2018년 인턴사원 교육

jQuery

GS ITM

GS ITM

1. 자바스크립트 DOM

- jQuery는 자바스크립트로 만들어진 라이브러리
- 자바스크립트 요소
  - 자바스크립트 Core 문법 : 기본문법, 구조, 데이터타입, 조건문, 반복문, 함수, 클래스
  - 자바스크립트 Core라이브러리 : 타이머, 문자열, 시간, 수학, 배열, 기타 전역함수
  - 자바스크립트 BOM : 브라우저와 관련된 window,Navigator,Location,History, Document, Screen 객체
  - 자바스크립트 DOM : 노드, 스타일, 속성, 이벤트, 위치 및 크기 등을 다룰수 있는 기능
- jQuery란 자바스크립트 DOM 부분을 좀더 쉽게 사용할 수 있게 도와주는 라이브러리 (새로운 언어가 아니다)
  - 자바스크립트 문법과 자바스크립트 DOM으로 작성된 일련의 작업들을 좀더 쉽게 사용하기 위해 감싸고 있는 껍데기

1. 자바스크립트 DOM

- 비교 – 메뉴항목의 글자색을 빨간색으로

```html
<ul id="menu">
    <li>menu1</li>
    <li>menu2</li>
    <li>menu3</li>
    <li>menu4</li>
    <li>menu5</li>
</ul>
```

```javascript
var menu = document.getElementById("mennu");
var liList = menu.getElementsByTagNames("li");
for (var i=0; i<liList.length; i++) {
    var li = liList[i];
    li.style.color = "#f00"
}
```

```javascript
$("#menu li").css("color", "#f00");
```

1. 자바스크립트 DOM

- 핵심 DOM 객체
  - Node 객체 : (노드)문서를 이루는 모든 요소를 통합하여 부르는 용어 (xml 포함)
    tag, text, 주석 (노드타입, 부모, 형제, 자식 노드 접근 및 추가/삭제/교체)
  - Element 객체 : Node중 주석과 text를 제외한 나머지 노드를 통합해서 부르는 용어
    태그이름, 속성 설정, 이벤트 제어
  - HTMLElement 객체 : 오직 HTML문서에만 있는 노드를 통합해서 부르는 용어
    id, className 속성, 오프셋 위치, 마우스 이벤트, 키보드 이벤트
  - Document 객체 : HTML문서와 XML문서의 root객체
    노드 생성, 이벤트 생성, id,tagname으로 특정노드 찾기 기능
  - HTMLDocument 객체 : HTML문서 전용 객체
    body, images 프로퍼티, write(), open() 기능, getElementsByName()

1. 자바스크립트 DOM

DOM객체	Node
상속구조	Node
기능	노드 탐색, 조작하는 프로퍼티와 메서드
주요 프로퍼티	`node.parentNode` 부모노드 탐색 `node.childNodes` 자식노드들 탐색 `node.firstChild` 첫번째 자식노드 탐색 `node.lastChild` 마지막 자식노드 탐색 `node.previousSibling` 이전 형제노드 탐색 `node.nextSibling` 다음 형제노드 탐색 `node.children` 그 안의 요소만 가져옴. 빈칸은 textnode인데 가져오지 않으므로 편리하다. `node.nodeName` 요소의 이름을 대문자로 반환 `node.nodeType` 요소노드는 1, 텍스트노드 3, 주석노드 8 `node.nodeValue` 텍스트노드에만 접근 가능. 텍스트 노드의 실제 값 반환. 요소노드의 경우는 null 반환 `node.hasChildNodes()` 자식이 있으면 true, 없으면 false **※ 아래는 IE8이하는 안되나 요소만 찾아줌** `node.parentElement` 부모요소 탐색 `node.firstElementChild` 첫 자식요소 노드 탐색 `node.lastElementChild` 마지막 자식 요소 노드 탐색 `node.previousElementSibling` 이전 형제요소 탐색 `node.nextElementSibling` 다음 형제요소 탐색

주요 메서드	`node.hasChildNodes()` true/false 반환
	`node.hasChildNodes()` true/false 반환
	`node.cloneNode(boolean)` false가 기본값. true면 자식까지 복제
	`부모노드.appendChild(자식노드)` 부모의 꽁지쪽에 붙이기
	`목표노드.부모노드.insertBefore(insert삽입할노드, target목표노드)`
	`node.removeChild(childnode)`
	`target_node.parentNode.replaceChild(replace_node, target_node)`
	노드 교체. 위치를 교체하는 것이 아니라, 이전 노드를 삭제 한다.
	이전 노드를 삭제하지만 결과 값으로 반환된다.

사용 예	`var el = document.getElementById('div-01').nextSibling;`

1. 자바스크립트 DOM

DOM객체	Document
상속구조	Node > Document
기능	Text node, Element node 생성
주요 프로퍼티	
주요 메서드	`document.createElement('element')` 요소 만들기. 실제 DOM에 붙는건 아님 `document.createTextNode('text')` 텍스트 노드 만들기 `document.createAttribute("name");` 잘쓰지않음 `document.getElementById("idname");` id로 대상(요소노드)을 선택 `document.getElementsByTagName("p");` 요소명으로 선택 `document.getElementsByClassName('classname');` 클래스명으로 선택 `document.querySelector(css selector);` 막강!!! IE8이상. 첫번째 하나만 반환 `document.querySelectorAll(css selector);` 상동. 전체 복수로 반환 createEvent() `target.addEventListener(type, listener[, options]);` dispatchEvent() removeListener()

1. 자바스크립트 DOM

DOM객체	HTMLDocument
상속구조	Node > Document > HTMLDocument
기능	HTML문서 전용 프로퍼티, 메서드
주요 프로퍼티	
주요 메서드	close() open() write() Element[] getElementByName()

1. 자바스크립트 DOM

DOM객체	Element
상속구조	Node > Element
기능	속성을 다루는 기능, 이벤트
주요 프로퍼티	`tagName` 요소의 이름반환. 예전방식

주요 메서드	Element[]
	ElementsByTagName()
	`element.hasAttribute(attName);` true/false 반환
	`element.getAttribute(attributeName);`
	`element.removeAttribute(attrName);`
	`element.setAttribute(name, value);`
	`target.addEventListener(type, listener[, options]);`
	dispatchEvent()
	removeListener()

사용 예	`var parent_el = document.getElementById('parent');` `console.log('data-con:', parent_el.getAttribute('data-con'));`

1. 자바스크립트 DOM

DOM객체	HTMLElement
상속구조	Node > Element > HTMLElement
기능	HTML요소 전용 프로퍼티, 메서드

주요 프로퍼티	`element.id`
	`element.className`
	`element.innerHTML = content;` 노드 동적생성을 쉽게해줌
	`element.style.color = "blue";`
	`element.offsetWidth` border까지의 width
	`element.offsetHeight` border까지의 height
	`element.offsetLeft`
	`element.offsetTop`

