jQuery(JS라이브러리)

쿼리(Query)는 **데이터베이스나 검색 엔진 등에서 원하는 정보를 요청하는 질문이나 명령**을 의미한다.

🔷 jQuery의 특징

1. **간단한 문법**
   - 자바스크립트의 긴 코드 몇 줄을 한 줄로 줄일 수 있다.
   - 예: DOM 요소 선택, 이벤트 처리, 애니메이션 등을 간결하게 작성 가능.
2. **DOM 조작이 편리**
   - HTML 문서의 요소를 쉽게 선택하고, 속성이나 스타일을 바꾸고, 추가/삭제 가능.
3. **이벤트 처리 단순화**
   - `onclick`, `onchange` 같은 이벤트를 더 직관적이고 짧게 작성 가능.
4. **브라우저 호환성 보장**
   - 예전에는 크롬, 익스플로러, 파이어폭스 등 브라우저마다 자바스크립트 동작이 달랐는데, jQuery가 이를 통일시켜줌.
5. **플러그인 생태계**
   - 슬라이더, 모달창, 달력 등 다양한 기능을 손쉽게 가져다 쓸 수 있음.

기본 사용 예시

✅ jQuery 불러오기 (CDN)

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>  //CDN 3.6.0버전
```

✅ 문법 예시

```
//문서가 준비되면 실행
//jQuery 단어를 축약해서 $(달러)로 표기
//jQuery === $
$(document).ready(function() {
  // 버튼 클릭 시 실행
  $("#myBtn").click(function() {
    $("#myText").text("버튼을 클릭했습니다!");
    $("#myText").css("color", "red");
  });
});
```

👉 위 코드는 **버튼을 클릭하면 글자가 바뀌고 빨간색으로 변하는 코드이다.**

🔷 요약

- **정의:** 자바스크립트를 더 쉽게 쓰게 해주는 라이브러리
- **장점:** 짧은 코드, 강력한 기능, 호환성, 다양한 플러그인
- **현황:** 요즘은 React, Vue 같은 프레임워크가 많이 쓰지만, 간단한 프로젝트나 레거시 코드에서는 여전히 많이 사용된다

📌1. 기초 선택자 실습

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>제이쿼리 기초 선택자 실습</title>
  <script src="./js/jquery-1.12.4.min.js"></script>
  <script src="./js/jquery-3.3.1.min.js"></script>
  <script>
    $(document).ready(function () {
      $('h2').css('color','red'), //일반 타입
      $('h3').css('color','green'), //일반 타입
        $('h3').css('background','pink'); //일반 타입
      $('ul').children('li:first-child').css('color','red'); //일반 타입

      $('ul').children().css({color:'blue', listStyle:'none'}); //객체 타입 (중괄호)
      $('.class1').css({fontSize:'36px'}); //객체 타입 (중괄호)
      $('#id1').css({color:'red',background:'yellow'}); //객체 타입 (중괄호)
    });
  </script>
  <style>
    ul{
      text-transform: uppercase;
    }
  </style>
</head>

<body>
  <h2>제이쿼리 선택자1</h2>
  <h3>제이쿼리 선택자2</h3>
  <ul>
    <li>one</li>
    <li>two</li>
    <li>three</li>
    <li>four</li>
  </ul>
  <h4 class="class1">제이쿼리 선택자3</h4>
```

```
  <h4 id="id1">제이쿼리 선택자4</h4>
</body>
```

1 실행 구조 (로컬방식 이용)

```
<script src="./js/jquery-1.12.4.min.js"></script>
<script src="./js/jquery-3.3.1.min.js"></script>
```

- jQuery 라이브러리 두 개를 불러옴 (사실 하나만 불러야 함 ⚠️).
- 두 개를 동시에 쓰면 충돌할 수 있다. → 보통 최신 버전 하나만 사용한다.

```
$(document).ready(function () {
  // jQuery 실행 영역
});
```

- 문서(HTML)가 모두 로딩된 뒤 실행할 코드를 넣는 부분.
- 순수 JS의 `window.onload`와 비슷하지만, 더 빠르게 실행됨.

---

2 태그 선택자

```
$('h2').css('color','red');
$('h3').css('color','green');
$('h3').css('background','pink');
```

- `<h2>` → 빨간색 글자
- `<h3>` → 초록색 글자 + 핑크 배경

---

3 자식 선택자

```
$('ul').children().css({
  color:'blue',
```

```
  listStyle:'none'
});
```

- `<ul>`의 자식 `<li>` 전부 선택
- 글자색 파랑 + 불릿 제거

```
$('ul').children('li:first-child').css('color','red');
```

- `<ul>`의 첫 번째 `<li>`만 빨간색

4️⃣ 클래스 선택자

```
$('.class1').css({
  fontSize:'36px'
});
```

- `.class1` (클래스명 class1 가진 요소) → 글자 크기 36px

5️⃣ 아이디 선택자

```
$('#id1').css({
  color:'red',
  background:'yellow'
});
```

- `#id1` (아이디가 id1인 요소) → 글자 빨강 + 배경 노랑

📌2. 선택자 정리 (자식 / 부모 / 형제)

First
Second
Third

jQuery Selector!!

```
<title>제이쿼리 선택자</title>
<script>
 $(document).ready(function(){
  /*1. 자식선택자 */
  // $('ul>li').css('color','blue');
  // $('ul>li').css('list-style','none');

  //객체 사용하여 같은 결과가 나오도록 코딩 | 소스캡쳐, 넘버링
  $('ul>li').css({color:'blue',listStyle:'none'});

  /*2. 부모 선택자 */
  $('li').parent('ul').css({fontWeight:'bold'}); // 'ul' 지칭

  /*3. 형제 선택자 */
  $('ul').siblings('h3').css({'background':'purple','font-size':36});
 });
</script>
</head>

<body>
 <ul>
  <li>First</li>
  <li>Second</li>
  <li>Third</li>
 </ul>
```

```
  <h3>
    jQuery Selector!!
  </h3>
</body>
```

1️⃣ 자식 선택자 (Child Selector)

```
$('ul>li').css({color:'blue', listStyle:'none'}); //다이렉트기법
```

- `ul > li` → `<ul>`의 **직계 자식 `<li>`*만 선택
- 적용 결과:
  - 글자색 파랑
  - 불릿(●) 제거

2️⃣ 부모 선택자 (Parent Selector)

```
$('li').parent('ul').css({fontWeight:'bold'}); //li의 부모(parent) bold
```

- `li` 요소의 **부모**가 되는 `ul`을 선택
- 적용 결과:
  - `<ul>` 안의 모든 글자가 **굵게(bold)** 표시됨

3️⃣ 형제 선택자 (Sibling Selector)

```
$('ul').siblings('h3').css({'background':'purple','font-size':36});
//siblings ul 의 친구or 형제
```

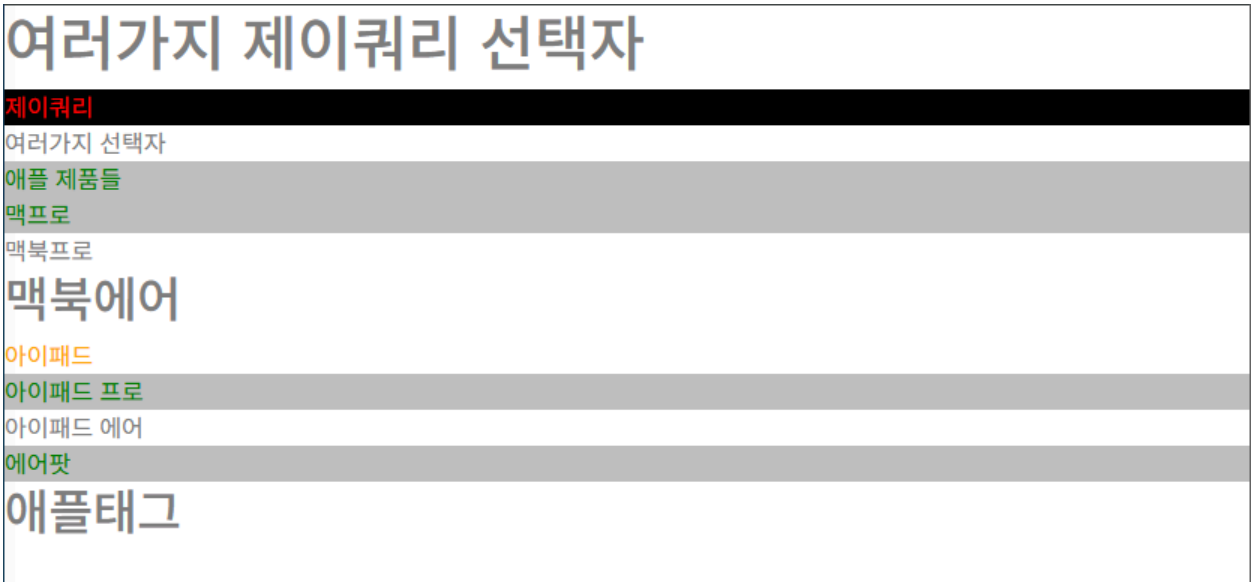
- `<ul>`과 같은 레벨에 있는 `<h3>` 선택
- 적용 결과:
  - 배경색 보라색

