int seed = 1;
101     return (seed * 1103515245 + 1) >> 16;
102 }
103
104 // 랜덤 숫자 생성 함수
105 int random() {
106     static int seed = 1;
107     return (seed * 1103515245 + 1) >> 16;
108 }
109
110 // 랜덤 숫자 생성 함수
111 int random() {
112     static int seed = 1;
113     return (seed * 1103515245 + 1) >> 16;
114 }
115
116 // 랜덤 숫자 생성 함수
117 int random() {
118     static int seed = 1;
119     return (seed * 1103515245 + 1) >> 16;
120 }
121
122 // 랜덤 숫자 생성 함수
123 int random() {
124     static int seed = 1;
125     return (seed * 1103515245 + 1) >> 16;
126 }
127
128 // 랜덤 숫자 생성 함수
129 int random() {
130     static int seed = 1;
131     return (seed * 1103515245 + 1) >> 16;
132 }
133
134 // 랜덤 숫자 생성 함수
135 int random() {
136     static int seed = 1;
137     return (seed * 1103515245 + 1) >> 16;
138 }
139
140 // 랜덤 숫자 생성 함수
141 int random() {
142     static int seed = 1;
143     return (seed * 1103515245 + 1) >> 16;
144 }
145
146 // 랜덤 숫자 생성 함수
147 int random() {
148     static int seed = 1;
149     return (seed * 1103515245 + 1) >> 16;
150 }
151
152 // 랜덤 숫자 생성 함수
153 int random() {
154     static int seed = 1;
155     return (seed * 1103515245 + 1) >> 16;
156 }
157
158 // 랜덤 숫자 생성 함수
159 int random() {
160     static int seed = 1;
161     return (seed * 1103515245 + 1) >> 16;
162 }
163
164 // 랜덤 숫자 생성 함수
165 int random() {
166     static int seed = 1;
167     return (seed * 1103515245 + 1) >> 16;
168 }
169
170 // 랜덤 숫자 생성 함수
171 int random() {
172     static int seed = 1;
173     return (seed * 1103515245 + 1) >> 16;
174 }
175
176 // 랜덤 숫자 생성 함수
177 int random() {
178     static int seed = 1;
179     return (seed * 1103515245 + 1) >> 16;
180 }
181
182 // 랜덤 숫자 생성 함수
183 int random() {
184     static int seed = 1;
185     return (seed * 1103515245 + 1) >> 16;
186 }
187
188 // 랜덤 숫자 생성 함수
189 int random() {
190     static int seed = 1;
191     return (seed * 1103515245 + 1) >> 16;
192 }
193
194 // 랜덤 숫자 생성 함수
195 int random() {
196     static int seed = 1;
197     return (seed * 1103515245 + 1) >> 16;
198 }
199
200 // 랜덤 숫자 생성 함수
201 int random() {
202     static int seed = 1;
203     return (seed * 1103515245 + 1) >> 16;
204 }
205
206 // 랜덤 숫자 생성 함수
207 int random() {
208     static int seed = 1;
209     return (seed * 1103515245 + 1) >> 16;
210 }
211
212 // 랜덤 숫자 생성 함수
213 int random() {
214     static int seed = 1;
215     return (seed * 1103515245 + 1) >> 16;
216 }
217
218 // 랜덤 숫자 생성 함수
219 int random() {
220     static int seed = 1;
221     return (seed * 1103515245 + 1) >> 16;
222 }
223
224 // 랜덤 숫자 생성 함수
225 int random() {
226     static int seed = 1;
227     return (seed * 1103515245 + 1) >> 16;
228 }
229
230 // 랜덤 숫자 생성 함수
231 int random() {
232     static int seed = 1;
233     return (seed * 1103515245 + 1) >> 16;
234 }
235
236 // 랜덤 숫자 생성 함수
237 int random() {
238     static int seed = 1;
239     return (seed * 1103515245 + 1) >> 16;
240 }
241
242 // 랜덤 숫자 생성 함수