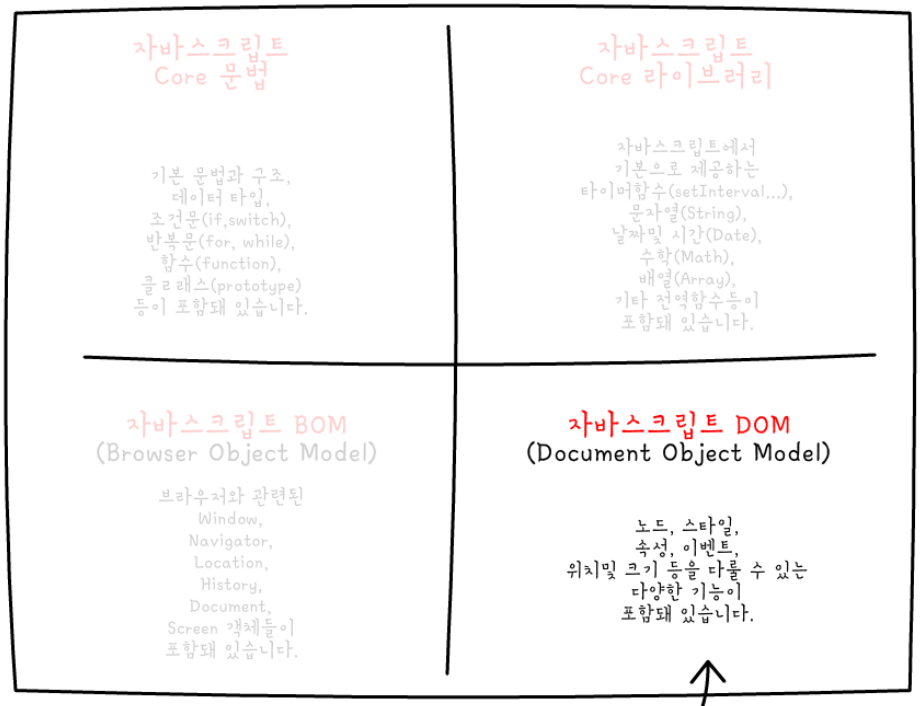
주요 메서드	onkeydown onkeypress onkeyup onclick ondbclick onmousedown onmousemove onmouseout onmouseover onmouseup

사용 예

```
var parent_el = document.getElementById('parent');
console.log('id:', parent_el.id);
console.log('class:', parent_el.className);
console.log('title:', parent_el.title);
```

2. jQuery 소개

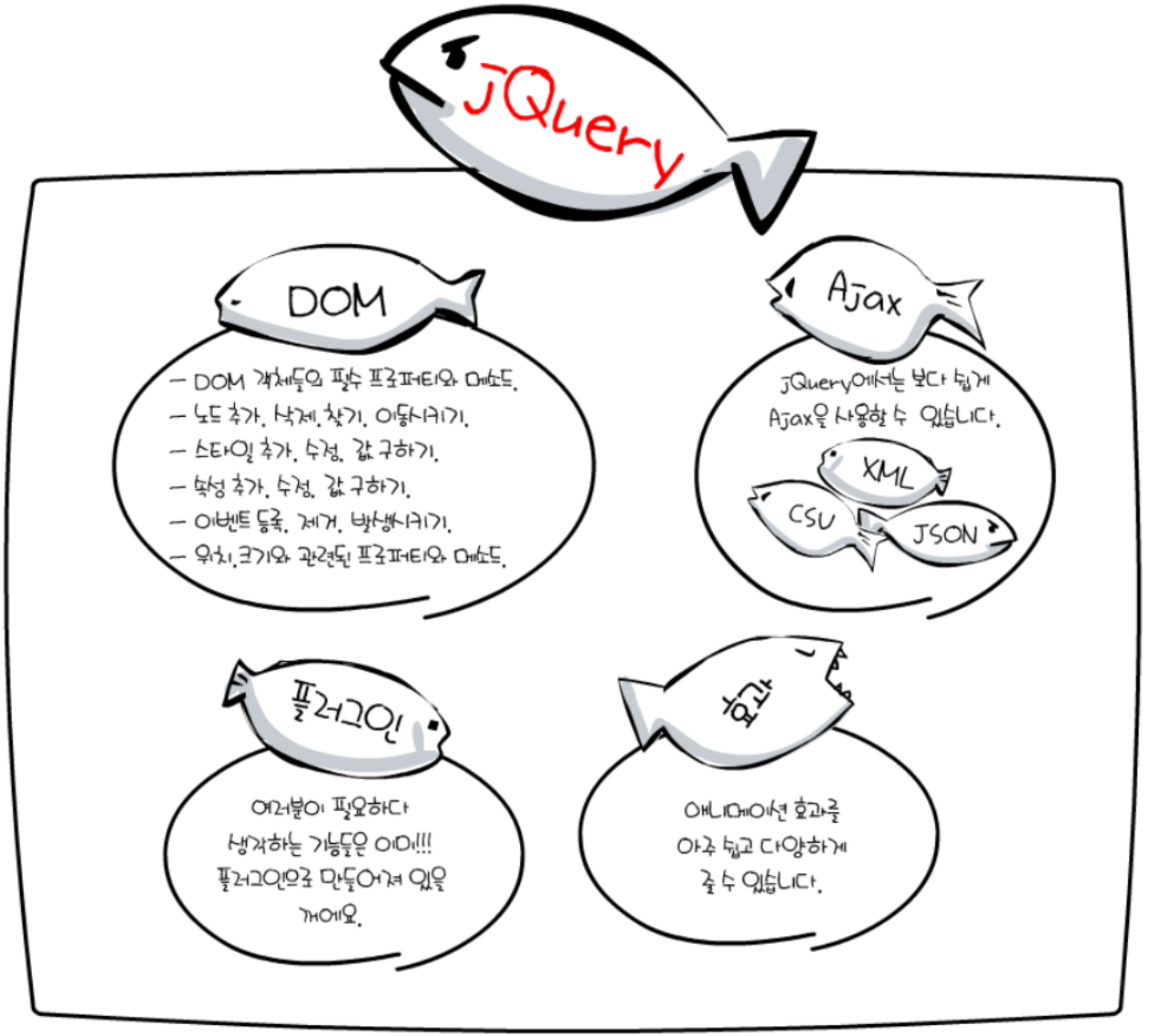
- jQuery란?
  jQuery는 크로스 브라우징 라이브러리의 한 종류



2. jQuery 소개

- jQuery 기능

  http://api.jquery.com/



2. jQuery 소개

- 예제 1. 문서 내용 중 div의 자식 p요소 중 클래스 test2이 적용된 요소를 모두 찾아 border를 4px solid #ff000으로 변경하기

```
window.onload = function() {
    var data2List = document.getElementsByClassName("test2");
    for (var i = 0; i < data2List.length; i++) {
        var data2 = data2List[i];

        if (data2.tagName == "P") {
            if (data2.parentNode.tagName == "DIV") {
                data2.style.border = "4px solid #ff0000";
            }
        }
    }

}
```

```
$(document).ready(function() {
    $("div>p.test2").css("border", "4px solid #ff0000");
})
```

2. jQuery 소개

- 예제 2. 버튼을 클릭하면 "안녕하세요" 메시지를 띄워 주세요

```
window.onload = function() {
    // DOM Level 2방식으로 이벤트 리스너 등록하기.
    var btn1 = window.document.getElementById("btn1");

    // 크로스 브라우징 처리.
    if (btn1.attachEvent){
        btn1.attachEvent("onclick", function() {
            alert("안녕하세요.");
        });
    } else{
        btn1.addEventListener("click", function() {
            alert("안녕하세요.");
        }, false);
    }
}
```

```
$(document).ready(function(){
    $("#btn1").on("click", function() {
        alert("안녕하세요.");
    });
});
```

2. jQuery 소개

- 예제 3. 버튼을 클릭하면 물고기를 왼쪽에서 오른쪽으로 움직여 주세요

```
window.onload = function() {
    //물고기 노드 구하기.
    var fish = document.getElementById("fish");
    var btnStart = document.getElementById("btnStart");
    var left = 50;
    var timerID = 0;
    btnStart.addEventListener("click", function() {
        // 물고기 움직이기
        timerID = setInterval(function() {
            left++;
            fish.style.left = left + "px";
            if (left >= 430) {
                clearInterval(timerID);
                timerID = 0;
            }
        }, 20)
    }, false)
}
```

```
$(document).ready(function() {
    //물고기 노드 구하기.
    var $fish = $("#fish");
    $("#btnStart").click(function() {
        // 물고기 움직이기
        $fish.animate({
            left : 430
        }, 5000);
    });
});
```

