

HTML5 & Java Script CSS & JQuery

목 차

1. HTML5
2. Java Script
3. DOM(Document Object Model)
4. Ajax
5. CSS
6. JQuery

1. HTML5

- HTML5 개요
- HTML5 추가된 태그

HTML5 개요

Hyper Text Markup Language 5의 약자
웹 문서 제작을 위한 표준 기술

HTML5 =
HTML + CSS3 + Javascript

분리되지 않은 하나의 통합된 체제

HTML5 개요-DOCTYPE

- HTML 4.01 Transitional DTD

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://  
www.w3.org/TR/html4/loose.dtd">
```

- XHTML 1.0 Strict DTD

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://  
www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

- HTML5 DTD

```
<!DOCTYPE html>
```

HTML5 개요-Semantic 태그

태그의 이름만 보아도 기능을 알 수 있는 태그
기존 html 태그와 차이

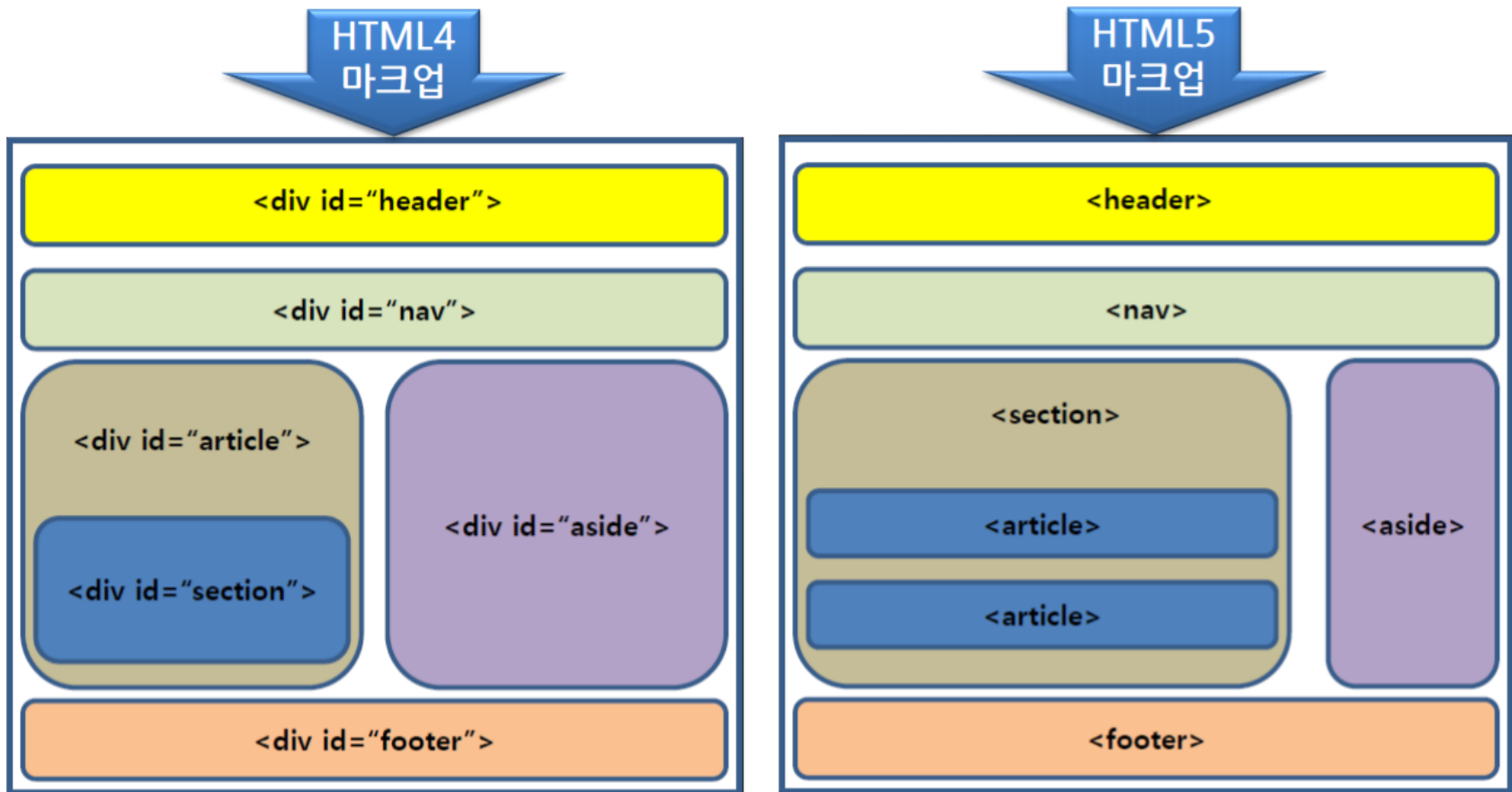
<header>

<nav>

<section>

<article>

HTML5 개요 - Semantic 태그



HTML5 추가된 태그 1

HTML 태그	속성
<article>	웹 페이지의 본문을 정의할 때 사용됩니다
<aside>	Article의 내용을 보충해 주는 역할을 하는 콘텐츠를 넣을 때 사용합니다.
<audio>	오디오를 재생할 때 사용합니다.
<canvas>	그래픽을 보여줄 때 사용합니다. (스크립트 언어를 사용해서 구현됩니다.)
<command>	명령 버튼을 만들 때 사용합니다.
<datalist>	드롭다운 리스트를 만들 때 사용합니다.
<details>	상세한 내용을 보여줄 때 사용합니다.
<embed>	플러그인이나 플래시 요소를 보여줄 때 사용합니다.
<figcaption>	<figure> 와 함께 사용되며, <figure>의 캡션을 추가할 때 사용됩니다.
<figure>	이미지나 사진, 코드 등을 보여줄 때 사용됩니다.

HTML5 추가된 태그 2

HTML 태그	속성
<footer>	푸터를 정의할 때 사용합니다.
<header>	헤더를 정의할 때 사용합니다.
<hgroup>	H1부터 H6 의 그룹을 만들 때 사용됩니다.
<keygen>	폼에서 사용되며, 로컬상에 보안 키를 저장하고 공개키는 서버로 보냅니다.
<mark>	텍스트에 마크 펜으로 칠한 효과를 줍니다.
<meter>	길이를 나타내줍니다. 그래픽적으로 어느 정도 길이인 것을 나타내 줍니다.
<nav>	메뉴를 정의할 때 사용합니다.
<output>	계산된 결과를 나타낼 때 사용됩니다.
<progress>	다운로드 같이 몇 %가 남아있는지 표시할 때 사용됩니다.
<section>	섹션을 정의할 때 사용합니다.

HTML5 추가된 태그 3

HTML 태그	속성
<source>	오디오, 또는 비디오 태그와 같이 사용되며, 소스를 나타낼 때 사용됩니다.
<summary>	Details 태그와 같이 사용되며, 상세한 내용의 요약을 나타냅니다.
<time>	문서 내부에 시간을 정의할 때 사용합니다.
<video>	비디오를 재생할 때 사용합니다.
<wbr>	문서의 내용이 길어서 다음 라인으로 표시될 때, 영문인 경우 같은 라인에 문장을 표시해야 하는 경우 사용 됩니다.

HTML5 추가된 태그 4

Input type	의미	Input type	의미
<input type="search">	검색	<input type="url">	URL
<input type="number">	숫자	<input type="email">	이메일
<input type="range">	범위	<input type="date">	일
<input type="color">	색상	<input type="month">	월
<input type="tel">	전화번호	<input type="week">	주
<input type="time">	시간	<input type="datetime">	날짜와 시간

2. Java Script

- JavaScript 개요
- JavaScript 문법
- 이벤트

JavaScript 문법 - HTML 문서에 적용 문법

- ▶ 외부 자바스크립트 파일을 HTML에 적용시

