

JavaScript

JQuery – 김근형 강사

JQuery

- JQuery



JQuery

- JQuery
 - 기존 JavaScript에서 쉽고 적게, 다양한 일을 수행할 수 있도록 고안하여 만든 함수들의 집합인 라이브러리 언어.
 - DOM(Document Object Model)을 더욱 쉽고 간편하게 쓰기 위해 고안된 라이브러리
- 사용 이유
 - 자바스크립트 보다 코드를 짧고 쉽게 구현할 수 있다.
 - 모든 브라우저에 표준화.

JQuery

- JQuery : <https://jquery.com/>



JQuery

- JQuery : <https://jquery.com/>

The screenshot shows the official jQuery website's "Downloading jQuery" page. At the top, there is a navigation bar with links for "Download", "API Documentation", "Blog", "Plugins", and "Browser Support". To the right of the navigation bar is a search bar with a magnifying glass icon. The main content area has a blue header titled "Downloading jQuery". Below the header, there is a paragraph of text explaining the availability of compressed and uncompressed files, mentioning source map files, and noting that sourceMappingURL comments are not included in compressed files. It also instructs users to right-click and save files. A section titled "jQuery" follows, with a note about upgrading and using the Migrate plugin. Several download links are provided, including compressed and uncompressed versions, and slim builds. At the bottom, there is a link to a blog post with release notes.

Download API Documentation Blog Plugins Browser Support

Search

Downloading jQuery

Compressed and uncompressed copies of jQuery files are available. The uncompressed file is best used during development or debugging; the compressed file saves bandwidth and improves performance in production. You can also download a [sourcemap file](#) for use when debugging with a compressed file. The map file is *not* required for users to run jQuery, it just improves the developer's debugger experience. As of jQuery 1.11.0/2.1.0 the `//# sourceMappingURL` comment is [not included](#) in the compressed file.

To locally download these files, right-click the link and select "Save as..." from the menu.

jQuery

For help when upgrading jQuery, please see the [upgrade guide](#) most relevant to your version. We also recommend using the [jQuery Migrate plugin](#).

[Download the compressed, production jQuery 3.6.0](#)

[Download the uncompressed, development jQuery 3.6.0](#)

[Download the map file for jQuery 3.6.0](#)

You can also use the slim build, which excludes the [ajax](#) and [effects](#) modules:

[Download the compressed, production jQuery 3.6.0 slim build](#)

[Download the uncompressed, development jQuery 3.6.0 slim build](#)

[Download the map file for the jQuery 3.6.0 slim build](#)

[jQuery 3.6.0 blog post with release notes](#)

JQuery

○ JQuery : <https://jquery.com/>

JQuery

○ JQuery 사용 방법

```
<!DOCTYPE html>
<html Lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="js/jquery-3.6.0.min.js"></script>
</head>

<body>
    <script>
        window.addEventListener('DOMContentLoaded', function(){
            document.write("<h1>이것이 Javascript 이다 !!</h1>");
        });
        $(document).ready(function () {
            document.write("<h1>이것이 JQuery이다 !!</h1>");
        });
        $(function(){
            document.write("<h1>이것이 JQuery이다 !!</h1>");
        });
    </script>
</body>

</html>
```

JQuery

- JQuery 사용 방법
 - 문단 요소를 먼저 로딩한 후 제이쿼리 실행문 실행시키기 위해서는 다음과 같이 코드를 작성한다.

```
<script type="text/javascript">
$(document).ready(function(){
    일련의 실행문;
});
</script>
```

```
<script type="text/javascript">
$(function(){
    일련의 실행문;
});
</script>
```

JQuery

- JQuery vs DOMContentLoaded
 - JQuery의 ready() 메서드는 Document 객체에서 발생하는DOMContentLoaded 이벤트를 포장한 메서드
 - DOMContentLoaded 이벤트는 웹 브라우저가 웹 페이지를 읽은 후 태그와 1:1 맵핑되는 DOM 객체를 생성한 후 발생한다.
 - 이미지나 동영상 등 용량이 많은 파일들은 DOMContentLoaded가 올라오고 난 후에 천천히 로드가 되며 이 과정에서 JQuery와의 동기화에 문제가 생긴다.
 - window의 load 이벤트는 무거운 콘텐츠가 모드 로드 되고 난 후 발생하는 이벤트
 - 상황에 따라 적절히 사용하는 것이 포인트

JQuery

○ JQuery

```
<script type="text/javascript">
function jQuery(){
    .....
}
jQuery.prototype.css = function(){}
jQuery.prototype.on = function(){}
jQuery.prototype.addClass = function(){}
.....
</script>
```

```
window.jQuery = window.$ = jQuery = $;
```

```
<script type="text/javascript">
function $(){
    .....
    return new jQuery();
}
</script>
```

JQuery

- JQuery 목차
 - 선택자(Selector)
 - 객체 조작 메서드
 - 이벤트
 - 효과
 - 애니메이션
 - Ajax

JQuery – 선택자(Selector)

- JQuery 기본 사용법

```
$( ".IDName" ).css("background","green");
```

jquery define

Selectors

Method

선언

선택지정

실행/적용

JQuery – 선택자(Selector)

○ 직접 선택자(Selector)

종류	사용법	설명
전체 선택자	<code>\$("*")</code>	모든 요소를 선택
아이디 선택자	<code>\$("#아이디 명")</code>	id 속성이 지정한 값을 가진 요소를 선택
클래스 선택자	<code>\$(".클래스 명")</code>	class 속성이 지정한 값을 가진 요소 선택
태그 선택자	<code> \$("요소명")</code>	지정한 요소명과 일치하는 요소들만 선택
그룹 선택자	<code> \$("선택1,선택2,...선택n")</code>	선택n 까지 지정된 요소들을 한번에 선택

JQuery – 선택자(Selector)

○ 직접선택자(Selector)

제이쿼리

직접 선택자랑 관계 선택자

직접 선택자

관계 선택자

```
<html Lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="js/jquery-3.6.0.min.js"></script>
    <script>
        $(function(){
            $("*").css("border", "1px solid blue");
            $("h3").css("backgroundColor", "yellow");
            $(".cl1").css("fontStyle", "italic");
            $("#id1").css("color", "red")
            $("h1, h2").css("transform", "translateX(50px)")
        });
    </script>
</head>
<body>
    <h1>제이쿼리</h1>
    <h2 class="cl1">직접 선택자랑 관계 선택자</h2>
    <h3 id="id1">직접 선택자</h3>
    <h3>관계 선택자</h3>
</body>
</html>
```

JQuery – 선택자(Selector)

○ 인접관계선택자

종류	사용법	설명
부모 요소 선택자	<code>\$("요소선택").parent()</code>	선택한 요소의 부모요소 선택
상위 요소 선택자	<code>\$("요소선택").parents([요소명])</code>	선택한 요소의 상위 요소를 모두 선택
자식 요소들 선택자	<code>\$("요소선택").children()</code>	선택한 요소의 모든 자식요소 선택
이전 요소 선택자	<code>\$("요소선택").prev()</code>	선택한 요소의 바로 이전 요소를 선택
이전 요소들 선택자	<code>\$("요소선택").prevAll()</code>	선택한 요소의 이전요소 모두를 선택
지정 이전 요소들 선택자	<code>\$("요소선택").prevUntil(요소명)</code>	선택한 요소로부터 위로 지정한 요소의 이전 요소까지 모두 선택
다음 요소 선택자	<code>\$("요소선택").next()</code>	선택한 요소의 다음 요소 선택
다음 요소들 선택자	<code>\$("요소선택").nextAll()</code>	선택한 요소의 다음 요소 모두 선택
지정 이후 요소들 선택자	<code>\$("요소선택").nextUntil(요소명)</code>	선택한 요소로부터 아래로 지정한 요소의 이전 요소까지 모두 선택
형제 요소 선택자	<code>\$("요소선택").siblings()</code>	선택한 요소로부터 위 아래의 형제 요소 모두 선택
인접 부모 요소 선택자	<code>\$("요소선택").closest([요소명])</code>	선택한 요소로부터 가장 가까운 상위 요소 하나를 선택

JQuery – 선택자(Selector)

○ 인접관계선택자

```
<body>
  <div>
    <ul class="children">
      <li><p class="parent">아이템1</p></li>
      <li><p class="parents">아이템2</p></li>
      <li class="prev"><p>아이템3</p></li>
      <li><p>아이템4</p></li>
      <li class="prevuntil"><p>아이템5</p></li>
    </ul>
    <ul>
      <li class="next"><p>아이템6</p></li>
      <li><p>아이템7</p></li>
      <li><p>아이템8</p></li>
      <li><p>아이템9</p></li>
      <li class="nextuntil"><p>아이템10</p></li>
    </ul>
  </div>
</body>
```

```
<script>
$(function(){
  $(".parent").parent().css("padding", "1rem");
  $(".parents").parents().css("border", "1px solid blue");
  $(".children").children().css("border", "3px dotted red");
  $(".prev").prev().css("background-color", "yellow");
  $(".prev").prevAll().css("font-size", "1.5rem");
  $(".prevuntil").prevUntil(".prev").css("font-style", "italic");
  $(".next").next().css("font-family", "궁서");
  $(".next").nextAll().css("border", "1px dashed green");
  $(".next").nextUntil(".nextuntil").css("background-color", "cyan");
});
</script>
```

JQuery – 선택자(Selector)

○ 인접관계선택자

- 아이템1

- 아이템2

- 아이템3

- 아이템4

- 아이템5

- 아이템6

- 아이템7

- 아이템8

- 아이템9

- 아이템10

JQuery – 선택자(Selector)

○ 위치 탐색 선택자

종류	사용법	설명
<code>\$("요소선택:first")/\$(요소선택).first()</code>	<code>\$("li:fisrt")/\$("li").first()</code>	전체 요소 중 첫번째만 검색
<code>\$("요소선택:last")/\$(요소선택).last()</code>	<code>\$("li:last")/\$("li").last()</code>	전체 요소 중 마지막 요소만 검색
<code>\$("요소선택:odd")</code>	<code>\$("li:odd")</code>	요소 그룹 중 홀수 요소만 선택
<code>\$("요소선택:even")</code>	<code>\$("li:even")</code>	요소 그룹 중 짝수 요소만 선택
<code>\$("요소선택:first-of-type")</code>	<code>\$("li:first-of-type")</code>	요소 무리 중 첫 번째 요소만 선택
<code>\$("요소선택:last-of-type")</code>	<code>\$("li:last-of-type")</code>	요소 무리 중 마지막 요소만 선택
<code>\$("요소선택:nth-child(숫자)")</code>	<code>\$("li:nth-child(3)")</code>	요소 무리 중 세 번째 요소만 선택
<code>\$("요소선택:nth-child(숫자n)")</code>	<code>\$("li:nth-child(3n)")</code>	요소 무리 중 3의 배수 번째 요소만 선택
<code>\$("요소선택:nth-last-of-type(숫자)")</code>	<code>\$("li:nth-last-of-type(2)")</code>	요소 무리 중 마지막 위치로부터 2번째 있는 요소만 선택
<code>\$("요소선택:only-child")</code>	<code>\$("li:only-child")</code>	부모 요소 내에 요소가 1개뿐인 요소만 선택
<code>\$("요소선택:eq(index)")/\$(요소선택).eq(index)</code>	<code>\$("li:eq(2)")/ \$("li").eq(2)</code>	 요소 중 인덱스 2에 참조하는 요소를 불러옴
<code>\$("요소선택:gt(index)")</code>	<code>\$("li:gt(1)")</code>	요소 중 인덱스 1보다 큰 인덱스가 참조하는 요소를 불러옴
<code>\$("요소선택:lt(index)")</code>	<code>\$("li:lt(1)")</code>	요소 중 인덱스 1보다 작은 인덱스가 참조하는 요소를 불러옴
<code>\$(요소선택).slice(index)</code>	<code>\$("li").slice(2)</code>	요소 중 인덱스 2부터 참조하는 요소를 불러옴

JQuery – 선택자(Selector)

○ 위치 탐색 선택자

```
$(function(){
    // $("li:first").css("color", "red");
    $("li").first().css("color", "red");
    // $("li:last").css("color", "blue");
    $("li").last().css("color", "blue");
    $("li:odd").css("text-decoration", "overline");
    $("li:even").css("text-decoration", "underline");
    $("li:first-of-type").css("background-color", "cyan");
    $("li:last-of-type").css("background-color", "pink");
    $("li:nth-child(3)").css("border", "5px dashed black");
    $("li:nth-child(3n)").css("font-family", "궁서");
    $("li:nth-last-of-type(2)").css("background-color", "purple");
    $("li:only-child").css("padding", "1rem");
    // $("li:eq(2)").css("transform", "translateX(3rem)");
    $("li").eq(2).css("transform", "translateX(3rem)");
    $("li:gt(12)").css("transform", "rotatez(-5deg)");
    $("li:lt(1)").css("transform", "translateX(2rem)");
    $("li").slice(2, 5).css("color", "green");
});
```

```
<body>
<ul>
<li>아이템1</li>
<li>아이템2</li>
<li>아이템3</li>
<li>아이템4</li>
<li>아이템5</li>
<li>아이템6</li>
<li>아이템7</li>
</ul>
<ul>
<li>아이템8</li>
<li>아이템9</li>
<li>아이템10</li>
<li>아이템11</li>
<li>아이템12</li>
<li>아이템13</li>
<li>아이템14</li>
</ul>
<ul>
<li>아이템15</li>
</ul>
</body>
```

