

jQuery

jquery.com에서 관련 자료를 우선 다운받으세요~

js폴더에 jQuery파일을 담으세요...(총 3개의 파일인거 아시죠? 물론 상황에 따라 더해지기도 줄기도 합니다.)

그리고 가져오세요 html문서로....(〈head〉 안에 link 다음줄에 넣으면 됩니다~)

〈script type="text/javascript" src="jQuery파일 경로 및 위치!!"〉〈/script〉 형식으로 작성하며,
jQuery는 별도로 작성된 〈script〉태그내에 작성하게 되어있습니다.

아래 내용을 각자 내용에 맞게 작성해보세요~

꽤 재미있는 현상을 맛볼 수 있습니다.

jQuery 파일을 가져왔다는 가정하에 아래내용을 작성해보세요.	
html 〈body〉〈/body〉 내에 작성	〈body〉 〈div id="red"〉〈/div〉 〈div id="blue"〉〈/div〉 〈div id="yellow"〉〈/div〉 〈div id="green"〉〈/div〉 〈/body〉
CSS 〈style〉〈/style〉 내에 작성	div { height:100px; width:100px; display: inline-block; } #red { background-color:#FF0000; } #blue { background-color:#0000FF; } #yellow { background-color:#E2BE22; } #green { background-color:#008800; }
jQuery 〈script〉〈/script〉 내에 작성 자바스크립트라이브러리이며, 이후 js라고 표기...	\$(document).ready(function() { \$('div').mouseenter(function() { \$(this).animate({ height: '+=10px' }); }); \$('div').mouseleave(function() { \$(this).animate({ height: '-=10px' }); }); \$('div').click(function() { \$(this).toggle(1000); }); });

이제 jQuery를 자세하게 익혀 봅시다~ 처음부터 차근차근 하나씩~~~

우선 아래 내용을 보고 이해를 해봅시다.

\$(무언가를 가져오기/선택하기).무언가하기(무언가 하는것에 대한 상세설명);

위의 사항이 기초 내용입니다.

그중에 첫부분 밑줄이 그어진 부분에 대해 알아 봅시다.

첫부분은 우리는 선택자 라고 합니다.(css와 유사한 형태를 띠고 있습니다.)

	jquery 선택자 이해하기
html	<div> </div>
css	div { height:100px; width:100px; background-color:#FF0000; border-radius:5px; }
js	<code>\$(document).ready(function() { \$().css({ width: '100px' , height: '100px' , backgroundColor : 'red'}) });</code>

위 항목은 div를 하나의 박스형태로 만들어놓은 자료입니다.

잘보시면 js영역에서 첫번째줄에 써있는_

`$(document).ready(function() {`

부분과, 마지막줄의

`});`

부분이 반복하고 있는것이 보이시나요? 네! jQuery 사용시의 필요한 부분입니다.

스타일에서는 <style></style>을 반드시 입력하고 내부에 작성해야하고,

자바스크립트는 <script></script>를 입력하고 내부에 작성해야 하는것처럼, jQuery는 script영역 내에

`$(document).ready(){ 작성 };` 를 입력하고 작성이라고 쓰여진 부분에 작업하는 것 입니다. 그러면 jQuery가 작동합니다.

단, jQuery는 자바스크립트 라이브러리이기 때문에 반드시 jQuery 파일을 가져와서 링크 처리하여야만 작동하나는 사실!!!!

jQuery 영역에서 어떠한 선택자를 표기해야 할까요?

기본은 css와 동일합니다.

`div{ width: 100px ; height : 100px ; background-color : red ; }` 와 같은 내용입니다.

단지 차이가 있다면 선택영역에서 div만 있는게 아니라

`$()` 를 감싸고 있다는 것입니다. 여기에 body에 있는 태그, class, id명에 따라서 . or # 의 표기를 더합니다.

우리가 알고있는 css 선택자와 동일합니다. 좀더 보겠습니다.

