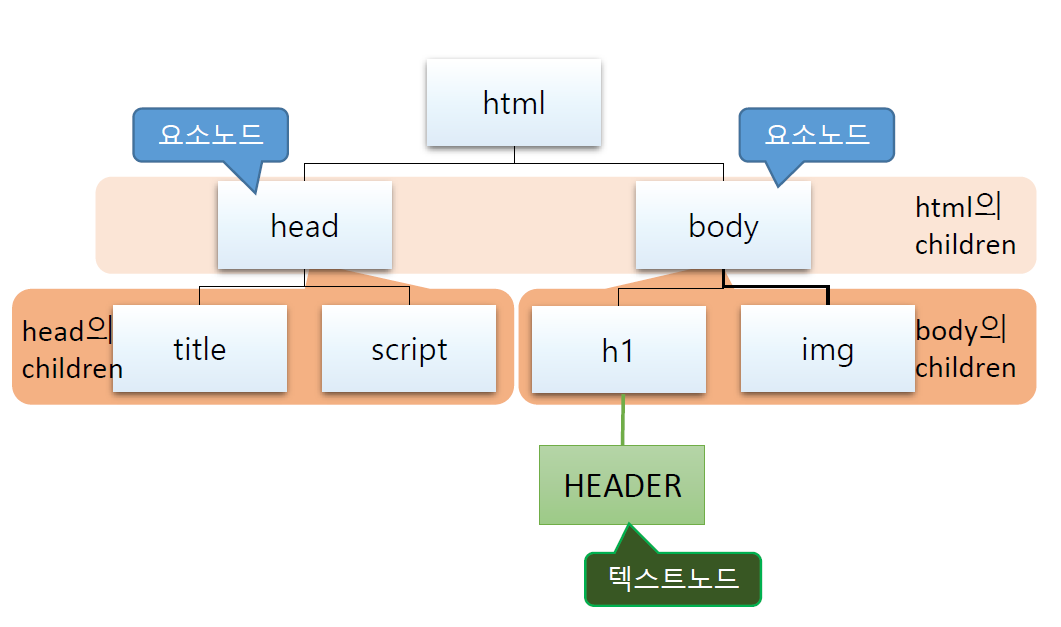
문서 객체 생성

위쪽 – 부모 노드

아래쪽 – 자식 노드

옆 – 형제 노드



문서 객체 노드 생성 메소드

document.createElement(“tagName”) – 요소 노드를 생성

document.createTextNode(“text”) – 텍스트 노드를 생성

문서 객체 연결 메서드

appendChild(node) – 객체에 노드를 연결, 마지막 자식 뒤에 새로운 자식 추가

insertBefore(node [, 기존 자식]) – 지정된 자식 앞에 새 자식 추가

문서 객체 속성 메서드

setAttribute(“id”, “value”) – 객체의 속성을 지정

getAttribute(“name”) – 객체의 속성을 가져옴

innerHTML을 이용해 텍스트 노드 추가 가능

style을 이용해 style 속성 추가 가능 (DIV.style.backgroundColor = “yellow”;)

문서 객체 삭제

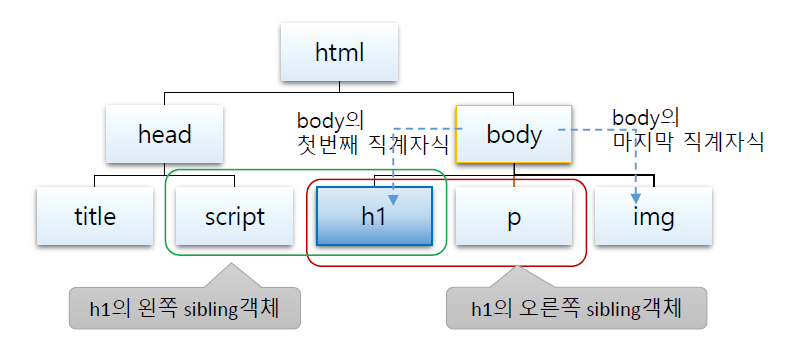
removeChlid(node) – 객체의 자식 노드를 제거

this

- 자기 자신을 가리키는 자바 스크립트 키워드

- 오브젝트에 자기 자신을 가리켜 자신에 대한 변화를 만들어 낼 수 있음

Dom 객체 사이의 관계



문서 객체 관계 메소드

parentElement – 부모 객체

childElementCount – 자식 객체 개수

firstElementChild – 첫번째 자식 객체

lastElementChild – 마지막 자식 객체

nextElementSibling – 다음 형재 객체(오른쪽 형제)

previousElementSibling – 이전 형재 객체(왼쪽 형제)

이벤트

- 웹 브라우저와 사용자간 커뮤니케이션

- 마우스 클릭, 키보드 입력, 이미지나 HTML 문서의 로딩, 타임아웃 등 브라우저가 자바스크립트에 알리는 통지

문제점 : 구버전의 웹 브라우저와 최신 버전의 웹 브라우저에 따라 이벤트 사용 방법이 다르다

해결책 : jQuery 등의 보완 라이브러리를 사용하여 각 브라우저마다 동일한 이벤트를 사용 가능하게 한다

이벤트 종류

onkeydown – 키보드의 아무 키나 누르는 순간

onkeypress – [F1]같은 키를 누르는 순간 (알파뉴메릭 키를 누르는 순간)

onkeyup – 누른 키를 놓는 순간

onclick – 객체에 클릭할 때

ondblclick – 객체에 더블 클릭될 때

oncontextmenu – 오른쪽 마우스 버튼이 클릭될 때

onmousedown – 마우스 버튼이 눌러지는 순간

onmouseenter – 마우스 커서가 객체 영역 안으로 들어가는 순간

onmouseleave – 마우스 커서가 객체 영역에서 벗어나는 순간

onmousemove – 객체 위에서 마우스가 움직이는 동안 계속 발생

onmouseover – 마우스 커서가 객체 영역 (자식객체도 포함) 안으로 들어가는 순간

onmouseout – 마우스 커서가 객체 영역 (자식객체도 포함)에서 벗어나는 순간

onmouseup – 마우스 버튼을 놓는 순간

onwheel – 마우스 휠을 굴리는 매 순간

onabort – 이미지나 문서의 로딩이 중단된 경우

onerror – 이미지나 문서, 리소스 로딩 시 오류가 발생할 때

onload – 웹 페이지가 로드된 후

onunload – 웹 페이지가 업로드 된 후

onresize – 윈도우, 프레임 객체의 크기가 변경될 때

onbeforeprint – 웹 페이지 프린트나 미리보기를 시작하기 전

onafterprint – 웹 페이지 프린트나 미리보기를 마치고 돌아갈 때

onfocus – 객체가 포커스를 가지게 되었을 때

onblur – 객체가 포커스를 잃을 때

onchange – 텍스트, 선택된 내용, 체크 상태 등이 변할 때

onreset – 사용자가 reset 버튼을 누르거나 자바스크립트 코드로 폼을 리셋 시켰을 때

onsearch – <input type=“search”> 에 검색 텍스트를 입력하고 엔터키를 누를 때

onselect – 입력된 텍스트를 사용자가 선택하거나 드래그할 때

onsubmit – 사용자가 submit 버튼을 클릭할 때

이벤트 리스너(event listener) 등록

- 이벤트 리스너란 이벤트가 발생했을 때 그 처리를 담당하는 함수

- 특정 요소에서 이벤트가 발생하면, 웹 브라우저는 그 요소에 등록된 이벤트 리스너를 실행시킨다

HTML 자체에도 추가 가능

<p onmouseover = “this.style.backgroundColor yellowgreen”>

자바 스크립트에서 일반적인 사용

p.onmouseover = function;

addEventListener() 메소드

- 새로운 이벤트 리스너 등록

- 여러 개의 이벤트 리스너 등록 가능

addEventListener(eventName , listener[, useCapture])

