



문서객체 선택과 탐색



- **document(문서객체)**를 선택하는 다양방법과 지원되는 함수의 기능을 파악한다.
 - 문서객체를 통해서 탐색되는 메서드들을 파악하고, 활용할 수 있다.
 - 문서객체에서 포함되거나, 수정, 삭제 처리되는 메서드들을 파악하고 활용할 수 있다.
-



기본 **filter** 처리 메서드:

- **filter()**

- 문서 객체를 필터링한다.
- **\$(selector).filter(selector);**
 - 해당 객체 검색
 - **\$("#h1").filter(":odd").css(처리내용);**
- **\$(selector).filter(function(index){**
 - 해당 객체를 검색했을 때, 처리할 기능 **list**
 - **return index%3==0;**
- **}).css("처리할 내용");**



확인예제 :

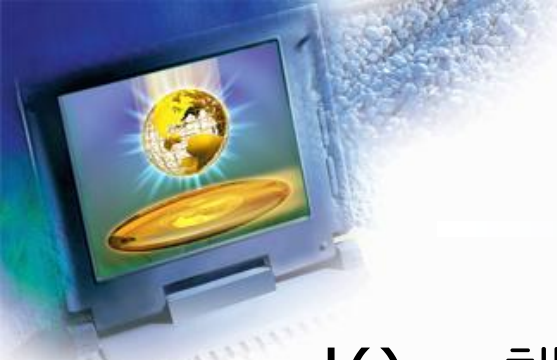
- filter를 활용하여.. 아래 화면 같이 구현하세요

판매 품목
오렌지(2칸씩)
사과(2칸씩)
바나나(3 칸씩) - 판매완료(글자추가)
딸기
수박



기본예제 :

```
$(document).ready(function){  
  // 데이터 입력처리..  
  var flist=["오렌지","사과","바나나","딸기","수박"];  
  var show= "<tr align='center'><td>판매품목</td></tr>";  
  for(var idx=0;idx<flist.length;idx++){  
    show+="<tr align='center'><td>"+flist[idx]+"</td></tr>";  
  }  
  $("table").html(show);  
  
  // title 처리 부분..  
  $("tr").filter(function(index){  
    return index==0;  
  }).css({background:"blue",color:"white"});
```



문서 탐색 filter

- `end()` : 체이닝(속성을 연결) 기능을 종료시키고, 다른 속성을 처리할 때, 활용된다.
 - 위치 지정 선택
 - `document`객체.`eq(index번호)` : 특정 `index`위치에 있는 `document` 객체 선택.
 - ex) `$("#h1").eq(5).html("데이터 변경");`
 - `document`객체.`first()` : 해당 객체들 중에 첫번째
 - `document`객체.`last()` : 해당 객체들 중에 마지막
 - `index` 번호 :
 - 양수 : 0 부터 앞에서 순서
 - 음수 : 뒤에서 부터 순서
-



확인예제 :

```
var point=0;
$(document).ready(function(){
    $("td").first().css("background","yellow");
    $("td").last().css("background","pink");
    // eq(index)
    setInterval(function(){
        // 초기화..
        $("td").html("");
        var ranIdx=parseInt(Math.random()*9);
        $("td").eq( ranIdx).html("두더지 출현");
        /* 시간 처리에 대한 부분으로 event가 반복적으로 처리..*/


    }, 1000); // 단계별로 게임 level upgrade
```



확인예제 :

```
$("#td").click(function(){  
    // 현재 문자열이 "두더지 출현"이라는 문자열이면  
    // 점수를 카운트up하게 처리..  
    // $(this) : td의 배열가운데 현재 클릭한 td를 지칭  
    // 이미지 $(this).html()!=""  
    if($(this).html()=="두더지 출현"){  
        point++;  
        $("#h1").html("점수:"+point)  
    }  
});
```


확인예제 :



```
<body>
<h1 align="center"> 점수 : 0</h1>
<table align="center" width="300pt" height="300pt" border>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
</table>
```



document 객체 처리 :

