**jQuery**

**1. 제이쿼리 사용시 장점**

**\* 호환성 문제 해결**

자바스크립트의 문서 객체모델(DOM, Document Object Model)은 브라우저간 호환성이 떨어져 크로스브라우징이 힘들다는 단점이 있습니다

(현재는 거의 대부분의 브라우저가 ECMAScript5까지는 지원을 하므로 크로스브라우징의 문제거 어느 정도 해결되었습니다 또한 개발시에 IE10 이하는 거의 고려하는 추세가 아닙니다.)

**\* 쉽고 편리한 애니메이션 효과 기능 구현**

자바스크립트로 애니메이션 효과를 구현하려면 많은 코드를 사용해야 하기 때문에 개발에 많은 시간이 필요했습니다. 그러나 제이쿼리는 애니메이션과 다양한 효과(Effect)를 지원하는 메서드를 제공하기 때문에 개발 시간을 많이 단축할 수 힜습니다.

**2. 제이쿼리 라이브러리 연동**

\* 다운로드 방식

제이쿼리 라이브러리를 제공하는 사이트에서 제이쿼리 라이브러리 파일을 직접 내려받아 HTML에 불러오는 방식

\* 네트워크 전송방식

온라인에서 제공하는 제이쿼리 라이브러리 파일을 네트워크를 통해 HTML에 불러오는 방식

**3. 선택자**

동적으로 스타일 및 동작을 적용하기 위해서는 HTML 요소를 선택해야 합니다.

**3.1 선택자 사용하기**

선택자를 사용하기 위해서는 문서 객체를 불러와야 합니다. 즉 HTML 태그가 모두 로딩이 되어야 HTML 요소를 선택할 수 있습니다.

$(document).ready(function() {

// jquery 선택자 소스

});

또는

$(function() {

});

상기 방식은 하기 소스와 동일하게 동작합니다.

window.addEventListener(”DOMContentLoaded”, function(e) {

});

**3.2 기본선택자**

|  |  |  |  |
| --- | --- | --- | --- |
| 구분 | 종류 | 사용법 | 설명 |
| 직접 선택자 | 전체선택자 | $(“\*”) | 모든 요소를 선택 |
| 아이디 선택자 | $(“#아이디명”) | id 속성에 지정한 값을 가진 요소를 선택 |
| 클래스 선택자 | $(“.클래스명”) | class 속성에 지정한 값을 가진 요소를 선택 |
| 요소 선택자 | $(“요소명”) | 지정한 요소명과 일치하는 요소들만 선택 |
| 그룹 선택자 | $(“선택1, 선택2, 선택3”) | 선택1, 선택2, 선택3 ... 선택 n에 지정된 요소들을 한꺼번에 선택 |
| 종속 선택자 | $(“p.txt\_1”) $(“p#txt\_1“) | <p> 요소 중 class 값이 txt\_1인 요소 또는 id 값이 txt\_1인 요소를 선택 |
| 인접  관계 선택자 | 부모요소 선택자 | $(“요소 선택”).parent() | 선택한 요소의 부모요소를 선택 |
| 상위요소 선택자 | $(“요소 선택”).parents() | 선택한 요소의 상위 요소를 모두 선택 |
| 가장 가까운 상위 요소 선택자 | $(“요소 선택”).closest(“div”) | 선택한 요소의 상위 요소 중 가장 가까운 <div>만 선택 |
| 하위요소 선택자 | $(“요소 선택 하위요소”) | 선택한 요소에 지정한 하위 요소를 선택 |
| 자식요소 선택자 | $(“요소 선택 > 자식요소”) | 선택한 요소를 기준으로 자식 관계에 지정한 요소만 선택 |
| 자식요소들 선택자 | $(“요소 선택”).children() | 선택한 요소의 모든 자식 요소를 선택 |
| 형(이전) 요소 선택자 | $(“요소 선택”).prev() | 선택한 요소의 바로 이전 요소를 선택 |
| 형(이전) 요소들 선택자 | $(“요소 선택”).prevAll() | 선택한 요소의 바로 이전 요소 모두를 선택 |
| 지정 형(이전) 요소들 선택자 | $(“요소 선택”).prevUntil(“요소명”) | 선택한 요소부터 지정한 요소의 이전 요소까지 모두 선택 |
| 동생(다음)요소 선택자 | $(“요소 선택”).next()  $(“요소 선택+다음요소”) | 선택한 요소의 다음 요소를 선택 |
| 동생(다음)요소들 선택자 | $(“요소 선택”).nextAll() | 선택한 요소의 다음 요소 모두를 선택 |
| 지정동생(다음) 요소들 선택자 | $(“요소 선택”).nextUntil(“h2”) | 선택한 요소부터 지정한 요소의 다음 요소까지 모두 선택 |
| 전체 형제 요소 선택자 | $(“.box\_1”).siblings() | class 값이 box\_1인 요소의 형제 요소 전체를 선택 |