JQuery – 선택자(Selector)

○ 위치 탐색 선택자

- 아이템1
- 아이템2
 - 아이템3
- 아이템4
- 아이템5
- 아이템6
- 아이템7
- 아이템8
- 아이템9
- 아이템10
- 아이템11
- 아이템12
- 아이템13
- 아이템14
- 아이템15

JQuery – 선택자(Selector)

○ 배열 관련 선택자

종류	사용법	설명
each()/.each()	<code>\$(“요소 선택”).each(function) \$.each(“요소 선택”), function)</code>	배열에 저장된 문서 객체만큼 메서드가 반복 실행됨. 배열에 저장된 객체의 인덱스 순서대로 하나씩 접근하여 객체를 선택하고 인덱스를 구함.
\$.map()	<code>\$.map(Array, function)</code>	배열에 저장된 데이터 수만큼 메서드가 반복 실행됨. 함수에서 반환된 데이터는 새 배열에 순서대로 저장됨. 새로 저장된 배열 객체를 반환함.
\$.grep()	<code>\$.grep(Array, function)</code>	배열에 저장된 데이터 수만큼 메서드가 반복 실행됨. 반환값이 true인 경우에만 배열의 데이터가 인덱스 오름차순으로 새 배열에 저장되며 그 배열을 반환함.
\$.inArray()	<code>\$.inArray(data, Array, start index)</code>	배열 안에서 데이터를 찾음. 데이터를 찾으면 가장 맨 앞 데이터의 인덱스를 반환하고, 찾지 못하면 -1을 반환함. start index의 값을 지정하면 해당 위치부터 데이터를 찾음.
\$.isArray()	<code>\$.isArray(object)</code>	입력한 객체가 배열 객체라면 true 아니면 false를 반환함.
\$.merge()	<code>\$.merge(Array1, Array2)</code>	인자값으로 입력한 2개의 배열 객체를 하나로 그룹화함.
index()	<code>\$(“요소 선택”).index(“지정 요소 선택”)</code>	선택자로 요소를 먼저 선택한 다음 지정한 요소의 인덱스 정보를 가져옴

JQuery – 선택자(Selector)

○ 배열 관련 선택자 – each()

```
<h1>탐색 선택자</h1>
<ul id="menu1">
    <li>내용1-1</li>
    <li>내용1-2</li>
    <li>내용1-3</li>
</ul>
<ul id="menu2">
    <li>내용2-1</li>
    <li>내용2-2</li>
    <li>내용2-3</li>
</ul>
```

```
$(function(){
    var obj = [
        {"area":"서울"}, 
        {"area":"부산"}, 
        {"area":"전주"}];
    
    $(obj).each(function(i, o){
        console.log(i + ":", o);
    });
    console.log("==== The End 1 =====");

    $.each($("#menu2 li"), function(i, o){
        console.log(i + ":", o);
    });
    console.log("==== The End 2 =====");

    $.each($("#menu2 li"), function(i){
        console.log(i + ":", $(this));
    });
});
```

```
0: ▼ Object ①
  area: "서울"
  ► __proto__: Object
1: ▼ Object ②
  area: "부산"
  ► __proto__: Object
2: ▼ Object ③
  area: "전주"
  ► __proto__: Object
==== The End 1 ====
▼<li>
  ::marker
  "내용2-1"
0: </li>
▼<li>
  ::marker
  "내용2-2"
1: </li>
▼<li>
  ::marker
  "내용2-3"
2: </li>
==== The End 2 ====
0: ▼ S.fn.init(1) ④
  ► 0: li
  length: 1
  ► __proto__: Object(0)
1: ▼ S.fn.init(1) ⑤
  ► 0: li
  length: 1
  ► __proto__: Object(0)
2: ▼ S.fn.init(1) ⑥
  ► 0: li
  length: 1
  ► __proto__: Object(0)
```

JQuery – 선택자(Selector)

○ 배열 관련 선택자 – map, grep

```
▼ Array(2) ⓘ
  ▶ 0: {area: "서울", name: "무대리"}
  ▶ 1: {area: "서울", name: "빅마마"}
    length: 2
  ▶ __proto__: Array(0)
===== first End =====
▼ Array(2) ⓘ
  ▶ 0: {area: "서울", name: "무대리"}
  ▶ 1: {area: "서울", name: "빅마마"}
    length: 2
  ▶ __proto__: Array(0)
===== Second End =====
```

```
$function(){
  var arr1 = [
    {"area":"서울", "name":"무대리"},
    {"area":"부산", "name":"홍과장"},
    {"area":"대전", "name":"박사장"},
    {"area":"서울", "name":"빅마마"}
  ];

  var arr2 = $.map(arr1, function(a, b){
    if(a.area == "서울") {
      return a;
    }
  });
  console.log(arr2);
  console.log("==== first End ====");

  var arr3 = $.grep(arr1, function(a, b){
    if(a.area == "서울") {
      return true;
    } else {
      return false;
    }
  });
  console.log(arr3);
  console.log("==== Second End ====");
});
```

JQuery – 선택자(Selector)

○ 배열 관련 선택자 – inArray, isArray, merge

```
2  
true  
false  
▼ Array(8) ⓘ  
 0: "한국"  
 1: "미국"  
 2: "일본"  
 3: "중국"  
 4: "서울"  
 5: "대전"  
 6: "부산"  
 7: "전주"  
 length: 8  
▶ __proto__ : Array(0)
```

```
$(function(){  
    var arr1 = ["서울", "대전", "부산", "전주"];  
    var arr2 = ["한국", "미국", "일본", "중국"];  
    var obj = {  
        "name": "정부장",  
        "area": "서울"  
    }  
  
    var idxNum = $.inArray("부산", arr1);  
    console.log(idxNum);  
  
    var okArray1 = $.isArray(arr1);  
    var okArray2 = $.isArray(obj);  
    console.log(okArray1);  
    console.log(okArray2);  
  
    $.merge(arr2, arr1);  
    console.log(arr2);  
});
```

JQuery – 선택자(Selector)

○ 배열 관련 선택자 – index

2
▶

```
<script>
  $(function(){
    var idxNum = $("li").index($("#list3"));
    console.log(idxNum);
  });
</script>
</head>
<body>
  <h1>배열 관련 메서드</h1>
  <ul>
    <li>내용1</li>
    <li>내용2</li>
    <li id="list3">내용3</li>
    <li>내용4</li>
  </ul>
</body>
```

JQuery – 선택자(Selector)

○ 속성 탐색 선택자 / 그 외 선택자

종류	사용법	설명
<code>\$("요소선택[속성]")</code>	<code>\$("li[title]")</code>	요소 중 title 속성이 포함된 요소만 선택
<code>\$("요소선택[속성=값]")</code>	<code>\$("li[title='리스트'])")</code>	요소 중 title 속성값이 '리스트'인 요소만 선택
<code>\$("요소선택[속성^=텍스트]")</code>	<code>\$("a[href^='http://'])")</code>	<a>요소 중 href 속성값이 'http://'로 시작하는 요소만 선택
<code>\$("요소선택[속성\$=텍스트]")</code>	<code>\$("a[href\$='.com'])")</code>	<a>요소 중 href 속성값이 '.com'으로 끝나는 요소만 선택
<code>\$("요소선택[href*=텍스트]")</code>	<code>\$("a[href*='aaaa'])")</code>	<a>요소 속성 중 'aaaa'라는 값을 포함하는 요소 선택
<code>\$("요소선택:hidden")</code>	<code>\$("li:hidden")</code>	요소 중 숨겨진 것만 선택
<code>\$("요소선택:visible")</code>	<code>\$("li:visible")</code>	요소 중 보이는 것만 선택
<code>\$(":text")</code>	<code>\$(":text")</code>	<input>요소 중 type='text' 만 선택
<code>\$(":selected")</code>	<code>\$(":selected")</code>	selected 속성이 적용된 요소만 선택
<code>\$(":checked")</code>	<code>\$(":checked")</code>	checked 속성이 적용된 요소만 선택
<code>\$("input요소:[타입이름]")</code>	<code>\$("#member_f :text")</code>	<input id="member_f"> 요소 중 해당 타입의 요소를 선택
<code>\$("요소선택:contains(내용1)")</code>	<code>\$("#p:contains(내용1)")</code>	#p 내에 "내용1"이라는 텍스트 요소가 있는 요소 선택
<code>\$("요소선택:has(요소명)")</code>	<code>\$("#p:has(strong)")</code>	#p 내에 엘리먼트가 있는 요소 선택

JQuery – 선택자(Selector)

○ 속성 탐색 선택자 / 그 외 선택자

종류	사용법	설명
<code>\$("요소선택").contents()</code>	<code>\$("#outer_wrap").contents()</code>	<code>id="outer_wrap" 내의 컨텐츠를 선택한다.</code>
<code>\$("요소선택").not("요소선택")</code>	<code>\$("#inner_2 p").not(":first")</code>	<code>선택한 요소 중 특정 선택자를 제외한 요소를 선택한다.</code>
<code>\$("요소선택").end()...</code>	<code>\$("#p").find("a").end()</code>	<code>선택한 요소의 이전 요소를 선택한다.</code>
<code>\$('요소선택').find(' 요소선택')</code>	<code>\$('h1').find('span')</code>	<code>어떤 요소의 하위 요소 중 특정 요소를 찾을 때 사용한다.</code>
<code>\$('요소선택').filter(' 요소선택')</code>	<code>\$("li").filter(":nth-child(2n)")</code>	<code>해당 요소 안에서 특정 요소를 찾을 때 사용한다.</code>
<code>\$(선택자).is("선택자"); \$(선택자).is("변수"); \$(선택자).is("함수");</code>	<code>\$("#chk1").is(":checked");</code>	<code>선택한 요소에서 주어진 선택자가 있는지 판별 때 사용한다.</code>

JQuery – 선택자(Selector)

○ 속성 탐색 선택자 – 속성탐색, input 타입

```
<div id="wrap">
  <p><a href="http://easyspub.co.kr" target="_blank">EasysPub</a></p>
  <p><a href="https://naver.com">Naver</a></p>
  <p><a href="http://daum.net">Google</a></p>
  <p><a href="http://google.co.kr">Daum</a></p>
</div>
<form action="#" method="get" id="member_f">
  <p>
    <label for="user_id">아이디</label>
    <input type="text" name="user_id" id="user_id">
  </p>
  <p>
    <label for="user_pw">비밀번호</label>
    <input type="password" name="user_pw" id="user_pw">
  </p>
</form>
```

```
$(function(){
  $("#wrap a[target]")
    .css({"color": "#f00"});

  $("#wrap a[href^=https]")
    .css({"color": "#0f0"});

  $("#wrap a[href$=.net]")
    .css({"color": "#00f"});

  $("#wrap a[href*=google]")
    .css({"color": "#000"});

  $("#member_f :text")
    .css({"background-color": "#ff0"});

  $("#member_f :password")
    .css({"background-color": "#0ff"});
});
```

EasysPub

Naver

Google

Daum

아이디



비밀번호



JQuery – 선택자(Selector)

- 속성 탐색 선택자 / 그 외 선택자
 - hidden, checked, selected

```
$(function(){  
    $("#wrap p:hidden").css({  
        "display": "block",  
        "background": "#ff0"  
    });  
  
    var z1 = $("#zone1 :selected").val();  
    console.log(z1);  
  
    var z2 = $("#zone2 :checked").val();  
    console.log(z2);  
  
    var z3 = $("#zone3 :checked").val();  
    console.log(z3);  
});
```

```
<div id="wrap">  
    <p>내용1</p>  
    <p style="display:none">내용2</p>  
    <p>내용3</p>  
</div>  
<form action="#">  
    <p id="zone1">  
        <select name="course" id="course">  
            <option value="opt1">옵션1</option>  
            <option value="opt2" selected>옵션2</option>  
            <option value="opt3">옵션3</option>  
        </select>  
    </p>  
    <p id="zone2">  
        <input type="checkbox" name="hobby1" value="독서"> 독서  
        <input type="checkbox" name="hobby2" value="자전거"> 자전거  
        <input type="checkbox" name="hobby3" value="등산" checked> 등산  
    </p>  
    <p id="zone3">  
        <input type="radio" name="gender" value="male"> 남성  
        <input type="radio" name="gender" value="female" checked> 여성  
    </p>  
</form>
```

내용1

내용2

내용3

옵션2 ▾

독서 자전거 등산

남성 여성

JQuery – 선택자(Selector)