- eventName : 이벤트 타입을 나타내는 문자열 (click, load, keydown 등)

- listener : 이벤트 리스너로 등록할 함수 이름

- useCapture : 이벤트 전파 방식 false면 버블링(bubbling) 방식 / true면 캡처링(capturing) 방식

p.addEventListener(“mouseover”, func1); -> on없이 사용

removeEventListener() 메소드

- 이벤트 리스너 삭제

이벤트 객체

이벤트 리스너 호출

- 이벤트가 발생하면, 브라우저는 등록된 이벤트 리스너를 호출한다. 이때 이벤트 리스너는 매개변수로 이벤트 객체(event object)를 전달받는다.

이벤트 객체(event object)

- 해당 타입의 이벤트에 대한 상세 정보를 저장하고 있다.

- 함수 내부 매개변수로 e 또는 event를 넣어 이벤트 내용을 확인 또는 사용 가능하다

function 이벤트함수명 (e) { }

이벤트 객체 공통 멤버

type – 현재 발생한 이벤트의 종류를 나타내는 문자열 (click, mouseover 등)

target – 이벤트를 발생시킨 객체

currentTarget – 현재 이벤트 리스너를 실행하고 있는 DOM 객체

defaultPrevented – 이벤트의 디폴드 행동이 취소되었는지를 나타내는 true/false 값

preventDefault – 이벤트의 디폴트 행동을 취소시키는 메소드

이벤트 흐름

캡쳐 단계

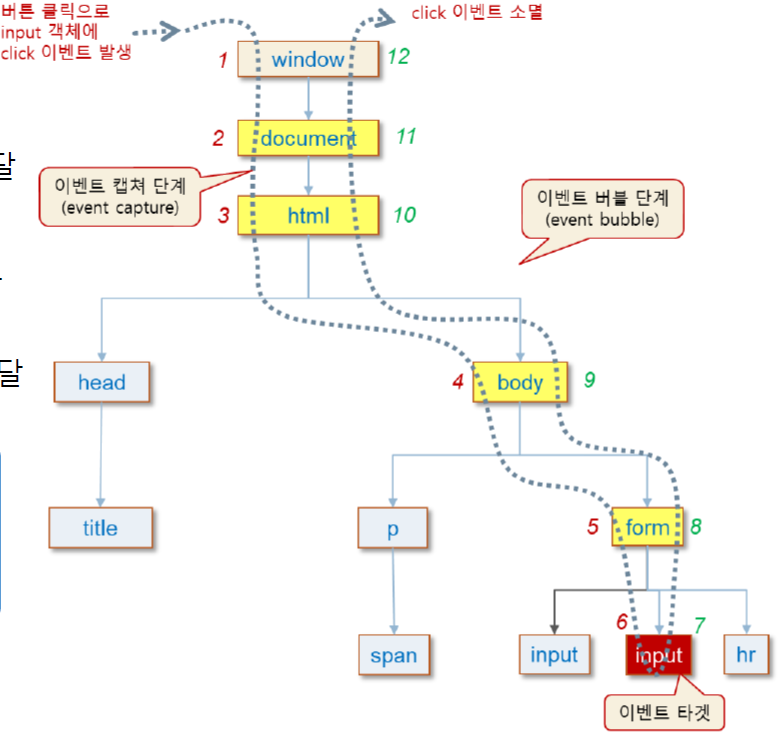
- DOM 트리의 최상위부터 이벤트가 발생한 요소까지 아래쪽으로 내려가면 전파되는 방식

- addEventListener() 메소드의 세 번째 인수에 true를 전달

버블 단계

- 이벤트가 발생한 요소부터 시작해서, DOM 트리를 따라 위쪽으로 올라가며 전파되는 방식

- addEventListener() 메소드의 세 번째 인수에 false를 전달



jQary

- 자바스크립트 요소 중 DOM 객체 부분을 좀 더 쉽게 사용할 수 있게 도와주는 라이브러리

- 자바스크립트로 만들어진 다양한 객체와 함수들의 집합

- 제이쿼리의 기능

-> 문서 객체 모델과 관련된 처리를 쉽게 구현

-> Ajax 애플리케이션을 쉽게 개발

-> 일관된 이벤트 연결을 쉽게 구현

-> 시각적 효과를 쉽게 구현

jQary 종류

jquery.js - 들여쓰기, 줄바꿈, 주석까지 정리된 비압축 파일

jquery.min.js - 들여쓰기, 줄바꿈, 주석이 무시된 압축 파일, 용량이 작아 로딩속도가 빠름

jQary 연동

<script src ="jquery 3.6.0. min.js"></script> 다운로드형식

<script src ="https://cdnjs.cloudflare.com/ajax/jquery/jquery.min.js"></script 네트워크형식>

$기호 – jQary 사용 선언

문서 객체 불러오기

Ex)

$(document).ready(function(){

alert('First Ready!');

});

$(document).ready(function(){

alert('Second Ready!!');

});

$(document).ready(function(){

alert('Thrid Ready!!!');

});

선택자

- 노드를 찾을 때 CSS 선택자를 활용한다

- 선택자는 $( )에 문자형 데이터로 CSS 선택자를 입력한다.

직접 선택자

전체 선택자 $(“\*”)

- 모든 요소를 선택 (HTML 문서의 모든 태그 선택)

아이디 선택자 $(“#아이디명”)

- id 속성에 지정한 값을 가진 요소 선택

클래스 선택자 $(“.클래스명”)

- class 속성에 지정한 값을 가진 요소 선택

요소 태그 선택자 $(“요소명”)

- 지정한 요소명과 일치하는 요소들만 선택

그룹 선택자 $(“선택 1, 선택 2, 선택 3, …”)

- 선택1, 선택 2, 선택 3,… 에 지정된 요소들을 한 번에 선택

종속 선택자 $(“요소명.class값”) $(“요소명#id값”)

- 요소 중 class 값이 같은 요소 선택, 요소 중 id 값이 같은 요소 선택

인접 관계 선택자

부모 요소 선택자 $(“요소선택”).parent()

- 선택한 요소의 부모 요소 선택

상위 요소 선택자 $(“요소선택“).parents()

- 선택한 요소의 상위 요소들 모두 선택

가장 가까운 상위 요소 선택자 $(“요소선택 “).closest(“요소명”)