```
<script language="스크립트 언어이름" src="외부 스크립트 파일의 URL">  
    var varName; functio  
    n 함수명(){ ... } 주석[  
    // or /* */ ]  
</script>
```

- ▶ HTML 내부에서의 자바스크립트

```
<script type="text/javascript">  
    var varName; functi  
    on 함수명(){...} 주석[  
    // or /* */ ]  
</script>
```

JavaScript 문법 – 주요 데이터 타입

데이터 타입		설명	
기본 데이터 타입	숫자	정수, 실수	
	문자열	문자열 타입, " " 또는 ' ' 표현	
	boolean	참(true)과 거짓(false)	
	null	아무런 값도 할당하지 않은 알 수 없다는 의미	
레퍼런스 타입	내장 객체	window	웹 브라우저 창 및 프레임을 표시하기 위한 최상위 객체
		document	HTML 파일 및 기술된 문서의 정보 제공 및 그에 대한 작업을 수행하는 객체로 Layer, Image, Form등의 하위 객체 보유
		Date	날짜와 시간의 정보를 얻어 낼 때 사용하는 객체
		Array	배열을 생성할 때 사용하는 객체
		RegExp	정규 표현식 객체로 정규 표현식의 패턴을 보유
		...	http://www.w2school.com 참조

JavaScript 문법 – 변수 문법과 특징

▶ 문법

- 변수 타입 선언 없이 변수명만 기술