**3.3 탐색 선택자**

탐색 선택자를 사용하시면 기본 선택자로 선택한 요소 중 원하는 요소를 한 번 더 탐색해서 좀 더 정확하게 선택할 수 있습니다

**3.3.1 위치 탐색 선택자**

기본 선택자로 선택한 요소는 배열에 담깁니다. 이때 배열의 인덱스를 사용하면 특정 요소를 좀 더 정확하게 선택 가능

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| $(“요소선택:first”) $(“요소선택”).fist() | $(“li:first”) $(“li“).first() | 전체 <li> 요소 중 첫 번째 요소만 선택합니다. |
| $(“요소선택:last”) $(“요소선택”).last() | $(“li:last”)  $(“li”).last() | 전체 <li> 요소 중 마지막 요소만을 선택합니다. |
| $(“요소선택:odd”) | $(“li:odd”) | <li>요소 무리 중 짝수 번째(홀수 인덱스)요소만 선택 |
| $(“요소선택:even”) | $(“li:even”) | <li>요소 무리 중 홀수 번째(짝수 인덱스)요소만 선택 |
| $(“요소선택:first-of-type”) | $(“li:first-of-type”) | <li>요소 무리 중 첫 번째 요소만 선택 |
| $(“요소선택:last-of-type”) | $(“li:last-of-type”) | <li>요소 무리 중 마지막 요소만 선택 |
| $(“요소선택:nth-child(숫자)”) | $(“li:nth-child(3)”) | <li>요소 무리 중 세번째 요소만 선택 |
| $(“요소선택:nth-child(숫자n)”) | $(“li:nth-child(3n)) | <li>요소 무리 중 3의 배수 번째 있는 요소만 선택 |
| $(“요소선택:nth-last-of-type(숫자)”) | $(“li:nth-last-of-type(2)”) | <li>요소 무리 중 마지막 위치로부터 두 번째 있는 요소만 선택 |
| $(“요소선택:only-child”) | $(“li:only-child”) | 부모 요소 내에 <li> 요소가 1개 뿐인 <li>요소만 선택 |
| $(“요소선택:eq(index)”)  $(“요소선택”).eq(index) | $(“li:eq(2)”)  $(“li”).eq(2) | <li>요소 중 인덱스 2가 참조하는 요소를 불러옵니다 |
| $(“요소선택:gt(index)”) | $(“li:gt(1)”) | <li>요소 중 인덱스 1보다 큰 인덱스가 참조하는 요소를 불러옵니다. |
| $(“요소선택:lt(index)”) | $(“li:lt(1)”) | <li>요소 중 인덱스 1보다 작은 인덱스가 참조하는 요소를 불러 옵니다. |
| $(“요소선택”).slice(index) | $(“li”).slice(2) | <li>요소 중 인덱스 2부터 참조하는 요소를 불러옵니다. |

**3.3.2 속성 탐색 선택자**

속성 탐색 선택자는 선택한 요소를 기준으로 일치하는 속성의 포함 여부를 따져 요소를 선택하는 선택자

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| $(“요소선택[속성]”) | $(“li[title]“) | <li>요소 중 title 속성이 포함된 요소만 선택 |
| $(“요소선택[속성=‘값’]”) | $(“li[title=’리스트‘]”) | <li>요소 중 title 속성 값이 ‘리스트’인 요소만 선택 |
| $(“요소선택[속성^=‘텍스트’]”) | $(“a[href^=’http://‘]”) | <li>요소 중 href 속성값이 ‘http://’로 시작하는 요소만 선택 |
| $(“요소선택[속성$=‘텍스트’]”) | $(“a[href$=’.com’]”) | <li>요소 중 href 속성값이 ‘.com’으로 끝나는 요소만 선택 |
| $(“요소선택[속성\*=‘텍스트’]”) | $(“a[href\*=’easypub’]”) | <li>요소 중 href 속성값 중에서 ‘easyspuyb’을 포함하는 요소만 선택 |
| $(“요소선택:hidden”) | $(“li:hidden”) | <li>요소 중 숨겨져 있는 요소만 선택 |
| $(“요소선택:visible”) | $(“li:visible”) | <li>요소 중 보이는 요소만 선택 |
| $(“:text”) | $(“:text”) | <input> 요소 중 type 속성값이 “text”인 요소만 선택 |
| $(“:selected”) | $(“:selected”) | selected 속성이 적용된 요소만 선택 |
| $(“:checked”) | $(“:checked”) | checked 속성이 적용된 요소만 선택 |

**3.3.3 컨텐츠 탐색 선택자**

콘텐츠 탐색 선택자는 요소 또는 속성의 포함 여부에 따라 특정 요소를 선택하는 선택자입니다.

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| $(“요소선택:contains(‘텍스트’)”) | $(“li:contains(‘내용2’)”) | <li> 요소 중 ‘내용2’라는 텍스트를 포함하는 요소만 선택 |
| $(“요소선택”).contents() | $(“p”).contents() | 선택한 요소의 하위 요소 중 가장 가까운 하위 요소를 선택 |
| $(“요소선택:has(‘요소명’)”)  $(“요소선택”).has(“요소명”) | $(“li:has(‘span’)”)  $(“li”).has(“span”) | <li>요소 중 <span>을 포함하는 요소만 선택 |
| $(“요소선택:not(‘제외요소’)”)  $(“요소선택”).not(“제외요소”) | $(“li:not(:first)”) | <li>요소 중 첫 번째 요소만 제외하고 선택 |
| $(“요소선택”).filter(“필터요소”) | $(“li”).filter(“.list2”) | <li>요소 중 class 값이 “list2”인 요소만 선택 |
| $(“요소선택1”).find(“요소선택2”) | $(“li”).find(“strong”) | <li>요소 중 하위 요소인 <strong>만 선택합니다. |
| $(“요소선택1”).closest(“요소선택2”) | $(“strong”).closest(“div”) | <strong>을 감싸는 <div> 요소 중 가장 까까운 상위 요소를 선택합니다. |
| end() | $(“li”).children(“a”).end() | 필터링이 실행되기 이전의 요소인 <li>가 선택됩니다. |