`nth-child()` , [href = “ ”] , ~ , + , > , , 와같은 nth, 속성, 동위, 자식, 복수선택자 등의 기능들도 포함합니다.

jQuery 기능에서 css 선택자기능은 대부분 가능합니다.

물론 사용의 편의를 위해서 jQuery 만의 선택자 기능도 포함되어져 있습니다.

대부분의 선택시에는

`$(" ")` 로 표기하여 해당하는 선택자를 사용하고 있습니다.

그러나 예외도 존재하는법. `$(this)`의 경우는 태그명도, class나 id도 아니기 때문에 ‘ ’ 를 표기하지 않습니다.

아래는 추가로 사용되는 선택자의 형태입니다.

선택자 추가기능	설명
:eq(index번호)	0부터 시작해서 선택하는 기능,nth-child(index번호)와 비슷한 기능이며 nth의 경우 1부터 메소드로도 가능합니다.!!
:even	짝수번째 선택하기 , nth-child(2n+1)과 비슷한기능, 0부터 시작한다는것을 염두
:odd	홀수번째 선택하기 , nth-child(2n)과 비슷한기능, 0부터 시작한다는것을 염두
:first	eq(0), first-child 와 동일한 기능
:last	last-child와 동일한기능
:contains('john')	특정단어가 포함된 요소를 찾기 원쪽에서는 john 이라는 글귀가 있는것을 찾습니다.
:checked	input에서 radio 또는 checkbox 형태에서 체크시에 발생하는 선택자입니다.

두번째 기능 무언가 하기.

\$(무언가를 가져오기/선택하기).무언가 하기 (무언가 하는것에 대한 상세설명);

jQuery에서 무언가를 선택 및 가져왔다면 이제 어떠한 행동을 진행시켜야합니다.

그것이 바로 무언가 하기입니다. 물론 올바르게 진행하려면 무언가하는 것에 대한 상세설명이 필요합니다만.

간단하게만 표현해 보겠습니다.

jQuery 시작부분 `$(document).ready(function() { 마지막줄의 })`; 표기는 이후 생략합니다!!!!!!

html	<div id="div1"> 클릭하세요 </div>
css	#div1{ width:100px; height:100px; background-color: lightblue; }
js	<code>\$("#div1").click(function() { alert("클릭 되었습니다."); });</code>

결과를 보면 아주 간단하게 해결 됩니다.

jQuery에서는 무언가 하는것에 대한 상세설명부분이 자연스럽게 따라오게되는데 이때! 추가 설명을 할때에는

function() {} 내용이 꼭 들어가게 됩니다.

실제로는 스크립트의 함수기능을 이용하는 것이기 때문, 상세내용은 생략하겠습니다.(당연히 들어가는 것 정도로 이해하세요!) 그렇다면 상세부분설명은 어디에 작성하는가!!!

`function() {}`에서 {}표기로 된곳에서 다시 작성하도록 되어져 있습니다.

위 내용에서 `alert()`는 메세지 창을 뜻 합니다.

아래내용으로 js영역을 다시 한번 작성해보세요.

이번에는 `$(document).ready(function() { })`; 영역을 제거하고 사용하세요!

```
var btn = document.getElementById("div1");
if(btn.addEventListener) {
    btn.addEventListener('click', doSomething, false);
} else if(btn.attachEvent) {
    btn.attachEvent('onclick', doSomething);
}
function doSomething() {
    alert('클릭되었다');
}
```

jQuery 기능을 javascript 기능으로 사용한 예시입니다..

실제로는 javascript로 사용해야하는 것을 간단하게 우리는 사용할 수 있습니다!!!! 아주 편하게!!!!

물론 javascript기능을 알아야하고 조금씩이라도 익혀야 하겠지만 앞서 익혔던 이해할 수 없는 문법들만으로도 우리는 충분히 익혔다고 생각합니다 그저 우리는 jQuery를 익히기 위해 형식만 필요할 뿐이었기에....