- 글자 크기 36px

🔷 선택자 개념 정리

- **자식 선택자 (>)** : 부모 바로 아래 있는 요소만 선택
- **부모 선택자 (.parent())** : 특정 요소의 부모를 선택
- **형제 선택자 (.siblings())** : 같은 부모를 가진 같은 레벨의 요소 선택

📌3. 조합 선택자




```
<script>
  $(document).ready(function () {
    // 문서전체 글자색 gray
    $('*').css('color', 'gray');
    // 아이디 값 a = 글자색 red , 배경색 black
    $('#a').css({ color: '#f00', backgroundColor: '#000' });
    // div태그중 id값이 b인 요소만 선택 = 글자색 오렌지
    $('div#b').css('color', 'orange');
```

```
    // div태그중 class값이 test인 요소만 선택 = 글자색 그린
    $('div.test').css({ color: 'green', backgroundColor: 'silver' });
  });
 </script>
</head>

<body>
 <h1>여러가지 제이쿼리 선택자</h1>
 <div class="content">
  <div id="a">
   제이쿼리
  </div>
  <div>
   여러가지 선택자
  </div>
  <div class="test">
   애플 제품들
  </div>
  <div class="test main">
   맥프로
  </div>
  <div>
   <span>맥북프로</span>
   <h2 id="c">맥북에어</h2>
  </div>
  <div id="b">
   아이패드
  </div>
  <div class="test">
   아이패드 프로
  </div>
  <div class="main">
   아이패드 에어
  </div>
  <div class="test">
   에어팟
```

```
    </div>
    <div>
      <h2 class="test">애플태그</h2>
    </div>
  </div>
</body>
```

1 전체 선택자

```
$('*').css('color','gray'); // * 에스터리스크
```

- : 문서 전체 모든 요소 선택
- 적용 결과 → **문서 전체 글자색 회색**

---

2 아이디 선택자

```
$('#a').css({color:'#f00', backgroundColor:'#000'});
```

- `#a` : id가 `a`인 요소 지정 선택
- 적용 결과 → **글자 빨강 + 배경 검정**

---

3 태그 + 아이디 선택자

```
$('div#b').css('color','orange');
```

- `div#b` : `div` 태그이면서 id가 `b`인 요소 지정 선택
- 적용 결과 → **글자 주황색**

---

4 태그 + 클래스 선택자

```
$('div.test').css({color:'green', backgroundColor:'silver'});
```

- `div.test` : `div` 태그이면서 class가 `test`인 요소 모두 선택
- 적용 결과 → **글자 초록 + 배경 은색**

🔷 선택자 개념 정리

- → 전체 선택자
- `#id` → 아이디 선택자
- `.class` → 클래스 선택자
- `태그#id` → 특정 태그 + 아이디 동시 지정
- `태그.class` → 특정 태그 + 클래스 동시 지정

📌4. 계층 선택자

1. 아침 바람 찬바람에

2. 울고 가는 저 기러기
3. 우리 선생님 계실 적에
4. 엽서 한장 써 주세요

---

5. 한 장 말고 두 장이요

---

6. 두 장 말고 세 장이요

좋아하는 음식들

ID	먹을거리	분류
F001	짜장면	중식
F002	라면	분식,한식
F003	김치찌개	한식
F004	돈까스	일식
F005	초밥	일식
F006	스테이크	양식

```
<script>
    $(document).ready(function () {
        /* 1. 자식 선택자 */
        $('#one > div').css('color', 'orange');
        /* 2. 인접 형제 선택자 */
        $('#one + div').css('border', '1px solid gray');
        /* 3. 형제 선택자 */
        $('#one ~ div').css('color', 'green');

        /* (실습) 자식 선택자 사용하여 #foods 테이블의 글자색 gray 적용 소스캡쳐, 캡쳐, 넘버링 */
        $('#foods > tbody > tr').css('color', 'gray');
    });
</script>
```

```
</head>

<body>
  <!-- 구조1 one -->
  <div class="card-body">
    1. 아침 바람 찬바람에
  </div>
  <div id="one" class="card-body"> //one
    <div>
      2. 울고 가는 저 기러기
    </div>
    <span>
      <div>
        3. 우리 선생님 계실 적에
      </div>
    </span>
    <div>
      4. 엽서 한장 써 주세요
    </div>
  </div>
  <div class="card-body">
    5. 한 장 말고 두 장이요
  </div>
  <div class="card-body">
    6. 두 장 말고 세 장이요
  </div>

  <!-- 구조2 foods -->
  <div class="card-body">
    <h3 class="card-title">좋아하는 음식들</h3>
    <table class="table" id="foods"> //foods
      <tr>
        <th>ID</th>
        <th>먹을거리</th>
        <th>분류</th>
      </tr>
```

```
        <tr>
          <td>F001</td>
          <td>짜장면</td>
          <td>중식</td>
        </tr>
        <tr>
          <td>F002</td>
          <td>라면</td>
          <td>분식,한식</td>
        </tr>
        <tr>
          <td>F003</td>
          <td>김치찌개</td>
          <td>한식</td>
        </tr>
        <tr>
          <td>F004</td>
          <td>돈까스</td>
          <td>일식</td>
        </tr>
        <tr>
          <td>F005</td>
          <td>초밥</td>
          <td>일식</td>
        </tr>
        <tr>
          <td>F006</td>
          <td>스테이크</td>
          <td>양식</td>
        </tr>
      </table>
    </div>
</body>
```

1️⃣ 자식 선택자 (`>`) #one

```
$('#one > div').css('color', 'orange');
```

- `#one` 요소의 **직계 자식 `<div>`*만 선택
- 결과: `2. 울고 가는 저 기러기`, `4. 엽서 한장 써 주세요` 글자 주황색
- ⚠️ `3. 우리 선생님 계실 적에`는 `<span>` 안에 들어가 있어서 "손자 요소"라 적용 안 됨

2️⃣ 인접 형제 선택자 (`+`) #one

```
$('#one + div').css('border', '1px solid gray');
```

- `#one` 바로 **뒤에 나오는 형제** `<div>` **하나** 선택
- 결과: `5. 한 장 말고 두 장이요`에 회색 테두리 적용

3️⃣ 일반 형제 선택자 (`~`) #one

```
$('#one ~ div').css('color', 'green');
```

- `#one` 뒤에 나오는 **모든 형제** `<div>` 선택
- 결과:
  - `5. 한 장 말고 두 장이요`
  - `6. 두 장 말고 세 장이요`

    → 글자 초록색

4️⃣ 실습: 자식 선택자로 테이블 행 스타일 적용 #foods

```
$('#foods > tbody > tr').css('color', 'gray');
```

- `#foods` 테이블의 직계 자식 `<tbody>` 안의 **모든** `<tr>` 선택
- 결과: 표의 모든 데이터 행 글자가 회색

📌5. 위치필터선택자 | 기호는 :(콜론)

좋아하는 음식들

ID	먹을거리	분류
F001	짜장면	중식
F002	라면	분식,한식
F003	김치찌개	한식
F004	돈까스	일식
F005	초밥	일식
F006	스테이크	양식