- 속성 탐색 선택자 / 그 외 선택자
 - contains, has, content, not

```
$("#inner_1 p:contains(내용1)"  
  .css({"background-color": "#ff0"});  
  
$("#inner_1 p:has(strong)"  
  .css({"background-color": "#0ff"});  
  
$("#outer_wrap").contents()  
  .css({"border": "1px dashed #00f"});  
  
$("#inner_2 p").not(":first")  
  .css({"background-color": "#0f0"});
```

```
<div id="outer_wrap">  
  <h1>콘텐츠 탐색 선택자</h1>  
  <section id="inner_1">  
    <h1>contains( ), contents( ), has( )</h1>  
    <p><span>내용1</span></p>  
    <p><strong>내용2</strong></p>  
    <p><span>내용3</span></p>  
  </section>  
  <section id="inner_2">  
    <h1>not( ), end( )</h1>  
    <p>내용4</p>  
    <p>내용5</p>  
    <p>내용6</p>  
  </section>  
</div>
```

콘텐츠 탐색 선택자

contains(), contents(), has()

내용1

내용2

내용3

not(), end()

내용4

내용5

내용6

JQuery – 선택자(Selector)

- 속성 탐색 선택자 / 그 외 선택자
 - end

```
$(function(){  
    $("ul.first")  
        .find(".foo")  
        .css("background-color", "red")  
        .end()  
        .find(".bar")  
        .css("background-color", "green");  
});
```

```
<ul class="first">  
    <li class="foo">list item 1</li>  
    <li>list item 2</li>  
    <li class="bar">list item 3</li>  
</ul>  
<ul class="second">  
    <li class="foo">list item 1</li>  
    <li>list item 2</li>  
    <li class="bar">list item 3</li>  
</ul>
```

- list item 1
- list item 2
- list item 3
- list item 1
- list item 2
- list item 3

JQuery – 선택자(Selector)

- 속성 탐색 선택자 / 그 외 선택자
 - find, filter

```
$(function(){  
    $("#inner_1").find(".txt1")  
    .css({"background-color": "#ff0"});  
  
    $("#inner_1 p").filter(".txt2")  
    .css({"background-color": "#0ff"});  
  
    $("#inner_2 p").filter(function(idx, obj){  
        console.log(idx);  
        return idx % 2 == 0;  
    })  
    .css({"background-color": "#0f0"});  
});
```

```
<div id="outer_wrap">  
    <h1>콘텐츠 탐색 선택자</h1>  
    <section id="inner_1">  
        <h2>find( ), filter( )</h2>  
        <p class="txt1">내용1</p>  
        <p class="txt2">내용2</p>  
    </section>  
  
    <section id="inner_2">  
        <h2>filter(function)</h2>  
        <p>index 0</p>  
        <p>index 1</p>  
        <p>index 2</p>  
        <p>index 3</p>  
    </section>  
</div>
```

콘텐츠 탐색 선택자

find(), filter()

내용1

내용2

filter(function)

index 0

index 1

index 2

index 3

JQuery – 선택자(Selector)

○ 속성 탐색 선택자 / 그 외 선택자

○ is

```
$(function(){
    var result_1 = $("#inner_1 p")
    .eq(0).is(":visible");
    console.log(result_1);

    var result_2 = $("#inner_1 p")
    .eq(1).is(":visible");
    console.log(result_2);

    var result_3 = $("#chk1").is(":checked");
    console.log(result_3);

    var result_4 = $("#chk2").is(":checked");
    console.log(result_4);
});
```

```
<div id="outer_wrap">
    <h1>is( )</h1>
    <section id="inner_1">
        <h2>문단 태그 영역</h2>
        <p>내용1</p>
        <p style="display:none;">내용2</p>
    </section>

    <section id="inner_2">
        <h2>폼 태그 영역</h2>
        <p>
            <input type="checkbox" name="chk1" id="chk1" checked>
            <label for="chk1">체크1</label>
        </p>
        <input type="checkbox" name="chk2" id="chk2">
        <label for="chk2">체크2</label>
    </section>
</div>
```

is()

문단 태그 영역

내용1

폼 태그 영역

체크1 체크2

true

false

true

false

JQuery – 객체 조작 메서드

○ 객체 조작 메서드

○ HTML 문서 객체를 생성, 복제, 삭제, 속성을 변환하는 메서드를 가리킨다.

- 생성

```
<div> </div> → <div><p>내용1</p></div>
```

- 복제

```
<div> 내용1 </div> → <div> 내용1 </div>  
                                <div> 내용1 </div>
```

- 삭제

```
<div> <p>내용1</p> </div> → <div> </div>
```

- 속성

```
 → 
```

JQuery – 객체 조작 메서드

○ 객체 조작 메서드 / 속성 조작 메서드

종류	사용법	설명
html()	<code>\$("요소선택").html(); \$("요소선택").html("새 요소");</code>	선택한 요소에 하위 요소들을 반환하거나 새 요소로 변환
text()	<code>\$("요소선택").text(); \$("요소선택").text("새 텍스트");</code>	선택한 요소에 포함하는 텍스트를 반환하거나 새 텍스트로 변환
css("속성")/ css("속성","값")	<code>\$("요소선택").css("color"); \$("요소선택").css("color","blue");</code>	선택한 요소에 스타일(CSS) 속성값을 반환 또는 변환
removeAttr("속성")	<code>\$("요소선택").removeAttr("title");</code>	선택한 요소의 지정한 속성을 삭제
attr("속성") / attr("속성","값")	<code>\$("요소선택").attr("href"); \$("요소선택").attr("href","main.html");</code>	선택한 요소에 속성값을 반환 또는 변환
addClass("클래스 값")	<code>\$("요소선택").addClass("abc");</code>	선택한 요소에 지정한 클래스 생성 또는 추가
removeClass("클래스 값")	<code>\$("요소선택").removeClass("abc");</code>	선택한 요소의 지정한 클래스 값 삭제
toggleClass("클래스 값")	<code>\$("요소선택").toggleClass("abc");</code>	선택한 요소에 지정한 클래스 값을 생성, 삭제를 반복
hasClass("클래스 값")	<code>\$("요소선택").hasClass("abc");</code>	선택한 요소에 지정한 클래스 값이 포함되어 있는지 검사. (true/false)
prop("속성")	<code>\$("요소선택").prop("tagName");</code>	선택한 태그의 속성들을 반환
val() / val(값)	<code>\$("요소선택").val(); \$("요소선택").val("홍길동");</code>	선택한 입력 요소에 지정한 value값을 반환 또는 변환

JQuery – 객체 조작 메서드

○ 객체 조작 메서드 / 속성 조작 메서드

○ html, text

```
$(function(){
    var result_1 = $("#sec_1").html( );
    console.log(result_1);
    $("#sec_1 p").html("<a href="#">Text1</a>");

    var result_2 = $("#sec_2").text( );
    console.log(result_2);
    $("#sec_2 h2").text("text()");
});
```

```
<h1><strong>객체 조작 및 생성</strong></h1>
<section id="sec_1">
    <h2><em>html()</em></h2>
    <p>내용1</p>
</section>
<section id="sec_2">
    <h2><em>텍스트()</em></h2>
    <p>내용2</p>
</section>
```

객체 조작 및 생성

html()

Text1

text()

내용2

JQuery – 객체 조작 메서드

- 객체 조작 메서드 / 속성 조작 메서드
 - attr, replace, removeAttr

```
$(function( ){
    var srcVal = $("#sec_1 img").attr("src");
    console.log(srcVal);

    $("#sec_1 img")
        .attr({
            "width":200,
            "src": srcVal.replace("1.jpg","2.jpg"),
            "alt": "바위"
        })
        .removeAttr("border");
});
```

```
<h1><strong>객체 조작 및 생성</strong></h1>
<section id="sec_1">
    <h2>이미지 속성</h2>
    <p></p>
</section>
```

객체 조작 및 생성

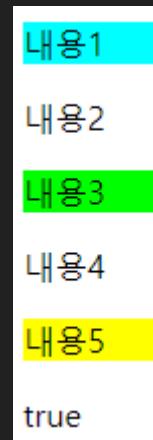
이미지 속성



가위 (2)

JQuery – 객체 조작 메서드

- 객체 조작 메서드 / 속성 조작 메서드
 - addClass, removeClass, toggleClass, hasClass



```
<script>
$(function( ){
    $("#p1").addClass("aqua");
    $("#p2").removeClass("red");
    $("#p3").toggleClass("green");
    $("#p4").toggleClass("green");
    $("#p6").text($("#p5").hasClass("yellow"));
});
</script>
<style>
.aqua{background-color: #0ff; }
.red{background-color: #f00; }
.green{background-color: #0f0; }
.yellow{background-color: #ff0; }
</style>
</head>
<body>
    <p id="p1">내용1</p>
    <p id="p2" class="red">내용2</p>
    <p id="p3">내용3</p>
    <p id="p4" class="green">내용4</p>
    <p id="p5" class="yellow">내용5</p>
    <p id="p6"></p>
</body>
```

JQuery – 객체 조작 메서드

- 객체 조작 메서드 / 속성 조작 메서드
 - val, prop

객체 조작 및 생성

이름

아이디

```
<script>
$(function( ){
    var result_1 = $("#user_name").val();
    console.log(result_1);

    $("#user_id").val("javascript");

    var result_2 = $("#user_id").prop("defaultValue");
    console.log(result_2);
});

</script>
</head>
<body>
    <h1>객체 조작 및 생성</h1>
    <form action="#" id="form_1">
        <p>
            <label for="user_name">이름</label>
            <input type="text" name="user_name"
                   id="user_name" value="용대리">
        </p>
        <p>
            <label for="user_id">아이디</label>
            <input type="text" name="user_id"
                   id="user_id" value="hello">
        </p>
    </form>
</body>
```

JQuery – 객체 조작 메서드

- 객체 조작 메서드 / 속성 조작 메서드
- val, prop

객체 조작 및 생성

chk1 chk2 chk3

option3 ▾

false
true
2

```
<script>
$(function( ){
    var result_1 = $("#chk1").prop("checked");
    console.log(result_1);

    var result_2 = $("#chk2").prop("checked");
    console.log(result_2);

    var result_2 = $("#chk3").prop("checked", true);

    var result_3 = $("#se_1").prop("selectedIndex");
    console.log(result_3);
});

</script>
</head>
<body>
<h1><strong>객체 조작 및 생성</strong></h1>
<form action="#" id="form_1">
<p>
    <input type="checkbox" name="chk1" id="chk1">
    <label for="chk1">chk1</label>
    <input type="checkbox" name="chk2" id="chk2" checked>
    <label for="chk2">chk2</label>
    <input type="checkbox" name="chk3" id="chk3">
    <label for="chk3">chk3</label>
</p>
<p>
    <select name="se_1" id="se_1">
        <option value="opt1">option1</option>
        <option value="opt2">option2</option>
        <option value="opt3" selected>option3</option>
    </select>
</p>
</form>
</body>
```