위의 js 내용을 간단하게 설명하자면 id값 div1을 호출합니다.

매번쓰기에는 길기때문에 변수 btn을 선언하고 이름을 축약하여 저장합니다.

이제 해당하는 div를 선택했을때 브라우저마다 javascript 기능이 조금씩 다르기때문에 조건문을 사용하여 선택사항을 입력합니다.

ie의 경우는 attachEvent, 기타브라우저는 addEventListener로 표기하고

해당하는 어떠한 이벤트기능을 했을때(위는 click했을때), doSomething 기능을 실행하라 라는 뜻입니다.

doSomething의 경우는 아래 별도로 function 기능으로 함수처리되어 jQuery와같이 메세지창을 띄우게 되어있습니다. javascript(자바스크립트, 이하 js)를 굳이 설명드린 이유는 jQuery코드가 짧고 직관적이다! 라는것을 설명함과 동시에 변수에 대해 설명하고 싶어서 입니다. 여러내용이 포함되어 있을 경우에는 별도의 변수를 만들어 사용 할 수 있게 해 놓았습니다. 좀더 간단하게 사용하기 위해서! 라고 익히면 되지 않을까요?

우리는 scss를 통해서 변수가 어떤 기능을 하는지 익혀놓았습니다.

길게 설명하지 않아도 되겠죠?^^

어려우신 분들이 계시다면 간단하게 설명드리겠습니다.

매번 id값을 호출하기위해 documet.getElementById("div1"); 을 쓰기에는 불편합니다. 그만큼 길어서겠죠?

이를 간단하게 정의 내릴 수 있습니다. 그것이 변수입니다.

이를태면 dBox 라는 이름이 documet.getElementById("div1"); 라는것과 같다면 여러분은 어떤것을 작성하시겠습니까?

전 당연히 dBox입니다. 쓰기도 기억하기에도 편하기 때문입니다.

이때 dBox를 변수라고 합니다. 하지만 미리만들어 놓은 값이 아니기때문에 존재할 수 없습니다. 컴퓨터가 인지하기 어렵다는것이 문제입니다.

이에 저는 컴퓨터에게 이렇게 말해줍니다.

나 dBox라는 것을 만들거야! dBox는 documet.getElementById("div1");라는 값이야! 라고 말입니다.

나 dbox라는 것을 만들거야! 라고하는 부분은 변수선언이라고 합니다.

var dbox; 라고 표기하는것입니다.

그리고 dBox가 documet.getElementById("div1"); 이다라고 설정하셔야 하는데 이때. = 기호를 표기해줍니다.

var dBox

```
dbox = documet.getElementById("div1");
```

이렇게 말이죠! 하지만 두번쓰는 이것도 불편하기에 한주로 정의합니다.

var dBox = documet.getElementById("div1"); 간단명료하게 끝납니다.

이제는 dBox라고만 표기하면 문제 없습니다.

변수명을 만들때에는 아래와 같은 규칙을 지켜야 합니다.(이미 여러분은 변수선언방법을 알고 있습니다.)

1.	영어로 이름을 작성합니다.
2.	띄어쓰기가 없습니다 단어를 구분시에는 _ 또는 새로운 단어의 첫글자를 대문자로 작성합니다. 대.소 문자를 정확하게 가리니 파악하세요.
3.	숫자, 특수문자가 맨앞글자에 오는것 올바르지 않습니다 (예외, \$ _ 두글자는 포함합니다.)
4.	특수문자는 3번에서 설명드린 것 외에는 불가능입니다.
5.	예약어를 사용하지 않습니다.(미리 만들어놓은 태그명, 속성, 스크립트 용어 등은 사용하지 않습니다.)

변수는 상황에 따라 변한다는 사실 기억하세요.~~~ (상세내용은 추후에 또...)

예약어 및 변수명을 사용하시기엔 무리가 있는 내용들을 정리해 보았습니다.