2. jQuery 소개

- 진입점 ready() 메서드 설정

```
// 방법1
jQuery(document).ready(function(){
    alert("안녕하세요. jQuery에 온 걸 환영합니다.");
});
// 방법2
// 방법1을 간소화
jQuery(function(){
    alert("안녕하세요. jQuery에 온 걸 환영합니다.");
});
// 방법3
// 방법1에서 jQuery함수 대신 $함수로 변경
$(document).ready(function(){
    alert("안녕하세요. jQuery에 온 걸 환영합니다.");
});
// 방법4
// 방법3을 간소화
$(function(){
    alert("안녕하세요. jQuery에 온 걸 환영합니다.");
});
```

2. jQuery 소개

- onload와 ready() 차이

```
$(document).ready(function() {
    console.log("width = "+ $("#target").width());
});
```

Width = 0 ➔ Dom content loaded (DOM이 사용할 준비가 된 경우)

```
window.onload=function(){
    console.log("width = "+ $("#target").width());
}
```

Width = 실제 너비 값 ➔ 이미지, 플래시 등 실제 content가 모두 로드된 후

2. jQuery 소개

- jQuery 정체

  ➔ window.jQuery = window.$ = jQuery;

  $는 함수이며 리턴값은 특정객체(jQuery 객체)

  var $div = $('div');
  $div.css('border', '4px solid #f00');

- 질문
  1번은 정상적으로 동작하지 않고 2번은 정상적으로 동작함. 그 이유는?

```
// 1번
$("#target").css('font-size').addClass('select');
// 2번
$("#target").css('font-size', 12).addClass('select');
```

2. jQuery 소개

- jQuery 정체

  jQuery는 자바스크립트 DOM을 좀 더 쉽게 다룰 수 있게 도와주는 기능들로 가득 찬 라이브러리.
  라이브러리는 아래와 같이 자바스크립트의 prototype이라는 클래스 제작 문법으로 만들어짐.

```
function jQuery() {

}
jQuery.prototype.css = function() {}
jQuery.prototype.on = function() {}
jQuery.prototype.addClass = function() {}
jQuery.prototype.click = function() {}
jQuery.prototype.animate = function() {}
```

- var 인스턴스 이름 = new 클래스이름();
- $() 에서 jQuery인스턴스를 자동으로 제공

```
function $() {
    ...............
    return new jQuery();
}
```

2. jQuery 소개

- jQuery 인스턴스

  아래의 구문이 실행되면 몇 개의 jQuery 인스턴스가 만들어질까?

```
    <script>
        $(document).ready(function() {
            var $divs = $('div');
            $divs.css("border", "4px solid #f00");
        })
    </script>
</head>
<body>
    <div>div-data1</div>
    <div>div-data2</div>
    <div>div-data3</div>
    <div>div-data4</div>
    <p>p-data1</p>
    <p>p-data2</p>
    <p>p-data3</p>
    <p>>p-data4</p>
</body>
```

3. 노드 다루기 – 노드 찾기

- 아이디 이름으로 노드 찾기
  - $('#아이디 이름')
  - 아이디는 문서 내 유일한 값

- 태그 이름으로 노드 찾기
  - $('태그 이름')

- 클래스 이름으로 노드 찾기
  - $('.클래스 이름')

- 속성으로 노드 찾기
  - $('속성 옵션')
  - $('E[A]') : 속성 A를 포함한 모든 E 노드 찾기
  - $('E[A=V]') : 속성 A의 값이 V인 모든 E 노드 찾기
  - $('E[A^=V]') : 속성 A의 값이 V로 시작하는 모든 E 노드 찾기
  - $('E[A$=V]') : 속성 A의 값이 V로 끝나는 모든 E 노드 찾기
  - $('E[A*=V]') : 속성 A의 값이 V를 포함하고 있는 모든 E 노드 찾기

3. 노드 다루기 – 노드 찾기

- 찾은 노드 다루기
  1) 찾은 노드의 개수 : $대상.length
  2) 찾은 노드 중 n번째 노드 접근하기 : $대상.eq(index)
  3) 자바스크립트 DOM 객체 접근하기 : $대상.get(index), $대상[index]
  4) 순차적으로 찾은 노드 접근하기 :

```
$대상.each(function(index) {
    var $target = $(this);
    // 또는
    var $target = $대상.eq(index);
})
```

예제1. each() 메서드의 index 값 확인하기 (this 의미 확인하기)

```
$(document).ready(function() {
    var $liList = $("ul.menu li");
    $liList.each(function(index) {
        console.log("index = " + index);
    })
});
```

3. 노드 다루기 – 노드 찾기

- 찾은 노드 다루기

  5) 찾은 노드 중에서 특정 노드만 찾기 : $대상.filter('선택자')

```
var $liList = $("ul.menu li");
$liList.filter(".select").css("border", "4px solid #f00");
```

  6) 찾은 노드의 자손(자식 포함) 노드 중 특정 노드 찾기 : $대상.find('선택자')

  예제. #content 노드의 자손(자식포함)노드 중 test1 클래스가 적용된 노드찾기

```
var $content = $("#content");
$content.find(".test1").css("border", "4px solid #f00");
```

  7) 인덱스 값 구하기 : $대상.index(), $목록.index($대상), $목록.index(대상DOM)

3. 노드 다루기 – 노드 찾기

- 자식 노드 찾기
  - 특정 노드(부모 노드)의 바로 아래에 위치하고 있는 노드
  1) 모든 자식 노드 찾기 : $대상.children() (오직 tag 노드만 찾기 - contents(): 포함
  2) 특정 자식 노드만 찾기 : $대상.children('선택자')
  3) 첫번째 자식 노드 찾기 : $대상.children().first(), .children().eq(0) , children(':first') , children(':eq(0)')

  예제. ul.menu의 자식 노드 중 첫번째 자식노드의 border변경

```
var $menu = $("ul.menu");
$menu.children(":first").css("border", "4px solid #f00");
```

```
var $menu = $("ul.menu");
$menu.children().first().css("border", "4px solid #f00");
```

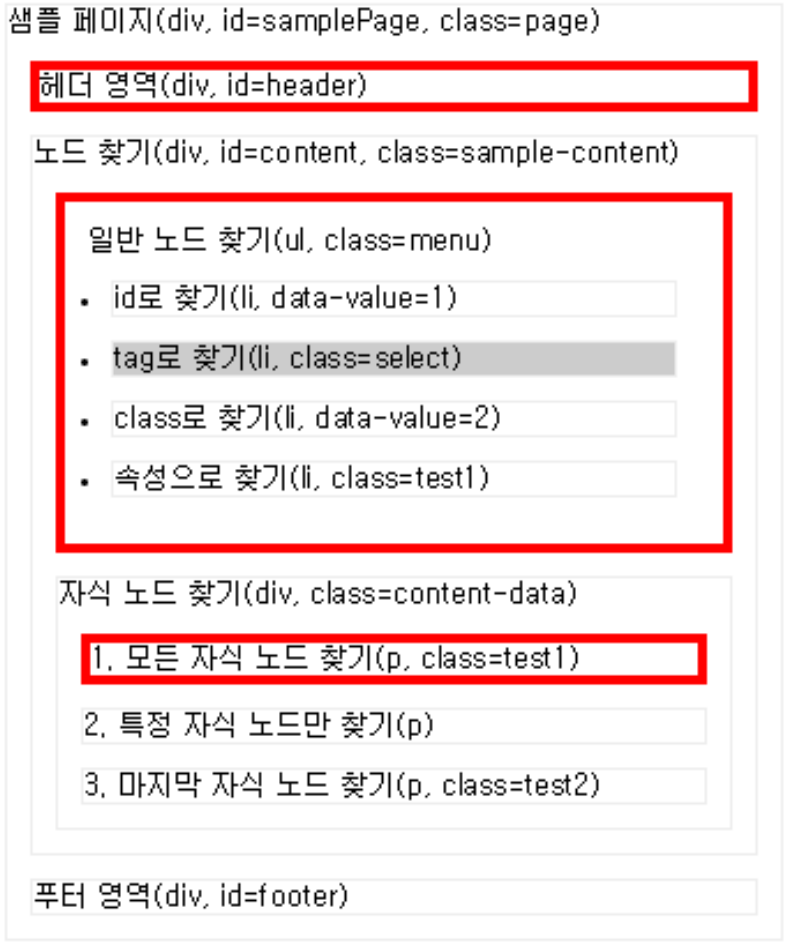
  예제. 해당 코드를 좀 더 효율적으로 변경해 주세요.

```
$(document).ready(function
    var $menu = $("ul.menu
    $menu.children(":first
    $menu.children(":last"
});
```

```
$(document).ready(function() {
    var $menu = $("ul.menu");
    var $children = $menu.children();
    $children.first().css("border", "4px solid #f00");
    $children.last().css("border", "4px solid #f00");
});
```