- 선택한 요소의 상위 요소 중 가장 가까운 요소만 선택

하위 요소 선택자 $(“요소선택 하위요소”)

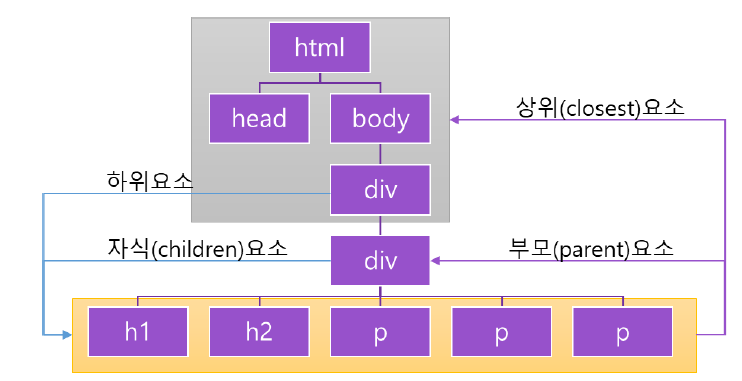
- 선택한 요소에 지정한 하위요소를 선택

자식요소 선택자 $(“요소선택>자식요소”) $(“요소선택”).children(“자식요소”)

- 선택한 요소를 기준으로 자식관계에 지정한 요소 선택

자식 요소들 선택자 $(“요소선택”).children()

- 선택한 요소의 모든 자식요소 선택



왼쪽 형제(이전) 요소 선택자 $(“요소선택”).prev()

- 선택한 요소의 바로 이전 요소 선택

왼쪽 형제(이전) 요소들 선택자 $(“요소선택”).prevAll()

- 선택한 요소의 이전 요소 모두 선택

지정 왼쪽 형제(이전) 요소들 선택자 $(“요소선택”).prevUntil(“요소명”)

- 선택한 요소부터 지정한 요소의 이전 요소까지 모두 선택

오른쪽 형제(다음) 요소 선택자 $(“요소선택”).next() $(“요소선택+다음요소”)

선택한 요소의 바로 다음 요소 선택

오른쪽 형제(다음) 요소들 선택자 $(“요소선택”).nextAll()

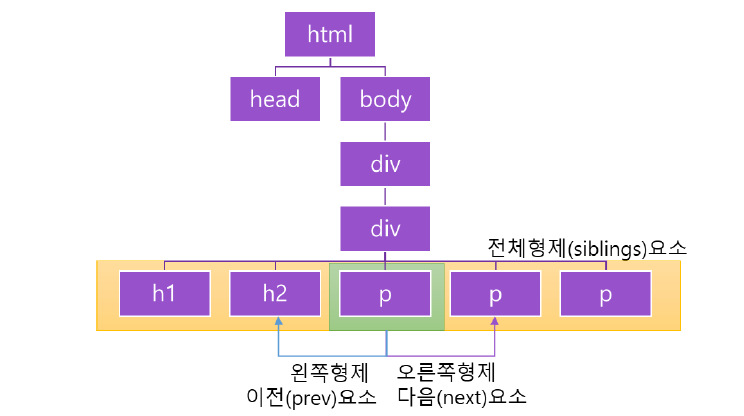
- 선택한 요소의 다음 요소 모두 선택

지정 오른쪽 형제(다음) 요소 선택자 $(“요소선택”).nextUntil(“요소명”)

- 선택한 요소부터 지정한 요소의 다음 요소까지 모두 선택

전체 형제 요소 선택자 $(“요소선택”).siblings()

- 선택한 요소의 형제 요소 전체를 선택



위치 탐색 선택자

$(“요소선택:first”) $(“요소선택“).first()

- 전체 “요소선택” 요소 중 첫번째 요소만 선택

$(“요소선택:last”) $(“요소선택“).last()

- 전체 “요소선택” 요소 중 마지막 요소만 선택

$(“요소선택:odd”)

- “요소선택” 요소 무리 중 홀수 인덱스 요소만 선택

$(“요소선택:even”)

- “요소선택” 요소 무리 중 짝수 인덱스 요소만 선택

$(“요소선택:first-of-type”)

- “요소선택” 요소 무리 중 첫번째 요소만 선택

$(“요소선택:last-of-type”)

- “요소선택” 요소 무리 중 마지막 요소만 선택

$(“요소선택:nth-child(숫자)”)

- “요소선택” 요소 무리 중 “숫자”번째 요소만 선택

$(“요소선택:nth-child(숫자n)”)

- “요소선택” 요소 무리 중 “숫자”의 배수 번째 있는 요소만 선택

$(“요소선택:nth-last-of-type(숫자)”)

- “요소선택” 요소 무리 중 마지막 위치로부터 “숫자”번째 요소만 선택

$(“요소선택:only-child”)

- 부모 요소 내에서 “요소선택” 요소가 1개뿐인 “요소선택” 요소만 선택

$(“요소선택:eq(index)”) $(“요소선택”).eq(index)

- “요소선택” 요소 중 인덱스 index가 참조하는 요소를 불러옴

$(“요소선택:gt(index)”)

- “요소선택” 요소 중 인덱스 index보다 큰 인덱스가 참조하는 요소를 불러옴

$(“요소선택:lt(index)”)

- “요소선택” 요소 중 인덱스 index보다 작은 인덱스가 참조하는 요소를 불러옴

$(“요소선택”).slice(index)

- “요소선택” 요소 중 인덱스 index부터 참조하는 요소를 불러옴

배열 관련 메소드

index : 인덱스번호 / object : 인덱스번호에 해당되는 배열객체

$(“요소선택”).each(function(index, object){ }) $.each(“요소선택“, function(index, object){ })

- 배열에 저장된 객체의 인덱스 순서대로 하나씩 접근하여 객체를 선택하고 인덱스를 구한다

$.map(Array, function(object, index){ })

- 함수에서 반환된 데이터는 새 배열에 순서대로 저장되고 새 배열을 반환한다

$.grep(Array, function(object, index){ })

- 반환값이 true인 경우에만 배열의 데이터가 인덱스 오름차순으로 새 배열에 저장되고 새 배열을 반환한다

$.inArray(data, Array[, start index])

- 배열 안에서 데이터를 찾음. 데이터를 찾으면 가장 맨 앞 데이터의 인덱스를 반환하고 찾지 못하면 1 반환

start index의 값을 지정하면 해당 위치부터 데이터를 찾음

$.isArray(object)

- 입력한 객체가 배열 객체라면 true, 아니면 false 반환

$.merge(Array1, Array2)

- 2개의 배열 객체를 하나로 그룹화

$(“요소선택”).index(“지정요소선택”)

- 선택한 요소에서 지정한 요소의 인덱스 정보 반환

속성 탐색 선택자

$(“요소 선택[속성]”)

- “요소선택” 요소 중 “속성” 속성이 포함된 요소만 선택

$(“요소 선택[속성=’값’]”)

- “요소선택” 요소 중 “속성” 속성값이 “값” 인 요소만 선택

$(”요소 선택[속성^=’텍스트’]”)

- “요소선택” 요소 중 “속성” 속성값이 “텍스트”로 시작하는 요소만 선택

$(”요소 선택[속성$=’텍스트’]”)