ID	먹을거리	분류
F001	짜장면	중식
F002	라면	분식,한식
F003	김치찌개	한식
F004	돈까스	일식
F005	초밥	일식
F006	스테이크	양식

```
<script>
 $(document).ready(function () {
    /* 인덱스 번호 eq() 는 0부터 시작*/
    $('#foods > tbody > tr:even').css({ background: 'gray' }); //짝수 선택
    $('#foods > tbody > tr:odd').css({ background: 'lightgray' }); // 홀수 선택

    // 첫번째 요소 선택
    $('#foods tr:first').css({ background: 'black', color: 'white' });

    // 마지막 요소 선택
    $('#foods tr:last').css({ background: 'orange', color: 'white' });

    // 인덱스 번호 (2) 선택
    $('#foods tr:eq(2)').css({ background: 'blue', color: 'white' });

    //less than ~ 미만 , grater than ~ 초과
    $("#foods2 tr:lt(3)").css({ background: 'gray' }); //미만
    $("#foods2 tr:gt(3)").css({ background: 'lightgray' }); //초과

    /* 김치찌개 배경색 오랜지  = 같은 맥락이지만 밖으로.eq매서드 이용 */
    //$("#foods2 tr:eq(3)").css({ background: 'orange' });
    $("#foods2 tr").eq(3).css({ background: 'orange' });
 });
</script>
</head>

<body>
   <div class="card-body">
      <h3 class="card-title">좋아하는 음식들</h3>
      <table class="table" id="foods">
         <tr>
            <th>ID</th>
            <th>먹을거리</th>
            <th>분류</th>
         </tr>
```

```
        <tr>
          <td>F001</td>
          <td>짜장면</td>
          <td>중식</td>
        </tr>
        <tr>
          <td>F002</td>
          <td>라면</td>
          <td>분식,한식</td>
        </tr>
        <tr>
          <td>F003</td>
          <td>김치찌개</td>
          <td>한식</td>
        </tr>
        <tr>
          <td>F004</td>
          <td>돈까스</td>
          <td>일식</td>
        </tr>
        <tr>
          <td>F005</td>
          <td>초밥</td>
          <td>일식</td>
        </tr>
        <tr>
          <td>F006</td>
          <td>스테이크</td>
          <td>양식</td>
        </tr>
      </table>
      <br><br><br>
      <table class="table" id="foods2">
        <tr>
          <th>ID</th>
          <th>먹을거리</th>
```

```
            <th>분류</th>
        </tr>
        <tr>
            <td>F001</td>
            <td>짜장면</td>
            <td>중식</td>
        </tr>
        <tr>
            <td>F002</td>
            <td>라면</td>
            <td>분식,한식</td>
        </tr>
        <tr>
            <td>F003</td>
            <td>김치찌개</td>
            <td>한식</td>
        </tr>
        <tr>
            <td>F004</td>
            <td>돈까스</td>
            <td>일식</td>
        </tr>
        <tr>
            <td>F005</td>
            <td>초밥</td>
            <td>일식</td>
        </tr>
        <tr>
            <td>F006</td>
            <td>스테이크</td>
            <td>양식</td>
        </tr>
    </table>
</div>
```

1 짝수/홀수 행 선택

```
$('#foods > tbody > tr:even').css({ background: 'gray' });  // 짝수
$('#foods > tbody > tr:odd').css({ background: 'lightgray' }); //홀수
```

- `:even` → 짝수 인덱스 행 (0, 2, 4 ...)
- `:odd` → 홀수 인덱스 행 (1, 3, 5 ...)
- 결과: 테이블 줄무늬(회색/연회색) 배경 적용

---

2 첫 번째 행 선택

```
$('#foods tr:first').css({ background: 'black', color: 'white' }); //first
```

- `:first` → 첫 번째 행 선택 (`<th>` 제목 행)
- 결과: 검정 배경 + 흰색 글자

---

3 마지막 행 선택

```
$('#foods tr:last').css({ background: 'orange', color: 'white' }); //last
```

- `:last` → 마지막 행 선택 (`스테이크`)
- 결과: 주황색 배경 + 흰색 글자

---

4 특정 인덱스 번호 선택

```
$('#foods tr:eq(2)').css({ background: 'blue', color: 'white' }); //eq
```

- `:eq(2)` → 인덱스 2번째 행 선택 (0부터 시작)
- 결과: **3번째 행 →** `라면` **줄** → 파란색 배경 + 흰색 글자

5️⃣ 미만(lt), 초과(gt) 선택

```
$("#foods2 tr:lt(3)").css({ background: 'gray' }); //lt
$("#foods2 tr:gt(3)").css({ background: 'lightgray' }); //gt
```

- `:lt(3)` → 인덱스 3 미만(0,1,2번 행 = 제목+짜장면+라면) → 회색 배경
- `:gt(3)` → 인덱스 3 초과(4번 이후 행 = 초밥, 스테이크 등) → 연회색 배경

6️⃣ .eq() 메서드 방식

```
$("#foods2 tr").eq(3).css({ background: 'orange' }); //eq매서드
```

- `.eq(3)` : `foods2`의 3번째 행 선택
- 결과: `김치찌개` 줄 → 주황색 배경

🔷 선택자 개념 정리

- `:even / :odd` → 짝수/홀수 행 선택 (0부터 시작)
- `:first / :last` → 첫 번째, 마지막 요소 선택
- `:eq(n)` → n번째 요소 선택
- `:lt(n)` → n 미만 요소 선택
- `:gt(n)` → n 초과 요소 선택

📌6. 자식필터선택자

번호	국가명	수도
1	미국	워싱턴DC
2	프랑스	파리
3	영국	런던
4	중국	베이징
5	태국	방콕
6	모로코	라바트
7	라오스	비엔티안
8	베트남	하노이
9	피지	수바
10	자메이카	킹스턴
11	나미비아	빈트후크
12	멕시코	멕시코시티

번호	국가명	수도
1	미국	워싱턴DC
2	프랑스	파리
3	영국	런던
4	중국	베이징
5	태국	방콕
6	모로코	라바트
7	라오스	비엔티안
8	베트남	하노이
9	피지	수바

```
<script>
$(document).ready(function () {
    $("#nations tbody tr:nth-child(even)").css({ background: 'lightgray' }); //짝수
    $("#nations tbody tr:nth-child(odd)").css({ background: 'gray' }); // 홀수
```

```
    $("#nations tbody tr:nth-child(3)").css({ background: 'orange' }); //3번째
    $("#nations tbody tr:first-child").css({ background: 'black', color: 'white' });
//첫번째
    $("#nations tbody tr:last-child").css({ background: 'black', color: 'white' });
//마지막

    $("#nations2 tbody tr:nth-child(3n)").css({ background: 'orange' }); //3의배
수
    $("#nations2 tbody tr:nth-child(3n-1)").css({ background: 'gray' }); //3의전
배수
});
</script>
</head>

<body>
    <div class="card-body">
        <table class="table" id="nations">
            <thead>
                <tr>
                    <th>번호</th>
                    <th>국가명</th>
                    <th>수도</th>
                </tr>
            </thead>
            <tr>
                <td>1</td>
                <td>미국</td>
                <td>워싱턴DC</td>
            </tr>
            <tr>
                <td>2</td>
                <td>프랑스</td>
                <td>파리</td>
            </tr>
            <tr>
                <td>3</td>
```

```
            <td>영국</td>
            <td>런던</td>
        </tr>
        <tr>
            <td>4</td>
            <td>중국</td>
            <td>베이징</td>
        </tr>
        <tr>
            <td>5</td>
            <td>태국</td>
            <td>방콕</td>
        </tr>
        <tr>
            <td>6</td>
            <td>모로코</td>
            <td>라바트</td>
        </tr>
        <tr>
            <td>7</td>
            <td>라오스</td>
            <td>비엔티안</td>
        </tr>
        <tr>
            <td>8</td>
            <td>베트남</td>
            <td>하노이</td>
        </tr>
        <tr>
            <td>9</td>
            <td>피지</td>
            <td>수바</td>
        </tr>
        <tr>
            <td>10</td>
            <td>자메이카</td>
```

```
            <td>킹스턴</td>
        </tr>
        <tr>
            <td>11</td>
            <td>나미비아</td>
            <td>빈트후크</td>
        </tr>
        <tr>
            <td>12</td>
            <td>멕시코</td>
            <td>멕시코시티</td>
        </tr>
    </table>
    <br><br><br>

    <table class="table" id="nations2">
        <thead>
            <tr>
                <th>번호</th>
                <th>국가명</th>
                <th>수도</th>
            </tr>
        </thead>
        <tr>
            <td>1</td>
            <td>미국</td>
            <td>워싱턴DC</td>
        </tr>
        <tr>
            <td>2</td>
            <td>프랑스</td>
            <td>파리</td>
        </tr>
        <tr>
            <td>3</td>
            <td>영국</td>
```

```
            <td>런던</td>
        </tr>
        <tr>
            <td>4</td>
            <td>중국</td>
            <td>베이징</td>
        </tr>
        <tr>
            <td>5</td>
            <td>태국</td>
            <td>방콕</td>
        </tr>
        <tr>
            <td>6</td>
            <td>모로코</td>
            <td>라바트</td>
        </tr>
        <tr>
            <td>7</td>
            <td>라오스</td>
            <td>비엔티안</td>
        </tr>
        <tr>
            <td>8</td>
            <td>베트남</td>
            <td>하노이</td>
        </tr>
        <tr>
            <td>9</td>
            <td>피지</td>
            <td>수바</td>
        </tr>
        <tr>
            <td>10</td>
            <td>자메이카</td>
            <td>킹스턴</td>
```

```
        </tr>
        <tr>
            <td>11</td>
            <td>나미비아</td>
            <td>빈트후크</td>
        </tr>
        <tr>
            <td>12</td>
            <td>멕시코</td>
            <td>멕시코시티</td>
        </tr>
    </table>
```