3. 노드 다루기 – 노드 찾기

- 자식 노드 찾기
  예제. 아래와 같이 각 div태그 노드의 첫번째 자식노드 border를 변경하세요.




```
$("div").find(":first").css("border", "4px solid #f00");
//또는
//$("div").find().first().css("border", "4px solid #f00");
```

```
$("div").children(":first                order", "4px solid #f00");
```

3. 노드 다루기 – 노드 찾기

- 자식 노드 찾기
  4) 마지막 번째 자식 노드 찾기

```
$대상.children().last()
$대상.children(':last')
$대상.children().eq($대상.children().length - 1)
$대상.children(':eq(' + ($대상.children().length - 1) + ')')
$대상.children().eq(-1)
$대상.children(':eq(-1)')
```

5) n번째 자식 노드 찾기 : $대상.children().eq(index), $대상.children(':eq(' + index + ')')

3. 노드 다루기 – 노드 찾기

- 부모 노드 찾기
  - 부모 노드 찾기 : $대상.parent()
  - 조상 노드 찾기 : $대상.parents(['선택자'])
- 형제 노드 찾기
  - 이전 형제 노드 찾기 : $대상.prev(), $대상.prevAll(['선택자'])
  - 다음 형제 노드 찾기 : $대상.next(), $대상.nextAll(['선택자'])

3. 노드 다루기 – 노드 찾기

- 미션
  1) 탭메뉴 아이템 개수 출력

```
$(document).ready(function() {
    alert("length = " + $("#tabMenu1 li").length)
});
```

  2) 클릭한 탭메뉴의 인덱스 값 출력

```
$("#tabMenu1 li").click(function(e) {
    alert("index " + $(this).index());
});
```

  3) 1초에 한번씩 탭메뉴 아이템에 select 적용

```
$(document).ready(function() {
    var currentIndex = 0;
    var $menuItems = $("#tabMenu1 li");

    var timerID = setInterval(function() {
        var $item = $menuItems.eq(currentIndex);
        $item.addClass("select");
        currentIndex++;
        if (currentIndex >= $menuItems.length)
            clearInterval(timerID);

        console.log("currentIndex " + currentIndex);
    }, 1000)
});
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 노드 생성 : var $신규노드 = $('신규DOM');

```
var count = 0;
$("#btnAdd").click(function() {
    count++;
    // 신규 노드 생성
    var $li = $("<li>new menu" + count + "</li>");
    console.log(
})
```

```
var count = 0;
$("#btnAdd").click(function() {
    count++;
    //신규 노드를 생성
    var newLI = document.createElement("li");
    // 신규 노드의 텍스트 노드 생성
    var liText = document.createTextNode("new menu" + count);
    // 텍스트 노드를 신규 노드에 추가
    newLI.appendChild(liText);

    console.log(newLI);

})
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 신규 노드를 첫 번째 자식 노드로 추가

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("ul.menu").prepend("<li>new menu"+count+"</li>");
})
```

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    var $newItem =$("<li>new menu"+count+"</li>");
    $("ul.menu").prepend($newItem);
})
```

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    var $newItem =$("<li>new menu"+count+"</li>");
    $newItem.prependTo("ul.menu");
})
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 신규 노드를 특정 노드의 마지막 번째 자식 노드로 추가

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("ul.menu").append("<li>new menu"+count+"</li>");
})
```

- 특정 노드의 이전 위치에 추가

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("<li>new menu"+count+"</li>").insertBefore("ul.menu li.select");
})
```

```
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("ul.menu li.select").before("<li>new menu"+count+"</li>");
})
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 특정 노드의 다음 위치에 추가

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("<li>new menu"+count+"</li>").insertAfter("ul.menu li.select");
})
```

```
var count=0;
$("#btnAdd").click(function(){
    count++;
    // 신규 노드 생성
    $("ul.menu li.select").after("<li>new menu"+count+"</li>");
})
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 노드 이동
  append(), appendTo(), insertAfter(), after(), insertBefore(), before() 사용
  - 신규노드가 아닌 기존 노드 이용

- 노드 삭제
  - 특정 노드 삭제 : $대상.remove()
  예제. 인덱스가 짝수인 메뉴만 삭제 (even:짝수, odd:홀수)

```
$("#remove").click(function() {
    $("ul.menu li:even").remove();
})
```

  - 모든 자식 노드 삭제 : $대상.children().remove()

4. 노드 다루기 – 노드 생성/추가/삭제/이동

- 노드 내용 읽기 및 변경
  - 노드 내용을 문자열로 읽기 : $대상.html(). $대상.text()

  - 노드 내용 수정하기 : $대상.html(수정할 태그 문자열), $대상.text(수정할 텍스트)

  - 노드 내용을 이용해 여러 개의 자식 노드 추가 : $대상.html('추가할 태그 문자열')
    예제. 메뉴아이템 10개 추가

  - 노드 내용을 이용해 모든 자식 노드 제거하기
    $대상.html('') ➔ remove는 루프를 통해 제거

- 미션. 아래처럼 작동하는 기능을 구현하세요.

4. 노드 다루기 – 노드 생성/추가/삭제/이동

```
// 전역 변수 선언 및 초기화
var $menu = null;
var $menuName = null;
var $selectedItem = null;
$(document).ready(function() {
    init();
    initEvent();
});
```

```
// 전역에서 사용할 요소 초기화
function init() {
    $menu = $("ul.menu");
    $menuName = $("#menuName");
}
```

```
// 이벤트 초기화
function initEvent() {
    // 메뉴 추가
    $("#add").click(function() {
        addMenu();
    })

    $menu.on("click", "li", function() {
        selectItem($(this));
    })
    // 업데이트
    $("#update").click(function() {
        updateMenuItem();
    })
    // 선택 항목 삭제
    $("#remove").click(function() {
        removeMenuItem();
    })
    // 선택 메뉴 항목을 위로 이동
    $("#up").click(function() {
        upMenuItem();
    })
    // 선택 메뉴 항목을 아래로 이동
    $("#down").click(function() {
        downMenuItem();
    })
}
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

```
// 메뉴 추가 처리
function addMenu() {
    // 텍스트 입력 값 구하기
    var menuName = $menuName.val();

    // 신규 메뉴 아이템 문자열 만들기
    var newMenuItem = "<li>" + menuName + "</li>";

    // 선택 메뉴 아이템이 있는 경우 신규 메뉴 아이템을 선택 메뉴 아이템 아래에 추가
    if ($selectedItem) {
        $selectedItem.after(newMenuItem);
    } else {
        // 메뉴에 신규 메뉴 아이템 추가
        $menu.append(newMenuItem);
    }
}
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

```
// 메뉴 선택 처리
function selectItem($item) {
    // 기존 선택 메뉴 아이템이 있는 경우 선택 효과 제거
    if ($selectedItem != null)
        $selectedItem.removeClass("select");

    // 신규 선택 메뉴 아이템 처리
    $selectedItem = $item;
    $selectedItem.addClass("select");

}
```

```
// 메뉴 항목 이름 수정 하기
function updateMenuItem() {
    if ($selectedItem) {
        var menuName = $menuName.val();
        $selectedItem.html(menuName);
    } else {
        alert("선택 메뉴가 존재 하지 않습니다.")
    }
}
```

4. 노드 다루기 – 노드 생성/추가/삭제/이동