**3.3.4 제이쿼리 배열관련 메서드**

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| each() / $.each() | $(“요소선택”).each(function)  $.each($(“요소선택”), function) | 배열에 저장된 문서 객체만큼 메서드가 반복 실행됩니다. 배열에 저장된 객체의 인덱스를 순서대로 하나씩 접근하여 객체를 선택하고 인덱스를 구합니다. |
| $.map() | $.map(Array, function) | 배열에 저장된 데이터 수 만큼 메서드가 반복 실행됩니다. 함수에 반환된 데이터는 새 배열에 순서대로 저장됩니다. 새로 저장된 배열 객체를 반환합니다. |
| $.grep() | $.grep(Array, function) | 배열이 지정된 데이터 수 만큼 메서드가 반복 실행됩니다. 반환값이 true인 경우에만 배열의 데이터가 인덱스 오름차순으로 새 배열에 저장되며, 그 배열을 반환합니다. |
| $.inArray() | $.inArray(data, Array, start index) | 배열 안에서 데이터를 찾습니다. 데이터를 찾으면 가장 맨 앞 데이터의 인덱스를 반환하고, 찾지 못하면 –1을 반환합니다. start index의 값을 지정하면 해당 위치부터 데이터를 찾습니다. |
| $.isArray() | $.isArray(object) | 입력한 객체가 배열 객체라면 true를, 아니면 false를 반환 |
| $.merge() | $.merge(Array1, Array2) | 인자값으로 입력한 2개의 배열 객체를 하나로 그룹화합니다. |
| index() | $(“요소선택”).index(“지정 요소선택”) | 선택자로 요소를 먼저 선택합니다. 그런 다음 지정한 요소의 인덱스 정보를 가져옵니다. |

**3.3.5 선택자와 함께 알아두면 유용한 메서드**

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| is(“요소 상태”) | $(“.txt1”).is(:visible“) | 선택한 요소가 보이면 true를 반환 |
| $.noConflict() | var 변수 = $.noConflict()  변수(“요소 선택”) | $.noConflict() 함수를 이용하면 현재 제이쿼리에서 사용 중인 $ 메서드 사용을 중단하고 새로 지정한 변수명 메서드를 사용 |
| get() | $(“요소 선택”).get(0)  var el = $(“요소선택).get(0);  el.style.color = “#f00”; | 선택자에 get(0)을 적용하면 자바스크립트 DOM 방식의 스타일을 적용할 수 있다  **(제이쿼리로 선택한 요소는 자바스크립트 DOM 방식의 스타일을 사용할 수 없습니다.)** |

is() 메서드

선택한 요소의 상태가 지정한 속성과 일치하면 true를 반환, 그렇지 않다면 false를 반환

체크 선택 보임 안보임 움직임

$(“요소선택”).is(“:[checked | selected | visible | hidden | animated]”)

**4. 객체 조작**

**4.1 속성 조작 메서드**

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| html()  html(“새요소”) | $(“요소선택”).html();  $(“요소선택”).html(“새 요소”) | 선택한 요소의 하위 요소를 가져옵니다.  선택한 하위 요소를 모두 제거하고, 그 위치에 지정한 새 요소를 생성합니다. |
| text()  text(“새 텍스트”) | $(“요소선택”).text();  $(“요소선택”).text(“새 텍스트”); | 선택한 요소가 감싸는 모든 텍스트를 가져옵니다.  선택한 요소의 하위요소를 모두 제거하고 그 위치에 지정한 새 텍스트를 생성합니다. |
| attr(“속성명”)  attr(“속성명”, “새 값”) | $(“요소선택”).attr(“속성명”);  $(“요소선택”).attr(“속성명”, “새 값”); | 선택한 요소의 지정한 속성(attribute)값을 가져옵니다.  요소를 선택하여 지정한 속성에 새 값을 적용합니다. |
| removeAttr(“속성명”) | $(“요소선택”).removeAttr(“속성명”); | 선택한 요소의 지정한 속성만 제거합니다. |
| prop(“상태 속성명”)  prop(“상태 속성명”, 새값) | $(“요소선택”).prop(“상태속성명”);  $(“요소선택”).prop(“상태속성명”,“새 값”); | 선택한 요소의 상태 속성값을 가져옵니다.  폼 요소를 선택하여 value 속성에 새 값을 적용 |
| val()  val(“새 값”) | $(“요소선택”).val();  $(“요소선택”).val(“새 값”); | 선택한 폼 요소의 value 값을 가져옵니다.  폼 요소를 선택하여 value 속성에 새 값을 적용 |
| css(“속성명”)  css(“속성명”, “새 값”) | $(“요소선택”).css(“속성명”);  $(“요소선택”).css(“속성명”, “새 값”); | 선택한 요소의 지정한 스타일(CSS) 속성값을 가져옵니다.  요소를 선택하여 지정한 스타일(CSS) 속성에 새 값을 적용 |
| addClass(“class 값”) | $(“요소선택”).addClass(“class값”); | 선택한 요소의 class 속성에 새 값을 추가합니다. |
| removeClass(“class 값”) | $(“요소선택”).removeClass(“class값”); | 선택한 요소의 class 속성에 지정한 값만 제거합니다. |
| toggleClass(“class 값”) | $(“요소선택”).toggleClass(“class값”); | 선택한 요소에 class값에 지정한 값이 포함되어 있으면 제거하고 속성값이 없으면 추가합니다. |
| hasClass(“class 값”) | $(“요소선택”).hasClass(“class 값”); | 선택한 요소의 class 값에 지정한 클래스 값이 포함되어 있으면 true, 없으면 false를 반환 |