예약어					
break	delete	function	return	typeo	try
Case	do	if	swich	var	null
catch	else	in	this	void	for
continue	false	instanceof	throw	while	default
debugger	finally	new	true	with	

미래 변수 예약어					
abstract	double	goto	native	static	volatile
boolean	enum	implements	package	super	short
byte	export	import	private	synchronized	long
char	extends	int	protected	throws	float
class	final	interface	public	transient	const

기타 식별자					
encodeURI	Infinity	Object	String	decodeURI	unescape
Array	Error	isFinite	parseFloat	SyntaxError	ReferenceError
Boolean	escape	isNaN	parseInt	TypeError	NaN
Date	eval	Math	RangeError	undefined	EvalError

\$(무언가를 가져오기/선택하기).무언가 하기(무언가 하는것에 대한 상세설명);

이제 우리는 무언가하기에 대한 다양한 용어 및 그에따른 기능들을 익히겠습니다.

정확하게 말하면 무언가 하기에대한 부분은 속성인 프로퍼티, 메소드 정도로 볼 수 있는데,
jQuery에서는 대부분이 메소드로 사용되고 있습니다.

메소드란, 어떠한 기능을 하기위한 것들입니다. 위에써있는 무언가 하기 기능 자체가 메소드라 보시면 됩니다.

메소드에 대한 기능에 대하여 익혀 보겠습니다.

메소드1(이벤트 타입) \$(‘선택자’).on(‘이벤트 타입’, function(){ }) ;

이벤트 타입	해당시점
mouseover	마우스 포인터를 올려놓았을때
mouseout	마우스포인터를 벗어날때
mousedown	마우스 포인터를 올려놓은 상태에서 해당요소를 클릭할때(누르고 있을때)
mouseup	마우스 포인터를 올려놓은 상태에서 해당요소를 클릭했다가 떨때(마우스버튼을 누르고있다가 띄는 순간)
click	요소를 클릭했을때
dblclick	요소를 더블클릭했을때
keydown	요소에 초첨을 맞춘 상태에서 키보드의 키를 눌렀을때
keyup	요소에 초첨을 맞춘 상태에서 키보드의 키를 떨때
focus	요소에 포커스가 일치했을때(a, input 등의 포커스가 있는요소. 즉 활성화되었을때)
blur	요소가 포커스를 잃어버렸을때(a, input 등의 요소에서 빠져나갔을때)
change	입력 내용이 변경되었을 때(textarea, input, select요소 등)
resize	요소의 크기를 다시 설정했을 때
scroll	요소를 스크롤했을 때

위 기능은 마우스 또는 키보드 또는 기타 특정기능을 사용할 때의 시점을 이용한 이벤트 기능입니다.

이를 **이벤트타입** 이라고 하며 이때 **매개변수는 이벤트 핸들러를 전달합니다.**

과거 매개변수에 대한 기능은 특별하게 없이 사용되어져 왔으나, 최근 들어 매개변수의 기능이 대두되고 있습니다.

매개변수라고해도 특별하게 어려운 것이 아니라 무언가 하기에 대한 메서드를

.on() 이라고 입력하면 됩니다.

그리고 이벤트타입의 작성되는 곳은 .on()에서 () 영역에 해당합니다.

.on(‘이벤트 타입’, ‘이벤트 핸들러’) 으로 작성합니다.

여기서 이벤트타입에 따른 기능을 하도록 처리해야하는데 우리는 그러한 기능을 실현하기 위해

.on(‘이벤트 타입’, function() {})으로 우선 입력합니다.(이벤트 핸들러 기능을 구현하기 위해 함수 function 을 입력해서 처리)

실제 이벤트 핸들러의 기능들은 과거 이벤트 타입으로 처리했던 부분들이라 생각하면 조금 쉽게 다가갈 수 있습니다.

이 부분은 책마다 조금씩 다르게 표현하고 있습니다. (아마도 번역에 대한 이해문제로 생각되는데, 아님 제이해 실력부족 ㅜㅜ 여러분은 너무 어렵게 다가가지 말고 “무언가하기 위한 기능이다” 라고 생각하세요.)