```
var 변수명;  
var 변수명 = 값;
```

▶ 특징

- 값 대입 시점에 변수 타입 자동 설정

JavaScript 문법 - 선언 위치에 따른 변수 종류

▶ 전역 변수

- 함수 외부에서 선언
- 함수 내에서 선언시 `var` 키워드 없이 선언
- 프로그램 전역에서 사용 가능

▶ 지역 변수

- 함수내에서 `var` 키워드 사용하여 선언
- 선언된 함수 내에서만 사용

Java script 문법 - 함수 종류

- ▶ 사용자 정의 함수
 - 발생한 이벤트를 처리하기 위해 사용자가 정의한 함수
- ▶ 내장 함수
 - 자체적으로 제공하는 특정 작업을 수행하는 내장 함수

Java script 문법 - 사용자 정의 함수

▶ 문법

[구현 문법]

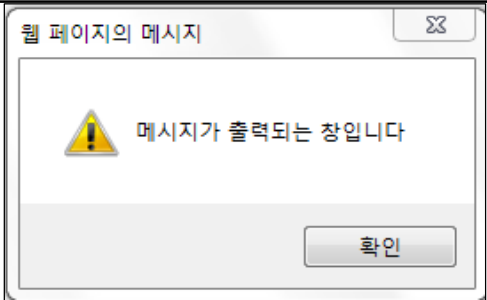
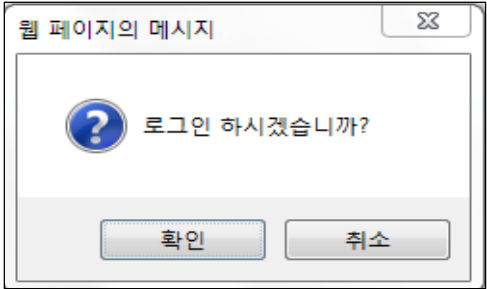
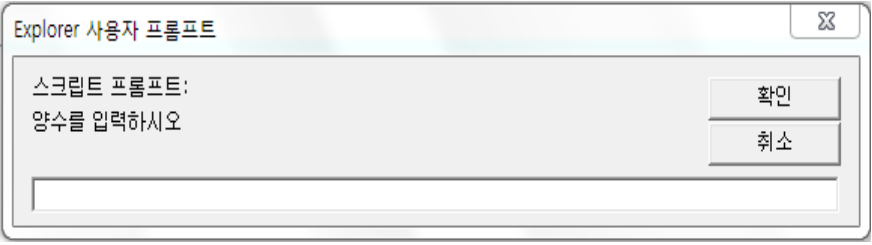
```
function 함수명( [args1, args2,..., argsn] ){  
    statement  
    [return expression;]  
}
```

[호출 문법]

1. 매개 변수가 없는 함수 호출 - 함수명();
2. 매개 변수가 있는 함수 호출 - 함수명("데이터");

- function은 함수 선언시 필수 키워드
- 함수명 임의 설정 가능
- 매개 변수 선언 가능
- 해당 함수를 호출한 코드에 "return expression" 로 값 반환

Java script 문법 – 주요 내장 함수(1/2)

함수명	설명	
alert()	메시지 출력 다이얼로그 창	
confirm()	특정 문자열을 표시하고 [확인],[취소] 버튼 제공 확인 클릭 – true 반환 취소 클릭 – false 반환	
prompt()	입력 상자를 화면에 표시해서 특정 문자열 또는 숫자값을 입력받 음	

Java script 문법 – 주요 내장 함수(2/2)

함수명	설명
eval()	특정 문자열을 객체로 변환 예 : "document.myForm" 이라는 문자열을 eval("document.myForm")으로 할 경우 document.myForm 객체로 자동 변환 따라서 myForm의 하위 객체에 access 가능
escape()	문자를 인코딩할 때 사용되는 함수 영문자와 숫자는 그대로 표시
encode URIComponent()	문자열을 utf-8형식으로 변환

이벤트 - 이벤트와 이벤트 핸들러

- ▶ 이벤트란?
 - 어플리케이션 상에서 발생하는 것으로, 어떤 작업을 수행하기 위한 동작
 - 예 : 사용자가 키 입력, 마우스 클릭...
 - 70여 가지의 이벤트 속성 제시됨
- ▶ 이벤트 핸들러
 - 이벤트가 발생되었을 때 상응하는 작용을 감지하고 처리하는 로직
 - 메소드로 구현됨

이벤트 - 주요 이벤트 속성

이벤트	설명
onBlur	입력 폼 등에서 특정 필드의 포커스가 다른 곳으로 이동했을 때
onChange	입력 폼 등에서 특정 필드의 내용이 변경 되었을 때
onClick	입력 폼 등에서 특정 필드나 단추를 클릭했을 때
onFocus	입력 폼 등에서 특정 필드의 문자 데이터가 선택되었을 때
onSubmit	입력 폼 등에서 "submit" 버튼을 클릭 했을 때
onSelect	입력 폼 등에서 특정 필드의 문자 데이터가 선택되었을 때
onLoad	웹 브라우저상에서 페이지가 읽혀졌을 때
...70여개	...

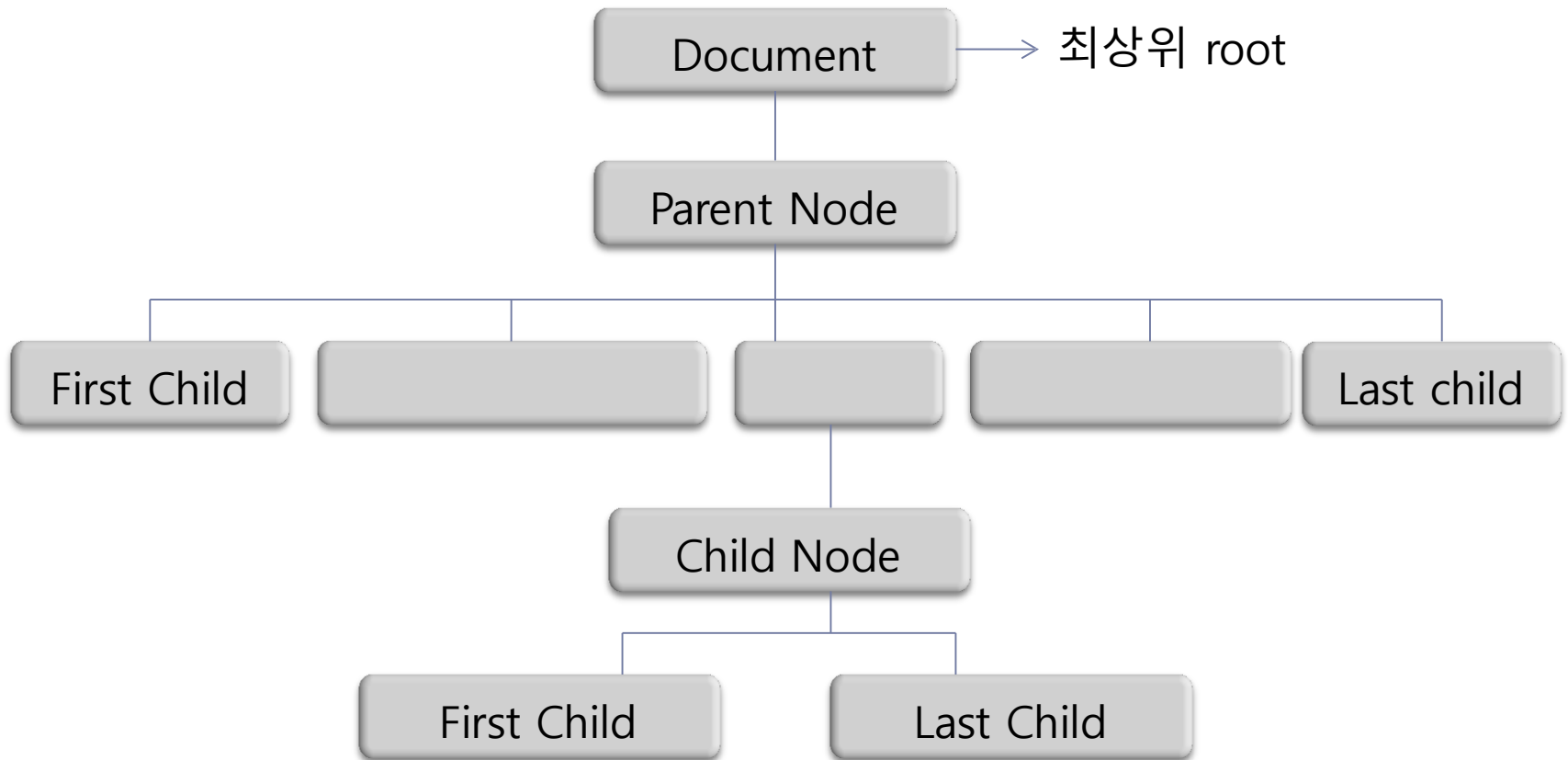
3. DOM(Document Object Model)

- DOM[도큐먼트 객체 모델]이란?
- DOM의 기본 구조
 - HTML DOM Tree
 - DOM tree 구조 이해하기
- DOM Parsing

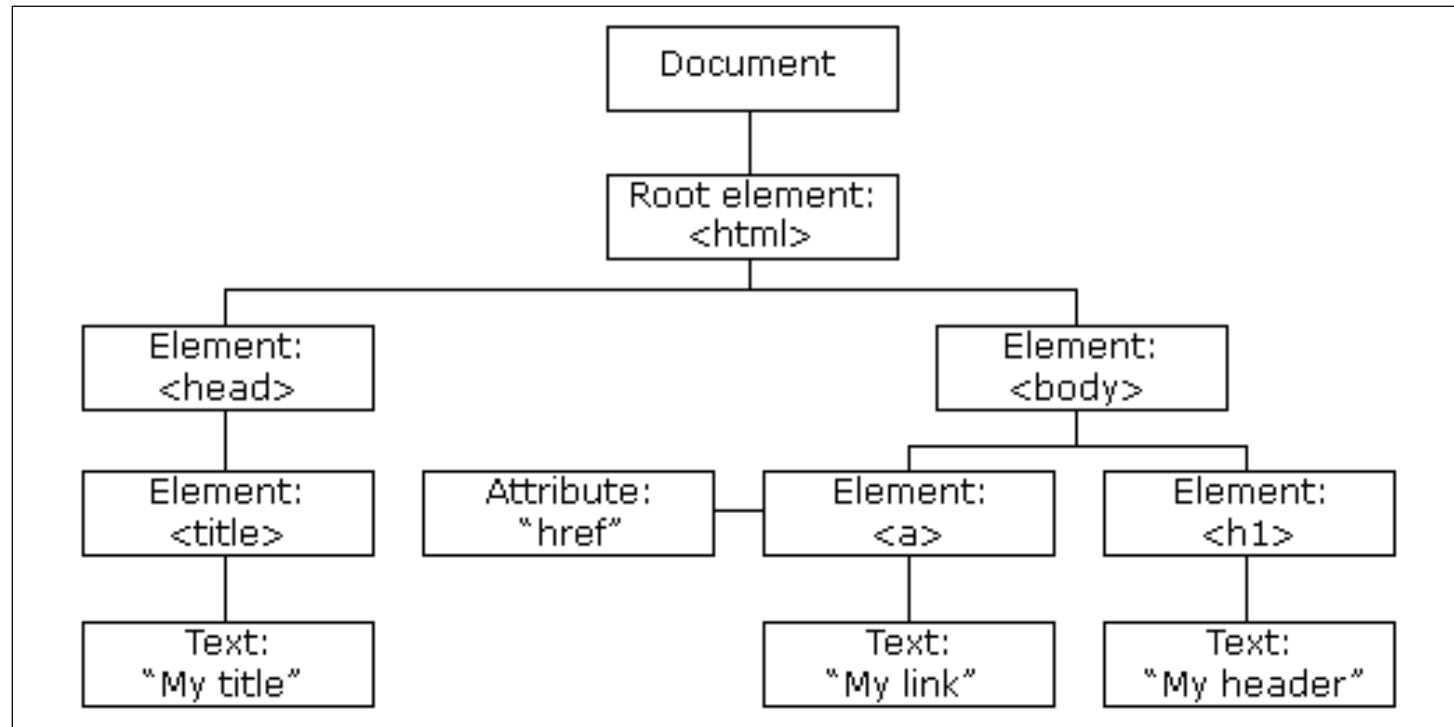
DOM 이란?

- ▶ 도큐먼트 객체 모델
- ▶ HTML, XML 문서를 표현하고 조작하는 표준적인 방법
- ▶ 문서를 객체 모델로 구현해서 문서의 구조와 내용에 access하기 위한 스펙
 - Random Access Mechanism
 - 동적으로 문서의 수정, 삭제, 삽입, 조회 가능
 - 메소드와 속성, 객체간의 관계를 정의하기도 함
- ▶ W3C(World Wide Web Consortium)에서 발표
- ▶ 플랫폼과 언어에 독립적
 - 다양한 환경과 어플리케이션에서 사용 가능한 표준 인터페이스 제공

DOM의 기본 구조



DOM의 기본 구조 - HTML DOM Tree



DOM의 기본 구조 - DOM tree 구조 이해하기