**4.2. 수치 조작 메서드**

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| height() | $(“요소선택”).height();  $(“요소선택”).height(100); | 안쪽 여백과 선을 제외한 높이값을 반환하거나 변환합니다. |
| width() | $(“요소선택”).width();  $(“요소선택”).width(100) | 안쪽 여백과 선을 제외한 너빗값을 반환하거나 변환합니다. |
| innerHeight() | $(”요소선택”).innerHeight();  $(”요소선택”).innerHeight(300); | 안쪽 여백을 포함한 높이값을 반환하거나 변환합니다. |
| innerWidth() | $(”요소선택”).innerWidth();  $(”요소선택”).innerWidth(100); | 안쪽 여백을 포함한 너비값을 반환하거나 변환합니다. |
| outerWidth() | $(”요소선택”).outerWidth();  $(”요소선택”).innerWidth(100); | 선과 안쪽 여백을 포함한 너비값을 반환하거나 변환합니다. |
| outerHeight() | $(”요소선택”).outerHeight();  $(”요소선택”).outerHeight(100); | 선과 안쪽 여백을 포함한 높이값을 반환하거나 변환합니다. |
| position() | $(”요소선택”).position().left;  $(”요소선택”).position().top; | 선택한 요소의 포지션 위칫값을 반환합니다. |
| offset() | $(”요소선택”).offset().left;  $(”요소선태”).offset().top; | 선택한 요소가 문서에서 수평/수직으로 얼마나 떨어져 있는지에 대한 값을 반환합니다. |
| scrollLeft() | $(”요소선택”).scrollLeft(); | 브라우저의 수직 스크롤 이동 너빗값을 반환합니다. |
| scrollTop() | $(”요소선택”).scrollTop(); | 브라우저의 수평 스크롤 이동 높이값을 반환합니다. |

**4.3 객체 편집 메서드**

선택한 요소를 복사하거나 새 요소를 생성하는 메서드와 복제하거나 새로 생성한 요소를 의도한 위치에 삽입하고 선택한 요소를 삭제

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| before() | $(”요소선택”).before(”새 요소”); | 선택한 요소의 이전 위치에 새 요소를 추가합니다. |
| after() | $(”요소선택”).after(”새 요소”); | 선택한 요소의 다음 위치에 새 요소를 추가합니다. |
| append() | $(”요소선택”).append(”새요소”); | 선택한 요소의 마지막 위치에 새 요소를 추가합니다. |
| appendTo() | $(”새요소”).appendTo(”요소선택”); | 선택한 요소의 마지막 위치에 새 요소를 추가합니다. |
| prepend() | $(”요소선택”).prepend(”새 요소”); | 선택한 요소의 맨 앞 위치에 새 요소를 추가합니다. |
| prependTo() | $(”새요소”).prependTo(”요소선택”) | 선택한 요소의 맨 앞 위치에 새 요소를 추가합니다. |
| insertBefore() | $(”새요소”).insertBefore(”요소선택”) | 선택한 요소의 이전 위치에 새 요소를 추가합니다. |
| insertAfter() | $(”새요소”).insertAfter(”요소선택”); | 선택한 요소의 다음 위치에 새 요소를 추가합니다. |
| clone() | $(”요소선택”).clone(true or false); | 선택한 문서 객체를 복사합니다. 이때 인자값이 true일 경우 하위 요소까지 모두 복제하고, false일 경우에는 선택한 요소만 복제합니다. |
| empty() | $(”요소선택”).empty(); | 선택한 요소의 하위내용들을 모두 삭제합니다. |
| remove() | $(”요소선택”).remove(); | 선택한 요소를 삭제합니다. |
| replaceAll()  replaceWith() | $(”새요소”).replaceAll(”요소 선택”);  $(”요소선택”).replaceWith(”새요소”); | 선택한 요소들을 새 요소로 교체합니다, |
| unwrap() | $(”요소선택”).unwrap(); | 선택한 요소의 부모요소를 삭제합니다. |
| wrap() | $(”요소선택”).wrap(”새요소”); | 선택한 요소를 새 요소로 각각 감쌉니다. |
| wrapAll() | $(”요소선택”).wrapAll(새 요소); | 선택한 요소를 새 요소로 한꺼번에 감쌉니다. |
| wrapInner() | $(”요소선택”).wrapInner(”새요소”); | 선택한 요소의 내용을 새 요소르 각각 감쌉니다. |