🔷 1. `#nations` 테이블

1. `$("#nations tbody tr:nth-child(even)")` 짝수 번째
2. `$("#nations tbody tr:nth-child(odd)")` 홀수 번째
3. `$("#nations tbody tr:nth-child(3)")` 3번째 행
4. `$("#nations tbody tr:first-child")` 첫 번째 행
5. `$("#nations tbody tr:last-child")` 마지막 행

🔷 2. `#nations2` 테이블

1. `$("#nations2 tbody tr:nth-child(3n)")` 3의 배수. 행
2. `$("#nations2 tbody tr:nth-child(3n-1)")` 3의 배수 바로 전. 행

📌7. 문서객체 선택

문서객체 선택자1

문서객체 선택자2

ONE

- TWO
- THREE
- FOUR
- FIVE

has 메서드

내용1	내용2
내용1	내용2
내용1	내용2
내용1	내용2

find메서드 - FOUR

find메서드 - FIVE

find메서드 - ONE

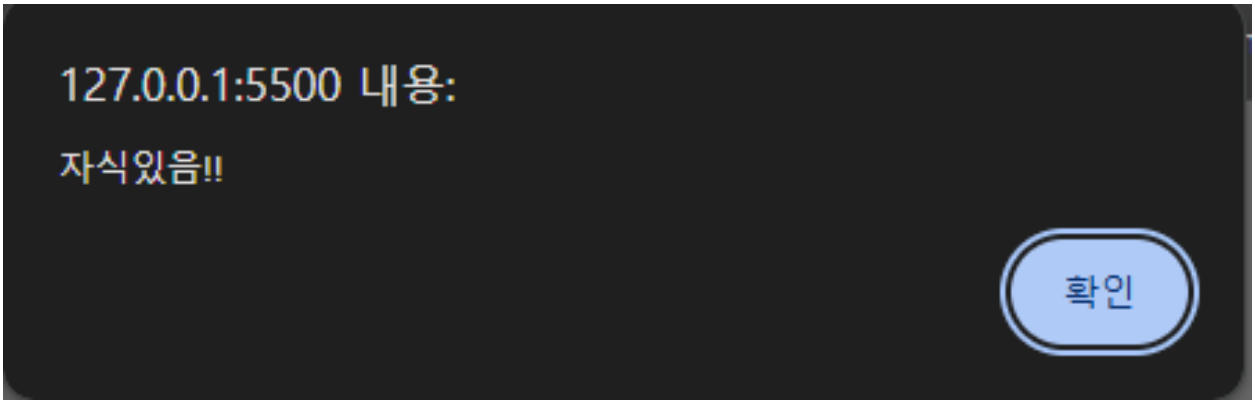
find메서드 - TWO

find메서드 - THREE

has메서드1

has메서드2

has메서드3




```
<script>
    $(document).ready(function(){
        /* 1. filter() : 기본 선택자를 사용해도 되지만, 배열과 관련된 DOM객체에서 특정 문서객체를 걸러내 선택할 목적으로 주로 사용 */
        $('li').css('color','red');
        $('li').filter(':even').css('color','blue');
```

```
        $('li').filter(':odd').css('color','red');

        /* 2. eq() : 요소들의 인덱스번호를 이용하여 선택 */
        $('li').eq(2).css('color','black');
        $('li').eq(-1).css({color:'orange', fontSize:24, fontWeight:'bold'});

        /* 3. first() / last() : 요소의 맨 처음과 맨 마지막을 선택 */
        $('li').first().css('background-color','green').css('color','white');
        $('li').last().css('background','yellow');

        /* 4. find() : 선택된 DOM객체의 (후손)문서객체를 선택 */
        $('body').find('span').css('color','cyan');
        $('h3').find('span').css('color','pink');
        $('h4').find('span').css('background','purple');
        $('h3,h4').find('span').css('border','1px solid #000');

        /* 5. is() : 주어진 매개변수와 하나라도 일치하면 true리턴 */
        /* 실습. h3,h4요소에 자식에 span 있는지를 판별하여
        자식으로 span이 있으면 (경고창)자식있음!! 출력
        없으면 (경고창)자식없음!! 출력
        is(), children(), 변수, if문 사용 | 소스캡쳐, 캡쳐 넘버링 */
        // 변수할당 = DOM 객체
        var isChild = $('h3,h4').children().is('span');
        if(isChild){
           alert('자식있음!!');
        }else{
           alert('자식없음!!');
        }

        /* 6. has() : 주어진 매개변수의 태그가 일치하면 선택후 적용 */
        $('div').css('color','lightgray').has('h2').css('color','blue');
    });
</script>
</head>
<body>
   <h2>문서객체 선택자1</h2>
```

```
    <h3>문서객체 선택자2</h3>
    <ul>
        <li>ONE</li>
        <li>TWO</li>
        <li><small>THREE</small></li>
        <li><small>FOUR</small></li>
        <li>FIVE</li>
        <h6>has 메서드</h6>
    </ul>
    <table class="table">
        <tr>
            <td>내용1</td>
            <td>내용2</td>
        </tr>
        <tr>
            <td>내용1</td>
            <td>내용2</td>
        </tr>
        <tr>
            <td>내용1</td>
            <td>내용2</td>
        </tr>
        <tr>
            <td>내용1</td>
            <td>내용2</td>
        </tr>
    </table>

    <h3>
        <span>find메서드 - FOUR</span>
    </h3>
    <h4>
        <span>find메서드 - FIVE</span>
    </h4>
    <h5>
        <span>find메서드 - ONE</span>
```

```
    </h5>
    <h5>find메서드 - TWO</h5>
    <h6>find메서드 - THREE</h6>

    <div>
      <h6>has메서드1</h6>
    </div>
    <div>
      <h6>has메서드2</h6>
    </div>
    <div>
      <h2>has메서드3</h2>
    </div>
</body>
```

1️⃣ .filter()

```
$('li').css('color','red');
$('li').filter(':even').css('color','blue');
$('li').filter(':odd').css('color','red');
```

- `filter("조건")` : 선택된 요소 중 특정 조건만 골라냄
- 모든 li → 빨강
- 짝수 li(0,2,4...) → 파랑
- 홀수 li(1,3,5...) → 빨강 유지

2️⃣ .eq()

```
$('li').eq(2).css('color','black');
$('li').eq(-1).css({color:'orange', fontSize:24, fontWeight:'bold'});
```

- `eq(n)` : 인덱스 n번째 요소 선택 (0부터 시작)

- `eq(2)` → 세 번째 li(THREE) → 검정색
- `eq(-1)` → 마지막 li(FIVE) → 주황색 + 크기 24px + 굵게

3️⃣ .first() / .last()

```
$('li').first().css('background-color','green').css('color','white');
$('li').last().css('background','yellow');
```

- `first()` → 첫 번째 li(ONE) → 초록 배경 + 흰 글자
- `last()` → 마지막 li(FIVE) → 노란 배경

4️⃣ .find()

```
$('body').find('span').css('color','cyan');
$('h3').find('span').css('color','pink');
$('h4').find('span').css('background','purple');
$('h3,h4').find('span').css('border','1px solid #000');
```

- `find("선택자")` : **후손 요소** 찾기
- body 안의 모든 span → 글자 청록
- h3 안 span → 글자 분홍
- h4 안 span → 배경 보라
- h3,h4 안 span → 검정 테두리

5️⃣ .is()

```
var isChild = $('h3,h4').children().is('span');
if(isChild){
    alert('자식있음!!');
}else{
```

```
    alert('자식없음!!');
}
```

- `is("조건")` : 요소 중 조건이 하나라도 맞으면 true 반환
- h3,h4의 직계 자식 중 span이 있는지 검사
- 있으면 → "자식있음!!"
- 없으면 → "자식없음!!"