243 int random() {
244     static int seed = 1;
245     return (seed * 1103515245 + 1) >> 16;
246 }
247
248 // 랜덤 숫자 생성 함수
249 int random() {
250     static int seed = 1;
251     return (seed * 1103515245 + 1) >> 16;
252 }
253
254 // 랜덤 숫자 생성 함수
255 int random() {
256     static int seed = 1;
257     return (seed * 1103515245 + 1) >> 16;
258 }
259
260 // 랜덤 숫자 생성 함수
261 int random() {
262     static int seed = 1;
263     return (seed * 1103515245 + 1) >> 16;
264 }
265
266 // 랜덤 숫자 생성 함수
267 int random() {
268     static int seed = 1;
269     return (seed * 1103515245 + 1) >> 16;
270 }
271
272 // 랜덤 숫자 생성 함수
273 int random() {
274     static int seed = 1;
275     return (seed * 1103515245 + 1) >> 16;
276 }
277
278 // 랜덤 숫자 생성 함수
279 int random() {
280     static int seed = 1;
281     return (seed * 1103515245 + 1) >> 16;
282 }
283
284 // 랜덤 숫자 생성 함수
285 int random() {
286     static int seed = 1;
287     return (seed * 1103515245 + 1) >> 16;
288 }
289
290 // 랜덤 숫자 생성 함수
291 int random() {
292     static int seed = 1;
293     return (seed * 1103515245 + 1) >> 16;
294 }
295
296 // 랜덤 숫자 생성 함수
297 int random() {
298     static int seed = 1;
299     return (seed * 1103515245 + 1) >> 16;
300 }
301
302 // 랜덤 숫자 생성 함수
303 int random() {
304     static int seed = 1;
305     return (seed * 1103515245 + 1) >> 16;
306 }
307
308 // 랜덤 숫자 생성 함수
309 int random() {
310     static int seed = 1;
311     return (seed * 1103515245 + 1) >> 16;
312 }
313
314 // 랜덤 숫자 생성 함수
315 int random() {
316     static int seed = 1;
317     return (seed * 1103515245 + 1) >> 16;
318 }
319
320 // 랜덤 숫자 생성 함수
321 int random() {
322     static int seed = 1;
323     return (seed * 1103515245 + 1) >> 16;
324 }
325
326 // 랜덤 숫자 생성 함수
327 int random() {
328     static int seed = 1;
329     return (seed * 1103515245 + 1) >> 16;
330 }
331
332 // 랜덤 숫자 생성 함수
333 int random() {
334     static int seed = 1;
335     return (seed * 1103515245 + 1) >> 16;
336 }
337
338 // 랜덤 숫자 생성 함수
339 int random() {
340     static int seed = 1;
341     return (seed * 1103515245 + 1) >> 16;
342 }
343
344 // 랜덤 숫자 생성 함수
345 int random() {
346     static int seed = 1;
347     return (seed * 1103515245 + 1) >> 16;
348 }
349
350 // 랜덤 숫자 생성 함수
351 int random() {
3
```

화면 캡처: 2022-10-18 오후 5:44

- 키보드 이벤트

메소드	설명
Keydown()	키보드 키를 눌렀을 때
Keypress()	키가 입력 되었을 때
Keyup()	키보드 키를 떼었을 때

- **마우스 이벤트**

메소드	설명
Click()	마우스를 클릭 했을 때
Dbclick()	마우스를 더블 클릭 했을 때
mousedown()	마우스 버튼을 눌렀을 때
mouseenter()	마우스 커서가 해당 태그로 들어갔을 때
mouseleave()	마우스 커서가 해당 태그에서 나갔을 때
Mousemove()	마우스가 움직일 때
Mouseup()	마우스 버튼을 땄 때

- 입력 양식 이벤트

메소드	설명
Blur()	입력 양식에 값 입력을 종료할 때
Change()	입력 양식의 값이 변경될 때
Focus()	입력 양식에 값 입력을 시작할 때
select()	Type 속성이 select 인 입력 양식의 목록에서 값을 선택했을 때
Submit()	Type 속성이 submit 인 입력 양식을 클릭 했을 때

- 웹 브라우저 이벤트

메소드	설명
Resize()	웹 브라우저의 크기를 변경할 때
Scroll?()	웹 브라우저를 스크롤할 때

*`$(document).ready(function(){})`의 `ready`도 이벤트이다

- 기본 이벤트 제거

- Event 객체의 `preventDefault()` 메소드를 사용한다