- ▶ Element와 Text를 tree구조로 관리
- ▶ newline과 blank도 하나의 Text 로 인식
- ▶ access 방식
 - 부모 -> 자식 -> 자손 순서로 access
 - 형제 관계에서 형, 동생 순서로 access
 - 특정 Element 를 access 할 수도 있음
 - 메소드를 통한 access도 가능

DOM parsing(1/8)

- ▶ 노드 개체의 속성
 - nodeName : 노드의 이름
 - nodeValue : 노드의 값. 일반 요소 노드인 경우에는 null
 - nodeType : 노드의 타입
(XML 문서 개체는 9,
Element 노드는 1,
Text 노드는 3)

DOM pasing(2/8)

- ▶ 노드와 노드 사이의 관계
 - childNodes – 자식 노드들 반환
 - firstChild: 첫 번째 자식 노드
 - lastChild: 마지막 자식 노드
 - parentNode: 부모 노드
 - previousSibling:
 - ▶ 현재 노드와 같은 부모를 갖는 자식 노드 중 동일한 레벨상의 앞단 노드
 - nextSibling:
 - ▶ 현재 노드와 같은 부모를 갖는 자식 노드 중 동일한 레벨상의 현재 노드 다음의 노드

DOM passing(3/8)

- ▶ 자식 노드에 접근하기
 - childNodes 속성: 자식 노드들을 항목으로 하는 배열
 - hasChildNodes() 메소드: 자식 노드가 있는 경우 true, 없으면 false.
 - 하위 노드들을 트리 형태로 출력할 때: 재귀 함수 사용
- ▶ 노드 id로 검색하기
 - getElementById () : 특정 id로 설정되어 있는 노드 검색
- ▶ 노드 이름으로 검색하기
 - getElementsByTagName() 메소드: 하위 노드 중에서 인자로 지정한 이름의 노드를 모두 찾아서 배열로 리턴
- ▶ 노드 생성: createElement(), createTextNode()
 - 요소 노드 생성: [XML문서개체].createElement("노드이름");
 - 텍스트 노드 생성: [XML문서개체].createTextNode("내용");

DOM passing(4/8)

- ▶ 노드 추가/이동: `appendChild()`, `insertBefore()`
 - `A.appendChild(B)`: A 노드의 맨 마지막 자식 노드 위치에 B 노드 추가
 - `A.insertBefore(B,C)` : A 노드의 자식 노드인 C 노드 앞에 B 노드를 추가
- ▶ 노드 복사하기: `cloneNode`
 - `A.cloneNode(true|false)` : A 노드를 복사해서 돌려줌, 인자가 true면 자식 노드도 복사
 - XML 문서 개체에 대해서 `cloneNode()` 메소드를 사용하면 XML 문서 개체도 복사할 수 있음
- ▶ 노드 삭제하기: `removeChild`
 - `A.removeChild(B)`: A 노드의 자식 노드인 B 노드를 삭제

DOM passing(5/8)

- ▶ 노드 바꾸기: `replaceChild()`
 - `A.replaceChild(B, C)`: A 노드의 자식 노드인 C 노드를 B 노드로 바꿈
 - 문서 개체 모델 상에 B 노드가 있다면 B 노드도 삭제된다는 것 주의
- ▶ 속성 읽기: `getAttribute()`, `getAttributeNode()`
 - `A.getAttribute(B)`: A 노드에서 이름이 B인 속성의 값을 문자열로 반환
 - `A.getAttributeNode(B)`: A 노드에서 이름이 B인 속성을 속성 반환

DOM passing(6/8)

- ▶ 속성 수정하기: `setAttribute()`, `setAttributeNode()`
 - `A.setAttribute(B,C)`: A 노드에서 이름이 B인 속성의 값을 C로 지정
 - `A.setAttributeNode(B)`:
 - ▶ A 노드에서 B 속성 개체를 적용
 - ▶ B 속성 개체는 다른 노드에 적용되어 있지 않아야 함
 - ▶ 다른 요소 노드의 속성 개체일 때에는 `cloneNode()` 메소드로 복사한 후에 추가해야 함
 - ▶ `[속성개체].value = "속성값"`

DOM passing(7/8)

- ▶ 속성 삭제하기

: removeAttribute(), removeAttributeNode()

- A.removeAttribute(B):
 - ▶ A 노드에서 이름이 B인 속성 삭제
- A.removeAttributeNode(B):
 - ▶ A 노드에서 속성 개체 B를 삭제

DOM parsing – example(8/8)