- “요소선택” 요소 중 “속성” 속성값이 “텍스트”로 끝나는 요소만 선택

$(”요소 선택[속성\*=’텍스트’]”

- “요소선택” 요소 중 “속성” 속성값 중 “텍스트”을 포함하는 요소만 선택

$(”요소 선택:hidden”)

- “요소선택” 요소 중 숨겨져 있는 요소만 선택

$(“요소 선택:visible”)

- “요소선택” 요소 중 보이는 요소만 선택

$(“:type속성”)

- “input” 요소(text, button) 중 type 속성값이 “type 속성”인 요소만 선택

$(“:selected”)

- “option” 요소 중 selected 속성이 적용된 요소만 선택

$(“:checked”)

- 체크박스 또는 라디오버튼 중 checked 속성이 적용된 요소만 선택

콘텐츠 탐색 선택자

$(“요소 선택:contains(‘텍스트’)”)

- “요소선택” 요소 중 “택스트” 텍스트를 포함하는 요소만 선택

$(“요소 선택”).contents()

- “요소선택” 요소의 하위요소 중 가장 가까운 하위 요소 선택

$(”요소 선택:has(‘요소명’)”) $(“요소 선택”).has(‘요소명’)

- “요소선택” 요소 중 “요소명”을 포함하는 요소만 선택

$(”요소 선택:not(:제외요소)”) $(“요소 선택”).not(:제외요소)

- “요소선택” 요소 중 첫번째 요소만 제외하고 선택

$(”요소 선택”).filter(“필터요소”) $(”요소 선택”).filter(function(){ })

- “요소선택” 요소 중 class 값이 “필터요소”인 요소만 선택

$(”요소 선택”).find(“하위요소”)

- “요소선택” 요소의 하위요소 중 “하위 요소”만 선택

$(“요소 선택”).closet(“상위요소”)

- “요소선택”의 상위 “상위 요소” 요소 중 가장 가까운 상위 요소만 선택

$(“요소 선택”).필터링().end() 적용 안됨

- 필터링이 실행되기 이전의 요소인 “요소선택” 가 선택

is() 메소드

$(“요소 선택”).is(“:요소 상태”)

- 선택한 요소의 상태가 지정한 속성과 일치하면 true 반환

상태: checked, selected, visible, hidden, animated

객체 조작 메서드

객체를 생성, 복제, 삭제, 속성 변환하는 메서드

속성 조작 메서드

$(“요소 선택”).html()

- 선택한요소의 하위 요소를 가져온다

$(“요소 선택”).html(“새 요소”)

- 선택한 요소의 하위 요소를 모두 제거하고 그 위치에 지정한 새 요소를 생성한다

$(“요소 선택”).text

- 선택한 요소의 텍스트만 가져온다

$(“요소 선택”).text(“새 텍스트”)

- 선택한 요소의 하위 요소를 모두 제거하고 그 위치에 지정한 새 텍스트를 생성한다

$(“요소 선택”).attr(“속성명”)

- 선택한 요소의 지정한 속성값을 가져온다

$(“요소 선택”).attr(“속성명”, “새 값”)

- 요소를 선택하여 지정한 속성에 새 값을 적용한다

attr({“속성명1”:”값1”, “속성명2”:”값2”, “속성명3”:”값3”,….}) – 여러 개 설정

$(“요소 선택”).removeAttr(“속성명”)

- 선택한 요소의 지정한 속성만 제거한다

$(“요소 선택”).prop(“상태 속성명”)

- 선택한 폼 요소 선택상자, 체크박스, 라디오버튼의 상태 속성값을 가져온다

$(“요소 선택”).prop(“상태 속성명”, “새 값”)

- 요소를 선택하여 상태 속성 (checked, selected)에 새 값 (true, false)을 적용한다

prop(“[ tagName | nodeType | selectedIndex | defaultValue ]”) – 이용하기도 함

$(“요소 선택”).val()

- 선택한폼 요소의 value 값을 가져온다

$(“요소 선택”).val(“새 값”)

- 요소를 선택하여 value 속성에 새 값을 적용한다

$(“요소 선택”).css(“속성명”)

- 선택한 요소의 지정한 스타일 속성값을 가져온다

$(“요소 선택”).css(“속성명”, “새 값”)

- 요소를 선택하여 지정한 스타일 속성에 새 값을 적용한다

$(“요소 선택”).addClass(“새 값”)

- 선택한 요소의 class 속성에 새 값을 추가한다

$(“요소 선택”).removeClass(“값”)

- 선택한 요소의 class 속성에서 지정한 값만 제거한다

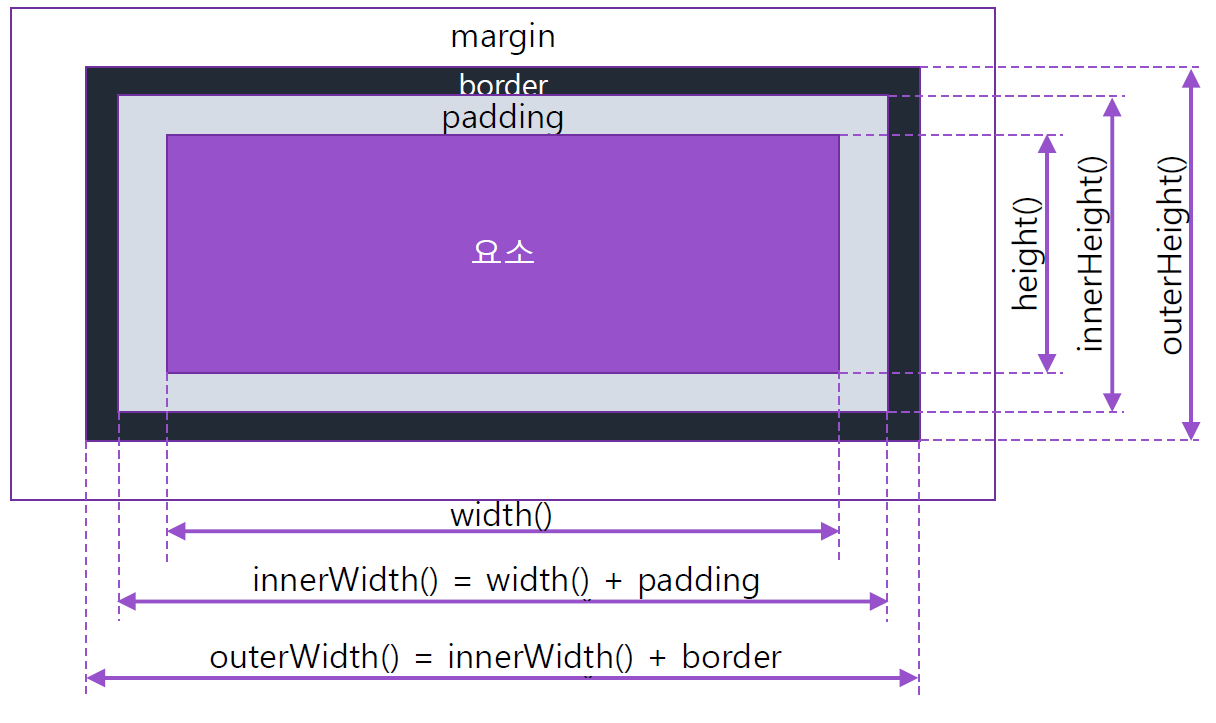
$(“요소 선택”).toggleClass(“값”)