```
// 선택 메뉴 항목 삭제
function removeMenuItem() {
    if ($selectedItem) {
        $selectedItem.remove();
        $selectedItem = null;
    } else {
        alert("선택 메뉴가 존재 하지 않습니다.")
    }
}

// 위로 이동
function upMenuItem() {
    if ($selectedItem) {
        var $prevItem = $selectedItem.prev();
        if ($prevItem)
            $selectedItem.insertBefore($prevItem);
    } else {
        alert("선택 메뉴가 존재 하지 않습니다.")
    }
}
```

```
// 아래로 이동
function downMenuItem() {
    if ($selectedItem) {
        var $nextItem = $selectedItem.next();
        if ($nextItem)
            $selectedItem.insertAfter($nextItem);
    } else {
        alert("선택 메뉴가 존재 하지 않습니다.")
    }
}
```

5. 스타일 다루기

- 스타일 종류
  - 내부 스타일
  - 외부 스타일
  - 인라인 스타일

- 스타일을 다루는 방법
  - 정적인 방법
  - 동적인 방법

5. 스타일 다루기

- jQuery를 활용한 스타일 다루는 방법의 특징

  1) 스타일 위치에 관계 없이 스타일 속성 접근 가능

```
var content = document.getElementById("test1");
// 인라인 스타일 접근 방법
console.log("1. color =", content.style.color);
// 인라인 스타일이 아닌 속성을 접근하는 경우
console.log("2. border =", content.style.border); // null;
// 내부, 외부 스타일 접근 방법
console.log("3. border =", window.getComputedStyle(content).border);
```

```
var $content = $("#test1");
// 인라인 스타일 접근 방법
console.log("1. color =", $content.css("color"));
// 내부, 외부 스타일 접근 방법
console.log("2. border =", $content.css("border"));
```

5. 스타일 다루기

2) 스타일 속성 이름 그대로 사용 가능

```
var content = document.getElementById("test1");
content.style.fontSize="12px";

content.style.font-size="12px"; // 에러
```

```
var $content = $("#test1");
$content.css({
    fontSize : "12px",
    //font-Size : "12px",
    border : "4px solid #f00"
});
```

5. 스타일 다루기

3) 여러 스타일 속성값을 쉽게 변경 가능

```
var content = document.getElementById("test1");
content.style.fontSize = "12px";
content.style.border = "4px solid #f00";
```

```
var $content = $("#test1");
$content.css({
    fontSize : "12px",
    border : "4px solid #f00"
});
```

5. 스타일 다루기

4) 단위 생략 가능

```
var content = document.getElementById("test1");
content.style.fontSize = "12px";
content.style.fontSize = 14; // 적용 안 됨
```

```
var $content = $("#test1");
$content.css("fontSize", "12px");
$content.css("fontSize", 14); // 적용 가능
```

5. 스타일 다루기

- 스타일 값 구하기
  - $대상.css('스타일 속성 이름')
  - $대상.css(['스타일 속성 이름',' 스타일 속성 이름'...])
- 스타일 값 설정하기
  - $대상.css('속성이름', '값')
  - $대상.css({

        속성이름: 값,

        속성이름: 값

        .....

    })

5. 스타일 다루기

- 클래스 추가

  - $대상.addClass('클래스 이름1 [클래스 이름2 ...]')

- 클래스 삭제

  - $대상.removeClass() // 모든 클래스 삭제

  - $대상.removeClass('클래스 이름1 [클래스 이름2 ...]')

  예제. 클릭한 메뉴 아이템에 select 클래스가 적용되어 있지 않은 경우 적용해주고 그렇지 않은 경우 제거

```
// 메뉴 아이템에 클릭 이벤트 리스너 등록
$("ul.menu li").click(function(){
    $(this).toggleClass("select");
})
```

```
// 메뉴 아이템에 클릭 이벤트 리스너 등록
$("ul.menu li").click(function(){
    // 클릭한 메뉴 아이템에 select 클래스가 있는 경우 -> 제거
    ;
    select")==false)
    "select");

        $item.removeClass("select");
})
```

5. 스타일 다루기

```
window.onload = function() {
    //물고기 노드 구하기.
    var $fish = $("#fish");
    // 패널의 너비와 높이 구하기
    var panelWidth = parseInt($("#panel").css("width"));
    var panelHeight = parseInt($("#panel").css("height"));
    // 물고기가 최대로 움직일 영역 구하기
    panelWidth = panelWidth - parseInt($fish.css("width"));
    panelHeight = panelHeight - parseInt($fish.css("height"));
    $("#btnStart").click(function() {
        // 랜덤하게 물고기 위치 구하기
        var left = parseInt(Math.random() * panelWidth);
        var top = parseInt(Math.random() * panelHeight);
        // 물고기 위치 설정하기
        $fish.css({
            left : left,
            top : top
        })
    });
}
```

게 만들어라.

5. 스타일 다루기

- 미션2. 랜덤하게 스타일 적용하기

생성한 후 글자 크기와 색이 랜덤하게 적용해 #panel의

```
var count = 0;
var $panel = null;
$(document).ready(function() {
    init();
    start();
});

function init() {
    $panel = $('#panel');
}
function start() {
    setInterval(addTag, 100);
}
```

```
function addTag() {
    var $span = $('<span></span>');
    var color = '#' + (parseInt(Math.random() * 0xffffff)).toString(16);
    var fontSize = (10 + parseInt(Math.random() * 40)) + 'px';

    $span.css({
        color: color,
        fontSize: fontSize,
        display: 'inline-block'
    });

    count++;
    $span.html(count);

    $panel.append($span);

    this.window.scrollTo(0, window.document.body.scrollHeight);
}
```

80 81 828

6. 속성 다루기

- 속성 값 구하기
  - $대상.attr('속성이름')
  - $대상.data('data-속성이름')
- 속성 값 설정하기
  - $대상.attr('속성이름', '값')
  - $대상.data('data-속성이름', '값')

6. 속성 다루기

- 미션 – 이미지 애니메이션 뷰어

  단계01 – 전역 요소 초기화

  단계02 – 이벤트 초기화

  단계03 – 다음 이미지 출력

  단계04 – 이전 이미지 출력

  단계05 – 리팩토링

  단계06 – 오토 플레이 구현

  단계07 – 오토 플레이 멈추기

```
// 다음 이미지
fun
}
```

```
// 오토 플레이 실행
function startAutoPlay(){
    if(timerID==0){
        timerID = setInterval(function(){
            nextImage();
        },100);
    }
}

// 오토 플레이 멈춤
function stopAutoPlay(){
    if(timerID!=0){
        clearInterval(timerID);
        timerID=0;
    }
}
```

```
// index 번째 이미지 출력
function showImage(index){
    $view.attr("src","./images/"+ind
    currentIndex = index;

    // 테스트를 위해 index 값 출력
    console.log(currentIndex);
}
```

7. 이벤트 다루기

- 이벤트 란?

  - 일종의 알림 기능.

  단계1) 클릭

  단계2) MouseEvent 객체 생성

  - 클릭한 마우스 버튼 정보
  - 클릭과 함께 누른 키보드 정보
  - HTML 버튼에서 클릭한 마우스 위치 정보

  단계3) 이벤트 리스너 실행

  - 이벤트가 발생하면 연결한 이벤트 리스너가 실행됨.
  - 단계2에서 생성한 MouseEvent객체가 리스너 함수로 전달됨.