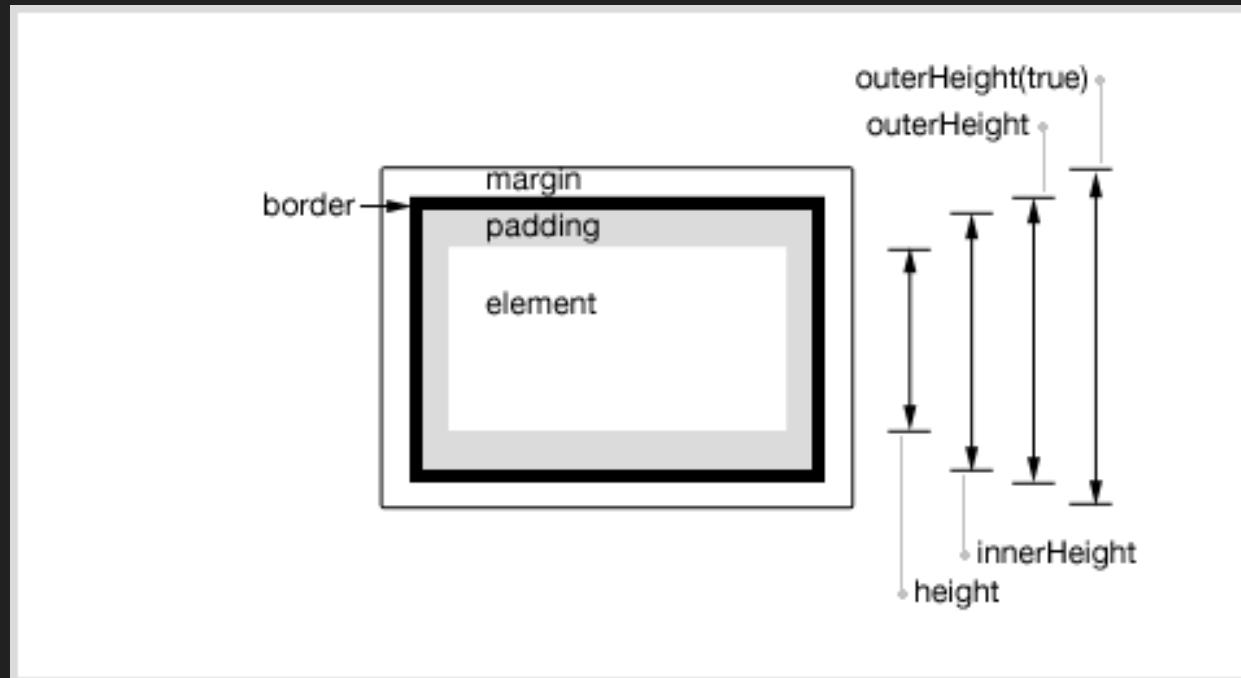
JQuery – 객체 조작 메서드

○ 수치 조작 메서드

종류	사용법	설명
.height()	<code>\$("#요소선택").height(); \$("#요소선택").height(100);</code>	안쪽 여백과 선을 제외한 높이값을 반환하거나 변환
.width()	<code>\$("#요소선택").width(); \$("#요소선택").width(100);</code>	안쪽 여백과 선을 제외한 너비값을 반환하거나 변환
.innerHeight()	<code>\$("#요소선택").innerHeight(); \$("#요소선택").innerHeight(100);</code>	안쪽 여백을 포함한 높이값을 반환하거나 변환
.innerWidth()	<code>\$("#요소선택").innerWidth(); \$("#요소선택").innerWidth(100);</code>	안쪽 여백을 포함한 너비값을 반환하거나 변환
.outerHeight()	<code>\$("#요소선택").outerHeight(); \$("#요소선택").outerHeight(100);</code>	선과 안쪽 여백을 포함한 높이값을 반환하거나 변환
.outerWidth()	<code>\$("#요소선택").outerWidth(); \$("#요소선택").outerWidth(100);</code>	선과 안쪽 여백을 포함한 너비값을 반환하거나 변환
position()	<code>\$("#요소선택").postion().left; \$("#요소선택").postion().top;</code>	선택한 요소의 포지션 위치값을 반환
offset()	<code>\$("#요소선택").offset().left; \$("#요소선택").offset().top;</code>	선택한 요소가 문서에서 수평/수직으로 얼마나 떨어져 있는지 떨어진 값을 반환
scrollLeft()	<code>\$(window).scrollLeft();</code>	브라우저의 수평 스크롤 이동 높이값을 반환
scrollTop()	<code>\$(window).scrollTop();</code>	브라우저의 수직 스크롤 이동 너비값을 반환

JQuery – 객체 조작 메서드

- 수치 조작 메서드
 - 요소의 높이/너비 메서드



JQuery – 객체 조작 메서드

- 수치 조작 메서드
- 요소의 높이/너비 메서드

수치 조작 메서드



50
90
100

```
<script>
$(function( ){
    var w1 = $("#p1").height();
    console.log(w1);

    var w2 = $("#p1").innerHeight();
    console.log(w2);

    var w3 = $("#p1").outerHeight();
    console.log(w3);

    $("#p2")
        .outerWidth(100)
        .outerHeight(100);
});
```

```
<style>
*{padding: 0;}
#p1, #p2{
    width: 100px;
    height: 50px;
    padding: 20px;
    border: 5px solid #000;
    background-color: #ff0;
}
</style>
</head>
<body>
    <h1>수치 조작 메서드</h1>
    <p id="p1">내용1</p>
    <p id="p2">내용2</p>
</body>
```

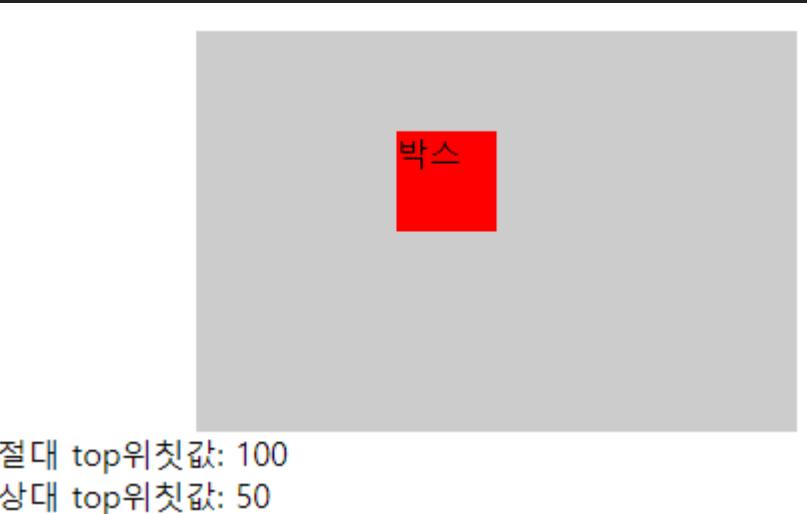
JQuery – 객체 조작 메서드

- 수치 조작 메서드
 - offset, position

```
<script>
$(function( ){
    var $txt1 = $(".txt_1 span"),
        $txt2 = $(".txt_2 span"),
        $box = $(".box");

    var off_t = $box.offset().top; //100
    var pos_t = $box.position().top; //50

    $txt1.text(off_t);
    $txt2.text(pos_t);
});
</script>
```



```
<style>
*{margin:0;padding:0;}
#box_wrap{
    width:300px;
    height:200px;
    margin:50px auto 0;
    position: relative;
    background-color:#ccc;
}
.box{
    width:50px;height:50px;
    position:absolute;
    left:100px;top:50px;
    background-color:#f00;
}
</style>
```

```
<body>
    <div id="box_wrap">
        <p class="box">박스</p>
    </div>
    <p class="txt_1">절대 top위치값: <span></span></p>
    <p class="txt_2">상대 top위치값: <span></span></p>
</body>
```

JQuery – 객체 조작 메서드

- 수치 조작 메서드
 - scrollTop

```
<script>
$(function( ){
    var topNum = $("h1").offset().top;
    $(window).scrollTop(topNum);

    var sct = $(window).scrollTop();
    console.log(sct);
});
</script>
<style>
*{margin:0;padding:0;}
body{line-height:1;}
#wrap{
    height:5000px;
    padding-top:2000px;
}
</style>
</head>
<body>
<div id="wrap">
    <h1>위치 메서드</h1>
</div>
</body>
```

JQuery – 객체 조작 메서드

○ 객체 편집 메서드

종류	사용법	설명
before()	<code>\$(“요소선택”).before(“새 요소”)</code>	선택한 요소 이전 위치에 새 요소를 추가함
after()	<code>\$(“요소선택”).after(“새 요소”)</code>	선택한 요소 다음 위치에 새 요소 추가
append()	<code>\$(“요소선택”).append(“새 요소”)</code>	선택한 요소의 마지막 위치에 새 요소 추가
appendTo()	<code>\$(“새 요소”).appendTo(“요소선택”)</code>	선택한 요소의 마지막 위치에 새 요소 추가
prepend()	<code>\$(“요소선택”).prepend(“새 요소”)</code>	선택한 요소의 맨 앞 위치에 새 요소를 추가
prependTo()	<code>\$(“새 요소”).prependTo(“요소선택”)</code>	선택한 요소의 맨 앞 위치에 새 요소 추가
insertBefore()	<code>\$(“요소선택”).insertBefore(“새 요소”)</code>	선택한 요소 이전 위치에 새 요소를 추가
insertAfter()	<code>\$(“요소선택”).insertAfter(“새 요소”)</code>	선택한 요소 다음 위치에 새 요소 추가
clone()	<code>\$(“요소선택”).clone(true or false)</code>	선택한 문서 객체를 복사.(인자값이 true 일 경우 하위 요소까지 모두 복제하고, false일 경우에 선택한 요소만 복제)
empty()	<code>\$(“요소선택”).empty()</code>	선택한 요소의 하위 내용들을 모두 삭제
remove()	<code>\$(“요소선택”).remove()</code>	선택한 요소를 삭제

JQuery – 객체 조작 메서드

○ 객체 편집 메서드

종류	사용법	설명
replaceAll() replaceWith()	<code>\$(“새 요소”).replaceAll(“요소선택”) \$(“요소선택”).replaceWith(“새 요소”)</code>	선택한 요소들을 새 요소로 교체
unwrap()	<code>\$(“요소선택”).unwrap();</code>	선택한 요소에 부모 요소를 삭제
wrap()	<code>\$(“요소선택”).wrap(새 요소);</code>	선택한 요소를 새 요소로 감쌈
wrapAll()	<code>\$(“요소선택”).wrapAll();</code>	선택한 요소를 새 요소로 한꺼번에 감쌈
wrapInner()	<code>\$(“요소선택”).wrapInner(새 요소);</code>	선택한 요소 내에 내용을 새 요소로 각각 감쌈

JQuery – 객체 조작 메서드

○ 객체 편집 메서드

○ after, insertAfter, before, insertBefore

```
<script>
$(function( ){
    $("#wrap p:eq(2)").after("<p>After</p>");
    "<p>insertAfter</p>").insertAfter("#wrap p:eq(1)");

    $("#wrap p:eq(1)").before("<p>Before</p>");
    "<p>insertBefore</p>").insertBefore("#wrap p:eq(0)");
});
</script>
</head>
<body>
    <div id="wrap">
        <p>내용1</p>
        <p>내용2</p>
        <p>내용3</p>
    </div>
</body>
```

insertBefore
내용1
Before
내용2
insertAfter
내용3
After

JQuery – 객체 조작 메서드

○ 객체 편집 메서드

○ appendTo, prepend

```
<script>
  $(function( ){
    $("<li>appendTo</li>").appendTo("#listZone");
    $("#listZone").prepend("<li>prepend</li>");
  });
</script>
</head>
<body>
  <ul id="listZone">
    <li>리스트</li>
  </ul>
</body>
```

- prepend
- 리스트
- appendTo

JQuery – 객체 조작 메서드

- 객체 편집 메서드
 - clone, remove, empty

```
<script>
$(function( ){
    var copyObj = $(".box1").children().clone();

    $(".box2").remove( );

    $(".box3").empty( );
    $(".box3").append(copyObj);
});

</script>
</head>
<body>
    <div class="box1">
        <p>내용1</p>
        <p>내용2</p>
    </div>
    <div class="box2">
        <p>내용3</p>
        <p>내용4</p>
    </div>
    <div class="box3">
        <p>내용5</p>
        <p>내용6</p>
    </div>
</body>
```

내용1	내용2
내용1	내용2

JQuery – 객체 조작 메서드

- 객체 편집 메서드

- replaceAll, replaceWith

```
<script>
  $(function( ){
    $("h2").replaceWith("<h3>replace method</h3>");
    $("<p>Change</p>").replaceAll("div");
  });
</script>
</head>
<body>
  <section class="box1">
    <h2>제 목1</h2>
    <div>내 용1</div>
    <div>내 용2</div>
  </section>
  <section class="box2">
    <h2>제 목2</h2>
    <div>내 용3</div>
    <div>내 용4</div>
  </section>
</body>
```

replace method

Change

Change

replace method

Change

Change

JQuery – 객체 조작 메서드

○ 객체 편집 메서드

- unwrap, wrap, wrapAll, wrapInner

```
<script>
$(function( ){
    $("strong").unwrap();
    $(".ct1").wrap("<div />");
    $(".ct2").wrapAll("<div />");
    $("li").wrapInner("<h3 />");
});
</script>
<style>
    div{background-color: #aqua;}
</style>
</head>
<body>
    <h1 id="tit_1">
        <strong>객체 조작 및 생성</strong>
    </h1>
    <p class="ct1">내용1</p>
    <p class="ct1">내용2</p>
    <p class="ct2">내용3</p>
    <p class="ct2">내용4</p>
    <ul>
        <li>내용3</li>
        <li>내용4</li>
    </ul>
</body>
```

객체 조작 및 생성

내용1

내용2

내용3

내용4

- 내용3
- 내용4

JQuery – 이벤트

- 이벤트
 - 사이트에 방문자가 취하는 모든 행위를 이벤트라 하며 이벤트가 발생했을 때 실행문을 이벤트 핸들러라 한다.



```
$("#btn1").click(function(){  
    $("p.blue").hide("slow");  
});
```

```
<p>내용1</p>  
<button id='btn1'>Hide Blue</button>
```

JQuery – 이벤트

○ 이벤트 등록 메서드 종류

구분	종류	설명
로딩이벤트	load()	선택한 이미지 또는 프레임 요소에 연동된 소스의 로딩이 완료 된 후 이벤트가 발생
	ready()	지정한 HTML 문서 객체의 로딩이 완료된 후 이벤트가 발생
	error()	이벤트 대상 요소에서 오류가 발생하면 이벤트가 발생

JQuery – 이벤트

○ 이벤트 등록 메서드 종류

구분	종류	설명
마우스이벤트	click()	선택한 요소를 클릭했을 때 이벤트 발생
	dblclick()	선택한 요소를 연속해서 두 번 클릭했을 때 이벤트 발생
	mouseout()	선택 요소의 영역에서 마우스 포인터가 벗어났을 때 이벤트가 발생(하위요소의 영향)
	mouseover()	선택 요소의 영역에 마우스 포인터 올렸을 때 이벤트 발생
	hover()	선택한 요소에 마우스 포인터를 올렸을 때와 벗어났을 때 각각 이벤트가 발생
	mousedown()	선택한 요소에서 마우스 버튼을 눌렀을 때 이벤트가 발생
	mouseup()	선택한 요소 범위에 마우스 버튼을 놓렀다 떼었을 때 이벤트가 발생
	mouseenter()	선택한 요소 범위에 마우스 포인터를 올렸을 때 이벤트가 발생
	mouseleave()	선택한 요소 범위에서 마우스 포인터가 벗어났을 때 이벤트가 발생
	mousemove()	선택한 요소 범위에서 마우스 포인터를 움직였을 때 이벤트가 발생
	scroll()	가로, 세로 스크롤바를 움직일 때마다 이벤트가 발생

JQuery – 이벤트

○ 이벤트 등록 메서드 종류

구분	종류	설명
로딩이벤트	focus()	선택한 요소에 포커스가 생성되었을 때 이벤트를 발생하거나 선택한 요소에 강제로 포커스 생성
	focusin()	선택한 요소에 포커스가 생성되었을 때 이벤트 발생
	focusout()	포커스가 선택한 요소에서 다른 요소로 이동되었을 때 이벤트 발생
	blur()	포커스가 선택한 요소에서 다른 요소로 이동되었을 때 이벤트가 발생하거나 선택한 요소의 포커스가 강제로 사라지도록 함
	change()	이벤트 대상인 입력 요소의 값이 변경되고, 포커스가 이동하면 이벤트가 발생. 그리고 강제로 change 이벤트를 발생시킬 때도 사용.
키보드이벤트	keyoress()	선택한 요소에서 키보드를 눌렀을 때 이벤트가 발생. 문자키를 제외한 키의 코드값 반환
	keydown()	선택한 요소에서 키보드를 눌렀을 때 이벤트 발생. 키보드의 모든 기의 코드값을 반환
	keyup()	선택한 요소에서 키보드에서 손을 떼었을 때 이벤트가 발생.