6️⃣ .has()

```
$('div').css('color','lightgray').has('h2').css('color','orange');
```

- `has("선택자")` : 특정 자식을 포함하는 요소 선택
- div 모두 회색 글자
- 그 중 h2를 포함한 div → 주황색 글자

📌8. 문서객체조작


```
<style>
  .change_01 {
    color: blue;
  }

  .change_02 {
    background: yellow;
  }
</style>
  <script>
    $(document).ready(function () {
      /*1. addClass(): 선택 DOM객체에 지정된 클래스 추가 */
      $('h5').eq(0).addClass('change_01');
```

```
        $('h5').eq(1).addClass('change_02');
        /*2. remove(): 선택 DOM객체에 지정된 클래스 삭제 */
        $('h5').eq(1).removeClass('change_02'); // 클래스는 삭제되지만 , attr속성은 남는다.
        /*3. attr(): 선택 DOM객체에 속성값을 확인하거나 속성을 추가*/
        $('img').attr('width',300);
        $('img').attr('height',300);

        /* 실습. 마지막 이미지에 너비 200 높이 200 대체텍스트 제니사진으로 변경 코딩 = 한줄표기법 */
         $('img').eq(-1).attr({width:200, height:200, alt:'제니사진'});
    });
  </script>
</head>

<body>
  <h5>
    분명중에서도 가장 어리석고 못난 변명은
  </h5>
  <h5>
    &quot;시간이 없어서 &quot; 라고 하는 변명입니다.
  </h5>
  <h6>
    By Edison
  </h6>
  <img src="./images/img1.jpg" alt="이미지1">
  <img src="./images/img2.jpg" alt="이미지2">
  <img src="./images/img3.jpg" alt="이미지3">
</body>
```

1️⃣ .addClass()

```
$('h5').eq(0).addClass('change_01');
$('h5').eq(1).addClass('change_02');
```

- 첫 번째 `<h5>`에 `change_01` 클래스 추가 → 글자 파란색
- 두 번째 `<h5>`에 `change_02` 클래스 추가 → 배경 노란색

2️⃣ .removeClass()

```
$('h5').eq(1).removeClass('change_02');
```

- 두 번째 `<h5>`에서 `change_02` 클래스 제거
- ⚠️ 클래스는 삭제되지만, 이미 `attr`로 추가된 속성이 있으면 그건 남음

3️⃣ .attr()

```
$('img').attr('width',300);
$('img').attr('height',300);
```

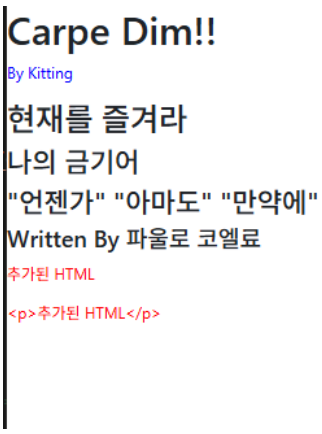
- 모든 `<img>` 요소에 **width=300, height=300** 속성 부여
- 결과: 모든 이미지 크기가 300×300으로 변경

4️⃣. 마지막 이미지 속성 한 줄 변경

```
$('img').eq(-1).attr({width:200, height:200, alt:'제니사진'});
```

- 마지막 `<img>`(세 번째 이미지) 선택
- width=200, height=200, alt="제니사진" 으로 변경
- 결과: 마지막 이미지는 크기가 200×200, 대체텍스트는 "제니사진"

📌9. 문서객체조작2




```
<script>
 $(document).ready(function () {
    /*1. html() : 선택된 DOM객체의 첫번째 요소, 문자열 반환 | 태그 인식*/
    var out = $('h3').html();
    //alert(out);

    /*2. text() : 선택된 DOM객체의 모든 문자열을 묶어서 반환 | 태그 인식 못함*/
    var out2 = $('h3').text();
    //alert(out2);

    //동적생성
    $('div').eq(-2).html('<p>추가된 HTML</p>').css({ color: 'red' });

    //동적생성2 text() 아코디언 내용
    $('div').eq(-1).text('<p>추가된 HTML</p>').css({ color: 'red' })

    // 실습. div태그의 자식으로 p태그사용하여 텍스트노드 By Kitting이 나오고, 글자색
블루
    $('div').eq(0).html('<p>By Kitting</p>').css({ color: 'blue' });
 });
</script>
</head>

<body>
   <h1>Carpe Dim!!</h1>
```

```
    <div></div>
    <h2>현재를 즐겨라</h2>
    <h3>나의 금기어</h3>
    <h3>
      &amp;quot;언젠가&amp;quot; &amp;quot;아마도&amp;quot; &amp;quot;만약에&amp;quot;
    </h3>
    <h4>
      Written By 파울로 코엘료
    </h4>
    <div></div>
    <div></div>
</body>
```

1 .html()

```
var out = $('h3').html();
//alert(out);
```

- `$('h3').html()` → **첫 번째 h3 요소 안의 HTML 코드**를 문자열로 가져옴.
- HTML 태그를 그대로 인식함.
- `alert(out)` 하면 `<h3>` 안의 내용이 태그 포함 형태로 출력됨.

---

2 .text()

```
var out2 = $('h3').text();
//alert(out2);
```

- `$('h3').text()` → **모든 h3의 텍스트만** 합쳐서 반환.
- 태그는 인식 못 하고 글자만 가져옴.
- `alert(out2)` 하면 `나의 금기어 "언젠가" "아마도" "만약에"` 같은 순수 텍스트만 출력됨.

---

3 동적생성 (html)

```
$('div').eq(-2).html('<p>추가된 HTML</p>').css({ color: 'red' });
```

- `eq(-2)` → 뒤에서 두 번째 div 선택
- 그 안에 `<p>추가된 HTML</p>` 삽입 (태그 인식 O)
- 글자색 빨강 적용

4️⃣ 동적생성2 (text)

```
$('div').eq(-1).text('<p>추가된 HTML</p>').css({ color: 'red' })
```

- `eq(-1)` → 마지막 div 선택
- `.text("<p>추가된 HTML</p>")` → HTML 태그를 단순한 **문자열**로 삽입 (태그 인식 X)
- 화면에는 `<p>추가된 HTML</p>` 그대로 글자로 출력되고 빨간색 적용됨

5️⃣ 실습

```
$('div').eq(0).html('<p>By Kitting</p>').css({ color: 'blue' });
```

- 첫 번째 div 선택
- 자식으로 `<p>` 태그 삽입, 내용 = `"By Kitting"`
- 글자색 파랑 적용

📌10. 문서객체제작

"언젠가" "아마도" "만약에"
Written By 파울로 코엘료
ONE
TOW
THREE
FOUR
Dead Poet Society

```
    <script>
        $(document).ready(function () {
            /* 1.remove() : 선택 DOM 객체 태그 제거 */

            $('h3').eq(0).remove(); //첫번째 h3 선택 삭제, 같은 삭제 기능 1
            //$('h3').first.remove();  // 같은 삭제 기능 2
            //$('h3:first').filter(0).remove(); //같은 삭제 기능 3

            /* 2.empty() : 자식노드를 제거 */
            $('div').empty(); // empty 비어있는 - ex) 쓰레기통은 남고 쓰레기는 node는 버리는

            /*
                3. $() = jQuery() //
                a. DOM객체 선택
                b. 태그 생성 : 매게변수로 HTML요소 문자열로 DOM객체 생성 가능
                    b-1. appendTo() : 매게변수의 맨 마지막 자식요소로 태그 생성
            */
            $('<h2>Dead Poet Society</h2>').appendTo('body').css({ color: 'purple' });
            $('<img>').attr({ src: './images/img1.jpg', alt: 'test', width: '300', height:
```

```
'300' }).appendTo('body');
      });
  </script>
</head>

<body>
  <div>
    <h1>Carpe Dim!!</h1>
    <h2>현재를 즐겨라</h2>
  </div>

  <h3>나의 금기어</h3> <!-- << 여기 해당 부분이 jquery로 인해 삭제됨 -->

  <h3>
    &quot;언젠가&quot; &quot;아마도&quot; &quot;만약에&quot;
  </h3>
  <h4>
    Written By 파울로 코엘료
  </h4>

  <h5>ONE</h5>
  <h5>TOW</h5>
  <h6>THREE</h6>
  <h6>FOUR</h6>

</body>
```

1 remove()

```
$('h3').eq(0).remove();  // 첫번째 요소 선택 이퀄(0); - 삭제
//$('h3').first.remove();  //  첫번째 요소 선택 first(); -  삭제
//$('h3:first').filter(0).remove();  첫번째 요소 선택 filter(0); - 삭제
```

- 첫 번째 `<h3>` 요소 자체를 통째로 삭제.