```
function (event) {
    event.이벤트정보
}
```

7. 이벤트 다루기

- 이벤트 종류
  - 마우스 이벤트 : 마우스 버튼, 클릭 위치, Ctrl/Alt 키를 누른 상태에서 눌렀는지
  - 키보드 이벤트 : 눌린 키보드 키에 대한 정보
  - 태그요소 고유 이벤트 : 엘리먼트마다 발생하는 고유 이벤트

    <img> : 이미지 모두 로드되었는지 load이벤트 등

    <input> : 입력한 정보가 바뀌면 change 이벤트 등
  - 사용자 정의 이벤트 : 개발자가 직접 만들어 사용하는 이벤트

7. 이벤트 다루기

- 이벤트 단계 (7.이벤트흐름샘플.html 확인)

  1) 캡처(Capture) 단계 – 가장 먼저 실행되는 이벤트 단계

  document 에서 시작 실제 이벤트 발생시킨 노드 전까지 흐름

  $대상.get(0).addEventListener(이벤트이름, 리스너, true); // jQuery에는 없음

  주로 캡처단계에서 이벤트를 막을때 사용 (e.stopPropagation())

  2) 타깃(Target) 단계

  이벤트를 발생시킨 노드에 머무르는 단계

  이벤트가 등록되어 있다면 리스너 실행

  $대상.on(이벤트이름, 리스너)

  3) 버블(Bubble) 단계

  캡처단계의 역순으로 흐르는 단계(흐름 = 버블링)

  모든 이벤트가 버블링이 발생하진 않는다.

  버블링이 발생하는 이벤트의 경우 버블링을 도중에 멈출 수도 있다.

7. 이벤트 다루기

- 버블링 발생여부와 기본행동의 취소가능 여부

  (e.preventDefault())

EventType	Bubbles	Cancelable
load	X	X
click	O	O
mousedown	O	O
mouseover	O	O
mousemove	O	X
blur	X	X
change	O	X
resize	O	X
scroll	O	X
focus	X	X

7. 이벤트 다루기

- 일반 이벤트 등록 : $대상.on('이벤트이름', 이벤트리스너)

```
<body>
    <ul class="menu">
        <li>menu1</li>
        <li>menu2</li>
        <li>menu3</li>
        <li>menu4</li>
        <li>menu5</li>
        <li>menu6</li>
    </ul>
</body>
```

```
// 메뉴 아이템에 클릭 이벤트 등록
$("ul.menu li").on("click", function(){
    // 클릭한 메뉴 항목 출력
    alert($(this).html());
})
```

```
$("ul.menu li").each(function(index){
    $(this).on('click', function() {
        alert($(this).html());
    });
})
```

7. 이벤트 다루기

- 단축 이벤트 등록 : $대상.단축이벤트(이벤트리스너)

```
$("ul.menu li").click(function(){
    // 클릭한 메뉴 항목 출력
    alert($(this).html());
})
```

- 단축이벤트 목록

  - blur, change, load, unload, resize, scroll, select, submit, click, dbclick, mousedown, mouseup, mousemove, mouseover, mouseout, mouseenter, mouseleave, focus, keydown, keypress, keyup....

7. 이벤트 다루기

- 등록한 이벤트 제거
  - $대상.off('click', 삭제하고 싶은 이벤트 리스너 명)
  - $대상.off('click')
  - $대상.off()
- 이벤트 한번만 실행
  - $대상.one(이벤트이름, 이벤트리스너)
- 기본 행동 취소
  - event.preventDefault()
  - 기본행동 : a 태그 클릭시 click 이벤트 발생 후 해당 링크로 이동되는데 이것을 기본행동이라 한다.
  - return false; 이용가능

7. 이벤트 다루기

- 버블링 멈추기

```html
<div id="panel">
    <button id="btn1">버블링처리</button>
    <button id="btn2">버블링중지</button>
</div>
```

```javascript
$(document).ready(function() {
    $("#panel").click(function(e) {
        console.log("01. panel에 등록한 클릭 이벤트.")
    });
    $("#btn1").click(function(e){
        console.log("02. 버블링 처리");
    });
    $("#btn2").click(function(e){
        console.log("03. 버블링 중지");
        e.stopPropagation();
    })
})
```

7. 이벤트 다루기

- 버블링 활용1 : 하나의 이벤트 리스너로 여러 이벤트 처리하기

```html
<ul class="menu">
    <li>menu1</li>
    <li>menu2</li>
    <li>menu3</li>
    <li>menu4</li>
</ul>
```

```javascript
$(document).ready(function() {
    // 메뉴 항목에 클릭 이벤트 등록
    $("ul.menu li").on("click", function() {
        // 클릭한 메뉴 아이템의 정보 출력
        alert($(this).html());
    })
});
```

```javascript
$(document).ready(function(){
    // jQuery 이벤트 메서드를 활용한 클릭 이벤트 처리
    $("ul").on("click", "li", function(e){
        console.log("테스트용 = "+e.target.tagName);
        // 클릭한 메뉴 아이템의 정보 출력
        alert($(e.target).html());
    })
});
```

7. 이벤트 다루기

- 버블링 활용2 : 라이브 이벤트

```
$(document).ready(function(){
    var $menu = $("ul.menu");
    var count=0;
    $("#add").click(function(){
        count++;
        $menu.append("<li>new"+count+"</li>");
    })
    $("ul li").click(function(){
        alert($(this).html());
    })
});
```

```
// #add 버튼에 클릭 이벤트 등록
$("#add").click(function(){
```

```
// jQuery on() 메서드를 활용해 li에 click live 이벤트 등록
$("ul").on("click", "li", function(){
    // 클릭한 메뉴의 정보 출력
    alert($(this).html());
})
```

```
});
```

7. 이벤트 다루기

- 이벤트 발생 시키기
  - $대상.trigger('이벤트이름')

```
<button id="btnA">버튼A</button>
<button id="btnB">버튼B</button>
```

```
$("#btnB").on("click",function(){
    console.log("버튼B가 눌렸습니다.");
})
```

  - 버튼 A를 누르면 버튼B가 눌린것처럼 이벤트 리스너가 실행되게 하세요

```
// #btnA에 클릭 이벤트 등록
$("#btnA").on("click",function(){
    //이벤트 발생
    $("#btnB").trigger("click");
})
```

7. 이벤트 다루기

- 사용자 정의 이벤트 만들기
  1) 이벤트 객체 생성
  2) 이벤트 발생 시 리스너에 보낼 데이터 생성
  3) 이벤트 발생