이제 function() {} 에 해당하는 부분을 작성해야 합니다.

물론 어떠한 이벤트 기능을 처리하기 전에 바로 하기 위해서라면 바로 처리해도 문제가 없겠지만, 우리는

무언가를 눌렀을 때, 발생시키건, 마우스가 올라갔을때 일어나는 일들을 구현하기 위해 있는 기능이니 당연히
어떠한 이벤트를 실행 했을때! 그와 관련된 기능이 작동하도록 만드는 것입니다.

그래서 function으로 만들어서 그 내부에 작동 가능하도록 처리합니다.

메소드2(작동하게하는것들) \$(‘선택자’).메소드({ });

범주	메소드	작동내용
스타일	css	스타일 속성변경하기
	width/ height	가로, 세로 속성값을 이용하여 변경
	show/ hide	보이기/ 숨기기(display:block / none과 유사)
애니메이션	fadeIn/ fadeOut	fade기능으로 나타나거나 사라지기
	slideDown/ slideUp	슬라이드 기능으로 나타나거나 사라지기
	animate	애니메이션기능
내용/속성변경	text/ html	글씨 및 html 기능 변경하기
	attr/ removeAttr	속성기능 변경하기/ 속성기능 삭제하기
	val	form 기능의 val 변경하기
	addClass/ removeClass/ toggleClass	class기능 추가/ 제거
새로운 요소만들기	append/ appendTo	해당 요소들 내부의 뒤에 추가하기
	prepend/ prependTo	해당 요소들 내부의 앞에 추가하기
	remove	지정요소를 삭제
	empty	지정요소의 내용을 비움
	eq()	nth-child와 같은기능(0)번부터 존재
각각 처리하기	\$(this)	선택한 그 자신
	each	지정된 요소 각각
기타기능	stop	반복 애니메이션 효과기능처리하기
	find	하위 내용 찾기
	next	같은 레벨 뒤(바로뒤)
	prev	같은 레벨 앞(바로앞)

작업의 결과물이 끝났을 경우 실행의 끝을 알리는 표기인 ; 를 꼭 입력해야하며,

단위가 px인경우에는 단위를 기입하지 않아도 무관합니다.

단, px의 단위를 입력할 경우에는 ‘ ’ or “ ” 로 감싸주어야 가능합니다.

하나이상의 속성을 변경할 경우에는 ()안에 {} 를 표기하며, ‘ ’ 의 표기를 생략해도 되며, 속성, 속성값 으로 구분한것을 :로 구분합니다.

단수 기능으로 사용시	복수기능으로 사용시
<pre>\$('.box').css('width', 500); \$('.box').css('height',500); \$('.box').css('background-color', 'red');</pre>	<pre>\$('.box').css({ width : 500 , height : 500, backgroundColor : 'red' });</pre>

css에서의 - 으로 구분되던 속성의 경우는 - 을 제거하고 뒤에오는 단어의 첫글자를 대문자로 표기하며,

숫자는 ‘ ’ 를 제거하여 표기, 문자의 경우는 ‘ ’ 를 표기하여 내부에 작업 하도록 한다.

animate의 경우는 top, left 같은 기능을 사용할때에는 position을 사용하는 것이 자연스럽게 진행됩니다.

아래내용은 메소드 기능들을 간략하게 정리해 놓았습니다. 참고하세요!!