**5. 제이쿼리 이벤트**

**5.1 이벤트 등록 메서드**

\* 이벤트 - 사이트에서 방문한 사용자가 취하는 모든 행위를 말함

\* 이벤트핸들러 - 이벤트가 발생했을 때 실행되는 코드

|  |  |  |
| --- | --- | --- |
| 구분 | 종류 | 설명 |
| 로딩 이벤트 | load() | 선택한 이미지 또는 프레임 요소에 연동된 소스의 로딩이 완료된 후 이벤트가 발생합니다. |
| ready() | 지정된 HTML 문서 객체의 로딩이 완료된 후 이벤트가 발생합니다. |
| error() | 이벤트 대상 요소에서 오류가 발생하면 이벤트가 발생합니다. |
| 마우스  이벤트 | click() | 선택한 요소를 클릭했을 때 이벤트가 발생합니다. |
| dblclick() | 선택한 요소를 연속해서 두번 클릭했을 때 이벤트가 발생합니다. |
| mouseout() | 선택한 요소의 영역에서 마우스 포인트가 벗어났을 때 이벤트가 밸생합니다. 이때 하위 요소의 영향을 받습니다. |
| mouseover() | 선택한 요소의 영역에 마우스 포인터가 벗어 났을 때 이벤트가 발생합니다. 이때 하위 요소의 영향을 받습니다. |
| hover() | 선택한 요소에 마우스 포인터를 올렸을 때와 벗어났을 때 각각 이벤트가 발생합니다. |
| mousedown() | 선택한 요소에서 마우스 버튼을 눌렀을 때 이벤트가 발생합니다. |
| mouseup() | 선택한 요소에서 마우스 버튼을 눌렀다 떼었을 때 이벤트가 발생합니다. |
| mouseenter() | 선택한 요소 범위에 마우스 포인터를 올렸을 때 이벤트가 발생합니다. |
| mouseleave() | 선택한 요소 범위에서 마우스 포인터가 벗어났을 때 이벤트가 발생합니다. |
| mousemove() | 선택한 요소 범위에서 마우스 포인터를 움직였을 때 이벤트가 발생합니다. |
| scroll() | 가로, 세로 스크롤바를 움직일 때마다 이벤트가 발생합니다. |
| 포커스  이벤트 | focus() | 선택한 요소에 포커스가 생성되었을 때 이벤트를 발생하거나 선택한 요소에 강제로 포커스를 생성합니다. |
| focusin() | 선택한 요소에 포커스가 생성되었을 때 이벤트가 발생합니다. |
| focusout() | 포커스가 선택한 요소에서 다른 요소로 이동되었을 때 이벤트가 발생합니다. |
| blur() | 포커스가 선택한 요소에서 다른 요소로 이동되었을 때 이벤트가 발생하거나 선택한 요소의 포커스가 강제로 사라지도록 합니다. |
| change() | 이벤트 대상인 입력 요소의 값이 변경되고, 포커스가 이동하면 이벤트가 발생합니다. 그리고 강제로 change 이벤트를 발생시킬 때도 사용합니다. |
| 키보드  이벤트 | keypress() | 선택한 요소에서 키보드를 눌렀을 때 이벤트가 발생합니다. 그리고 문자 키를 제외한 키의 코드 값을 반환합니다. |
| keydown() | 선택한 요소에서 키보드를 눌렀을 때 이벤트가 발생합니다. 키보드의 모든 키의 코드값을 반환합니다. |
| keyup() | 선택한 요소에서 키보드에서 손을 떼었을때 발생합니다. |

**5.2 이벤트 등록방식**

**5.2.1 단독 이벤트 등록 메서드**

단독 이벤트 등록 메서드는 대상에 한 가지 동작에 대한 이벤트만 등록할 수 있습니다.

$(“이벤트 대상 선택”).이벤트 등록 메서드(function() {

자바스크립트 코드;

});

**5.2.2 그룹 이벤트 등록 메서드**

그룹 이벤트 등록 메서드는 대상에 한 가지 동작 이상의 이벤트를 등록할 수 있습니다.

on() 메서드를 사용하여 이벤트를 등록

**등록방식1**

$(“이벤트 대상 선택”).on(“이벤트 종류1 이벤트 종류2 ... 이베트 종류n”, function() {

자바스크립트 코드;  
});

**등록방식2**

$(“이벤트 대상 선택”).on({  
 “이벤트 종류1 이벤트 종류2... 이벤트 종류n ” : function() {

자바스크립트 코드;

}  
});

**등록방식3**

$(”이벤트 대상 선택”).on({

“이벤트 종류1” : function() { 자바스크립트 코드1; },

“이벤트 종류2” : function() { 자바스크립트 코드2; },

...

“이벤트 종류n” : function() { 자바스크립트 코드n; }

});

**5.2.3 강제로 이벤트 발생시키기**

단독 이벤트 등록 메서드를 사용하거나 trigger() 메서드를 사용하면 강제로 이벤트를 발생시킬 수 있습니다.

$(”이벤트 대상”).단독 이벤트 등록 메서드();

$(”이벤트 대상”).trigger(”이벤트 종류”);

**5.2.4 이벤트 제거 메서드**

이벤트를 제거하는 메서드로 off()가 있습니다.

$(”이벤트 대상”).off(”제거할 이벤트 종류”);

**5.2.5 로딩 이벤트 메서드**

로딩 이벤트 메서드는 사용자가 브라우저에서 HTML 문서를 요청하여 HTML 문서의 로딩이 완료되면 이벤트 핸들러를 실행합니다. 로딩 이벤트 메서드에는 ready()와 load()가 있습니다.

$(document).ready(function() { 자바스크립트 코드; });

$(document).on(”ready”, function() { 자바스크립트 코드; });

$(window).load(function() { 자바스크립트 코드; });

$(window).on(”load”, function() { 자바스크립트 코드; });

**5.2.6 이벤트 객체와 종류**

사용자가 이벤트를 발생시킬 때마다 이벤트 핸들러의 매개변수에는 이벤트 객체가 생성이 되며, 이벤트 객체에는 이벤트 타입에 맞는 다양한 정보를 제공하는 속성과 메서드가 포함되어 있습니다.

$(”이벤트 대상 선택”).mousemove(function(매개변수) {

매개변수(이벤트 객체).속성;

});

