

# 문서객체 설정

# 기본설정





## each메소드

- 배열을 관리하는 for in문과 비슷한 메소드로 객체나 배열의 요소를 검사하는 메소드
- index : 객체, 배열의 순번
- item : 값을 보관하는 변수(객체) ' . '으로 접근

메소드	내용
<code>\$.each(배열이름, function(index, item) { })</code>	지정한 배열을 0번 부터 자동으로 불러와 순번을 index, 값을 item에 넣는 메소드
<code>\$( 's' ).each(function(index,item) { })</code>	선택자로 선택한 요소를 index순번으로 item에 요소 값을 수정시 사용하는 메소드





# jQuery배열관리

```
$.each(배열(객체)이름, function(index, item){  
    배열(객체) 차례로 처리할 때 사용  
})  
);
```

```
$('선택자').each(function(index, item){  
    선택된 요소들을 차례로 조작할때 사용  
})  
);
```





## extend 메소드

- jQuery에서 객체의 내용을 추가하는 메소드

```
$.extend(객체이름, {  
    추가할 객체내용,  
    추가할 객체내용2,  
    .....  
});
```





## noConflict메소드

- jQuery변수를 \$외에 다른것으로 지정하는 메소드
- 다른 javascript플러그인을 사용시 특수문자 \$변수를 중복되어 발생하는 에러방지를 위해 사용하기도 함

```
var 변수명 = $.noConflict();
```

변수명을 \$대신 사용



# 클래스 / 속성 설정





# jQuery class 추가

## addClass()

- 선택자에 의해 선택된 요소에 클래스를 추가하는 메소드

1. `$('.s').addClass("클래스이름");`

//클래스를 여러 개 추가는 띄어쓰기로 구분

2. `$('.s').addClass(function(n){`

인자는 선택된 요소들의 번호;

`return 값; // 리턴된 값을 클래스로 추가`

`});`







## removeClass()

- 선택자에 의해 선택된 요소에 클래스를 삭제하는 메소드

1. `$('.s').removeClass("클래스이름");`

//클래스를 여러 개 추가는 띄어쓰기로 구분

2. `$('.s').removeClass(function(n){`

인자는 선택된 요소들의 번호;

`return 값; //리턴된 값과 일치하는 클래스삭제`

`});`





# jQuery class 추가

## toggleClass()

- 선택된 요소에 class가 있으면 삭제하고 없으면 추가하는 메소드

1. `$( 's' ).toggleClass( "클래스이름" );`

//클래스를 여러 개 추가는 띄어쓰기로 구분

2. `$( 's' ).toggleClass( '클래스명', [function(index,curclass){  
}], [true|false]);`

☞ true : 추가만, false : 삭제만





# jQuery 속성 추가

## attr()

- 선택자에 의해 선택된 요소에 속성을 속성값확인, 속성값 변경하는 메소드

1. \$('s').attr("속성명");

//속성명의 속성값을 리턴

2. \$('s').attr("속성명","속성값");

//선택자의 속성과 속성값을 추가





## jQuery 속성 추가

3. `$( 's' ).attr( "속성명", function( index, value ) { } );`  
//index : 선택된 요소 set의 index값, value : 현재 값
  
4. `$( 's' ).attr( { 속성명: "속성값", 속성명: "속성값" ..... } );`  
//다중으로 속성값 설정시 사용, 객체사용 가능





# jQuery 속성 추가

## removeAttr()

- 선택자에 의해 선택된 요소에 속성을 제거하는 메소드

1. `$( 's' ).removeAttr( "속성명" );`

//속성명의 속성을 제거, 여러개일 경우 띄어쓰기 사용





# jQuery 속성 추가

## css()

- 선택자에 의해 선택된 요소의 속성값을 가져오거나 설정하는 메소드
  - ☞ 단축속성설정(예 : border, background)제대로 지원 않고 브라우저별 상이

1. `$('#s').css("속성명");`  
//속성명의 속성값을 리턴
2. `$('#s').css("속성명","속성값");`  
//속성명의 속성값을 설정