- 웹 페이지의 로딩이 완료하면ダイアル로그를 표시

```
$(function() {  
    alert('페이지 로딩완료');  
})
```

- id="foo" 인 요소를 표시

```
$('#foo').show();
```

- 클래스 foo가 지정된 요소를 표시

```
$('.foo').show();
```

- 모든 div 요소(태그)를 표시

```
$('.div').show();
```

- id='box'인 요소를 취득, 변수 box에 적용

```
var box = $('#box');
```

- div1 안에 있는 클래스 foo가 지정된 요소를 표시

```
div1.find('.foo').show();
```

- id="foo"인 요소의 스타일, font-size를 20픽셀로 변경

```
$('#foo').css('font-size','20px');
```

- id="foo"인 요소의 스타일, font-size의 값을 변수 size에 보관

```
var size = $('#foo').css('font-size');
```

- id="foo"인 요소를 표시시키기

```
$('#foo').show();
```

- id="foo"인 요소를 감추기

```
$('#foo').hide();
```

- id="foo"인 요소의 폭을 400픽셀로 하기

```
$('#foo').width(400);
```

- id="foo"인 요소의 높이를 400픽셀로 하기

```
$('#foo').height(400);
```

- id="foo"인 요소를 페이드인

```
$('#foo').fadeIn();
```

- id="foo"인 요소를 페이드아웃

```
$('#foo').fadeOut();
```

- id="foo"인 요소를 1초로 페이드인

```
$('#foo').fadeIn(1000);
```

- id="foo"인 요소를 1초로 페이드인한 후에ダイ얼로그를 표시

```
$('#foo').fadeIn(1000,function(){  
    alert('페이드인 완료');  
});
```

- id="foo"인 요소를 슬라이드 다운

```
$('#foo').slideDown();
```

- id="foo"인 요소를 슬라이드 업

```
$('#foo').slideUp();
```

- id="foo"인 요소를 top:200px, left:400px에 애니메이션

```
$('#foo').animate({  
    top: 200,  
    left: 400  
});
```

- id="foo"인 요소를 top:200px, left:400px에 1초로 애니메이션한 후ダイ얼로그를 표시

```
$('#foo').animate({  
    top: 200,  
    left: 400  
}, 1000, function() {  
    alert('애니메이션 완료');  
});
```

- id="foo"인 요소의 내용을 'jQuery책'으로 변경

```
$('#foo').text('jQuery책');
```

- id="foo"인 요소의 내용을 변수 text에 보관

```
var text = $('#foo').text();
```

- id="foo"인 요소의 내용을 '<p>jQuery책</p>'으로 변경

```
$('#foo').html('<p>jQuery책</p>');
```

- id="foo"인 요소의 내용을 html에 보관

```
var html = $('#foo').html();
```

- id="foo"인 요소의 내용을 비움

```
$('#foo').empty();
```

- id="foo'인 a 요소의 target 속성값에 _blank를 지정

```
$('#a#foo').attr('target', '_blank');
```

- id="foo"인 요소의 target 속성값을 변수 target에 보관

```
var target = $('#a#foo').attr('target');
```

- id="foo"인 input 요소의 입력 값을 'jQuery책'으로 변경

```
$('#input#foo').val('jQuery책');
```

- id="foo"인 input 요소에 입력되어 있는 값을 변수 value에 보관

```
var value = $('#input#foo').val();
```

- id="foo"인 요소의 클래스에 bar를 추가

```
$('#foo').addClass('bar');
```

- id="foo"인 요소의 클래스로부터 bar를 삭제

```
$('#foo').removeClass('bar');
```

- id="bar"인 요소를 id="foo"인 요소의 안으로 이동

```
$('#foo').append($('#bar'));
$('#bar').appendTo($('#foo'));
```

- p요소를 만들어 id="foo"인 요소의 안에 추가하기

```
$(‘

새로운요소</p>’).appendTo($('#foo'));


```

- id="foo"인 요소를 삭제

```
$('#foo').remove();
```

- id="foo"인 요소가 클릭되면ダイアル로그를 표시

```
$('#foo').click(function() {
    alert('클릭되었다');
});
```

- id="foo"인 요소 위에 마우스 포인터가 있으면ダイアル로그를 표시

```
$('#foo').mouseenter(function() {
    alert('마우스 포인터가 위에 있다');
});
```

- id="foo"인 요소 위로부터 마우스 포인터가 벗어나면ダイアル로그를 표시

```
$('#foo').mouseleave(function() {
    alert('마우스 포인터가 벗어났다');
});
```