```
5<script language="javascript">
6  function guess() {
7      var element = document.documentElement.lastChild;
8      alert("document의 마지막 자식의 nodeName? " + element.nodeName);//BODY
9
10     var anotherElement = document.getElementsByTagName("h1")[0];
11     alert("h1 tag의 첫번째 tag의 nodeValue 값은? " + anotherElement.nodeValue);//null
12
13     var child = anotherElement.firstChild;
14     alert("h1 tag의 첫번째 tag의 firstChild의 nodeValue는? " + child.nodeValue);//애완 동물 이야기
15
16     element = document.getElementById("cat").lastChild;
17     alert("cat id값을 보유한 tag의 nodeValue는? " + element.nodeValue);//고양이
18
19     alert("부모 tag의 id 속성 value는? "
20           + element.parentNode.getAttributeNode("id").nodeValue);//cat
21 }
22 </script>
23 </head>
24<body>
25  <h1>애완 동물 이야기</h1>
26  <div id="message">
27      내 애완 동물을 강아지야 너의 애완 동물은
28      <span id="cat">
29          고양이
30      </span>
31      였으면 좋겠어
32  </div>
33  <form> <input type="button" value="애완동물은?" onClick="guess();" /> </form>
34 </body>
```

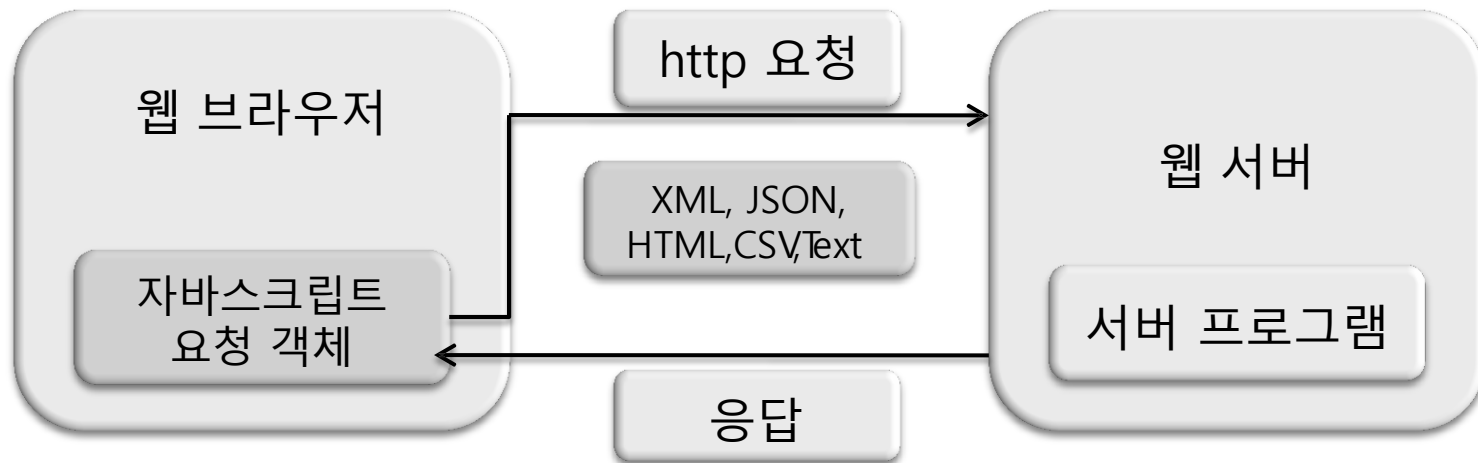
4. Ajax

- Ajax란?
- 동기 통신 : 비동기 통신
- Ajax 적용 기술

Ajax란? (1/2)

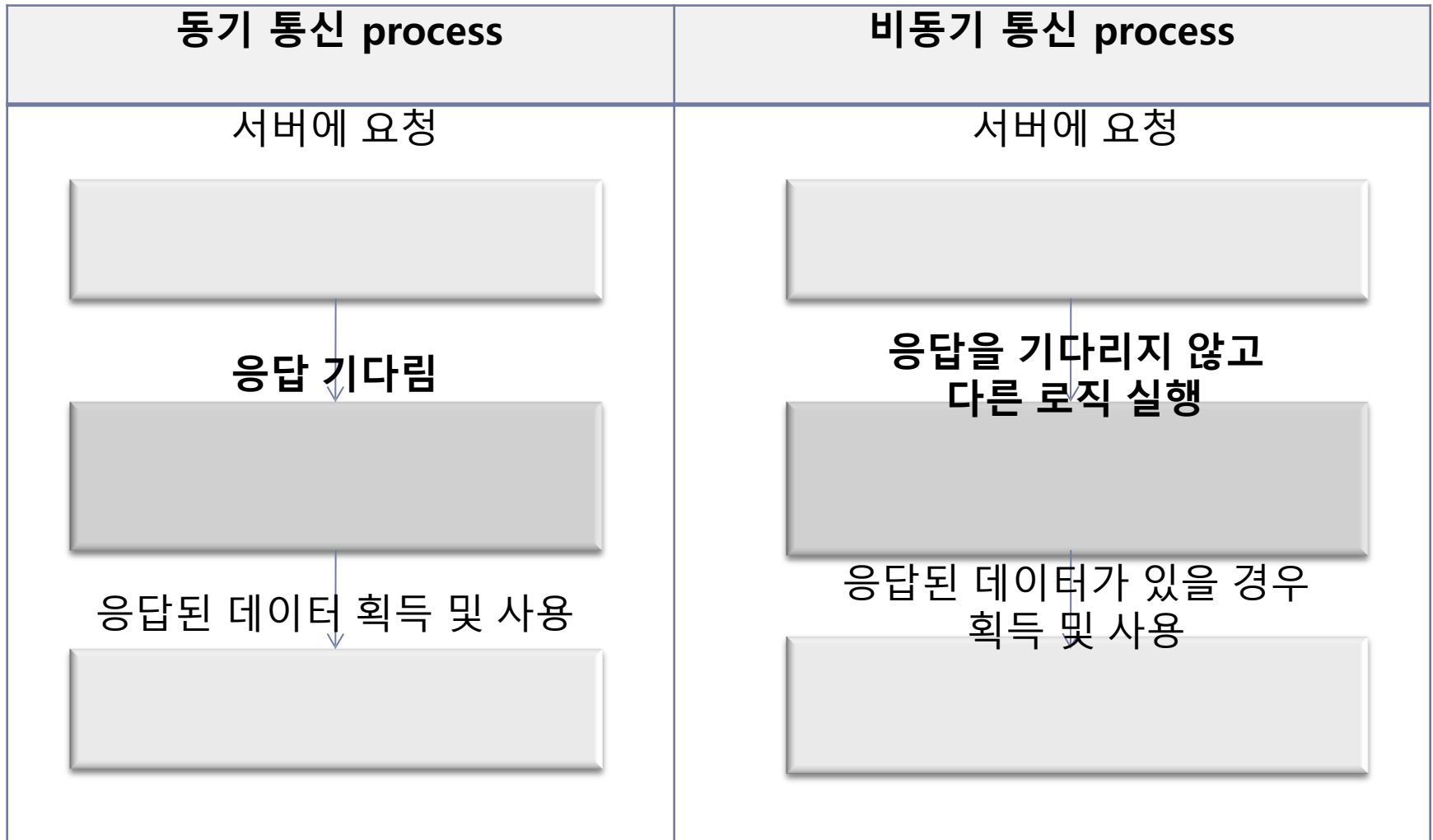
- ▶ Asynchronous JavaScript & XML
 - Java Script에 의한 비동기적인(Asynchronous) 통신으로 XML기반인 데이터를 클라이언트인 웹 브라우저와 서버 사이에서 교환하는 방식 의미
 - 주 목적
 - ▶ 웹 어플리케이션을 데스크톱 어플리케이션처럼 빠른 응답을 갖는 구조로 개발하고자 함

Ajax란? (1/2)