- 결과: `<h3>나의 금기어</h3>` 사라짐.

2️⃣ empty()

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

- `<div>` 태그 자체는 남기고, **자식 요소(h1, h2)**만 삭제.
- 결과: `<div>`는 존재하지만 안이 비어 있음.

3️⃣ $('<태그>...</태그>') : 동적 생성 (제작!!)

- `$('<h2>내용</h2>')` → 새로운 DOM 요소를 생성 가능.

```
$('<h2>Dead Poet Society</h2>').appendTo('body').css({ color: 'purple' });
```

- `<h2>` 요소를 **새로 만들어서 body 마지막에 추가**
- 글자색 보라색 적용

```
$('<img>').attr({ src: './images/img1.jpg', alt: 'test', width: '300', height: '300' }).appendTo('body');
```

- `<img>` 요소를 새로 만들어 `attr` 속성매서드안에 소스 넣고 body 마지막부분에 appendTo로 추가
- 속성 지정:
  - `src="./images/img1.jpg"`
  - `alt="test"`
  - `width=300`, `height=300`

📌11. 문서객체실습 (.eq(0) 이용하여_롤링)

```
    <style>
        img{
            width:240px;
            width:160px;
        }
    </style>
    <script>
        $(document).ready(function () {
            setInterval(function(){
                $('img').eq(0).appendTo('body');
                //이미지의 eq(첫번째)를 선택 바디 마지막 부분으로 요소로 이동
            },2000)
            // 2초마다 함수실행
        });
    </script>
</head>

<body>
    <img src="./images/img1.jpg" alt="이미지1">
    <img src="./images/img2.jpg" alt="이미지2">
    <img src="./images/img3.jpg" alt="이미지3">
</body>
```

- `setInterval(function(){...}, 2000)`

    - 2초(2000ms)마다 안의 함수 실행하며
- `$('img').eq(0)`
    - 현재 문서 안에서 **첫 번째 이미지(img)** 선택
- `.appendTo('body')`
    - 선택된 이미지를 `<body>`의 **맨 마지막 자식 요소로 이동**

📌12. 문서객체실습 (이미지_하나에 롤링)


```
<style>
    div {
        width: 960px;
        margin: 0 auto;
        position: relative;
    }

    img {
        width: 100%;
```

```
            position: absolute;
            left: 0;
            top: 0;
            z-index: 1;
        }
    </style>
    <script>
        $(document).ready(function () {
                // 이미지를 "어디에" 추가 한다.
            setInterval(function () {
                $('img').first().appendTo('div'); //첫번째 이미지를 div에 추가
            }, 2000); //2초마다

                    // "어디에"  이미지를 추가한다.
            /* setInterval(function () {
             $('div').append($('img').first());   //div에 첫번째 이미지를 추가
            }, 2000); */ //2초마다
        });
    </script>
</head>

<body>
    <div>
        <img src="./images/slam_glass.jpg" alt="슬램덩크1">
        <img src="./images/slam_jung.jpg" alt="슬램덩크2">
        <img src="./images/slam_kang.jpg" alt="슬램덩크1">
    </div>
</body>
```

1. `$('img').first().appendTo('div')`

- `$('img').first()` → 첫 번째 `<img>` 선택
- `.appendTo('div')` → 그 이미지를 `<div>`의 마지막 자식으로 이동

👉 **자식(img) 먼저 선택 → 부모(div) 안으로 이동**

2. `$('div').append($('img').first())`

- `$('div')` → 부모 `<div>` 선택
- `.append($('img').first())` → 그 안에 첫 번째 이미지를 추가

👉 **부모(div) 먼저 선택 → 자식(img)을 집어넣음**

✅ 실행 결과

- 2초마다 첫 번째 이미지가 맨 뒤로 이동
- 이미지들이 차례대로 순환하면서 보임 → **롤링 슬라이드 효과**

👏오늘 선택자 요약

- **태그 선택자** : `$("div")` → 특정 태그
- **부모 선택자** : `$("li").parent()` → 부모 요소
- **자식 선택자** : `$("ul>li")` → 직계 자식
- **형제 선택자** :
  - 인접 형제 : `$("#one+div")`
  - 모든 형제 : `$("#one~div")`
- **클래스 선택자** : `$(".class")`
- **아이디 선택자** : `$("#id")`
- **조합 선택자** :
  - 전체 : `$("*")`
  - 태그+아이디 : `$("div#id")`
  - 태그+클래스 : `$("div.class")`
- **계층 선택자** : `A > B` , `A + B` , `A ~ B`
- **필터 선택자** :
  - 짝수 : `tr:even`

- 홀수 : `tr:odd`
- 첫번째 : `tr:first`
- 마지막 : `tr:last`
- 특정 인덱스 : `tr:eq(n)`
- n 미만 : `tr:lt(n)`
- n 초과 : `tr:gt(n)`

📌 jQuery 주요 메서드

🔷 탐색 & 조건

- **.find("선택자")** → 하위(후손) 요소 찾기
- **.filter("조건")** → 선택된 요소 중 일부만 걸러내기
- **.has("선택자")** → 특정 자식을 포함하는 요소 선택

🔷 속성 & 값

- **.attr("속성","값")** → 속성 읽기/설정하기
- **.removeAttr("속성")** → 속성 제거
- ~~**.prop("속성")** → 속성(checked, selected 등) 상태 확인/변경~~
- ~~**.val()** → 폼 입력 값 가져오기/설정~~

🔷 인덱스 & 위치

- **.eq(n)** → n번째 요소 선택 (0부터 시작)
- **.eq(-1)** → 마지막 요소
- **.eq (-2)면 마지막요소의 두번째**
- **.first()** → 첫 번째 요소
- **.last()** → 마지막 요소

🔷 내용 & 구조

- **.text()** → 요소의 텍스트 가져오기/변경하기
- **.html()** → 요소의 HTML 가져오기/변경하기
- **.append("내용")** → 해당 (”내용”) 요소 **뒤쪽에 추가**
- **.appendTo("선택자")** → 해당 특정 요소에 (”선택자”) **자신을 붙이기**
- ~~**.prepend("내용")** → 선택한 요소 **앞쪽에 추가**~~
- ~~**.before("내용")** → 요소 **앞에 형제 추가**~~
- ~~**.after("내용")** → 요소 **뒤에 형제 추가**~~
- **.remove()** → 요소 삭제
- **.empty()** → 요소 내부 내용만 삭제

🔷 클래스 조작

- **.addClass("클래스명")** → 클래스 추가
- **.removeClass("클래스명")** → 클래스 제거
- **.toggleClass("클래스명")** → 클래스 토글