JQuery – 이벤트

- 메서드 실행 방식

```
$("#btn1").click(function(){
    $("p.blue").hide("slow");
});
```

JQuery – 이벤트

○ 메서드 등록 방식

1. \$("이벤트 대상 선택").on("이벤트 종류1, 이벤트종류2....이벤트 종류n", function(){자바스크립트 코드;});
2. \$("이벤트 대상 선택").on({"이벤트 종류1 이벤트 종류2 ... 이벤트 종류n" : function() {자바스크립트 코드;}});
3. \$("이벤트 대상 선택").on({
 "이벤트 종류1" : function() {자바스크립트 코드;1},
 "이벤트 종류2" : function() {자바스크립트 코드;2},
 ...
 "이벤트 종류n" : function() {자바스크립트 코드;n},
});

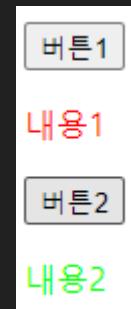
JQuery – 이벤트

○ 이벤트 메서드 예제

```
$(function () {
    $(".btn1").click(function () {
        $(".btn1").parent().next()
            .css({ "color": "#f00" });
    });

    $(".btn2").on({
        "mouseover focus": function () {
            $(".btn2").parent().next()
                .css({ "color": "#0f0" });
        },
        "mouseout blur": function () {
            $(".btn2").parent().next()
                .css({ "color": "#000" });
        },
    });
});
```

```
<body>
    <p>
        <button class="btn1">버튼1</button>
    </p>
    <p>내용1</p>
    <p>
        <button class="btn2">버튼2</button>
    </p>
    <p>내용2</p>
</body>
```



JQuery – 이벤트

○ 이벤트 강제 실행

- \$("이벤트 대상").단독 이벤트 등록 메서드();
- \$("이벤트 대상").trigger("이벤트 종류");

```
$(function () {
    $(".btn1").click(function () {
        $(".btn1").parent().next()
            .css({ "color": "#f00" });
    });

    $(".btn2").on({
        "mouseover focus": function () {
            $(".btn2").parent().next()
                .css({ "color": "#0f0" });
        }
    });

    $(".btn1").click();
    $(".btn2").trigger("mouseover");
});
```

```
<p>
    <button class="btn1">버튼1</button>
</p>
<p>내용1</p>
<p>
    <button class="btn2">버튼2</button>
</p>
<p>내용2</p>
```



JQuery – 이벤트

- 이벤트 제거 메서드

- \$("이벤트 대상").off("제거할 이벤트 종류");

```
$(function () {
    $(".btn1").click(function () {
        $(".btn1").parent().next()
            .css({ "color": "#f00" });
    });

    $(".btn2").on({
        "mouseover focus": function () {
            $(".btn2").parent().next()
                .css({ "color": "#0f0" });
        }
    });

    $(".btn1").off("click");
    $(".btn2").off("mouseover focus");
});
```

```
<p>
|   <button class="btn1">버튼1</button>
</p>
<p>내용1</p>
<p>
|   <button class="btn2">버튼2</button>
</p>
<p>내용2</p>
```

JQuery – 이벤트

- 이벤트 메서드
 - ready, load

```
<script>
$(function () {
    $(document).ready(function () {
        var h1 = $(".img1").height();
        console.log("ready :", h1);
    });

    $(window).on('load', function () {
        var h2 = $(".img1").height();
        console.log("load :", h2);
    });
});
</script>
```

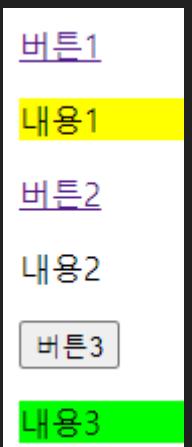
```
<body>
  <p>
    
  </p>
</body>
```



```
ready : 0
load : 300
```

JQuery – 이벤트

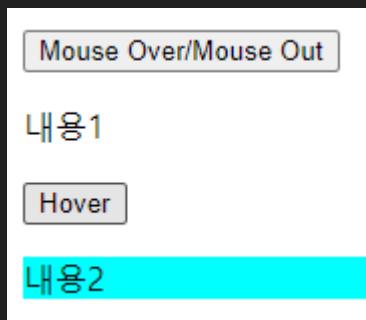
- 이벤트 메서드
- click, dblclick



```
<script>
$(function( ) {
    $(".btn1").on("click", function(e){
        e.preventDefault();
        $(".txt1")
            .css({"background-color":"#ff0"});
    });
    $(".btn2").on("click", function(e){
        $(".txt2")
            .css({"background-color":"#0ff"});
    });
    $(".btn3").on("dblclick", function(){
        $(".txt3")
            .css({"background-color":"#0f0"});
    });
});
</script>
</head>
<body>
    <p><a href="https://www.google.com/" class="btn1">버튼1</a></p>
    <p class="txt1">내용1</p>
    <p><a href="https://www.google.com/" class="btn2">버튼2</a></p>
    <p class="txt2">내용2</p>
    <p><button class="btn3">버튼3</button></p>
    <p class="txt3">내용3</p>
</body>
```

JQuery – 이벤트

- 이벤트 메서드
 - mouseout, mouseover, hover



```
<script>
$(function( ) {
    $(".btn1").on({
        "mouseover" : function( ) {
            $(".txt1")
                .css({"background-color":"yellow"});
        },
        "mouseout" : function( ) {
            $(".txt1")
                .css({"background":"none"});
        }
    ));

    $(".btn2").hover(function( ) {
        $(".txt2")
            .css({"background-color":"aqua"});
    }, function(){
        $(".txt2")
            .css({"background":"none"});
    });
});
</script>
</head>
<body>
    <p><button class="btn1">Mouse Over/Mouse Out</button></p>
    <p class="txt1">내용1</p>
    <p><button class="btn2">Hover</button></p>
    <p class="txt2">내용2</p>
</body>
```

JQuery – 이벤트

- 이벤트 메서드
 - mouseout, mouseleave



```
<script>
$(function () {
    $("#box_1").on("mouseout", function () {
        $("#box_1")
            .css({ "background-color": "yellow" });
    });

    $("#box_2").on("mouseleave", function () {
        $("#box_2")
            .css({ "background-color": "pink" });
    });
</script>
</head>

<body>
<h1>mouseout</h1>
<div id="box_1">
    <p><a href="#">내용1</a></p>
    <p><a href="#">내용2</a></p>
    <p><a href="#">내용3</a></p>
</div>

<h1>mouseleave</h1>
<div id="box_2">
    <p><a href="#">내용4</a></p>
    <p><a href="#">내용5</a></p>
    <p><a href="#">내용6</a></p>
</div>
</body>
```

JQuery – 이벤트

- 이벤트 메서드
 - mousemove

```
<script>
$(function( ) {
    $(document).on("mousemove", function(e) {
        $(".posX").text(e.pageX);
        $(".posY").text(e.pageY);
    });
});
</script>
</head>
<body>
<h1>mousemove</h1>
<p>X : <span class="posX">0</span>px</p>
<p>Y : <span class="posY">0</span>px</p>
</body>
```

mousemove

X : 218px

Y : 178px

JQuery – 이벤트

- 이벤트 메서드
- scrollTop, scrollLeft

scrollTop: 883px
scrollLeft: 788px

```
<script>
  $(window).on("scroll",function(){
    var sc_top = $(this).scrollTop();
    var sc_left = $(this).scrollLeft();

    $(".top").text(sc_top);
    $(".left").text(sc_left);
  });
</script>
<style>
  body{
    height:1000px;
    width:500px;
  }
  #wrap{
    position: fixed;
    left: 10px; top: 10px;
  }
</style>
</head>
<body>
<div id="wrap">
  <p>scrollTop: <span class="top">0</span>px</p>
  <p>scrollLeft: <span class="left">0</span>px</p>
</div>
```

JQuery – 이벤트

○ 이벤트 메서드

○ focus, blur, focusin, focusout

focus / blur

ID

PW

focusin / focusout

ID

PW

```
<script>
  $(function () {
    $("#user_id_1, #user_pw_1").on("focus",
      function () {
        $(this)
          .css({ "background-color": "pink" });
      });
    $("#user_id_1, #user_pw_1").on("blur",
      function () {
        $(this)
          .css({ "background-color": "#fff" });
      });

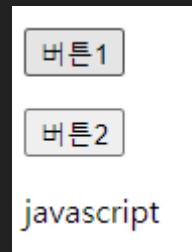
    $("#frm_2").on("focusin",
      function () {
        $(this)
          .css({ "background-color": "pink" });
      });
    $("#frm_2").on("focusout",
      function () {
        $(this)
          .css({ "background-color": "#fff" });
      });
  });
</script>
```

```
<body>
  <h1>focus / blur</h1>
  <form action="#">
    <p>
      <label for="user_id_1">ID</label>
      <input type="text" name="user_id_1" id="user_id_1">
    </p>
    <p>
      <label for="user_pw_1">PW</label>
      <input type="password" name="user_pw_1" id="user_pw_1">
    </p>
  </form>
  <h1>focusin / focusout</h1>
  <form action="#" id="frm_2">
    <p>
      <label for="user_id_2">ID</label>
      <input type="text" name="user_id_2" id="user_id_2">
    </p>
    <p>
      <label for="user_pw_2">PW</label>
      <input type="password" name="user_pw_2" id="user_pw_2">
    </p>
  </form>
</body>
```

JQuery – 이벤트

○ 이벤트 메서드

- `data()` : 해당 엘리먼트에 Javascript Type의 value를 <Key, Value>로 저장할 수 있으며, 값으로 저장되어 있는 데이터를 읽어올 때 사용 한다.



```
<body>
  <p><button id="btn1">버튼1</button></p>
  <p><button id="btn2">버튼2</button></p>
  <p class="txt"></p>
</body>

<script>
$(function () {
  $("#btn1")
    .data({ "text": "javascript" })
    .on({
      "mouseover": overFnc,
      "mouseout": outFnc
    });

  $("#btn2")
    .data({ "text": "welcome!" })
    .on({
      "mouseover focus": overFnc,
      "mouseout blur": outFnc
    });

  function overFnc() {
    $(".txt").text($(this).data("text"));
  }
  function outFnc() {
    $(".txt").text("");
  }
});
</script>
```

JQuery – 이벤트

- 이벤트 메서드

- change

```
<script>
$(function(){
    $("#rel_site").on("change", function(){
        $(".txt").text($(this).val());
    });
});
</script>
</head>
<body>
<select id="rel_site">
    <option value="">사이트 선택</option>
    <option value="www.google.com">구글</option>
    <option value="www.naver.com">네이버</option>
    <option value="www.daum.net">다음</option>
</select>
<p class="txt"></p>
```



JQuery – 이벤트

- 키 코드를 받는 이벤트 파라미터
- 키 이벤트 시 파라미터를 통해 키 이벤트를 받는 함수 만들기

```
<body>
|   <p><input type="text" name="user_id" id="user_id"></p>
</body>
```

```
<script>
$(function () {
    $(document).on("keydown", keyEventFnc);
    function keyEventFnc(e) {
        var direct = "";

        switch (e.keyCode) {
            case 37: direct = "LEFT";
            break;
            case 38: direct = "TOP";
            break;
            case 39: direct = "RIGHT";
            break;
            case 40: direct = "BOTTOM";
            break;
        }

        if (direct) $("#user_id").val(direct);
    });
}</script>
```