- 상기 2개를 한꺼번에

```
$('#foo').hover(function() {
    alert('마우스 포인터가 위에 있다');
}, function() {
    alert('마우스 포인터가 벗어났다');
});
```

- id="foo"인 input요소에 포커스를 얻으면ダイアル로그를 표시

```
$('#input#foo').focus(function() {
    alert('포커스를 얻었다');
});
```

- id="foo"인 input요소로부터 포커스가 벗어나면ダイアル로그를 표시

```
$('#input#foo').blur(function() {
    alert('포커스가 벗어났다');
});
```

- 클래스 foo가 지정된 요소가 클릭되면 내용물의 텍스트를 변경

```
$('.foo').click(function() {
    $(this).text('클릭되었다');
});
```

- 클래스 foo가 지정된 요소 각각에 대해 내용물의 텍스트를ダイアル로그 표시

```
$('.foo').each(function() {
    var text = $(this).text();
    alert(text);
})
```

위의 기능들은 많이 사용하는 기능들입니다.

정리된 기능을 보면서 한번씩 꼭 보세요 잊을만하면 한번씩이라도 봐야 조금은 눈에 익겠죠?

물론!!! 자주 사용해서 자다가도 쓸 수 있으면 완전 금상첨화겠지만요.

하지만, 모두 완전하게 익히기엔 너무 많습니다.(사실 저도 완전히 못익힐 정도로 많습니다. ㅜㅜ)

간단하지만 자주 사용하는 기능들을 익혀보아요~(문제도 포함되어 있습니다.!!!)

1. 나타나기/ 숨기기의 기능

영역	내용
html	<pre><body> <button class="hidden"> 숨기기 </button> <button class="view"> 나타나기 </button> <button class="dual"> 보이기/숨기기 </button> <li id="red">css 수정하기 <li id="blue">들어있는 내용바꾸기 <li id="yellow"> <li id="green"> <div class="gallery"></div> </body></pre>
css	<pre>button{width:100px; height:50px; background-color:#ccc; padding:5px; } .insert{border-top:3px solid hsla(240, 50%, 70%, 0.5); border-bottom:5px solid rgba(50, 50, 200, 0.5); padding:50px; background-color:#f1f1f1;} li{width:100px; height:50px; line-height:50px;} #red{background-color:#ffa;} #blud{background-color:#ccf;} .gallery{width:500px; height:500px; border:2px solid #ccc; margin:10px;}</pre>
jQuery	<pre>\$('.hidden').on('click', function() { \$('#ul').hide(); // 1번 }); \$('.show').on('click', function() { \$('#ul').show(); // 1번 }); \$('#red').on("",function() { \$(this).css('background-color', '#f33'); // 2번 }); \$('#blue').on('click', function(){ \$(this).text('변경된 내용을 입력해서 결과값을 확인하세요'); // 3번 }); \$('#yellow').on('mouseover', function(){ //4번 \$(this).find('img').attr("src", "change.jpg") }); \$('a').on('click', function(){ \$('.gallery').attr('src', /* 5번 */ .attr("href")) return false; });</pre>
문제	<ol style="list-style-type: none">좀 더 이쁘게 숨기거나 나타나게 하는 것은 없을까요? .dual 버튼을 이용하며 나타나거나 숨기기를 모두 가능하게 만들어 보세요.선택영역에 무엇을 입력하여 #red의 배경색상이 변경될까요? 하나뿐이아닌 여러기능을 사용할 수 있도록 만들어 보세요.메서드 키체인을 이용하여 css를 만들어보세요.무엇이 문제일까요? 마우스가 벗어날때에는 원래의 이미지를 바꾸려면 어떻게 해야할까요?무엇을 입력해야 a태그를 클릭했을때 div에 원하는 이미지로 바뀔까요?ul에 별도의 css기능을 추가해 보세요. (class="insert")