📌13. 효과 메서드 + 기본 이벤트

제이쿼리 효과

제이쿼리에서 화면전환, 슬라이드 다운과 업효과 등
다양한 효과 메서드를 제공
제이쿼리 코어 파일 버전에 따라 동작이 안될 경우도 있음

show / hide / toggle 효과

HIDE SHOW TOGGLE

DOM선택요소를 숨기거나 노출하고 싶을 경우에 사용
형식 : hide('시간') | show('시간') | toggle('시간') |
매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능

slideDown / slideUp / slideToggle 효과

slideDown slideUp slideToggle

slideUp은 DOM선택요소를 위로 미끄러지듯 말아 올리면서 숨김
slideDown DOM선택요소를 아래로 미끄러지듯 말아 내리면서 노출시킴
형식 : slideDown('시간') | slideUp('시간') | slideToggle('시간') |
매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능

fadeIn / fadeOut / fadeToggle 효과

fadeOut fadeIn fadeToggle

fadeOut DOM선택요소를 천천히 사라지게 함
fadeIn은 DOM선택요소를 천천히 나타나게 함
형식 : fadeIn('시간') | fadeOut('시간') | fadeToggle('시간') |
매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능

```
<script>
    $(document).ready(function () {
        /* show() / hide() / toggle() */
        $('#btn1').click(function () {
            $('.box1').hide();
        });
        $('#btn2').click(function () {
            $('.box1').show();
        });
        $('#btn3').click(function () {
            $('.box1').toggle();
        });
```

```
        $('#btn4').click(function () {
            $(this).parent().next().slideUp();
        });
        $('#btn5').click(function () {
            $(this).parent().next().slideDown();
        });
        $('#btn6').click(function () {
            $(this).parent().next().slideToggle();
        });

        $('#btn7').click(function () {
            $(this).parent().next().fadeOut();
        });
        $('#btn8').click(function () {
            $(this).parent().next().fadeIn();
        });
        $('#btn9').click(function () {
            $(this).parent().next().fadeToggle();
        });
    });
</script>
</head>

<body>
<div id="wrap">
  <h1>제이쿼리 효과</h1>
  <p>
      제이쿼리에서 화면전환, 슬라이드 다운과 업효과 등 <br>
      다양한 효과 메서드를 제공 <br>
      제이쿼리 코어 파일 버전에 따라 동작이 안될 경우도 있음
  </p>

  <h2>show / hide / toggle 효과</h2>
  <div>
      <button id="btn1">HIDE</button>
      <button id="btn2">SHOW</button>
```

```
    <button id="btn3">TOGGLE</button>
  </div>
  <p class="box1">
    DOM선택요소를 숨기거나 노출하고 싶을 경우에 사용<br>
    형식 : hide('시간') | show('시간') | toggle('시간') | <br>
    매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능
  </p>

  <h2>slideDown / slideUp / slideToggle 효과</h2>
  <p>
    <button id="btn4">slideDown</button>
    <button id="btn5">slideUp</button>
    <button id="btn6">slideToggle</button>
  </p>
  <p class="box2">
    slideUp은 DOM선택요소를 위로 미끄러지듯 말아 올리면서 숨김<br>
    slideDown DOM선택요소를 아래로 미끄러지듯 말아 내리면서 노출시킴<br>
    형식 : slideDown('시간') | slideUp('시간') | slideToggle('시간') | <br>
    매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능
  </p>

  <h2>fadeIn / fadeOut / fadeToggle 효과</h2>
  <p>
    <button id="btn7">fadeOut</button>
    <button id="btn8">fadeIn</button>
    <button id="btn9">fadeToggle</button>
  </p>
  <p class="box3">
    fadeOut DOM선택요소를 천천히 사라지게 함<br>
    fadeIn은 DOM선택요소를 천천히 나타나게 함<br>
    형식 : fadeIn('시간') | fadeOut('시간') | fadeToggle('시간') | <br>
    매개변수 '시간' = 'fast' 200, 'nomal' 400, 'slow' 600, 밀리세컨 입력 가능
  </p>
</div>
</body>
```

📌 jQuery 효과 메서드 요약

- **show()** → `display: block;` (보이게)
- **hide()** → `display: none;` (숨김)
- **toggle()** → `show ↔ hide;`(전등 "딸깍" 생각하면 편함)

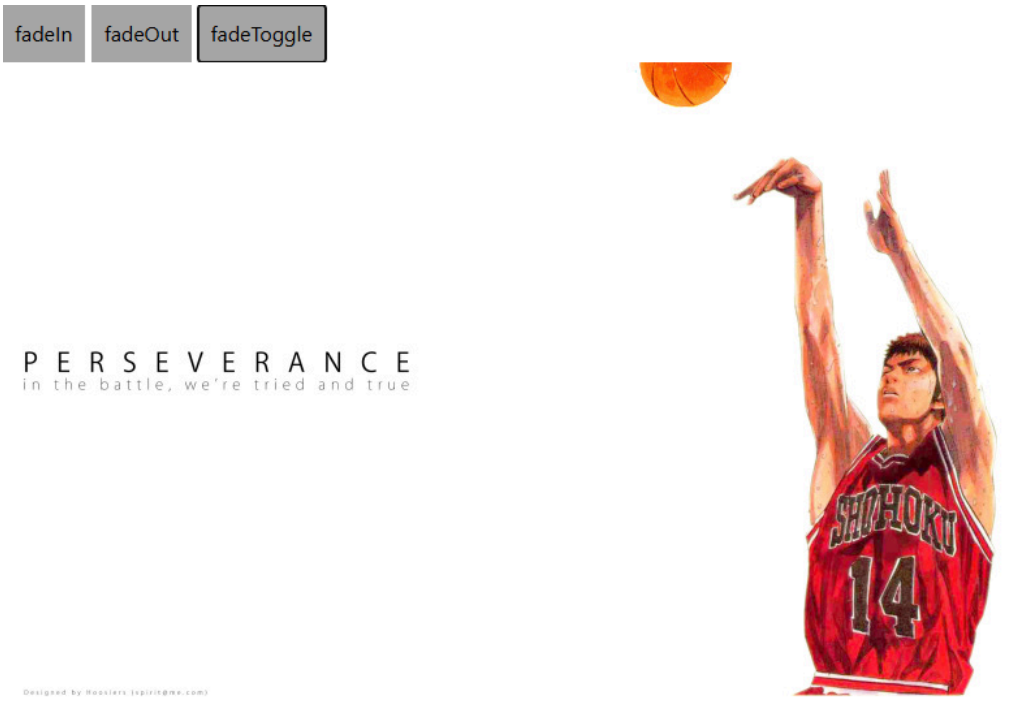
- **slideDown()** → 요소를 아래로 미끄러지듯 나타냄
- **slideUp()** → 요소를 위로 말아 올리며 숨김
- **slideToggle()** → slideDown ↔ slideUp 전환

- **fadeIn()** → 서서히 나타남 (opacity 0 → 1)
- **fadeOut()** → 서서히 사라짐 (opacity 1 → 0)
- **fadeToggle()** → fadeIn ↔ fadeOut 전환

👉 핵심

- show/hide = 단순 노출/숨김
- slide = 위/아래 슬라이드 효과
- fade = 투명도 변화 효과

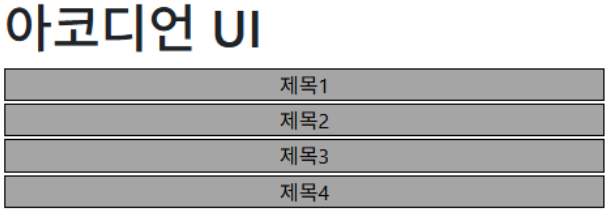
📌 1.4 클릭 이미지(IN,OUT)




```
<style>
    button {
        padding: 10px;
        background-color: darkgray;
        border: none;
        outline: none;
        color: #000;
        font-size: 1rem;
        cursor: pointer;
    }
    img {
        display: none;
    }
</style>
<script>
    $(document).ready(function(){
        $('button').eq(0).click(function(){ //버튼을 클릭했을때 eq 순서 따른 이미지
를
```

```
            $('p>img').fadeIn();// fadeIn
        });
        $('button').eq(1).click(function(){
            $('p>img').fadeOut(); //fadeOut
        });
        $('button').eq(2).click(function(){
            $('p>img').fadeToggle();//fadeToggle을 주는것
        });
    });
  </script>
</head>
<body>
  <button>fadeIn</button>
  <button>fadeOut</button>
  <button>fadeToggle</button>
  <p>
    <img src="./images/slam_jung.jpg" alt="효과이미지">
  </p>
</body>
```