LEFT

JQuery – 이벤트

- o this와 index(this)를 활용한 예제

\$this

- 메뉴1
- 메뉴2
- 메뉴3

Index()

- 메뉴4
- 메뉴5
- 메뉴6

0

```
<h2>$this</h2>
<ul class="menuWrap_1">
  <li><a href="#">메뉴1</a></li>
  <li><a href="#">메뉴2</a></li>
  <li><a href="#">메뉴3</a></li>
</ul>
<h2>Index()</h2>
<ul class="menuWrap_2">
  <li><a href="#">메뉴4</a></li>
  <li><a href="#">메뉴5</a></li>
  <li><a href="#">메뉴6</a></li>
</ul>
<p class="idxNum"></p>
```

```
$(function () {
  $(".menuWrap_1 a").on("click", function (e) {
    e.preventDefault();

    $(".menuWrap_1 a")
      .css({ "background-color": "#fff" });

    $(this)
      .css({ "background-color": "#ff0" });
  });

  $(".menuWrap_2 a").on("click", function (e) {
    e.preventDefault();

    $(".menuWrap_2 a")
      .css({ "background-color": "#fff" });

    var idx = $(".menuWrap_2 a").index(this);
    $(".menuWrap_2 a").eq(idx)
      .css({ "background-color": "#0ff" });

    $(".idxNum").text(idx);
  });
});
```

JQuery – 이벤트

- 전역 동적 이벤트 리스너
 - 이벤트를 통해 엘리먼트가 추가되거나 삭제되는 경우 기존에 동일하게 막혔던 이벤트가 동작하지 않는 경우가 대다수 발생한다.
 - 이럴 경우 추가되는 엘리먼트에 동적으로 이벤트를 추가하도록 설정할 수 있다.
 - document 자체에 on을 이용하여 이벤트를 추가하는 방식이며 엘리먼트가 추가되어도 해당 엘리먼트는 동적 이벤트 추가 없이 그대로 이벤트를 사용할 수 있다.
 - 이것을 전역 동적 이벤트 리스너라 하며 쓰는 방법은 아래와 같다.

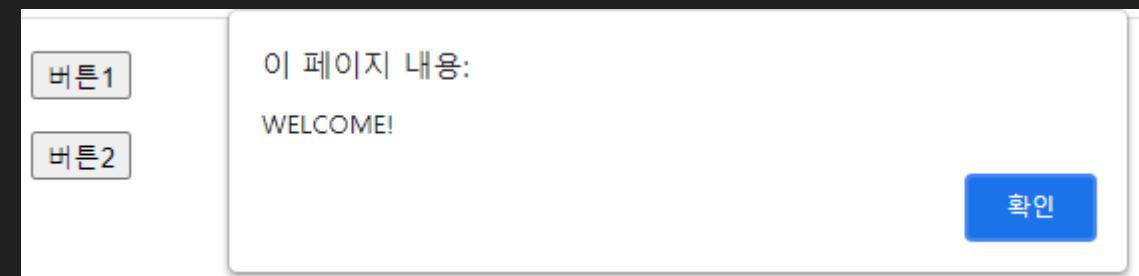
```
$(document).on("이벤트","셀렉터", function(){ ... });
```

JQuery – 이벤트

- 전역 동적 이벤트 리스너 예제

```
<script>
$(function(){
    $(".btn_1.on").on("mouseover focus", function() {
        alert("HELLO!");
    });
    $(".btn_1").addClass("on");

    $(document).on("mouseover focus", ".btn_2.on", function() {
        alert("WELCOME!");
    });
    $(".btn_2").addClass("on");
});
</script>
</head>
<body>
    <div id="wrap">
        <p><button class="btn_1">버튼1</button></p>
        <p><button class="btn_2">버튼2</button></p>
    </div>
</body>
</html>
```



JQuery – 이벤트

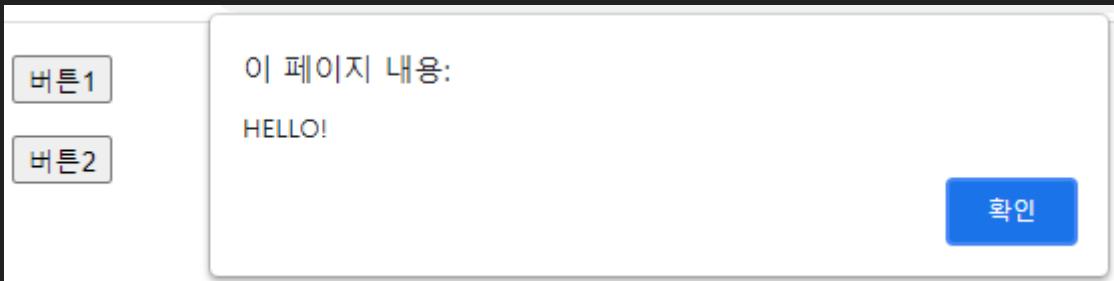
○ 그룹 이벤트

- 이벤트를 다수 등록하는 방식에는 on이라는 메서드 말고 상당히 다양한 메서드가 있다.
- 각 메서드는 아래와 같다.

메서드	설명
.on()	한 개 이상의 이벤트 등록 시 사용, 사용방식에 따라 이벤트를 등록한 이후에도 동적으로 생성된 이벤트를 등록한 요소와 동일한 새 요소에도 이벤트가 등록
.delegate()	선택한 요소에 지정한 하위 요소에 이벤트 등록. 등록한 후에 동적으로 생성된 이벤트를 등록한 요소와 동일한 새 요소에도 이벤트 등록
.one()	선택한 요소에 이벤트를 등록하며, 단 한번만 이벤트 발생

JQuery – 이벤트

○ 그룹 이벤트 예제



```
<script>
$(function(){
    $(".btn_wrap").delegate(".btn_1.on",
        "mouseover focus", function() {
            alert("HELLO!");
        });
    $(".btn_1").addClass("on");

    $(document).one("mouseover focus",
        ".btn_2.on", function() {
            alert("WELCOME!");
        });
    $(".btn_2").addClass("on");
});
</script>
</head>
<body>
<div id="wrap">
    <p class="btn_wrap">
        <button class="btn_1">버튼1</button>
    </p>
    <p><button class="btn_2">버튼2</button></p>
</div>
```

JQuery – 이벤트

○ 전역 동적 이벤트 리스너의 삭제



```
<script>
$(function(){
    $(".btn_1").on("mouseover", function() {
        alert("HELLO!");
    });
    $(document).on("mouseover", ".btn_2", function() {
        alert("WELCOME!");
    });
    var btn_2 = $("<p><button class=\"btn_2\">버튼2</button></p>");
    $("#wrap").append(btn_2);

    $(".del_1").on("click", function(){
        $(".btn_1").off("mouseover");
    });
    $(".del_2").on("click", function(){
        $(document).off("mouseover", ".btn_2");
    });
});
</script>
</head>
<body>
<div id="wrap">
    <p><button class="btn_1">버튼1</button></p>
</div>
<p>
    <button class="del_1">버튼1 이벤트 해지</button>
    <button class="del_2">버튼2 이벤트 해지</button>
</p>
</body>
```

JQuery – 효과

○ 효과(Effect)

- 효과(Effect)란 스타일(CSS)을 이용해 보였다, 숨겼다 하는 속성을 보다 역동적으로 모션을 적용할 수 있는 메서드를 가리킨다.
- 사용방법은 아래와 같다.

```
$(“요소선택”).효과 메서드(효과 소요 시간, 가속도, 콜백 함수);
```

효과 소요 시간
"slow", "normal", "fast"
1,000(1초), 500(0.5초)

가속도
"swing" 시작과 끝은 느리게, 중간은 빠르게 (기본값) "linear" 일정한 속도로 움직입니다.



JQuery – 효과

○ 효과(Effect) 이벤트 종류

메서드	설명
<code>\$(selector).hide()</code>	선택한 요소를 숨깁니다.
<code>\$(selector).show()</code>	선택한 요소를 노출시킵니다.
<code>\$(selector).toggle()</code>	선택한 요소를 hide() and show() 합니다.
<code>\$(selector).slideDown()</code>	선택한 요소를 밑으로 펼쳐지며 노출시킵니다.
<code>\$(selector).slideUp()</code>	선택한 요소를 위로 접으며 숨깁니다.
<code>\$(selector).slideToggle()</code>	선택한 요소를 slideDown() and slideUp() 합니다.
<code>\$(selector).fadeIn()</code>	선택한 요소를 투명도를 조절하며 나타냅니다.
<code>\$(selector).fadeOut()</code>	선택한 요소를 투명도를 조절하며 숨깁니다.
<code>\$(selector).fadeTo()</code>	선택한 요소를 지정한 투명도까지 숨깁니다.
<code>\$(selector).fadeToggle()</code>	선택한 요소를 fadeIn() and fadeOut() 합니다.

JQuery – 효과

○ 효과(Effect) 예제

slideUp

toggleSide

fadeTo(0.3)

fadeTo(1)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas feugiat consectetur nibh, ut luctus urna placerat non. Phasellus consectetur nunc nec mi feugiat egestas. Pellentesque et consectetur mauris, sed rutrum est. Etiam odio nunc, ornare quis urna sed, fermentum fermentum augue. Nam imperdiet vestibulum ipsum quis feugiat. Nunc non pellentesque diam, nec tempor nibh. Ut consequat sem sit amet ullamcorper sodales.

```
<style>
    .content{
        width:400px;
        background-color: #eee;
    }
</style>
</head>
<body>
    <p>
        <button class="btn1">slideUp</button>
        <button class="btn2">fadeIn</button>
    </p>
    <p>
        <button class="btn3">toggleSide</button>
    </p>
    <p>
        <button class="btn4">fadeTo(0.3)</button>
        <button class="btn5">fadeTo(1)</button>
    </p>
    <div class="box">
        <div class="content">
            Lorem ipsum dolor sit amet, consectetur
        </div>
    </div>
</body>
```

```
<script>
$(function( ) {
    $(".btn2").hide();

    $(".btn1").on("click", function( ) {
        $(".box").slideUp(1000, "linear",
            function( ) {
                $(".btn1").hide( );
                $(".btn2").show( );
            });
    });

    $(".btn2").on("click", function( ) {
        $(".box").fadeIn(1000, "swing",
            function( ) {
                $(".btn2").hide( );
                $(".btn1").show( );
            });
    });

    $(".btn3").on("click", function( ) {
        $(".box").slideToggle(1000, "linear");
    });

    $(".btn4").on("click", function( ) {
        $(".box").fadeTo("fast", 0.3);
    });

    $(".btn5").on("click", function( ) {
        $(".box").fadeTo("fast", 1);
    });
});
```

JQuery – 애니메이션

- 애니메이션
- css 요소에 동작(Motion)을 적용하는 것을 가리킨다.



```
$(“선택요소”).animate({ 변화될 속성 }, 적용속도, 콜백함수 );
```

JQuery – 애니메이션

○ 애니메이션 메서드 종류

메서드	설명
<code>\$(selector).animate()</code>	선택한 요소에 애니메이션을 적용
<code>\$(selector).stop()</code>	실행중인 효과를 모두 정지
<code>\$(selector).delay()</code>	스택에 대기중인 애니메이션 효과를 지연(지연 시간 후 발생)
<code>\$(selector).queue()</code>	스택에 대기중인 큐를 반환 혹은 함수의 실행문을 큐로 추가
<code>\$(selector).clearQueue()</code>	첫번째 효과를 제외한 나머지 애니메이션 큐를 전부 제거
<code>\$(selector).dequeue()</code>	스택에 쌓인 큐를 전부 제거
<code>\$(selector).finish()</code>	선택 요소의 진행중인 애니메이션이 강제로 완료 시점으로 간 후 종료

JQuery – 애니메이션

○ 애니메이션 예제

버튼1

"500px" 이동

버튼2

"50px"씩 이동

```
<script>
$(function(){
    $(".btn1").on("click", function( ) {
        $(".txt1").animate({
            marginLeft:"500px",
            fontSize:"30px"
        }, 2000, "linear", function( ) {
            alert("모션 완료!");
        });
    });

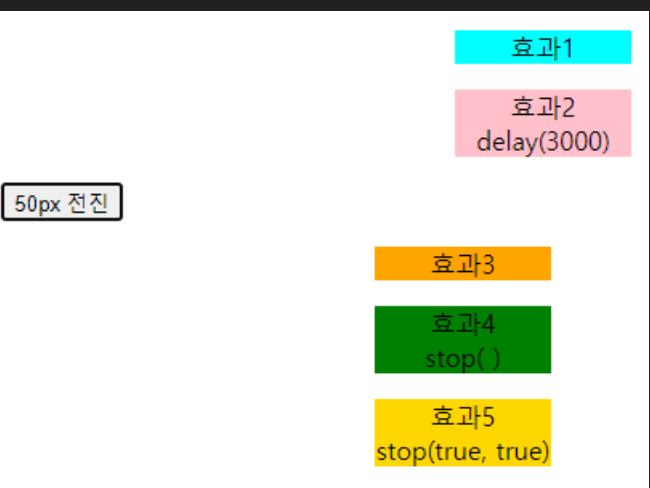
    $(".btn2").on("click", function( ) {
        $(".txt2").animate({
            marginLeft:"+=50px"
        }, 1000);
    });
</script>
<style>
    .txt1{background-color: aqua;}
    .txt2{background-color: pink;}
</style>
</head>
<body>
    <p><button class="btn1">버튼1</button></p>
    <span class="txt1">"500px" 이동</span>
    <p><button class="btn2">버튼2</button></p>
    <span class="txt2">"50px"씩 이동</span>
</body>
```