- 선택한 요소의 class 값에 지정한 값이 포함되어 있으면 제거하고 없으면 추가한다

$(“요소 선택”).hasClass(“값”)

- 선택한 요소의 class 값에 지정한 값이 포함되어 있으면 true, 없으면 false 반환한다

수치 조작 메서드



$(“요소 선택”).height() $(“요소 선택”).height(값)

- 안쪽 여백과 선을 제외한 높이 값을 반환하거나 변환한다

$(“요소 선택”).width() $(“요소 선택”).width(값)

- 안쪽 여백과 선을 제외한 너비 값을 반환하거나 변환한다

$(“요소 선택”).innerHeight() $(“요소 선택”).innerHeight(값)

- 안쪽 여백을 포함한 높이 값을 반환하거나 변환한다 (height + padding)

$(“요소 선택”).innerWidth() $(“요소 선택”).innerWidth(값)

- 안쪽 여백을 포함한 너비 값을 반환하거나 변환한다 (width + padding)

$(“요소 선택”).outerHeight() $(“요소 선택”).outerHeight(값)

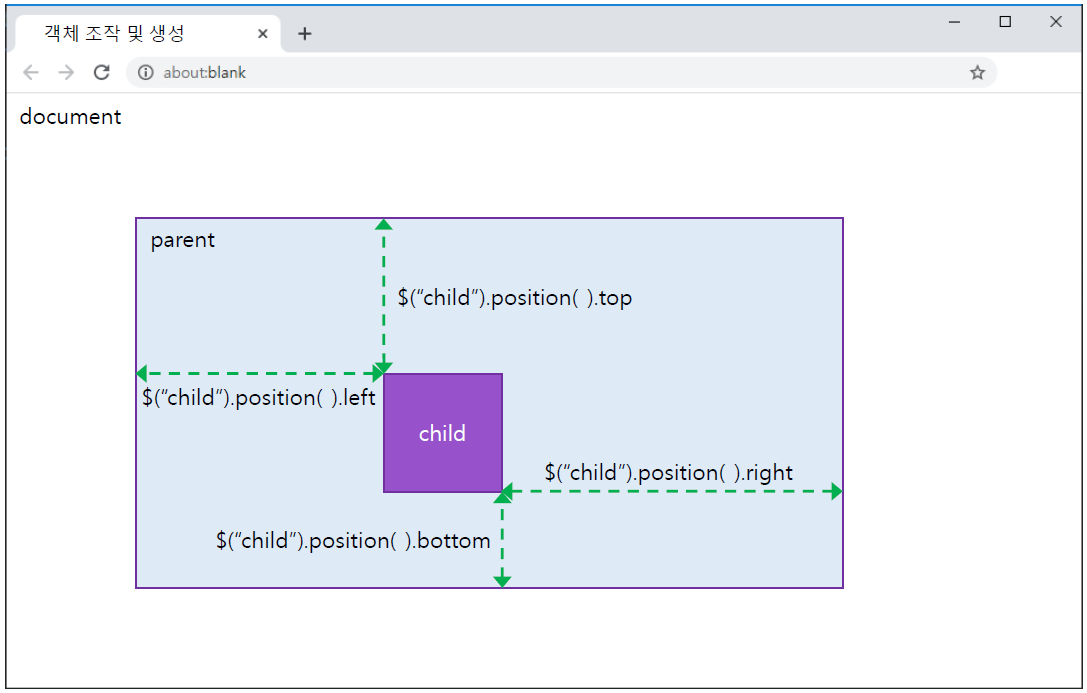
- 안쪽 여백과 선을 포함한 높이 값을 반환하거나 변환한다 (height + padding + border)

$(“요소 선택”).outerWidth() $(“요소 선택”).outerWidth(값)

- 안쪽 여백과 선을 포함한 너비 값을 반환하거나 변환한다 (width + padding + border)

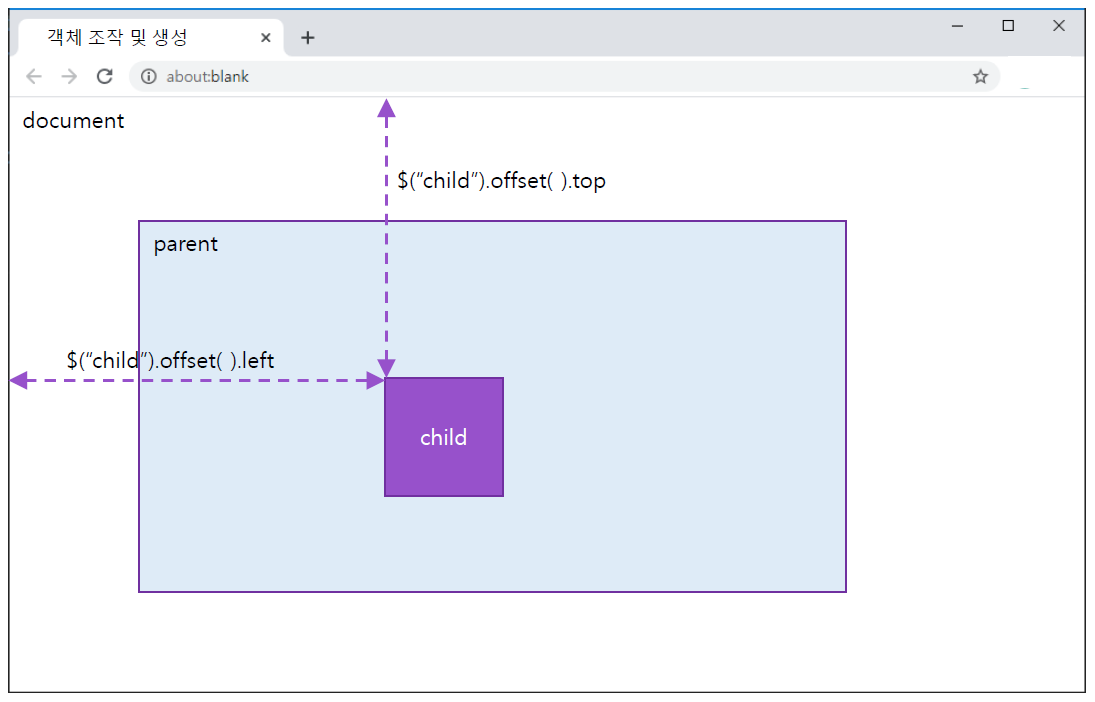
$(“요소 선택”).position().left $(“요소 선택”).position().top

- 선택한 요소의 위치 (left, right, top, bottom) 좌표 값을 반환한다



$(“요소 선택”).offset().left $(“요소 선택”).offset().top

- document를 기준으로 선택한 요소의 위치 좌표 값을 반환한다



$(window).scrollLeft()