📌15. 메뉴 실습(아코디언효과)



```
    <style>
        h2 {
            font-size: 1rem;
            font-weight: normal;
        }
        .accordion {
            width: 500px;
            font-size: 14px;
        }
        .header {
            border: 1px solid #000;
            text-align: center;
            padding: 2px;
            margin-bottom: 2px;
            background-color: darkgray;
            color: #000;
            cursor: pointer;
        }
        .content {
            border: 1px solid #000;
            padding: 5px;
            height: 100px;
        }
    </style>
    <script>
        $(document).ready(function(){
            $('#acc>.content').hide(); // 로드시 내용부분 먼저 가리기

            $('#acc>.header').click(function(){ // 제목 클릭시 이벤트 연결
                $(this).next().stop().slideToggle();
            });
        });
    </script>
</head>
<body>
```

```
    <div id="acc" class="card-body accordion">
      <h1>아코디언 UI</h1>
      <h2 class="header">제목1</h2>
      <p class="content">내용1</p>

      <h2 class="header">제목2</h2>
      <p class="content">내용2</p>

      <h2 class="header">제목3</h2>
      <p class="content">내용3</p>

      <h2 class="header">제목4</h2>
      <p class="content">내용4</p>
    </div>
  </body>
```

📌 16. 버튼_아코디언(클릭제외닫기)

버튼1
버튼2
버튼3
내용3

```
    <script>
      $(document).ready(function () {

        $("h2").click(function () {  // h2 클릭시
```

```
            $("p").slideUp(); // p 모든것을 먼저 slideUp 닫고,
            $(this).next("p").slideToggle(); // 클릭한 this 의 p를  slideToggle
        });

    });
  </script>
</head>

<body>
  <div id="wrap">
    <h2>
      <a href="#">버튼1</a>
    </h2>
    <p>
      내용1
    </p>
    <h2>
      <a href="#">버튼2</a>
    </h2>
    <p>
      내용2
    </p>
    <h2>
      <a href="#">버튼3</a>
    </h2>
    <p>
      내용3
    </p>
  </div>
</body>
```

📌 THIS 란?


```
<style>
   .clickTest {
     background: yellowgreen;
     padding: 10px;
     cursor: pointer;
     margin-bottom: 10px;
     border: solid 1px #000;
   }
  </style>
  <script>
   $(document).ready(function () {

     var elm = $('.clickTest');

     // elm.on('click', function () {
     //   elm.text('클릭하였습니다!!');
     //   // $(this).text('클릭하였습니다!!');
     // });

     //on() 사용 = id 속성 이용하여 이벤트 각각 연결
     // $("#div1").on("click", function () { $(this).text("클릭하였습니다!!"); });
```

```
    // $("#div2").on("click", function () { $(this).text("클릭하였습니다!!"); });
    // $("#div3").on("click", function () { $(this).text("클릭하였습니다!!"); });
    // $("#div4").on("click", function () { $(this).text("클릭하였습니다!!"); });
    // $("#div5").on("click", function () { $(this).text("클릭하였습니다!!"); });

    /*※중요※ 위의 불편한 중복을 해결한 방법이 this(클릭한:자기자신)임 */
    elm.on('click', function () {
      $(this).text($(this).attr('id') +'클릭하였습니다.!!');
    });

  });
 </script>
</head>

<body>
 <div id="div1" class="clickTest">div!!</div>
 <div id="div2" class="clickTest">div!!</div>
 <div id="div3" class="clickTest">div!!</div>
 <div id="div4" class="clickTest">div!!</div>
 <div id="div5" class="clickTest">div!!</div>

 <!-- div#div$@1*5 div를#div id값으로1부터 5개까지 만들겠다.-->

</body>
```

위의 불편한 중복을 해결한 방법이 this(자기자신)임

📌 애니메이션


```
<style>
 button {
   cursor: pointer;
 }

 div {
   width: 50px;
   height: 50px;
   margin: 20px;
   background-color: skyblue;
   cursor: pointer;
   position: absolute;
   left: 50px;
   top: 50px;
   z-index: 1;
 }
 .delay{
   width: 300px;
   height: 100px;
   background: purple;
   cursor: pointer;
   position: absolute;
   left: 50px;
   top: 150px;
```

```
    z-index: 1;
    display: none;
  }
 </style>
 <script>
  $(document).ready(function () {
   /*
    1.animation({속성명:속성값(숫자),시간,속도,콜백함수})
     a. 속성 : (반드시)매게변수값으로 입력(=>속성값이 숫자값으로 사용되는 속성들만 사용가능)
          => 제이쿼리 코어 플로그인일 경
     b. 시간 : 'slow', nomal','fast' 밀리세컨
     c. 속도 : swing(기본값), linear(동일속도)
     d-1. 콜백함수 : 해당 메서드가 실행이 된 이후 호출하여 사용되는 함수
     d-2. 인자로 쓰이는 함수가 콜백
   */

   // $('div').mouseover(function () {
   //   $(this).animate({ width: 100, height: 100 }, 'slow', 'linear');
   // });
   // $('div').mouseout(function () {
   //   $(this).animate({ width: 50, height: 50 }, 'slow', 'linear');
   // });

   /* 실습1. 버튼1 클릭할때마다 50px씩 오른쪽 */
   $('button').eq(0).click(function () {
    $('div').stop().animate({ left: '+=50' }, 'slow');
   })

   /* 실습2. 버튼2을 클릭할때마다 50px씩 왼쪽으로 */
   $('button').eq(1).click(function () {
    $('div').stop().animate({ left: '-=50' }, 'slow');
   })

   /* 실습3. 버튼3을 클릭하면 초기화*/
   $('button').eq(2).click(function () {
```

```
      $('div').animate({ left: '50px' }, 'slow');
    })

    /* 실습4 . 버튼4를 클릭하면 delay 1초 뒤 애니메이션 실행*/
    // $('button').eq(-1).click(function () {
    //   $('.delay').stop().delay(1000).animate({left:300},'slow');
    // })

    /* 실습4. DELAY 버튼 클릭시 delay클래스 박스 나오도록 코딩 | on() 사용 = 캡쳐, 넘버링 */
    $('button').eq(-1).on('click',function () {
      $('.delay').stop().show().delay(1000).animate({left:300},'slow');
    });


  });
 </script>

<body>
 <div></div>
 <div class="delay"></div>
 <!-- button{버튼$}*3 -->
 <button>버튼1</button>
 <button>버튼2</button>
 <button>버튼3</button>

 <button>DELAY</button>
 <!-- button{delay} -->


</body>
```

1. **.animate({속성명: 값}, 시간, 속도, 콜백)**

- **지정한 CSS 속성**을 **숫자 단위(px, %, em 등)** 로 부드럽게 변화시킴
- **형식**

```
$(선택자).animate(
  { width: "300px", opacity: 0.5 }, // 변화시킬 속성
  1000,                          // 시간(ms) → 1000 = 1초
  "swing",                        // 속도(easing) → "swing"(기본), "linear"
  function() {                    // 콜백
    alert("애니메이션 완료!");
  }
);
```

- **설명**
  - `{속성명: 값}` : CSS 속성(숫자 단위만 가능. 예: width, height, margin, padding, opacity 등)
  - `시간` : 밀리세컨드(ms) 또는 `"fast"(200ms), "normal"(400ms), "slow"(600ms)`
  - `속도` : easing 효과 ("swing"(기본, 가속/감속), "linear"(일정))
  - `콜백` : 애니메이션 완료 후 실행할 함수

---

2. .stop()

- 현재 실행 중인 애니메이션을 **즉시 멈춤**
- 기본적으로 **해당 요소의 모든 애니메이션 큐를 제거**
- 예:

```
$("#box").stop();          // 현재 애니메이션만 중지
$("#box").stop(true);      // 큐에 대기 중인 애니메이션도 전부 제거
$("#box").stop(true, true);// 즉시 현재 상태를 최종값으로 설정
```

---

3. .delay(시간)

- **애니메이션 실행 전 지연 시간**을 추가

- 시간 단위: ms(밀리세컨드)
- 예:

```
$("#box")
  .delay(1000)          // 1초 기다렸다가
  .fadeIn(500);         // 이후 실행
```

🚀 핵심 요약

- `.animate()` → 숫자 단위 속성 변화를 애니메이션으로 실행
- `.stop()` → 애니메이션 중단 (대기열 제거 옵션 있음)
- `.delay(1000)` → 실행 전 1000ms(=1초) 지연