JQuery – 애니메이션

- delay, stop
 - stop(clearQueue, jumpToEnd)
 - clearQueue : 대기중인 애니메이션 전부 제거
 - jumpToEnd : 진행중인 애니메이션을 강제 종료하고 애니메이션을 종료한 맨 끝 상태로 이동

JQuery – 애니메이션

○ delay, stop



```
<style>
  p{width: 110px;text-align: center;}
  .txt1{background-color: aqua;}
  .txt2{background-color: pink;}
  .txt3{background-color: orange;}
  .txt4{background-color: green;}
  .txt5{background-color: gold;}
</style>
</head>
<body>
  <p class="txt1">효과1</p>
  <p class="txt2">효과2<br> delay(3000)</p>

  <p><button class="btn1">50px 전진 </button></p>
  <p class="txt3">효과3</p>
  <p class="txt4">효과4<br>stop( )</p>
  <p class="txt5">효과5<br>stop(true, true)</p>
</body>
```

```
<script>
$(function(){
  $(".txt1")
    .animate({marginLeft:"300px"},1000);

  $(".txt2").delay(3000)
    .animate({marginLeft:"300px"},1000);

  $(".btn1").on("click", moveElement);

  function moveElement( ) {
    $(".txt3")
      .animate({marginLeft:"+=50px"},800);

    $(".txt4")
      .animate({marginLeft:"+=50px"},800);
    $(".txt4").stop()

    $(".txt5")
      .animate({marginLeft:"+=50px"},800);
    $(".txt5").stop(true, true)
  }
});
```

JQuery – 애니메이션

- queue

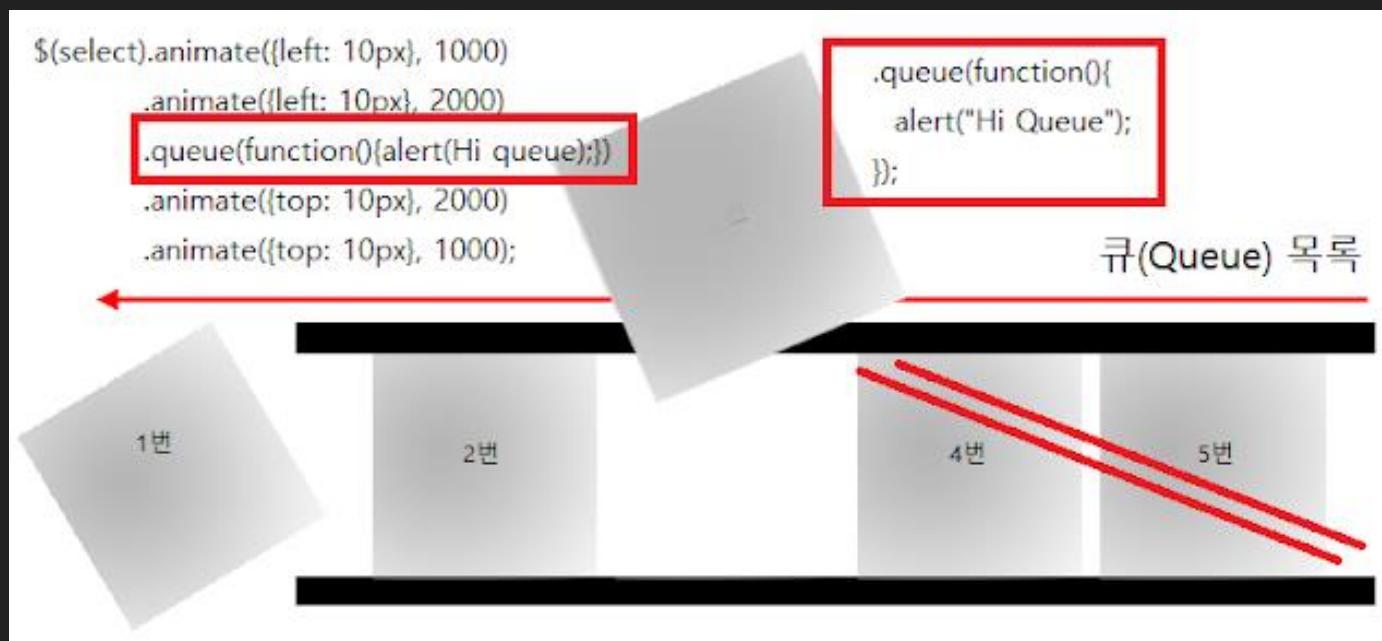
- 큐(Queue)란 특정 요소에서 애니메이션이 실행되는 순서로 저장되어 있는 공간을 의미한다.
- 큐에 저장된 목록에서 순번대로 애니메이션을 하나씩 꺼내어 실행하고, 실행된 애니메이션은 제거한다.



JQuery – 애니메이션

- queue

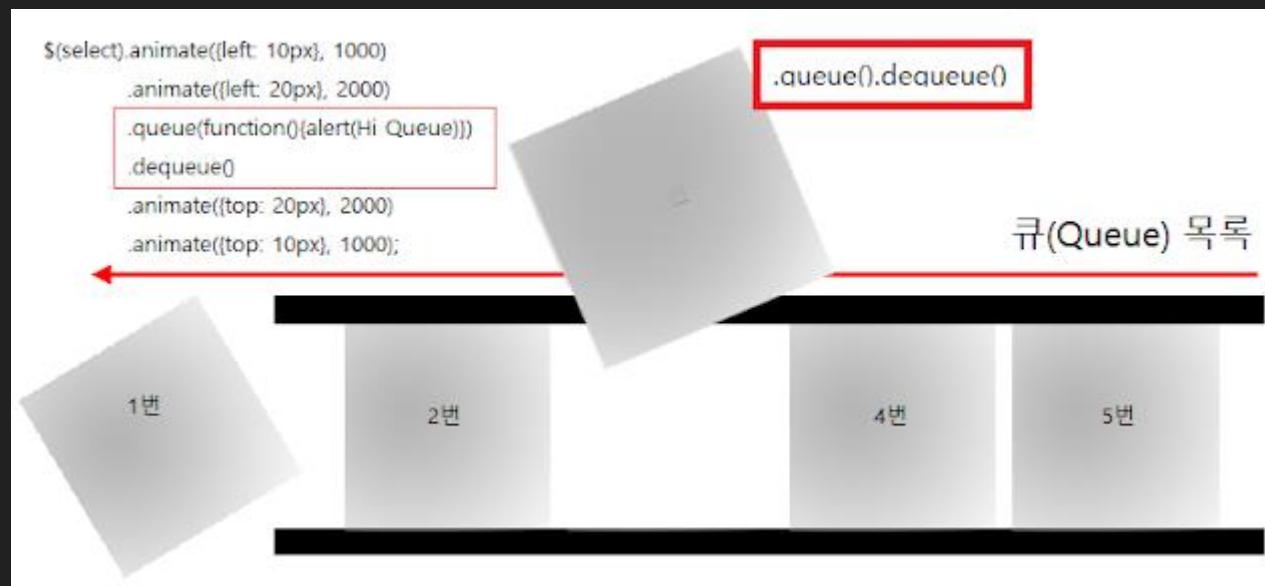
- .queue() 메소드는 큐에 대기중인 함수를 반환하거나 새로운 함수를 추가한다.



JQuery – 애니메이션

○ queue

- 하지만 .queue() 메소드를 .animate() 중간에 추가하게 되면 .queue() 까지만 실행되고 그 뒤에 대기중인 대기열은 자동으로 취소가 된다.
- 그렇지만 뒤에 .dequeue() 메소드를 추가해 주면 취소되지 않고 연결되어 애니메이션이 실행되게 할 수 있다.



JQuery – 애니메이션

- queue, dequeue

```
<script>
$(function(){
    $(".txt1")
        .animate({marginLeft:"200px"},1000)
        .animate({marginTop:"200px"},1000)
        .queue(function() {
            |   |   $(this).css({background:"red"});
            |   |   $(this).dequeue();
        })
        .animate({marginLeft:0},1000)
        .animate({marginTop:0},1000);
});
</script>
<style>
    *{margin:0;}
    .txt1{width:50px;text-align:
        center;background-color: #aqua;}
</style>
</head>
<body>
    <p class="txt1">내용1</p>
</body>
```

JQuery – 애니메이션

○ clearQueue



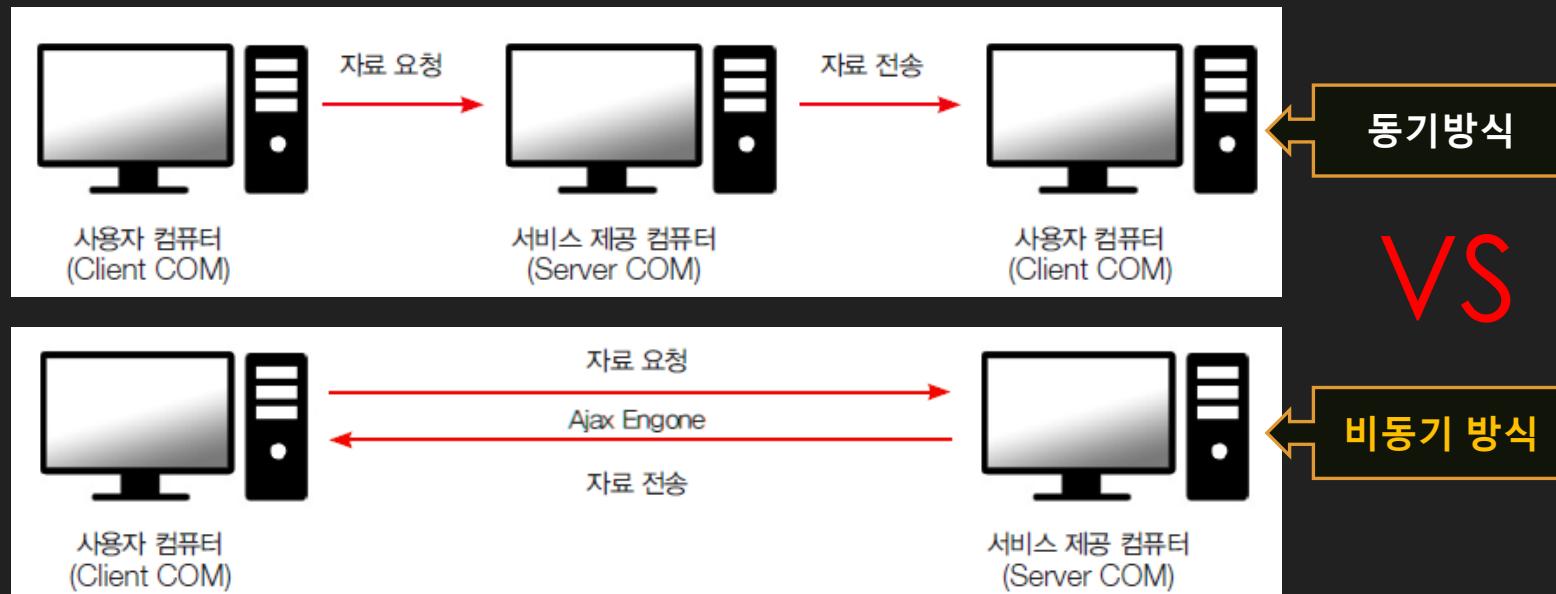
```
<script>
$(function() {
    $(".txt1")
        .animate({marginLeft:"100px"},1000)
        .animate({marginLeft:"300px"},1000)
        .animate({marginLeft:"400px"},1000);

    $(".txt2")
        .animate({marginLeft:"100px"},1000)
        .animate({marginLeft:"300px"},1000)
        .animate({marginLeft:"400px"},1000);
    $(".txt2").clearQueue();
});
</script>
<style>
.txt1, .txt2{width:50px; text-align:center; background-color: #aqua;}
.txt2{background-color:#orange;}
</style>
</head>
<body>
    <p class="txt1">내용1</p>
    <p class="txt2">내용2</p>
</body>
```

Ajax

- ajax

- Ajax(Asynchronous Javascript and XML)란 자바스크립트를 이용해 비동기 방식으로 자료를 전송 요청하는 것을 말한다.



