Jquery

1) 개요

- (바닐라) 자바스크립트를 가지고 편리한 사용을 지원 https://jquery.com/download/
- 가장 많이 사용된 Javascript 라이브러리, 브라우저 호환성이 뛰어나다. (Cross Browsing)[1]
- 장단점 : DOM 탐색과 변경이 쉽고 간편, 크로스 브라우징 지원, 애니메이션과 대화형 웹페이지의 쉬운 구현, 다양하고 풍부한 플러그인 제공, 모바일 기기 지원[2]
- 적용 : 다운하거나 아래의 구문을 추가하면 사용가능

Other CDNs > Google CDN >
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

2) DOM이 로드되었을 때 jQuery 실행 //교안 JQ100~103, 3page

- 웹페이지에 jquery 적용 - $ 표시가 jquery를 의미한다.

```
<script>
 $() = jQuery();
</script>
```

- 넷다 동일, 맨아래가 간결하니 많이 쓰인다.

```
window.onload = function(){ };
window.addEventListener("DOMContentLoaded", function(){ });
$(document).ready(function(){ });
$(function(){ });
```

3) 바닐라 자바스크립트와 제이쿼리의 비교

기존의 바닐라보다 훨씬 간결해진 것을 확인 가능하다.

```
    document.querySelector("#content");
    $("#content")
```

text() 메소드는 지정한 요소 안에 텍스트를 설정하거나 반환하는 메소드로 아래 코드는 문서에서 전체 p 태그의 텍스트를 반환한다.

```
    document.querySelector("p").innerText = "<h2>하하하</h2>";
    $("p").text("<h2>하하하</h2>");
```

html() 메소드는 지정한 요소 안에 태그를 포함한 내용을 설정하거나 반환하는 메소드로 아래 코드는 문서에서 첫 번째 나오는 p 태그 안의 내용을 태그를 포함해 반환한다.

```
    document.querySelector("p").innerHTML= "<h2>하하하</h2>";
    $("p").html("<h2>하하하</h2>");
```

[1] 이제는 퇴색된 의미, 안해주는 것 없는 편
[2] PhoneGap으로 하이브리드 지원이 되었다고 한다.

3-1) CSS 의 적용

css() 메서드는 현재 jQuery로 선택한 문서 객체에 스타일을 지정하거나 이미 지정된 스타일 정보를 읽어 올 수 있다. // 'margin' 으로만 두면 기본속성을 적용한다고 함

```
  $('#result1').text(result1)
    .css('margin', '15px')
    .css('border', '1px solid red')
    .css('padding', '15px')
    .css('width', '500px');
```

한줄도 가능

```
$('#result2').html(result2).css({"margin" : "15px", "border" : "1px solid red"});
```

4) JQuery 의 다양한 선택자 사용하기 //JQ 201

$간결

```
  document.querySelector("#btnNameOrder").addEventListener("click", ()=> {});
$("#btnNameOrder").click(function(){ });
```

$간결 - 4줄이 한줄로!

```
    let h4 = document.createElement("h4");
    h4.textContent = "과일이름순정렬"
    let ul = document.createElement("ul");
    h4.append(ul);
    $("<h4>과일이름순정렬</h4>")
```

$간결

```
let h4 = document.createElement("h4");
    h4.textContent = "과일이름순정렬"
    let ul = document.createElement("ol");
    ol.setAttribute("id","fruitName")
    h4.append(ol);
    $('<h4>과일 이름순 정렬</h4><ol id="fruitName"></ol>').appendTo('#nameOrder');
```

4-1) $(selector).each() 메소드 //for 반복문의 간결화

```
$('#fruitList').children().each(function(index, value) {
```