|  |  |  |
| --- | --- | --- |
| 구분 | 종류 | 설명 |
| 마우스 이벤트 | clientX | 마우스 포인터의 X좌푯값 반환(스크롤 이동 거리 무시) |
| clientY | 마우스 포인터의 Y좌표값 반환(스크롤 이동 거리 무시) |
| pageX | 스크롤 X축의 이동한 거리를 계산하여 마우스 포인터의 X좌표값을 반환 |
| pageY | 스크롤 Y축의 이동한 거리를 계산하여 마우스 포인터의 Y기본값을 반환 |
| screenX | 화면 모니터를 기준으로 마우스 포인터의 X 좌표값을 반환 |
| screenY | 화면 모니터를 기준으로 마우스 포인터의 Y 좌표값을 반환 |
| layerX | position을 적용한 요소를 기준으로 마우스 포인터의 X 좌표값을 반환 |
| layerY | position을 적용한 요소를 기준으로 마우스 포인터의 Y 좌표값을 반환 |
| button | 마우스 버튼의 종류에 따라 값을 반환(왼쪽 0, 휠 1, 오른쪽 2) |
| 키보드 이벤트 | keyCode | 키보드의 아스키 코드값을 반환 |
| altKey | 이벤트 발생 시 Alt키가 눌렸으면 true, 아니면 false를 반환 |
| ctrlKey | 이벤트 발생 시 Ctrl키가 눌렸으면 true, 아니면 false를 반환 |
| shiftKey | 이벤트 발생 시 Shift키가 눌렸으면 true, 아니면 false를 반환 |
| 전체 이벤트 | target | 이벤트가 전파된 마지막 요소를 가리킵니다. |
| cancelBubble | 이벤트 전파를 차단하는 속성으로 기본값은 false며, true로 설정하면 전파가 차단됩니다. |
| stopPropagation | 이벤트 전파를 차단합니다. |
| preventDefault() | 기본 이벤트를 차단합니다. 예를 들어 <a>에 클릭 이벤트를 적용하고 사용자가 이벤트를 발생시키면 기본 이벤트가 등록되어 있어 링크 주소로 이용하는데, 이런 기본 이벤트를 차단할 수 있습니다. |

**5.2.7 이벤트가 발생한 요소 추적하기**

**5.2.7.1 $(this) 선택자**

이벤트 핸들러에서 $(this)를 사용하면 이벤트가 발생한 요소를 선택하여 이벤트가 발생한 요소를 추적할 수 있습니다.

**5.2.7.2 index() 인덱스 반환 메서드**

index() 인덱스 반환 메서드는 이벤트를 등록한 요소 중 이벤트가 발생한 요소의 인덱스값을 반환 합니다.

$(”이벤트 대상 선택”).on(”이벤트 종류”, function() {  
 $(”이벤트 대상 선택”).index(this);  
});

**5.2.8 그룹 이벤트 등록 및 삭제하기**

그룹 이벤트 등록 메서드를 사용하면 한 번에 2개 이상 이벤트를 등록할 수 있습니다.

**\* 그룹 이벤트 등록 메서드**

|  |  |
| --- | --- |
| 종류 | 설명 |
| on() | 이벤트 대상 요소에 2개 이상의 이벤트를 등록합니다. 사용 방식에 따라 이벤트를 등록한 이후에도 동적으로 생성되거나 복제된 요소에도 이벤트가 적용됩니다.  새롭게 생성, 복제된 요소에 이벤트 등록  $([document|”이벤트 대상의 상위 요소”]).on(”이벤트 종류”, “이벤트 대상 선택”, function() {  자바스크립트 코드; }); |
| bind() | 이벤트 대상 요소에 2개 이상의 이벤트를 등록합니다. |
| delegate() | 선택한 요소의 하위 요소에 이벤트를 등록합니다. 이벤트를 등록한 이후에도 동적으로 생성되거나 복제된 요소에도 이벤트가 적용됩니다.  $([document|”이벤트 대상의 상위 요소 선택”]).delegate(”이벤트 대상 요소”, “이벤트 종류”, function() {  자바스크립트 코드;  }); |
| one() | 이벤트 대상 요소에 1개 이상의 이벤트를 등록합니다. 지정한 이벤트가 1회 발생하고 자동으로 해제 됩니다.  \* 기본 이벤트 등록 방식  $(”이벤트 대상 선택”).one(”이벤트 종류”, function() {  자바스크립트 코드; });  \* one() 라이브 이벤트 등록 방식  $([document | “이벤트 대상의 상위 요소 선택”]).one(”이벤트 종류”, “이벤트 대상요소 선택”, function() {  자바스크립트 코드;  }); |

**\* 이벤트 제거 메서드**

|  |  |
| --- | --- |
| 종류 | 설명 |
| off() | on() 메서드로 등록한 이벤트를 제거합니다.  **\* 기본 이벤트 제거 방식**  $(”이벤트 대상 요소 선택”).off(”이벤트 종류”);  **\* 라이브 이벤트 제거 방식**  $([document|“이벤트 대상 상위 요소 선택”]).off(”이벤트 종류”, “이벤트 대상 요소 선택”); |
| unbind() | bind() 메서드로 등록한 이벤트를 제거합니다.  $(”이벤트 대상 요소 선택”).unbind(”이벤트 종류”); |
| undelegate() | delegate() 메서드로 등록한 이벤트를 제거합니다.  **\* 기본 이벤트 제거 방식**  $(”이벤트 대상 요소 선택”).delegate(”이벤트 종류”);  **\* 라이브 이벤트 제거 방식**  $([document | “이벤트 대상의 상위 요소 선택”]).undelegate(”이벤트 대상 선택”, “이벤트 종류”); |

**6. 다양한 효과와 애니메이션**

**6.1 효과 메서드**

선택한 요소를 역동적으로 숨겼다가 보이게 만드는 기능을 가진 메서드

|  |  |  |
| --- | --- | --- |
| 구분 | 종류 | 설명 |
| 숨김 | hide() | 요소를 숨깁니다. |
| fadeOut() | 요소가 점점 투명해지면서 사라집니다. |
| slideUp() | 요소가 위로 접히면서 숨겨집니다. |
| 노출 | show() | 숨겨진 요소가 노출됩니다. |
| fadeIn() | 숨겨진 요소가 점점 선명해집니다. |
| slideDown() | 숨겨진 요소가 아래로 펼쳐집니다. |
| 노출, 숨김 | toggle() | hide(), show() 효과를 적용합니다. |
| fadeToggle() | fadeIn(), fadeOut() 효과를 적용합니다. |
| slideToggle() | slideUp(), slideDown() 효과를 적용합니다. |
| fadeTo() | 지정한 투명도를 적용합니다 |

\* 효과 메서드의 기본형

$(”요소선택”).효과 메서드(효과 소요시간(1), 가속도(2), 콜백함수(3));

(1) 효과 소요시간

방법1 - “slow”, “normal”, “last”

방법2 - 1,000(1초), 500(0.5초)

(2) 가속도

방법1 - “swing” 시작과 끝은 느리게, 중간은 빠른 속도로 움직입니다(기본값).