- ▶ 자바스크립트로 HTTP 요청을 보내고 XML, JSON, Text 등으로 응답받아 사용하는 비동기 통신 개발 기술

동기 통신 : 비동기 통신



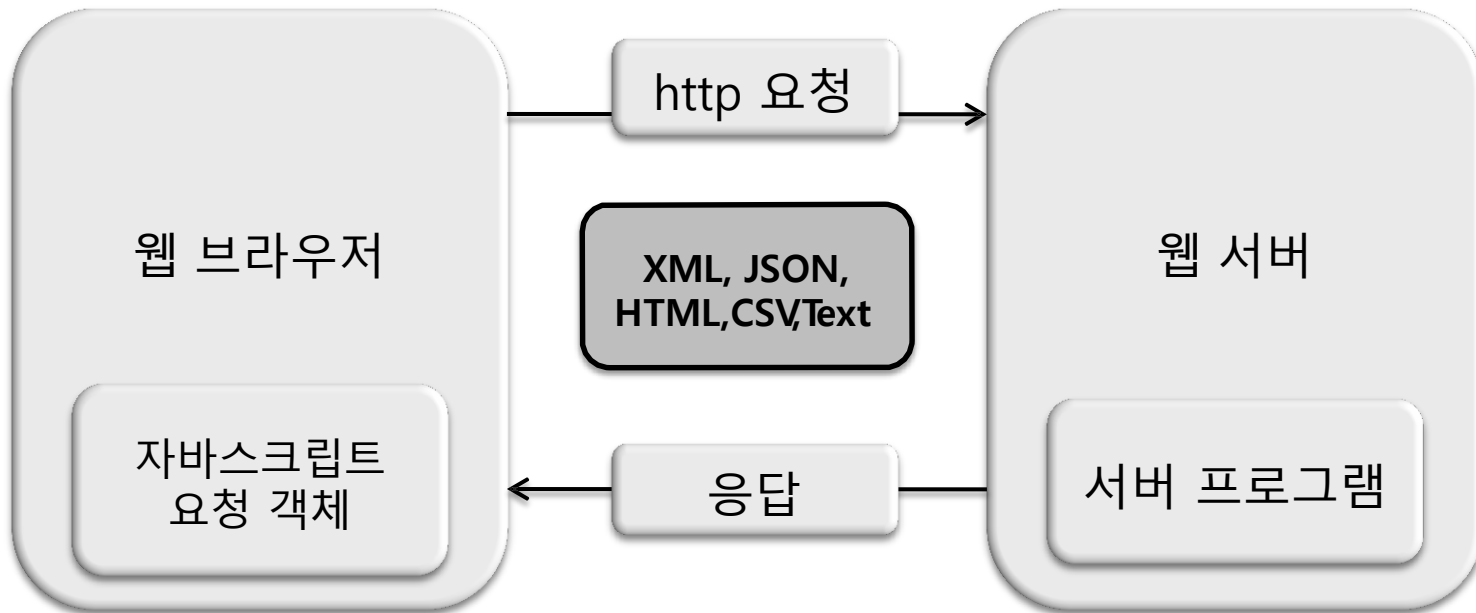
동기 통신 : 비동기 통신

동기 통신 특징	비동기 통신 특징
<p>발생 가능한 문제점</p> <ul style="list-style-type: none">- 서버의 응답이 지연될 경우 그 시스템에서 계속 딜레이가 발생	<p>개발 코드</p> <ul style="list-style-type: none">- 응답을 기다리지 않고 다른 일을 할 수도 있기 때문에 비 동기적으로 응답 이벤트를 처리할 수 있는 핸들러 개발 <p>웹 브라우저</p> <ul style="list-style-type: none">- 서버로 부터 전체 페이지를 반환 받지 않기 때문에 페이지의 이동 및 새로고침 현상이 없음 <p>발생 가능한 문제점</p> <ul style="list-style-type: none">- 웹 페이지와 서버 간에 벌어지는 대부분의 상호 작용들은 사용자에게 보이지 않음. 100% 신뢰 할수 없다는 단점- 요청 후 응답 유무와 상관없이 다른 로직을 실행할 수 있기 때문에 순차적으로 무언가 하기에는 부적합

4-1. Ajax 데이터

- Ajax에서 사용되는 데이터 형식
- JSON의 개요
- JSON 데이터 형식
- JSON 데이터 사용

Ajax에서 사용되는 데이터 형식(1/2)



Ajax에서 사용되는 데이터 형식(2/2)

데이터 포맷	설명
XML	확장 가능한 마크업 언어로 데이터를 구조화 할 수 있는 언어
JSON	JavaScript를 기반으로 만들어졌으나, 프로그래밍 언어 및 이종 시스템간 데이터 교환시에 사용 할 수 있는는 이상적인 text 포맷
HTML	웹 브라우저상에 문자나 이미지 등을 표현하는 마크업 언어
CSV	데이터를 , 표기를 기준으로 구분한 text 데이터
TEXT	일반 문자열

JSON의 개요

- ▶ JavaScript Object Notation
- ▶ JSON 은 텍스트 기반의 경량(lightweight) 데이터 변환 포맷
- ▶ 사용자가 읽고, 쓰기 쉬워 데이터 처리가 쉬움
- ▶ JavaScript를 기반으로 만들어졌으나, 프로그래밍 언어에 독립적인 text 형식
- ▶ 기기종간의 데이터 교환에 적합

JSON 데이터 형식

▶ 컬렉션데이터 구조

- 이름/값 쌍(pair)으로 이루어진 데이터 집합
- 중괄호 내에 key와 value 로 구성
- key와 value 사이에는 : 표기로 구분
- key는 문자열 타입, value는 JSON 데이터
- 다수의 데이터 존재할 경우 key와 value 값들은 , 표기로 구분