- 스크롤바가 수평으로 이동한 위치 값을 반환한다

$(window).scrollLeft(값)

- 입력한 값만큼 수평으로 스크롤바를 이동시킨다

$(window).scrollTop()

- 스크롤바가 수직으로 이동한 위치 값을 반환한다

$(window).scrollTop(값)

- 입력한 값만큼 수직으로 스크롤바를 이동시킨다

객체 편집 메소드

$(“요소 선택”).before(“새 요소”) $(“새 요소”).insertBefore(“요소 선택”)

- 선택한 요소의 이전 위치에 새 요소를 추가한다

$(“요소 선택”).after(“새 요소”) $(“새 요소”).insertAfter(“요소 선택”)

- 선택한 요소의 다음 위치에 새 요소를 추가한다

$(“요소 선택”).append(“새 요소”) $(“새 요소”).appendTo(“요소 선택”)

- 선택한 요소의 마지막 위치에 새 요소를 추가한다

$(“요소 선택”).prepend(“새 요소”) $(“새 요소”).prependTo(“요소 선택”)

- 선택한 요소의 맨 앞 위치에 새 요소를 추가한다

$(“요소 선택”).clone(true or false)

- 선택한 문서 객체를 복제한다(기본값은 false)

true 인 경우 하위 요소까지 모두 복제 / false 인 경우 선택한 요소만 복제

$(“요소 선택”).empty()

- 선택한 요소의 하위 내용들을 모두 삭제한다

$(“요소 선택”).remove()

- 선택한 요소를 삭제한다

$(“새 요소”).replaceAll(“요소 선택”) $(“요소 선택”).replaceWith(“새 요소”)

- 선택한 요소들을 새 요소로 교체한다

$(“요소 선택“).unwrap()

- 선택한 요소의 부모 요소를 삭제한다

$(“요소 선택“).warp(“새 요소”)

- 선택한 요소를 새 요소로 각각 감싼다

$(“요소 선택”).wrapAll(“새 요소”)

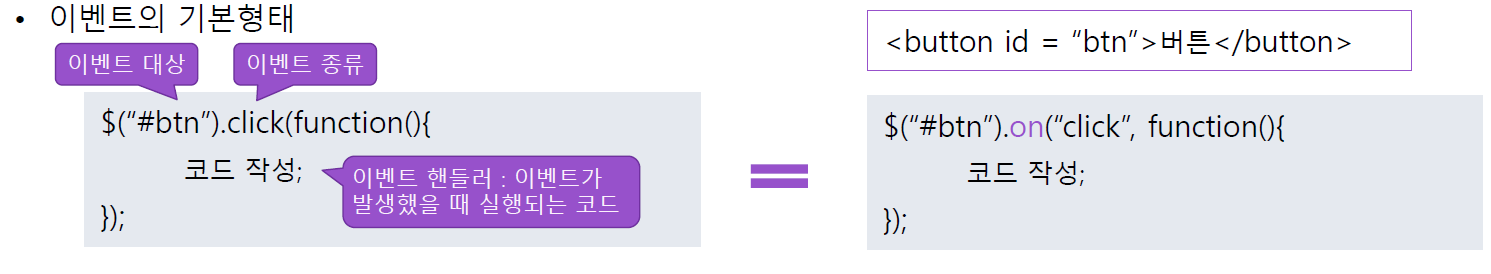
- 선택한 요소를 새 요소로 한꺼번에 감싼다

$(“요소 선택”).wrapInner(“새 요소”)

- 선택한 요소의 모든 하위 요소를 새 요소로 각각 감싼다

이벤트 등록 메서드

이벤트: 사이트에서 방문자가 취하는 모든 행위



이벤트 등록 메소드

- 단독 이벤트 등록 메소드: 하나의 이벤트만 등록

- 그룹 이벤트 등록 메소드: 2개 이상의 이벤트를 등록 on() 메소드 사용

이벤트 제거 메소드 off()

텍스트이(가) 표시된 사진

자동 생성된 설명

로딩 이벤트 메소드

load()

- 외부에 연동된 소스 (iframe, img, video)의 로딩이 완료된 후 이벤트 발생

ready()

- 지정한 HTML 문서 객체의 로딩이 완료된 후 이벤트 발생

$(document).ready(function(){ 코드 })

error()

- 이벤트 대상 요소에서 오류가 발생하면 이벤트 발생

마우스 이벤트 메소드

click()

- 선택한 요소를 클릭했을 때 이벤트 발생

dblclick()

- 선택한 요소를 더블 클릭했을 때 이벤트 발생

mouseout()

- 선택한 요소에서 마우스 포인터가 벗어났을 때 이벤트 발생

mouseover()

- 선택한 요소에 마우스 포인터를 올렸을 때 이벤트 발생

hover()

- 선택한 요소에 마우스 포인터를 올렸을 때와 벗어났을 때 각각 이벤트 발생

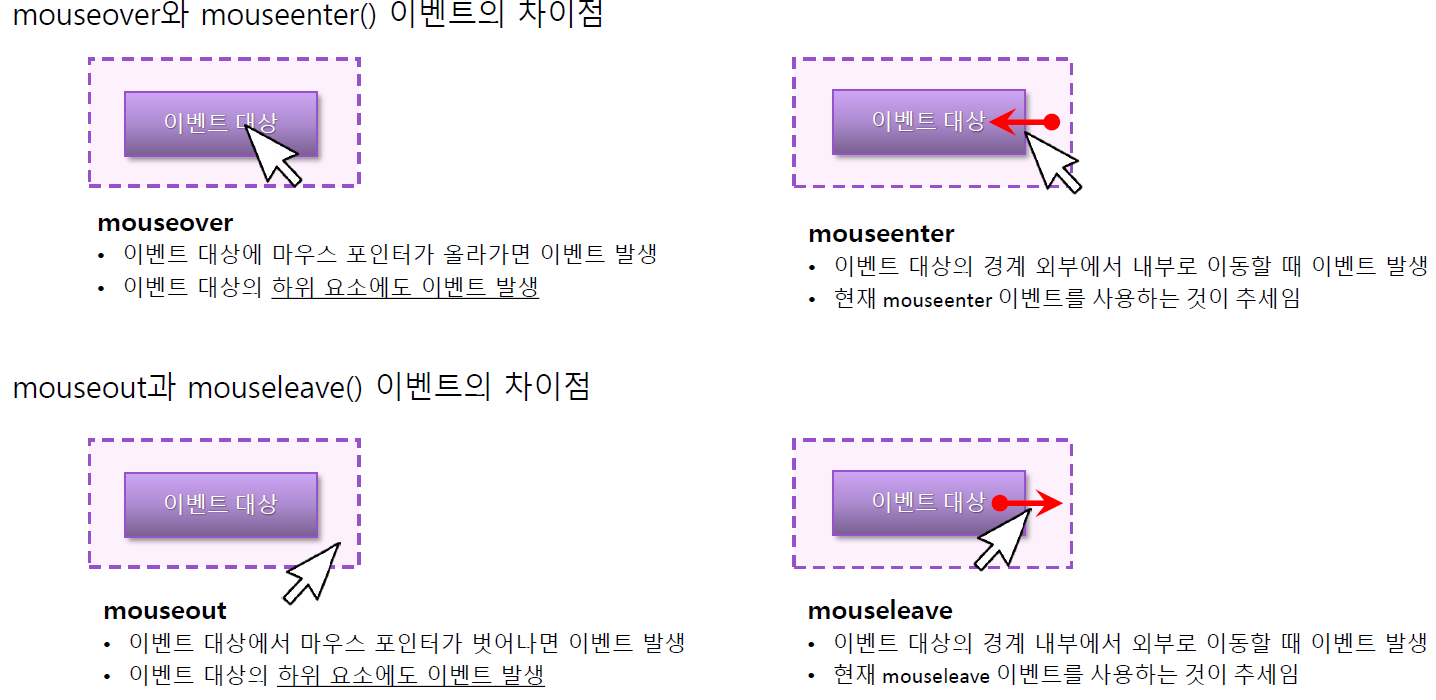
|  |
| --- |
| $(“이벤트 대상”).hover(  function(){ 마우스 over 일 때 실행될 코드 };  function(){ 마우스 out 일 때 실행될 코드 };  ); |