```
//submit 버튼의 기본 이벤트를 모두 제거한다.
$('input[type=submit]').submit(function (event){
    event.preventDefault();
});
```

화면 캡처: 2022-10-18 오후 6:13

- 이벤트가 발생하면 새로운 객체 생성

[illegible]

- 코드를 실행하고 첫 번째 h1 태그를 클릭하면 h1 태그를 생성해서 body 태그 아래에 추가한다
- 하지만 새로 생성된 h1 태그에는 이벤트가 연결되어 있지 않다
- 이벤트 직접 연결은 연결 하는 시점에 존재하는 문서 객체에만 이벤트를 연결한다
- 문서 객체를 생성 했을 때 이벤트를 따로 연결해서 해결 할 수 있다.

- 이벤트 추가연결

```

//클래스 생성
class MyNumber {
public:
    static MyNumber *myNumber;
    MyNumber(int value):value(value){}
    ~MyNumber(){}
    static MyNumber *getInstance()
    {
        if(myNumber == 0)
            myNumber = new MyNumber(0);
        return myNumber;
    }
    static int getNumber()
    {
        return value;
    }
    static void setNumber(int value)
    {
        value = value;
    }
};

```

화면 캡처: 2022-10-18 오후 6:24

- 자주사용해야 하는 코드인데 형태가 복잡함
- 그래서 이벤트 간접 연결 기능을 제공한다

2. 이벤트 간접 연결

- 부모에게 이벤트를 위임 해서 부모가 이벤트를 처리하게 하는것
- 예제 : 이벤트 간접 연결, body 태그 내부에 있는 h1 태그에서 click 이벤트가 발생했을 때 h1 태그를 생성해 추가한다

[illegible]

- Body 태그 내부에서 h1 태그를 클릭 했을 때 구현할 수 있다.

3. 이벤트 제거

- H1 태그를 클릭하면 새로 h1 태그를 생성해 추가하고 기존 태그에서는 이벤트를 제거한다

[illegible]

```
// 002. 태그의 모든 이벤트를 제거
$('x1').off();

// 01. 태그로 click 이벤트를 모두 제거
$('x1').off('click');

// 03. 태그로 click 이벤트로 전달된 event 객체 중 특정은 event 객체는 제거한다
$('x1').off('click', handler);
```

- 이벤트를 한 번만 연결하는 메소드

메소드	설명
One()	이벤트를 한 번만 연결한다

애니메이션

2022년 10월 18일 화요일 오후 6:41

- 애니메이션 기능은 대부분 외부 라이브러리를 사용한다
- 애니메이션 메소드

메소드	설명
Animate()	애니메이션을 적용한다

- [\$(선택자).animate(속성, 시간, 콜백함수)]

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>jQuery Event</title>
<style>
  .box {
    /* 크기*/
    width: 100px; height: 100px;
    /* 색상*/
    background-color: red;
    /* 위치*/
    position: absolute;
    left: 10px; top: 10px;
  }
</style>
<script src="jquery-3.6.1.min.js"></script>
<script>
  // 웹 페이지를 모두 불러오려면
  $(document).ready(function () {
    //box 태그를 클릭해서
    $('.box').click(function () {
      //1초 동안 left를 1000픽셀로 변경한다
      $(this).animate({
        left: 1000
      }, 1000);
    })
  });
</script>
</head>
<body>
<div class="box">박스</div>

</body>
</html>
```

화면 캡처: 2022-10-18 오후 6:48