```
{  
  "id" : id,  
  "pw" : pw  
}
```

예 1

```
{  
  "conversation":{ "decimal":"103",  
    "hyper":"&0x67"  
  }  
}
```

예 2

JSON 데이터 형식

▶ 배열 구조

- 순서가 있는 데이터들의 목록
- [(left bracket) 으로 시작해서](right bracket)으로 표현
- 각 데이터들은 , 표기로 구분

예 1

```
var arrayName=[id, pw]
```

예 2

```
[  
    [0, -1, 0],  
    [1, 0, 0],  
    [0, 0, 1]  
]
```

JSON 데이터 사용

▶ eval() 함수

- JSON 형식의 문자열 데이터를 자바스크립트 객체로 변환
- 이 함수 생략시 접근자(.) 및 []로 특정 데이터 access 불가
- 문법

```
eval( "(" + JSON형식의 데이터 + ")" );
```

JSON 데이터 사용

- ▶ [] 브래킷 연산자 와 . 접근자를 활용하여 JSON 데이터 access
 - [] 연산자
 - ▶ 예 : 변수['key']
 - . 접근자
 - ▶ 예 : 변수.key

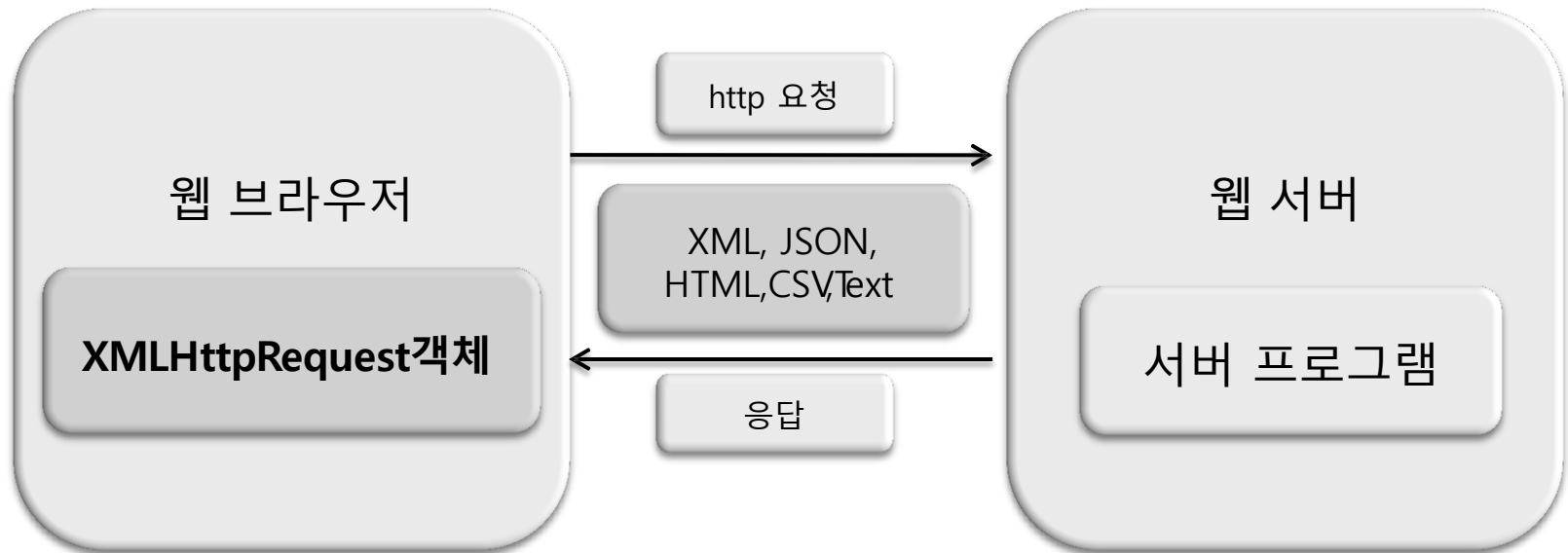
```
var json = eval({num:'111', name:'tom', tel:'12345'});  
var num = json['name']; //tom  
Var tel = json.tel;//12345
```

4-2. Ajax Programming

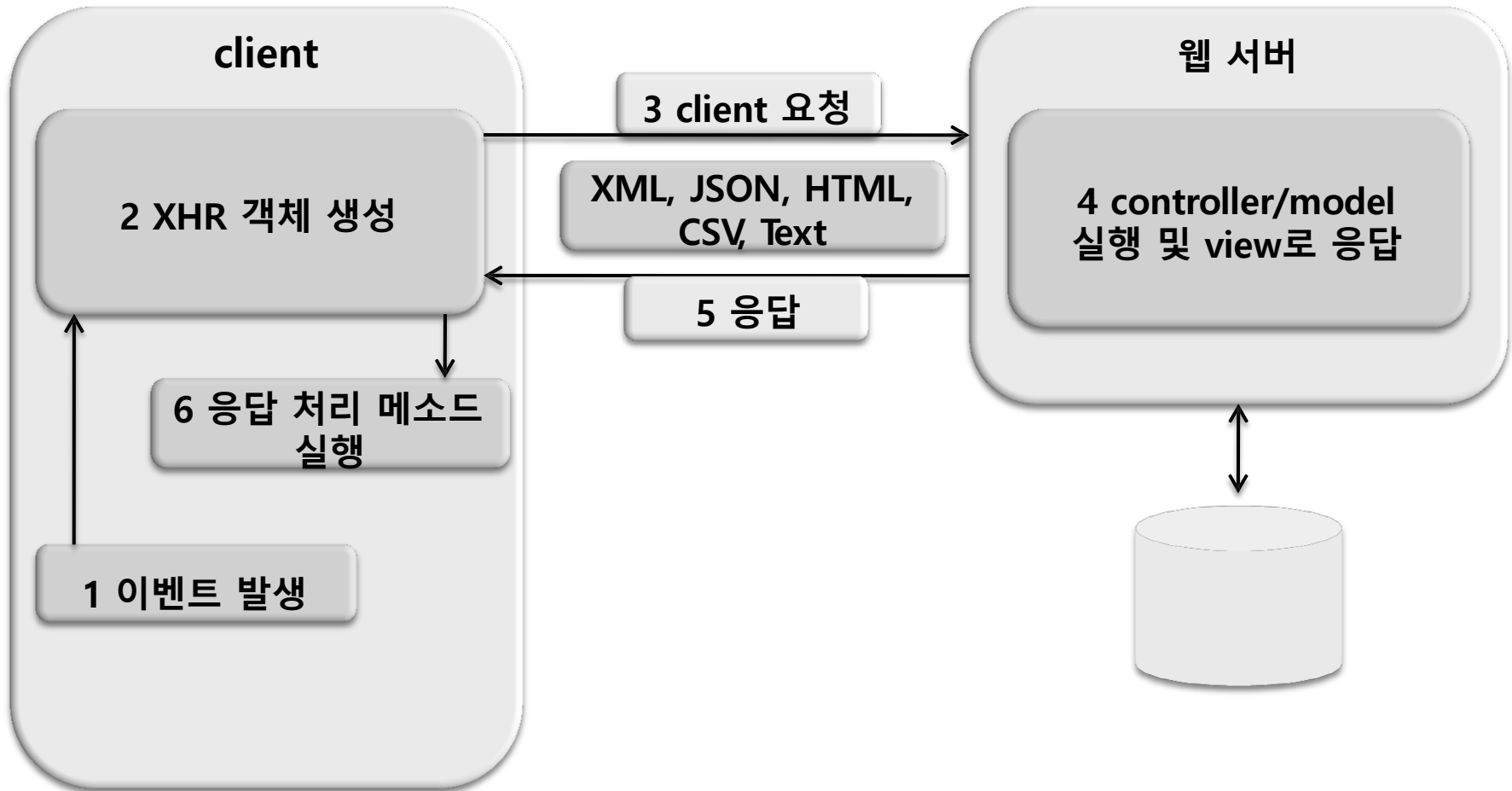
- Ajax 통신 원리
- Ajax 실행 process
- XHR 객체의 개요
- XHR 객체의 속성
- XHR 객체의 메소드

Ajax 통신 원리

- ▶ 폼 전송 방식이 아닌 자바스크립트 객체 (XMLHttpRequest)를 이용한 요청과 응답 처리
- ▶ 웹 브라우저가 비동기 요청 받으면 콜백 함수 실행



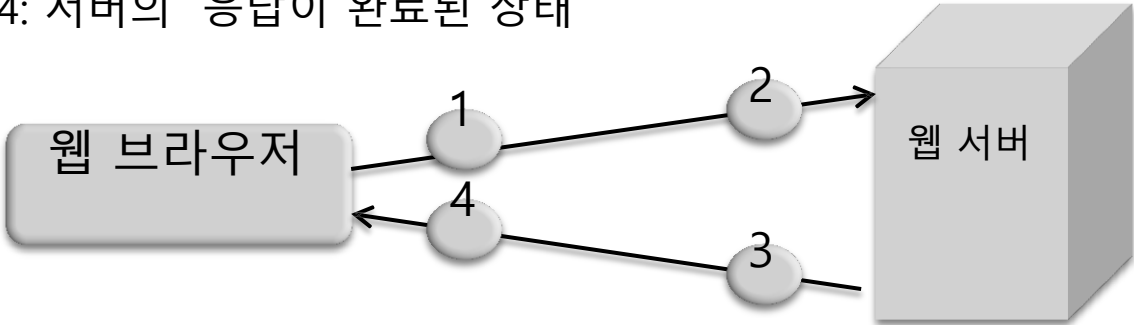
Ajax 실행 process



XMLHttpRequest(XHR) 객체의 개요

- ▶ Ajax에서 서버와 클라이언트 간에 통신을 담당하는 주요 객체
- ▶ 요청과 응답을 처리하는 필수 객체
- ▶ 전체 페이지의 새로 고침 없이 서버와의 통신을 가능하게 함

XHR 객체의 속성(1/3)

속성	설명	읽기/쓰기