- 문서 객체 추가..
 - **add(추가할 객체 지정)** : 객체를 추가 처리하는 기능을 할 수 있다..

```
$(document).ready(function){  
// add(객체) : 객체를 지정해서 해당 속성을 변경 처리..  
$("h1").css("background","gray").add("h2").css("float","left");  
});
```

- **is("객체")** : 문서 객체의 특징을 판별할 때, 활용하는 메서드.. **boolean**값 **return**, 주로 조건문에서 활용 된다.
 - 객체의 값 : .클래스명



확인예제 :

```
$(document).ready(function(){  
    $("td").each(function(){  
        // $(this) : td의 단위 객체  
        // td 중에 class가 ckcls인것은..  
        if( $(this).is(".ckcls") ){  
            $(this).css("background","orange"  
            );  
        }  
    });  
});
```



확인예제 :


```
<table width="300" align="center" border>
<tr><th>NO</th><th>이름</th><th>포인트
점수</th></tr>
<tr><td>1</td><td class="ckcls">홍길동</td>
<td >3000점</td></tr>
<tr><td>2</td><td
class="ckcls">김길동</td><td >5000 점</td></tr>
<tr><td>3</td><td >신길동</td><td
class="ckcls">7000 점</td></tr>
</table>
```



xml과 find() :

- xml : extensible markup language
 - 계층구조의 tag값과 각 tag의 속성값으로 구성
 - ex)
 - <products>
 - <product id="prod01">사과</product>
 - <product id="prod02">바나나</product>
 - <product id="prod03">딸기</product>
 - </products>
- 위와 같은 xml 데이터 형식을 `$.parseXML(xmldata)`를 통해 객체단위로 변수로 설정이 가능하다.
 - 객체.**find**("찾고자하는 xml계층구조 하위 객체")

기본예제 :




```
var xml="";
xml += "<friends>";
xml += "    <friend>";
xml += "        <name>연희경</name><age>29</age><loc>서울  
강남</loc>";
xml += "    </friend>";
xml += "    <friend>";
xml += "        <name>윤영희</name><age>32</age><loc>서울  
잠실</loc>";
xml += "    </friend>";
xml += "    <friend>";
xml += "        <name>이희우</name><age>25</age><loc>인천  
계양</loc>";
xml += "    </friend>";
xml += "</friends>";
// {"name":"연희경","age":"29"}
var json="[{"name":"연희경","age":"29","loc":"서울  
강남"},  
{"name":"윤영희","age":"32","loc":"서울 잠실"} ]";
json += "    {"name":"윤영희","age":"32","loc":"서울 잠실"} ]";
```



기본예제 :

```
// XML문자데이터 ==> 객체데이터 변환 $.parseXML("문자열")
// JSON문자데이터 ==> 객체데이터 변화 $.parseJSON("문자열")
var xmlObj = $.parseXML(xml);
// $("객체").find("해당계층구조이름")
//alert("friend객체의 갯수:"+$(xmlObj).find("friend").length);
$(xmlObj).find("friend").each( function( index ){
  // $(this) : friend의 단위 객체..
  // text() 해당 내용 안에 있는 문자열..<name>문자열</name> ==>
  text()
  var show="";
  show+="<div>";
  show+="<h1>"+ $(this).find("name").text()+"</h1>";
  show+="<p>"+$(this).find("age").text()+", "
  +$(this).find("loc").text()+"</p>";
  show+="</div>";
  document.body.innerHTML+=show;
});
```

기본예제 :



```
var jsonObj = $.parseJSON(json);  
$(jsonObj).each(function(index,item){  
  var show="";  
  show+="  // json객체의 단위 객체 내용..  
  show+="

# 

  show+="    +item.loc+"</p>";  
  show+="  document.body.innerHTML+=show;  
});
```



xml, json 데이터 확인 예제 (속제) :

- 현재 타격 list(xml)
 - rank, 팀명, 선수이름, 타율, 홈런 (3명)
 - xml 문자열로 데이터를 만들고,

rank	팀명	선수이름	타율	홈런

- 방어율 list(json)
 - rank, 팀명, 선수이름, 방어율, 피안타율(3명)
 - json 문자열로 데이터 만들고.

rank	팀명	선수이름	방어율	피안타율



감사합니다 !