## jQuery 속성 추가

3. `$('#s').css("속성명", function(index, 현재값) { } );`

//함수에서 리턴되는 값으로 속성을 설정

4. `$('#s').css({오브젝트});`

//오브젝트에서 설정한 다중 속성을 설정



# content 설정







# jQuery content 추가

## html()

- 선택된 요소의 content영역(innerHTML/텍스트노드)을 리턴 / 설정하는 메소드, html태그를 태그로 인식
- 리턴시 첫번째 요소 텍스트노드 리턴 / 전체 요소 리턴시 함수 사용

1. \$('s').html();

//선택된 요소의 첫번째 텍스트 노드를 리턴

2. \$('s').html("text노드 내용");

//선택된 요소의 text노드 내용을 대입

3. \$('s').html(function(index, 현재값) { } );





# jQuery content 추가

## text()

- 선택된 요소의 content영역(innerHTML/텍스트노드)을 리턴 / 설정하는 메소드, html태그를 문자로 인식
- 리턴시 전체 요소 텍스트노드를 리턴

1. `$( 's' ).text();`

//선택된 요소의 모든 텍스트 노드를 리턴

2. `$( 's' ).text("text노드 내용");`

//선택된 요소의 text노드 내용을 대입

3. `$( 's' ).text(function(index, 현재값) { } );`



# 객체 생성 및 제거





## 문서객체 생성방법

구분	내용
html 태그	직접 html구문을 문자열로 작성 예) '<p>직접 생성하는 방식</p>'
jQuery 방식	jQuery 객체 이용 구문작성 \$(생성태그).css()   html()   text()   attr()사용하여 생성
DOM 방식	document.createElement('태그명').innerHTML('텍스트노드내용'); document.createElement('태그명').setAttribute('속성명','속성값');





## 객체 생성된 객체 추가/이동

- html내부에 생성한 요소를 추가하는 메소드

메소드	내용
<code>\$( 's' ).append(생성구문)</code> <code>\$( 's' ).append(function([n],[html]){ })</code>	생성된 요소를 선택자 뒤에 추가. 함수의 리턴된 값을 선택자 뒤에 추가. ☞ n : 선택된 요소set의 인덱스 번호 ☞ html : 현재 선택된 요소
<code>\$( 's' ).after(생성구문)</code> <code>\$( 's' ). after(function([n],[html]){ })</code>	
<code>\$( '생성구문 태그명' ).appendTo('s')</code>	선택자에 생성된 요소나 태그를 뒤에 추가
<code>\$( '생성구문 태그명' ).insertAfter('s')</code>	☞ 만일 추가되는 태그명이 문서에 있으면 위치를 변경시킴(이동)





# jQuery 객체 생성/이동

메소드	내용
<code>\$('#s').prepend(생성구문)</code> <code>\$('#s').prepend(function([n],[html]){ })</code>	생성된 요소를 선택자 앞에 추가. 함수의 리턴된 값을 선택자 앞에 추가. ☞ n : 선택된 요소set의 인덱스 번호 ☞ html : 현재 선택된 요소
<code>\$('#s').before(생성구문)</code> <code>\$('#s').before(function([n],[html]){ })</code>	
<code>\$('#생성구문 태그명').prependTo('s')</code>	선택자에 생성된 요소나 태그를 앞에 추가 ☞ 만일 추가되는 태그명이 문서에 있으면 위치를 변경시킴(이동)
<code>\$('#생성구문 태그명').insertBefore('s')</code>	





## clone 메소드

- html내부에 있는 요소를 복사하는 메소드

```
$('#태그명').clone([true | false]).appendTo("추가될위치");
```

```
// 태그명의 태그를 복사하여 추가될 위치에 넣기
```

```
// 인자값 이벤트 핸들러까지 복사할지 여부까지
```





## document에 있는 객체 제거

- html내부에 요소객체를 제거하는 메소드

메소드	내용
<code>\$( 's[,s...]').remove(['s']);</code>	문서 내에 선택자를 제거 후 삭제된 요소를 리턴 ☞ 인자값을 기준으로 삭제가능 ☞ 이벤트핸들러, 데이터까지 모두 삭제
<code>\$( 's').detach();</code>	문서 내에 선택자를 제거 후 삭제된 요소를 리턴 ☞ 이벤트핸들러, 데이터는 삭제 안함 ☞ 값을 리턴받아 나중에 활용 가능
<code>\$( 's').empty();</code>	문서 내에 선택자를 제거 후 삭제된 요소를 리턴 ☞ 선택자 자신과 속성은 삭제 안하고 내용만 삭제