index는 순서대로 0,1,2,3 으로 넘어갈것이고 , value인 사과 : 500원 , 배 : 700원 순서대로 들어간다

5) 이벤트 등록 메소드(JS : 이벤트 속성)와 이벤트 객체

jQuery는 다양한 이벤트에 대응하는 이벤트 핸들러를 등록할 수 있도록 각각의 이벤트 이름과 동일한 이벤트 핸들러 등록 메서드를 제공하고 있다.
구조 : $(selector).이벤트 등록 메서드(function(event) { });
이벤트 등록 메소드는 JS의 이벤트 속성(ex) on+click)을 의미한다.

```
 $(function() {
//이벤트 핸들러 안에서 이벤트 객체에 접근하려면 아래와 같이 첫 번째 매개변수를
    * 지정하고 그 매개변수로 이벤트 객체의 속성이나 메서드를 사용할 수 있다.
     $('img').mouseenter(function(e) {
// 이벤트 핸들러 함수 안에서 this는 이벤트가 발생한 문서 객체를 의미한다.
     $(this).removeClass('mouseout')
       .addClass('mouseover');

// is(selector) 메소드는 지정한 객체가 selector와 일치하는지 판단해 boolean 값으로 반환하는 메소드
이벤트 속성에서 target은 이벤트가 발생한 문서 객체를 의미한다.
     if($(e.target).is('#img3')) {
       alert('세 번째 이미지로 마우스가 들어 왔네요');
     }});
```

메소드()는 단일지원 메소드, on+메소드()는 통합지원 메소드라고 구분한다. (+이벤트 위임까지 3개)
//개념은 똑같은데 표현하는 방법이 다른 것, 편한거 쓰셈

```
// $("선택자").on("이벤트명", function(e) {}) 단일 이벤트 핸들러 등록
   $('img').on('mouseenter', function(e) {
     // 이벤트 핸들러 안에서 this는 이벤트가 발생한 문서 객체를 의미한다.
     $(this).removeClass('mouseout')
       .addClass('mouseover');
   });
   $('img').on('mouseleave', function(e) {
     $(this).removeClass('mouseover')
       .addClass('mouseout');
   });
   $('#img1').on('click', function(e) {
     alert($('div#container1').next().attr('id'));
   });

   // $("선택자").on({"이벤트명" : function(e){}, ...}) 다중 이벤트 핸들러 등록
   $('img').on({
     /* is(selector) 메소드는 지정한 객체가 selector와 일치하는지 판단, boolean 으로 반환하는 메소드
     dblclick: function(e) {
       if($(this).is('#img2')) {
         alert('두 번째 이미지를 더블 클릭 하셨네요');
       }},
```

```
  mouseenter: function(e) {
    if($(e.target).is('#img3')) {
      console.log('세 번째 이미지로 마우스가 들어 왔네요');
    }},
  mouseleave: function(e) {
    if($(this).is('#img1')) {
      $(this).css('border', '2px solid red');
    }}});
```

위에서 아래로, 아래에서 위로 접근방식으로 둘다 동일하다.

```
$('#img1').on('click', function(e) {
  $(this).parent().clone().appendTo('body');
  $("body").append($(this).parent().clone());
});
```

off를 쓰면 이벤트가 단 한번 적용됨

```
  $(this).off("click");
```

one() 메소드를 이용한 click 이벤트에 대한 단일 이벤트 등록 이벤트를 한 번만 실행하려면 one() 메소드를 이용하면 편리하다.

```
$('#img2').one('click', function(e) {
  $(this).parent().clone().appendTo('body');
});
```

이벤트 위임 방식 이벤트 핸들러 등록 : 이벤트 위임 방식은 상위 요소에 이벤트 핸들러를 등록하는 방식으로 이벤트 버블링을 활용하기 때문에 동적 생성해 추가되는 문서 객체나 기존 객체를 복제한 문서 객체 등에 이벤트 핸들러를 등록할 수 있다. (아래의 소스는 둘다 유사하게 작동됨)

```
$(document).on('mouseenter', '#img3', function(e) {
  $(this).parent().clone().appendTo('body');
})
$("body").on("mouseenter", "#img3", function(e) {
  $("body").append($(this).parent().clone());
});
```

6) 체크박스 다루기

라벨을 통해 체크박스에 정확히 클릭하지 않고 텍스트를 눌러서 편하게 접근

```
<div class="checkWrap">
  <label><input type="checkbox" id="pizza"
    name="pizza" value="3500">조각피자 3500원</label>
</div>
```

form 안에서 find 는 자식요소를 의미
modal : 동일 화면내에 modal 창을 닫지 않으면 기존 창으로 접근 불가