```
var $menu = $("ul.menu");
```

```
// 신규 메뉴 항목 생성
var $newItem = $("<li>new"+count+"</li>");
```

```
// 이벤트 객체 생성
var event = jQuery.Event("addItem");
// 이벤트 데이터 생성
var data = [$newItem, $newItem.text(), $menu.children().length];
// 이벤트 발생
$menu.trigger(event, data);
```

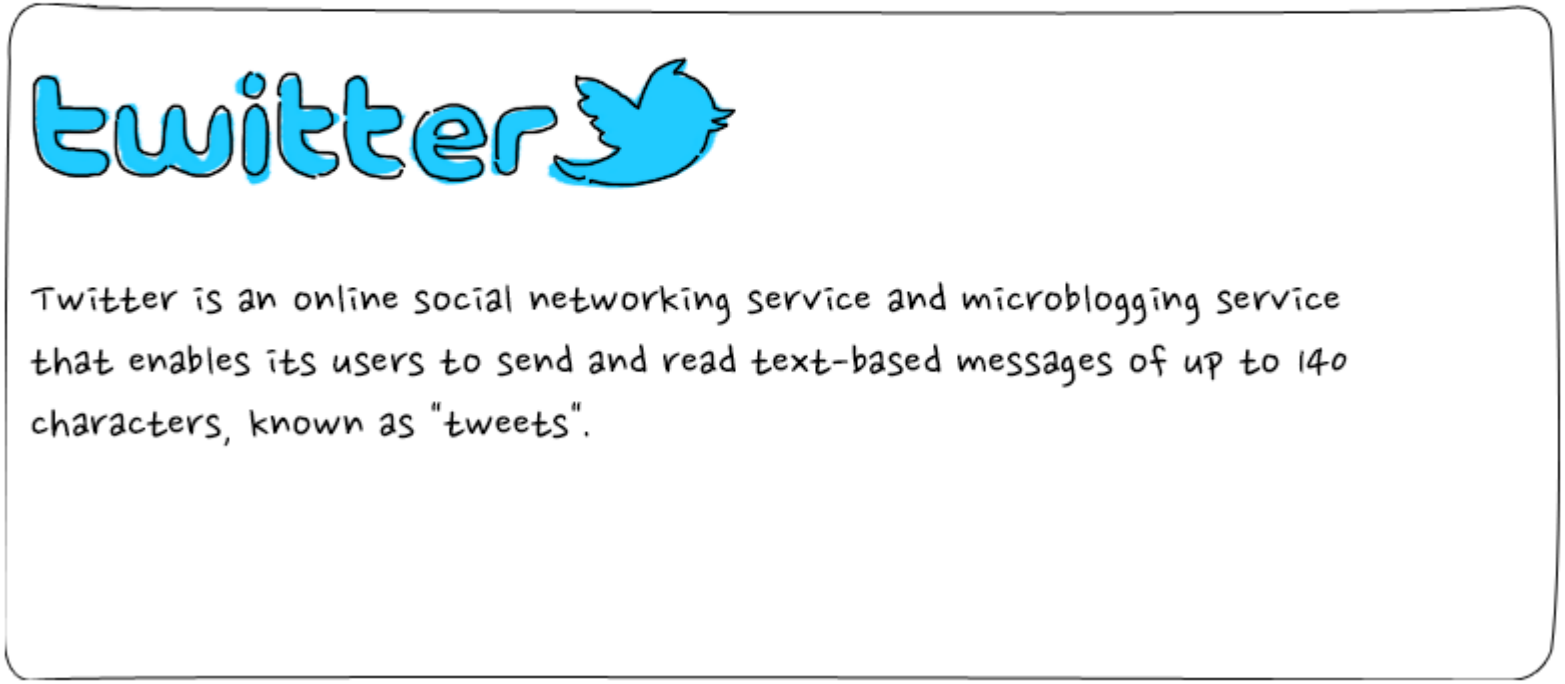
```
$menu.on("addItem", function(e, $item, itemName, itemLength){
    $output.prepend("<p>추가 아이템 = "+itemName+", 아이템 개수 = "+itemLength+"</p>");

})
```

```
// addItem 이벤트 리스너 등록
$menu.on("addItem", function(e){
    $output.prepend("<p>추가 아이템 = "+e.itemName+", 아이템 개수 = "+e.itemLength+"</p>");
})
```

7. 이벤트 다루기

- 미션1
  - 탭 메뉴 클릭 시 탭 패널에 이미지 변경



7. 이벤트 다루기

1) 탭 메뉴 구현하기

```
$(document).ready(function(){
    tabMenu("#tabMenu1");
});
```

```
function tabMenu(selector){
    var $tabMenu = null;
    var $menuItems = null;
    var $selectMenuItem =null;

    function init(){
        $tabMenu = $(selector);
        $menuItems =$tabMenu.find("li");
    }

    function initEvent(){
        $menuItems.click(function(){
            setSelectItem($(this));
        });
    }

    function setSelectItem($item){
        if($selectMenuItem){
            $selectMenuItem.removeClass("select");
        }
        $selectMenuItem = $item;
        $selectMenuItem.addClass("select");
    }

    init();
    initEvent();
}
```

7. 이벤트 다루기

2) 탭 패널 구현하기

```
$(document).ready(function(){
    tabMenu("#tabMenu1");
    var tabPanel1 = tabPanel(".tab-contents");
    //1번째 탭 패널 활성화
    tabPanel1.setSelectPanel(1);
});
```

```
// 탭패널 기능 구현하기
function tabPanel(selector){
    var $tabPanels = null;
    var $selectPanel = null;
    function init(selector){
        $tabPanels = $(selector).find(".content");
    }

    function setSelectPanel(index){
        if($selectPanel){
            $selectPanel.removeClass("select");
        }
        $selectPanel = $tabPanels.eq(index);
        $selectPanel.addClass("select");
    }

    init(selector);

    return {
        setSelectPanel:setSelectPanel
    }
}
```

7. 이벤트 다루기

3) 탭 메뉴와 탭 패널 연동

- 탭 메뉴에서 직접 탭 패널 접근

```
var tabPanel1 = null;
$(document).ready(function(){
    tabMenu("#tabMenu1");
    var tabPanel1 = tabPanel(".tab-contents");
    //1번째 탭 패널 활성화
    //tabPanel1.setSelectPane
});
```

```
// 선택 메뉴 아이템 만들기
function setSelectItem($item){
    if($selectMenuItem){
        $selectMenuItem.removeClass("select");
    }
    $selectMenuItem = $item;
    $selectMenuItem.addClass("select");

    // index에 맞는 탭패널 내용 활성화하기
    tabPanel1.setSelectPanel($item.index());

}
```

7. 이벤트 다루기

3) 탭 메뉴와 탭 패널 연동

- 콜백함수를 활용한 연동 처리

```
var tabPanel1 = null;
$(document).ready(function(){
    tabMenu("#tabMenu1",function(index){
        tabPanel1.setSelectPanel(index);
    });
    tabPanel1 = tabPanel(".tab-contents");
});
```

```
function tabMenu(selector, callback){
```

```
// 선택 메뉴 아이템 만들기
function setSelectItem($item){
    if($selectMenuItem){
        $selectMenuItem.removeClass("select");
    }
    $selectMenuItem = $item;
    $selectMenuItem.addClass("select");

    // 콜백함수가 있는 경우 콜백함수 실행
    if(callback)
        callback($item.index());
}
```

7. 이벤트 다루기

3) 탭 메뉴와 탭 패널 연동

- 사용자 정의 이벤트를 활용한 연동 처리

```
var tabPanel1 = null;
$(document).ready(function() {
    var tabMenu1 = tabMenu("#tabMenu1");
    tabMenu1.$tabMenu.on("tabSelect", function(e) {
        tabPanel1.setSelectPanel(e.selectIndex);
    })
    tabPanel1 = tabPanel(".tab-contents");
});
```

```
function tabMenu(selector) {
    function setSelectItem($item) {
        if ($selectMenuItem) {
            $selectMenuItem.removeClass("select");
        }
        $selectMenuItem = $item;
        $selectMenuItem.addClass("select");

        // 이벤트 발생
        dispatchSelectEvent();
    }
```

```
    // 이벤트 발생
    function dispatchSelectEvent() {
        // 이벤트 객체 생성
        var event = jQuery.Event("tabSelect");
        // 이벤트에 담아 보낼 데이터 연결
        event.selectIndex = $selectMenuItem.index();
        event.$selectItem = $selectMenuItem;

        $tabMenu.trigger(event);
    }
```

```
    return {
        $tabMenu : $tabMenu,
        setSelectItemAt : setSelectItemAt
    }
```

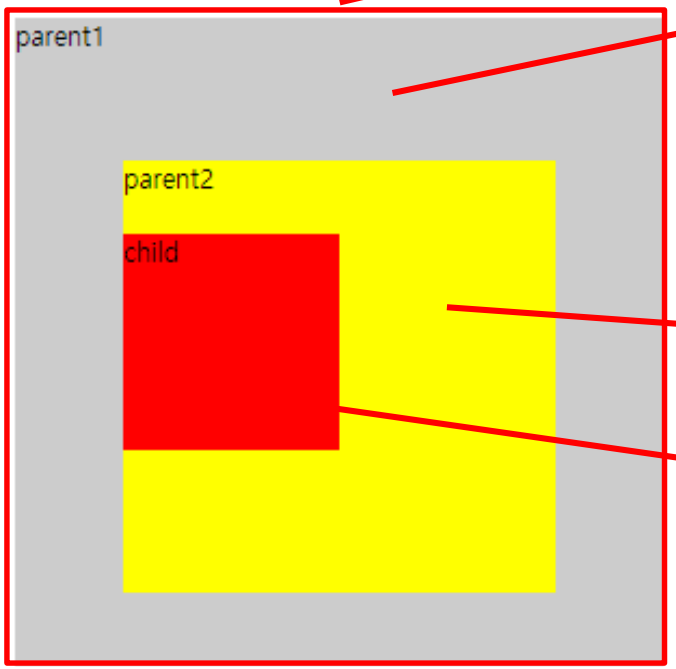
8. 위치/크기 관련 기능 다루기

- 물고기 위치 변경