방법2 - “linear” 일정한 속도로 움직입니다.

(3) 콜백함수 - 노출과 숨김효과가 끝난 후에 실행할 함수입니다. 콜백 함수는 생략할 수 있습니다.

\* fadeTo 메서드

$(”요소 선택”).fadeTo (효과 소요 시간, 투명도(1), 콜백함수);

(1) 투명도 - 0 ~ 1까지의 값을 입력할 수 있습니다. 1에 가까울수록 선명하게 보입니다.

**6.2 애니메이션 메서드**

**6.2.1 animate() 메서드**

스타일 속성에 따른 동적 효과를 적용할 수 있습니다.

$(”요소 선택”).animate({스타일 속성(1)}, 적용시간(2), 가속도(3), 콜백 함수(4));

(1) 스타일 속성 - 애니메이션으로 적용할 스타일 속성입니다.

(2) 적용 시간 - 동작에 반응하는 데 소요되는 시간

**6.2.2 애니메이션 효과 제어 메서드**

|  |  |
| --- | --- |
| 종류 | 설명 |
| stop() | 현재 실행 중인 효과를 모두 정지시킵니다.  (1) $(”요소선택”).stop();  (2) $(”요소선택”).stop(clearQueue, finish);  (1) 진행 중인 첫 번째 애니메이션만 정지시킵니다. 큐에 대기 중인 애니메이션은 계속 실행합니다.  (2) clearQueue가 true면 큐에서 대기 중인 애니메이션을 모두 제거, finish가 true면 진행 중인 애니메이션을 강제로 종료 |
| delay() | 지정한 시간만큼 지연했다가 애니메이션을 진행합니다.  $(”요소 선택”).delay(지연시간).애니메이션 효과 메서드(); |
| queue() | 큐에 사용자 정의 함수를 추가하거나 큐에 대기 중인 함수를 배열에 담아 반환합니다. 그리고 queue() 메서드 이후의 애니메이션 효과 메서드는 모두 취소합니다.  \* 큐(Queue)의 함수 반환  $(”요소 선택”).queue();  \* 큐(Queue)에 함수 추가  $(”요소 선택”).queue(function() { 자바스크립트 코드 }); |
| clearQueue() | 큐에서 처음으로 진행하고 있는 애니메이션만 제외하고 대기 중인 애니메이션은 모두 제거합니다.  $(”요소선택”).clearQueue(); |
| dequeue() | queue() 메서드를 이용하면 대기하고 있는 애니메이션 메서드는 제거됩니다. 하지만 dequeue() 메서드를 이용하면 메서드가 제거되는 것을 막을 수 있습니다.  $(”요소 선택”).dequeue(); |
| finish() | 선택한 요소의 진행 중인 애니메이션을 강제로 완료 시점으로 보낸 후 종료합니다. |

**7. 제이쿼리 비동기 방식 연동**

**7.1 Ajax**

Ajax(Asynchronous Javascript and XML)란 비동기 방식의 Javascript와 XML을 말합니다.

**동기방식** - 서버에 신호를 보냈을 때 응답이 돌아와야 다음 동작을 수행할 수 있습니다.

**비동기방식** - 신호를 보냈을 때 응답상태와 상관 없이 다음 동작을 수행 할 수 있습니다.

\* AJax를 이용하는 이유는 화면 전환 없이 클라이언트(사용자 컴퓨터)와 서버(서비스를 제공하는 컴퓨터) 간에 XML, JSON(JavaScript Object Notation), 텍스트, HTML 등의 정보를 교환하기 위해서 입니다(이용자가 서버에 자료를 요청할 때 화면 전환 없이 요청한 자료를 전송받을 수 있습니다.).

|  |  |
| --- | --- |
| 종류 | 설명 |
| load() | 외부 콘텐츠를 가져올 때 사용합니다.  $(요소 선택).load(url, data, 콜백함수); |
| $.ajax() | 데이터 서버에 HTTP, POST, GET 방식으로 전송할 수 있으며, HTML, XML, JSON 텍스트 유형에 데이터를 요청힐 수 있는 통합적인 메서드 입니다. 이 표에 있는 $.post(), $.get(), $.getJSON() 메서드의 기능을 하나로 합쳐 놓은 것이라 보면 됩니다.  $.ajax({  url : “전송 페이지(action url)”,  type : “전송 방식(get, post)”,  data : “전송할 데이터”,  dataType : “요청한 데이터 형식(”html”, “xml”, “json”, “text”, “jsonp”)”,  success : function(data) {  전송에 성공하면 실행될 코드;  },  error : fuiinction() {  전송에 실패하면 실행될 코드;  } }); |
| $.post() | 데이터를 서버에 HTTP POST 방식으로 전송한 후 서버 측의 응답을 받을 때 사용합니다. |
| $.get() | 데이버를 서버에 HTTP GET 방식으로 전송한 후 서버 측의 응답을 받을 때 사용합니다. |
| $.getJSON() | 데이터를 서버에 HTTP GET 방식으로 전송한 후 서버 측의 응답을 JSON 형식으로 받을 때 사용합니다. |
| $.getScript() | Ajax를 이용하여 외부 자바스크립트를 불러옵니다.  $(”button”).click(function() {  $.getScript(”demo\_ajax\_script.js”); }); |
| ajaxStop(function(){...}) | 비동기 방식으로 서버에 응답 요청이 완료되었을 때 함수가 실행됩니다. |
| ajaxSuccess(function(){...}) | ajax 요청이 성공적으로 완료되면 함수가 실행됩니다. |
| ajaxComplete(function(){...}) | Ajax 통신이 완료되면 함수가 실행됩니다. |

**\* JSON이란?**

자바스크립트의 객체 표기법을 JSON(JavaScript Object Notation)이라고 부릅니다. JSON은 데이터를 전달할 때 사용하는 표준 형식입니다. JSON은 속성(key)과 값(value)이 하나의 쌍을 이루고 있습니다.