mouseenter()

- 선택한 요소가 포함된 경계 범위에 마우스 포인터가 들어왔을 때 이벤트 발생

mouseleave()

-선택한 요소가 포함된 경계 범위에서 마우스 포인터가 벗어났을 때 이벤트 발생



mousedown()

- 선택한 요소에 마우스 버튼을 눌렀을 때 이벤트 발생

mouseup()

- 선택한 요소에서 마우스 버튼을 눌렀다 떼었을 때 이벤트 발생

mousemove()

- 선택한 요소 범위에서 마우스 포인터를 움직였을 때 이벤트 발생

scroll()

- 스크롤바를 움직일 때마다 이벤트 발생

마우스 이벤트 객체의 속성 종류

clientX

- 마우스 포인터의 x좌표 값 반환 스크롤 이동 거리 무시

clientY

- 마우스 포인터의 y좌표 값 반환 스크롤 이동 거리 무시

pageX

- 스크롤 x축의 이동한 거리를 계산하여 마우스 포인터의 x 좌표 값을 반환

pageY

- 스크롤 y축의 이동한 거리를 계산하여 마우스 포인터의 y 좌표 값을 반환

screenX

- 모니터를 기준으로 마우스 포인터의 x 좌표 값을 반환

screenY

- 모니터를 기준으로 마우스 포인터의 y 좌표 값을 반환

layerX

- position을 적용한 요소를 기준으로 마우스 포인터의 x 좌표 값을 반환

layerY

- position을 적용한 요소를 기준으로 마우스 포인터의 y 좌표 값을 반환

button

- 마우스 버튼의 종류에 따라 값을 반환 (왼쪽: 0, 휠: 1, 오른쪽: 2)

포커스 이벤트 메소드

focus()

- 선택한 요소로 포커스가 이동하면 이벤트 발생

blur()

- 포커스가 선택한 요소에서 다른 요소로 이동하면 이벤트가 발생

focusin()

- 선택한 요소의 하위 요소로 포커스가 이동하면 이벤트 발생

focusout()

- 포커스가 선택한 요소의 하위 요소에서 외부 요소로 이동되었을 때 이벤트 발생

change()

- 이벤트 대상인 입력 요소의 값이 변경되고 포커스가 이동하면 이벤트 발생

키보드 이벤트 메소드

keypress()

- 키보드를 눌렀을 때 이벤트 발생 .

기능키

(([F1]~[F12], [Alt], [Ctrl], [Shift], [Backspace], [Caps Lock], [Tab], [한/영], [방향키], 한글키 등)를 제외한 키보드의 모든 키의 코드 값을 반환

keydown()

- 키보드를 눌렀을 때 이벤트 발생 .

한글키를 제외한 키보드의 모든 키의 코드 값을 반환

keyup()

- 키보드에서 손을 떼었을 때 이벤트 발생

키보드 이벤트 객체의 속성 종류

keycode

- 키보드의 아스키 코드 값을 반환

altKey

- 이벤트 발생시 [Alt] 키가 눌렸으면 true, 아니면 false를 반환

ctrlKey

- 이벤트 발생시 [Ctrl] 키가 눌렸으면 true, 아니면 false를 반환

shiftKey

이벤트 발생시 [Shift] 키가 눌렸으면 true, 아니면 false를 반환

그룹 이벤트 등록

on()

- 선택한 요소에 2개 이상의 이벤트를 등록. 라이브 이벤트 등록 가능

off()

- on() 메소드로 등록한 이벤트를 제거한다.

one()

- 지정한 이벤트가 1회 발생하고 자동으로 해제. 라이브 이벤트 등록 가능

라이브 이벤트

- 새로 생성되거나 복제된 요소에 이벤트 등록

|  |
| --- |
| $(“document | 이벤트 대상의 상위요소”).on(“이벤트 종류”, “이벤트 대상”, function(){  코드 작성;  });  $(“document | 이벤트 대상의 상위요소”).one(“이벤트 종류”, “이벤트 대상”, function(){  코드 작성;  });  $(“document | 이벤트 대상의 상위요소”).off(“이벤트 종류”, “이벤트 대상“); //라이브 이벤트 제거 |

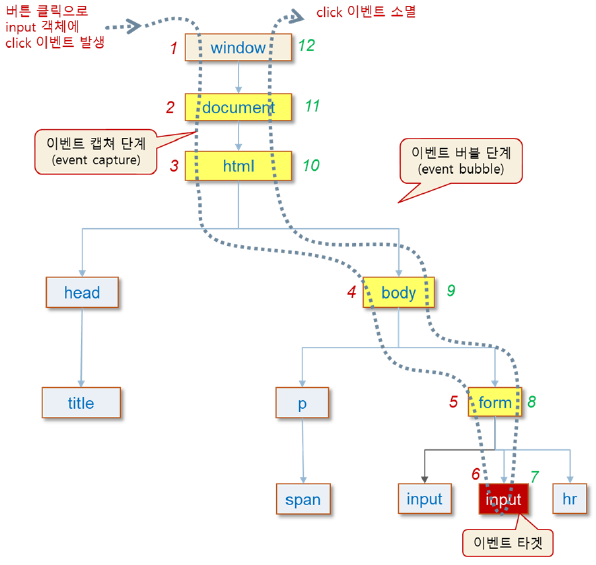
이벤트 흐름

이벤트 단계

- 단계1 캡쳐 단계: jQuery 에서는 기본적으로 타깃 버블 단계에서 이벤트가 등록되도록 되어있다

- 단계2 타깃 단계: 이벤트를 발생시킨 대상

- 단계3 버블 단계: 타깃에서 document까지 상위로 전파되는 단계



이벤트 객체의 속성 종류 function(e)

target

- 실제 이벤트를 발생시킨 대상

cancelBubble()

- 이벤트의 전파를 차단하는 속성으로 기본값은 false. true로 설정하면 전파가 차단됨

stopPropagation()

- 이벤트가 상위로 전파되는 것을 차단

preventDefault()