```
$(document).ready(function(){
    // 물고기 노드 구하기.
    var $fish = $("#fish");

    // 버튼에 이벤트 걸기.
    $("#move").click(function(){
        // 물고기 위치 값 구하기
        var currentX = $fish.position().left+10;
        // 위치 값 설정하기
        $fish.css("left", currentX);
    });
})
```

8. 위치/크기 관련 기능 다루기




```
<div id="parent1">
    parent1
#parent1 {
    position:absolute;
    width:300px;
    height:300px;
    left:100px;
#parent2 {
</    margin:50px;
#child {                    f0;
    position:absolute;
    width:100px;
    height:100px;
    left:50px;
    top:100px;
    background-color:#f00;
}
```

8. 위치/크기 관련 기능 다루기

- 지역(부모 노드 기준) 좌표 위치 다루기

  1) 지역 위치 구하기 : $대상.position().left, top

  예제. 클릭한 물고기의 지역 위치값을 #info에 출력하세요. (8.지역위치구하기.html)

  2) 지역 위치 설정하기 : $대상.css('left 또는 top', 값)

  $대상.css({left:값, top:값})

  예제.클릭시 물고기를 정렬하세요. (8.지역위치설정하기.html)

- 전역 좌표 위치 다루기

  1) 전역 위치 값 구하기 : $대상.offset().left, top

  2) 전역 위치 설정하기 : $대상.offset({left:값, top:값})

  예제. 클릭한 마우스의 위치로 물고기를 이동해 주세요.(8.물고기전역위치이동.html)

  tip. event.pageX, pageY

8. 위치/크기 관련 기능 다루기

- 요소의 크기 다루기

  1) 기본크기 구하기 : $대상.width(), height()

  2) 기본크기 + padding영역 크기 구하기 : $대상.innerWidth(), innerHeight()

  3) 기본크기 + padding + border : $대상.outerWidth(), outerHeight()

  4) 기본크기 + padding + border + margin : $대상. outerWidth(true), outerHeight(true)

- 예제. 물로기를 패널의 오른쪽에 정렬

```
$("#move").click(function(e) {
// 물고기가 이동할 end 위치 구하기
var endX = $("#parent1").innerWidth()-$fish.outerWidth()-parseInt($fish.css("margin-right"));
// 물고기 움직이기
$fish.css("left", endX);
})
```

8. 위치/크기 관련 기능 다루기

- 요소 스크롤 위치 다루기

  1) 스크롤 위치 구하기 : $대상.scrollLeft(), scrollTop()

  8.스크롤위치샘플.html

```
$("#btn").click(function() {
    // 스크롤 위치 값 구하기
    var strInfo = "scrollLeft = " + $container.scrollLeft() + "<br>";
    strInfo += "scrollTop = " + $container.scrollTop();

    // #info에 정보 출력
    $info.html(strInfo);
})
```

8. 위치/크기 관련 기능 다루기

```
// 스크롤 마지막 위치 구하기
var scrollEnd = $container.find("img").width() - $container.width();

$("#btn").click(function(){
    // 스크롤 위치 값 구하기
    var left =$container.scrollLeft();
    // 카운트 변수 초기화
    var count=0;
    // 타이머 실행
    var timerID = setInterval(function(){
        // 0.1초 마다 5px만큼 왼쪽으로 스크롤
        left+=5;
        $container.scrollLeft(left);
        // 스크롤 정보 출력
        count++;
        $info.html(count+", left = "+left+", scrollEnd = "+scrollEnd);
        // 오른쪽 끝까지 스크롤되면 타이머 종료.
        if(left>=scrollEnd){
            clearInterval(timerID);
        }
    },100);
})
```

있게 0.1초에

8. 위치/크기 관련 기능 다루기

- 문서의 위치 및 크기 관련 기능
  - $(document).width(), height()
- 화면의 위치 및 크기 관련 기능
  - 전체 화면 크기 : screen.width, height
  - 유효한 전체 화면 크기 : screen.availWidth, availHeight (작업 표시줄 제외 영역)
- 윈도우의 위치 및 크기 관련 기능(웹브라우저)
  - 기본크기 : window.innerWidth, innerHeight (메뉴바, 툴바, 스크롤바 제외)
  - 기본크기 + 메뉴바 + 툴바 : $(window).width(), $(window).height
  - 기본크기 + 메뉴바 + 툴바 + 스크롤바 : window.outerWidth, outerHeight
- 윈도우 크기 설정
  - window.resizeTo(width, height)

8. 위치/크기 관련 기능 다루기

- 윈도우 리사이징 이벤트 처리

```
$(window).on("resize",function(){
    // 윈도우 크기 구하기
    var strInfo="";
    strInfo += "window.outerWidth = "+window.outerWidth+"<br>";
    strInfo += "window.outerHeight = "+window.outerHeight+"<br>";

    // 윈도우 정보 출력
    $('#info').html(strInfo);
})
```

- 윈도우 위치 다루기 (window.open으로 만들어진 윈도우만 가능)
  - 위치 구하기 : window.screenLeft, screenTop
  - 위치 설정하기 : window.moveTo(dx, dy), window.moveBy(dx, dy)

8. 위치/크기 관련 기능 다루기

- 윈도우 스크롤 다루기
  - 스크롤 위치 구하기 : window.pageXOffset, pageYOffset
  - 스크롤 위치 설정하기 : window.scrollTo(x, y), scrollBy(x, y)
  - 스크롤 이벤트 처리하기 : $(window).on('scroll', function(){})

8. 위치/크기 관련 기능 다루기

- 마우스의 위치 및 크기 관련 기능
  - 윈도우를 기준으로 한 좌표 정보 위치 pageX, pageY, clientX, clientY
  - 문서를
  - 클릭한

var

var

예제

표기

```
// 클릭한 위치 값 알아내기
var offsetX = e.pageX - $(this).offset().left;
var offsetY = e.pageY - $(this).offset().top;

// 위치 값을 문자열로 만들기
var strInfo = "offsetX = "+offsetX+", offsetY = "+offsetY;

// 클릭한 위치 값 출력
$info.html(strInfo);

// 클릭한 위치로 이동
$fish.css({
    left:offsetX,
    top:offsetY
})
```

8. 위치/크기 관련 기능 다루기

- 미션1. 심플 배너 만들기

  left속성을 이용

```
// .banner-container 구하기
var $banner = $(".banner-container");
// index 번째 배너 이미지 출력
// 이전 버튼(#prev)에 클릭 이벤트 등록
// 다음 버튼(#next)에 클릭 이벤트 등록
$("#next").click(function(){
    // 인덱스 값 구하기
    currentIndex++;
    if(currentIndex>=bannerLength)
        currentIndex= 0;

    // currentIndex 번째 배너 이미지 출력
    showImage(currentIndex);
})
```

8. 위치/크기 관련 기능 다루기

- 미션2. 스크롤 스파이 만들기

```
$(document).ready(function(){
    // .tab-menu 구하기
    var $tabMenu = $(".tab-menu");
    // 탭메뉴 위치 구하기
    var tabMenuPos = $tabMenu.offset().top;

    // 스크롤 이벤트 등록
    $(window).on("scroll",function(){
        // 스크롤 위치 값 구하기
        var scrollY = window.pageYOffset;

        // 스크롤 위치 값이 탭메뉴 위치 보다 큰 경우만 탭메뉴에 fixed 클래스 적용
        // 그렇지 않은 경우 fixed 클래스 제거
        if(scrollY>tabMenuPos){
            $tabMenu.addClass("fixed");
        }else {
            $tabMenu.removeClass("fixed");
        }
    })
});
```