Ajax

○ 비동기 관련 메서드 종류

종류	풀이
load()	외부 컨텐츠를 가져올 때 사용
\$.ajax()	데이터를 서버에 HTTP POST, GET 방식으로 전송이 가능하며. (X)HTML, XML, JSON, 텍스트 유형에 데이터를 요청할 수 있는 통합적인 메서드.
\$.post()	데이터 서버에 HTTP POST 방식으로 전송한 후 서버 측의 응답을 받을 때 사용
\$.get()	데이터 서버에 HTTP GET 방식으로 전송한 후 서버 측의 응답을 받을 때 사용
\$.getJSON	데이터 서버에 HTTP GET 방식으로 전송한 후 서버 측으로부터 JSON 형태로 응답요청을 받을 때 사용
\$.getScript	Ajax를 사용하여 외부 자바 스크립트를 불러올 때 사용
\$.ajaxStop()	비동기 방식으로 서버에 응답 요청이 완료되었을 때 함수에 실행문이 수행 됨.
\$.ajaxSuccess()	ajax 요청이 성공적으로 완료되면 함수에 실행문을 수행

Ajax

- load()
 - 외부의 컨텐츠를 요청하여 가져오고자 할 경우 사용
 - 로컬 영역에서 쓰이고자 할 컨텐츠를 가져올 때 자주 쓰임

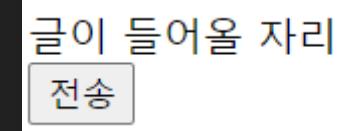
```
$(selector).load(url[,data[,function(response,status,xhr)]])
```

- url : 로드할 URL을 지정
- data : 요청과 함께 서버로 보낼 데이터를 지정한다.
- function(response,status,xhr) : 메서드가 완료될 때 실행할 콜백 함수를 지정한다.
 - response : 요청의 결과 데이터를 포함한다.
 - status : 요청의 상태("success", "notmodified", "error", "timeout", "parsererror")
 - xhr : XMLHttpRequest 객체를 포함한다.

Ajax

- load()

```
<div id="div1">글이 들어올 자리</div>
<button>전송</button>
<script>
    $(function(){
        $("button").click(function(){
            $("#div1").load("demo_test1.txt");
        });
    });
</script>
```



AJAX 로드 성공

txt 파일을 로드했습니다



Ajax

- getJSON()
 - AJAX HTTP GET 요청을 사용하여 JSON 데이터를 가져오는 데 사용

```
$(selector).getJSON(url[,data[,success(data,status,xhr)]])
```

- url : 로드할 URL을 지정
- data : 요청과 함께 서버로 보낼 데이터를 지정한다.
- function(response,status,xhr) : 메서드가 완료될 때 실행할 콜백 함수를 지정한다.
 - response : 요청의 결과 데이터를 포함한다.
 - status : 요청의 상태("success", "notmodified", "error", "timeout", "parsererror")
 - xhr : XMLHttpRequest 객체를 포함한다.

Ajax

- getJSON()

```
<div></div>
<button>요청</button>
<script>
$(function(){
    $("button").click(function(){
        $.getJSON("json_demo_text.json", function(result){
            $.each(result, function(i, field){
                $("div").append(field + " ");
            });
        });
    });
}</script>
```

Ajax

- **get()**

- HTTP GET 요청을 사용하여 서버에서 데이터를 로드한다.

```
$.get(URL[,data[,function(data,status,xhr)[,datatype]]])
```

- url : 로드할 URL을 지정
 - data : 요청과 함께 서버로 보낼 데이터를 지정한다.
 - function(response,status,xhr) : 메서드가 완료될 때 실행할 콜백 함수를 지정한다.
 - response : 요청의 결과 데이터를 포함한다.
 - status : 요청의 상태("success", "notmodified", "error", "timeout", "parsererror")
 - xhr : XMLHttpRequest 객체를 포함한다.

Ajax

- get()
 - dataType : 서버 응답에 필요한 데이터 유형을 지정한다. 가능한 유형은 아래와 같다
 - "xml" - XML 문서
 - "html" - HTML을 일반 텍스트로 사용
 - "text" – 일반 텍스트 문자열
 - "script" – 응답을 JavaScript로 실행하고 일반 텍스트로 반환한다.
 - "json" – 응답을 JSON으로 실행하고 JavaScript 객체를 반환한다.
 - "jsonp" – JSONP를 사용하여 JSON 블록에 로드한다.

Ajax

○ get() 예제 - 1

```
<button>전송받기</button>
<script>
  $(document).ready(function(){
    $("button").click(function(){
      $.get("http://localhost:2000/get_request1", function(data, status){
        alert("Data: " + data + "\nStatus: " + status);
      });
    });
  });
</script>
```

```
app.get("/get_request1", function(req,res){
  res.send("이것은 ajax get 메서드를 활용한 비동기 통신입니다.");
});
```

127.0.0.1:5500의 메시지

Data: 이것은 ajax get 메서드를 활용한 비동기 통신입니다.
Status: success

확인

Ajax

○ get() 예제 - 2

```
<button>전송받기</button>
<script>
    $(document).ready(function(){
        $("button").click(function(){
            $.get("http://localhost:2000/get_request2", function(data, status){
                $("div").html(`name : ${data.name}, age : ${data.age}`);
            });
        });
    });
</script>
```

```
app.get("/get_request2", function(req,res){
    const data = {
        name : "kim",
        age : 23
    }
    res.json(data);
});
```

name : kim, age : 23
전송받기

Ajax

○ get() 예제 - 3

```
<div></div>
<button>전송받기</button>
<script>
    $(document).ready(function(){
        $("button").click(function(){
            const obj = {
                name : "kgh",
                address : "수원시"
            }
            $.get("http://localhost:2000/get_request3", obj ,function(data, status){
                $("div").html(` ${data.name}, ${data.address}`);
            });
        });
    });
</script>
```

```
app.get("/get_request3", function(req,res){
    console.log(req.query);
    const data = {
        name : `내가 보낸 이름 : ${req.query.name}`,
        address : `내가 보낸 주소 : ${req.query.address}`
    }
    res.json(data);
});
```

내가 보낸 이름 : kgh, 내가 보낸 주소 : 수원시
전송받기

Ajax

- Post()

- HTTP POST 요청을 사용하여 서버에서 데이터를 로드한다.

```
$.post(URL[,data[,function(data,status,xhr)[,datatype]]])
```

- url : 로드할 URL을 지정

- data : 요청과 함께 서버로 보낼 데이터를 지정한다.

- function(response,status,xhr) : 메서드가 완료될 때 실행할 콜백 함수를 지정한다.

- response : 요청의 결과 데이터를 포함한다.

- status : 요청의 상태("success", "notmodified", "error", "timeout", "parsererror")

- xhr : XMLHttpRequest 객체를 포함한다.

Ajax

- post()
 - dataType : 서버 응답에 필요한 데이터 유형을 지정한다. 가능한 유형은 아래와 같다
 - "xml" - XML 문서
 - "html" - HTML을 일반 텍스트로 사용
 - "text" – 일반 텍스트 문자열
 - "script" – 응답을 JavaScript로 실행하고 일반 텍스트로 반환한다.
 - "json" – 응답을 JSON으로 실행하고 JavaScript 객체를 반환한다.
 - "jsonp" – JSONP를 사용하여 JSON 블록에 로드한다.

Ajax

o post() 예제 - 1

```
<button>클릭</button>
<script>
    $(function () {
        $("button").click(function () {
            $.post("http://localhost:2000/post_request1",
                function (data, status) {
                    alert("Data: " + data + "\nStatus: " + status);
                });
        });
    });
</script>
```

```
app.post("/post_request1", function(req,res){
    res.send("이것은 ajax post 메서드를 활용한 비동기 통신입니다.");
});
```

127.0.0.1:5500의 메시지

Data: 이것은 ajax post 메서드를 활용한 비동기 통신입니다.
Status: success

확인

Ajax

o post() 예제 - 2

```
<div></div>
<button>클릭</button>
<script>
    $(function () {
        $("button").click(function () {
            $.post("http://localhost:2000/post_request2",
                function (data, status) {
                    $("div").html(`name : ${data.name}, age : ${data.age}`);
                });
        });
    });
</script>
```

```
app.post("/post_request2", function(req,res){
    const data = {
        name : "kim",
        age : 23
    }
    res.json(data);
});
```

name : kim, age : 23
클릭

Ajax

○ post() 예제 - 3

```
<div></div>
<button>클릭</button>
<script>
    $(function () {
        $("button").click(function () {
            const obj = {
                "name": "kgh",
                "address": "수원시"
            }
            $.post("http://localhost:2000/post_request3", obj,
                function (data, status) {
                    console.log(data);
                    $("div").html(` ${data.name}, ${data.address}`);
                });
        });
    });
</script>
```

```
app.post("/post_request3", function(req,res){
    const data = {
        name : `보낸 이름 : ${req.body.name}`,
        address : `보낸 주소 : ${req.body.address}`
    }
    res.json(data);
});
```

보낸 이름 : kgh, 보낸 주소 : 수원시
클릭

Ajax

- ajax()
 - ajax() 메서드는 AJAX(비동기 HTTP) 요청을 수행하는 데 사용한다.
 - 모든 jQuery AJAX 메소드는 ajax() 메소드를 사용한다.
 - 이 방법은 다른 방법을 사용할 수 없는 요청에 주로 사용된다.

Ajax

- ajax()

```
$.ajax({  
    url:"자료를 전송할 페이지",  
    type:"전송방식(get or post)",  
    data:"전송할 데이터",  
    dataType:"요청한 데이터 타입(html, xml, json, text, jsonp)",  
    success:function(매개변수){  
        전송 성공시 실행문;  
    }  
});
```

Ajax

○ ajax()

Name	Value/Description
async	요청을 비동기식으로 처리할지 여부를 나타내는 Boolean 값. 기본값은 true.
beforeSend(xhr)	요청을 전송하기 전에 실행할 함수
cache	브라우저가 요청된 페이지를 캐시해야 하는지 여부를 나타내는 Boolean 값입니다. 기본값은 true
complete(xhr,status)	요청이 완료되면 실행할 기능(성공 및 오류 기능 후)
contentType	서버로 데이터를 보낼 때 사용되는 컨텐츠 유형. 기본값은 "application/x-www-form-urlencoded"입니다.
context	모든 AJAX 관련 콜백 함수에 대해 "this" 값을 지정한다.
data	서버로 보낼 데이터를 지정한다.
dataFilter(data,type)	XMLHttpRequest의 원시 응답 데이터를 처리하는 데 사용되는 함수
dataType	서버 응답에 필요한 데이터 유형
error(xhr,status,error)	요청이 실패할 경우 실행할 함수
global	요청에 대해 글로벌 AJAX 이벤트 핸들을 트리거할지 여부를 지정하는 Boolean 값. 기본값은 true
ifModified	마지막 요청 이후 응답이 변경된 경우에만 요청이 성공하는지 여부를 지정하는 Boolean 값. 기본값은 false.
jsonp	jsonp 요청에서 콜백 함수를 재정의하는 문자열
jsonpCallback	jsonp 요청의 콜백 함수 이름을 지정한다.
password	HTTP 액세스 인증 요청에 사용할 암호를 지정한다.
processData	요청과 함께 전송된 데이터를 쿼리 문자열로 변환할지 여부를 지정하는 부울 값. 기본값은 true.
scriptCharset	요청에 대한 문자 집합을 지정한다.
success(result,status,xhr)	요청이 성공할 때 실행할 함수
timeout	요청에 대한 로컬 시간 초과(밀리초)
traditional	기존의 파라미터 직렬화 스타일을 사용할지 여부를 지정하는 Boolean 값
type	요청 유형을 지정한다(GET 또는 POST).
url	요청을 보낼 URL을 지정한다. 기본값은 현재 페이지
username	HTTP 액세스 인증 요청에 사용할 사용자 이름을 지정한다.
xhr	XMLHttpRequest 객체를 생성하는 데 사용되는 함수

Ajax

- ajax() - json 타입 요청

```
<div></div>
<button>요청</button>
<script>
  $(function () {
    $("button").click(function () {
      $.ajax({
        type: "get",
        url: "json_demo_text.json",
        dataType: "json",
        success: function (result) {
          $.each(result, function (i, field) {
            $("div").append(field + " ");
          });
        }
      });
    });
  });
</script>
```

Ajax

- ajax() – get 요청

```
<button>전송받기</button>
<script>
$(document).ready(function () {
    $("button").click(function () {
        $.ajax({
            type: "get",
            url: "http://localhost:2000/get_request1",
            dataType: "text",
            success: function(data, status){
                alert("Data: " + data + "\nStatus: " + status);
            }
        });
    });
}</script>
```

Ajax

- ajax() - post 요청

```
<div></div>
<button>클릭</button>
<script>
    $(function () {
        $("button").click(function () {
            const obj = {
                "name": "kgh",
                "address": "수원시"
            }
            $.ajax({
                type: "post",
                url: "http://localhost:2000/post_request3",
                data : obj,
                dataType: "json",
                success: function (data, status) {
                    console.log(data);
                    $("div").html(` ${data.name}, ${data.address}`);
                }
            });
        });
    });
</script>
```