{”속성1” : “값1”, “속성2 : “값2”, “속성3” : “값3” }

**$.ajax() 메서드의 옵션 종류**

|  |  |
| --- | --- |
| 종류 | 설명 |
| async | 통신을 동기 또는 비동기 방식으로 설정하는 옵션입니다. 기본값은 비동기 통신 방식인 true로 설정되어 있습니다. 만일 비동기 방식으로 설정되어 있다면 사용자 컴퓨터에서 서버로 데이터를 전송하고 요청하는 동안에도 다른 작업을 할 수 있습니다. |
| beforeSend | 요청하기 전에 함수를 실행하는 이벤트 핸들러입니다. |
| cache | 요청한 페이지를 인터넷에 캐시(저장)할지의 여부를 설정합니다. 기본값은 true 입니다. |
| complete | Ajax가 완료되었을 때 함수를 실행하는 이벤트 핸들러입니다. |
| data | 서버로 전송할 데이터를 지정합니다. |
| dataType | 서버에서 받아올 데이터의 형식을 지정합니다. 생략하면 요청한 자료에 맞게 자동으로 형식이 설정됩니다. |
| error | 통신에 문제가 발생했을 때 함수를 실행합니다. |
| success | Ajax로 통신이 정상적으로 이뤄지면 이벤트 핸들러를 실행합니다. |
| timeout | 통신 시간을 제한합니다. 시간 단위는 밀리초입니다. |
| type | 데이터를 전송할 방식(get, post)를 설정합니다. |
| url | 데이터를 전송할 페이지를 설정합니다. 기본값은 현재 페이지 입니다. |
| username | HTTP 액세스를 할 떄 인증이 필요한 경우 사용자 이름을 지정합니다. |
|  |  |

**Ajax 전송 데이터 가공 메서드의 종류**

|  |  |  |
| --- | --- | --- |
| 종류 | 사용법 | 설명 |
| serialize() | $(”fom”).serialize() | 사용자가 입력 요소에 값을 입력한 데이터의 전송방식을 'name1=value1 & name2=value2,...'와 같은 쿼리 스트링 형태의 데이터로 변환해 반환합니다. |
| serializeArray() | $(”form”).serializeArray() | 사용자가 입력요소에 값을 입력한 데이터의 전송 방식을[{name1 : value1}, {name2 : value2}]와 같은 배열 객체로 변환해 반환합니다. |
| $.param() | $.param( Object ); | {name1 : value1, name2 : value2}와 같이 작성된 객체를 가공해 'name1=value1&name2=value2, ..'와 같은 쿼리 스트링 형식의 데이터로 변환해 반환합니다. |
| JSON.parse() | JSON.parse('{”name”:”value”}'); | 객체 형태로 작성한 문자열 데이터를 객체로 가공하여 변환합니다, |
| JSON.stringify() | JSON.stringify({”name1”:”value1”, “name2” : “value2”}); | 객체를 문자열 데이터로 가공하여 반환합니다. |

**\* 자바스크립트의 보안 정책과 외부 데이터 바인딩하기**

자바스크립트의 보안 정책은 자바스크립트로 A 사이트(www.aaa.com)의 데이터를 B사이트(www.bbb.com)로 불러오는 것을 허용하지 않습니다. 이 정책을 동일 출저 원칙(Same-origin policy)이라 합니다. 즉, ajax() 메서드로는 서로 다은 도메인의 데이터를 전송할 수 없을 뿐만 아니라 교차 도메인(Cross Domain)도 허용하지 않습니다.

서로 다른 사이트의 데이터를 서로 주고 받으려면 서버 언어를 사용해 데이터를 가져오거나 JSONP를 사용하는 방법이 있습니다. 서버 언어를 사용하는 방법은 서로 다른 사이트에서 데이터를 요청하고 이를 ajax로 가공하면 됩니다. 이 방법은 주로 관리자가 직접 접근할 수 없는 외부 서버의 데이터를 가져올 때 사용합니다. JSONP를 사용하는 방법은 '동일 출처 보안 정책'을 피해야 하고 관리자의 외부 서버 접근이 가능해야 한다는 전제조건이 필요합니다. 이 방법은 교차 도메인으로 데이터를 불러올 때 사용합니다.

**8. 플러그인 제작하기**

**$.fn.extend()** 메서드

$.fn.extend() 메서드는 개발자가 직접 제작한 함수를 jQuery에 확장시켜 주는 역할을 합니다.

방법1

$.fn.extend({

확장 함수명 : function(매개변수1, 매개변수2, ...) {

$(this); (this)는 플러그인을 적용한 요소를 가리킵니다.

자바스크립트 코드;

}

});

$(”요소 선택”).확장 함수명(인자값1, 인자값2); // 플러그인 적용

방법2

$.fn.확장 함수명 = function(매개변수1, 매개변수2, ...) {

}