- 기본 이벤트를 차단

Ex) <a>에 클릭 이벤트가 발생하면, href속성의 링크 주소로 이동하는데 이런 기본 이벤트를 차단할 수 있다

효과 메서드

선택한 요소를 숨겼다가 보이게 만드는 기능을 가진 메소드

숨김

hide()

- 요소를 숨긴다

fadeOut()

- 요소가 점점 투명해지면서 사라진다

slideUp()

- 요소가 위로 접히며 숨겨진다

노출

show()

- 숨겨진 요소가 나타난다

fadeIn()

- 숨겨진 요소가 점점 선명해진다

slideDown()

- 숨겨진 요소가 아래로 펼쳐진다

노출, 숨김

toggle()

- hide(), show() 효과를 적용한다

fadeToggle()

- fadeIn(), fadeOut () 효과를 적용한다

slideToggle()

- slideUp(), slideDown () 효과를 적용한다

fadeTo()

- 지정한 투명도를 적용한다

효과 메서드 기본형

$(“요소선택”).효과메소드(효과소요시간, 가속도, 콜백함수);

효과소요시간 : 요소를 숨기거나 노출할 때 소요되는 시간

“slow”, “normal”, “fast” or 1000(1s)

가속도 : 숨기거나 노출하는 동안의 가속도. 생략가능

“swing” (시작과 끝은 느리게, 중간은 빠른 속도로, 기본값)

“linear” (일정한 속도로 움직임)

콜백함수 : 노출과 숨김효과가 끝난 후에 실행할 함수. 생략가능

fadeTo() 메소드의 기본형

$(“요소선택”).fadeTo(효과소요시간, 투명도, 콜백함수);

투명도 : 0~1까지의 값 입력, 1에 가까울수록 선명함.

애니메이션 메소드

animate() 메소드

- 선택한 요소에 다양한 동작(Motion) 효과를 적용한다.

animate() 메소드의 기본형

$(“요소선택”).animate({스타일 속성}, 적용시간, 가속도, 콜백함수);

스타일 속성 : 애니메이션으로 적용할 스타일 속성

opacity, height, width, top, bottom, left, right, margin, padding

효과소요시간 : 요소를 숨기거나 노출할 때 소요되는 시간

“slow”, “normal”, “fast” or 1000(1s)

가속도 : 숨기거나 노출하는 동안의 가속도. 생략가능

“swing” (시작과 끝은 느리게, 중간은 빠른 속도로, 기본값)

“linear” (일정한 속도로 움직임)

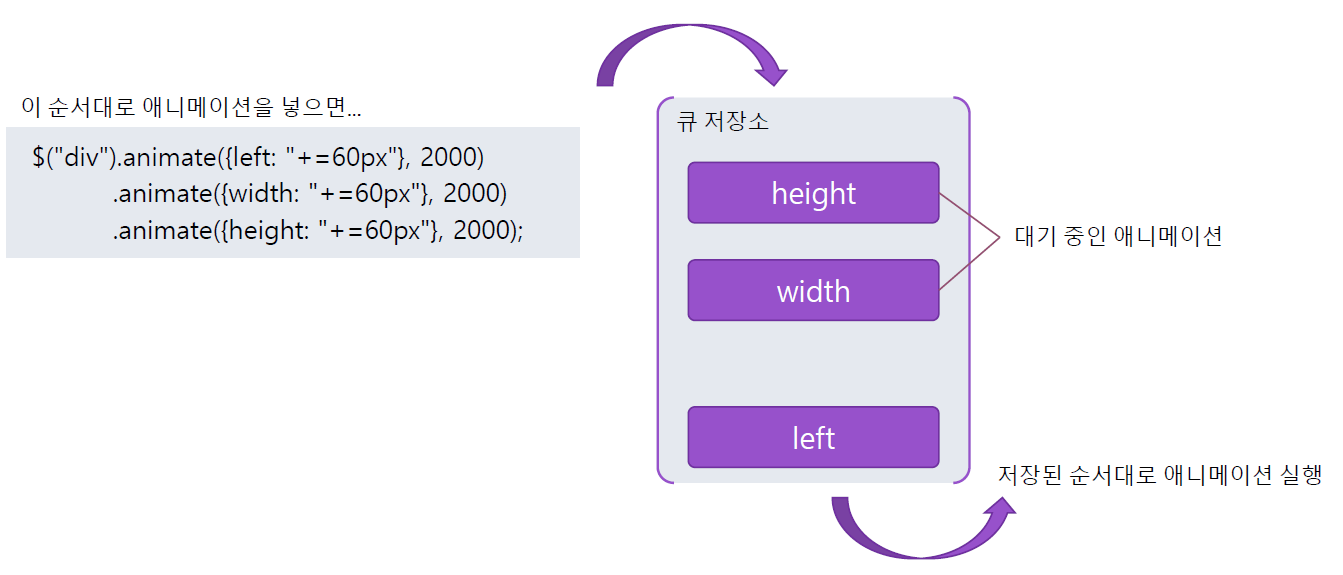
콜백함수 : 노출과 숨김효과가 끝난 후에 실행할 함수. 생략가능

애니메이션 큐

애니메이션 적용 원리와 큐의 개념

- 애니메이션 메소드는 함수가 적용된 순서대로 큐(queue)라는 저장소에 저장된다.

- 큐에 저장된 애니메이션은 먼저 들어간 애니메이션이 먼저 실행된다. (FIFO:First In First Out)



애니메이션 효과 제어 메소드

효과 또는 애니메이션이 적용된 요소의 동작을 제어하는 메소드

stop()

- 현재 실행 중인 애니메이션을 정지시킨다

delay()

- 지정한 시간만큼 지연했다가 애니메이션을 진행한다

queue()

- 큐에 사용자 정의 함수를 추가하거나 큐에 대기 중인 함수를 배열에 담아 반환하고 queue() 메소드 이후의 애니메이션은 모두 취소한다

dequeue

- queue() 메소드 실행 이후에 대기 중인 애니메이션이 취소되지 않게 연결한다

clearQueue()

- 처음으로 진행 중인 애니메이션을 제외하고 큐에 대기 중인 모든 애니메이션을 제거한다

finish()

- 선택한 요소의 모든 애니메이션을 강제로 완료 시점으로 보낸 후 종료한다

stop(clearQueue, finish)

- clearQueue, finish는 true, false의 값을 입력함. 기본값은 false

- clearQueue가 true면 큐에서 대기 중인 애니메이션을 모두 제거

- finish가 true면 진행 중인 첫번째 애니메이션을 강제로 종료

stop()

- 큐에 대기 중인 애니메이션은 계속 실행, 진행 중인 첫번째 애니메이션 정지

stop(true)

- 큐에 대기 중인 애니메이션 모두 제거, 진행 중인 첫번째 애니메이션 정지

stop(false, true)

- 큐에 대기 중인 애니메이션은 계속 실행, 진행 중인 첫번째 애니메이션 강제 종료(종료 시점)

stop(true, true)

- 큐에 대기 중인 애니메이션 모두 제거, 진행 중인 첫번째 애니메이션 강제 종료(종료 시점)