readyState	<p>AJAX 객체의 요청, 응답의 상태를 나타내는 숫자</p> <p>-로딩중, 처리중, 처리완료등의 상태 표현</p> <p>0: request 가 초기화 되지 않음(open() 호출 전)</p> <p>1: request 가 셋업되고 보내지 않음(send() 호출 전)</p> <p>2: request 가 보내지고, 진행 중 일 경우(send() 호출후)</p> <p>3: response 에서 부분적인 데이터를 받으며, 완료가 아직 안 된 경우</p> <p>4: 서버의 응답이 완료된 상태</p>  <pre> graph LR Browser[웹 브라우저] -- 1 --> Server[웹 서버] Server -- 2 --> Browser Browser -- 3 --> Server Server -- 4 --> Browser </pre>	읽기 전용

XHR 객체의 속성(2/3)

속성	설명	읽기/쓰기
status	서버로부터 응답된 HTTP 상태 코드를 나타내는 숫자 200 : 정상적으로 응답을 받은 경우 404 : 페이지를 찾지 못한 경우 405 = 요청 방식 다름 500 = 서버 오류 발생	읽기 전용
onreadystatechange	요청에 대한 응답을 받을때마다(readyState 속성이 바뀔때마다) 호출됨 콜백 함수 지정 속성	읽기/쓰기

XHR 객체의 속성(3/3)

속성	설명	읽기/쓰기
statusText	서버로부터 받은 응답의 상태를 나타내는 문자열 정상적으로 응답을 받으면 'OK' 파일을 찾지 못하면 'Not Found'	읽기 전용
responseText	서버로부터 응답된 문자열 데이터를 보유	읽기 전용
responseXML	서버로부터 응답된 xml 데이터를 보유 DOM 객체로 파싱할 수 있음	읽기 전용

XHR 객체의 메소드(1/2)

메소드	설명
<code>void open(String method, String url, [boolean async])</code>	<p>AJAX 요청을 초기화하면서 요청 방식, 주소, 동기화 여부를 지정</p> <p>[매개변수]</p> <ol style="list-style-type: none">1. method<ul style="list-style-type: none">- http 요청 방식을 나타내며 "get" 또는 "post" 방식을 사용2. url<ul style="list-style-type: none">- 요청할 페이지의 주소를 지정3. aysnc<ul style="list-style-type: none">- 비동기 통신 여부를 나타내며 true 또는 false- default값 true

XHR 객체의 메소드(2/2)

메소드	설명
<code>void send(Object content)</code>	사용자의 요청을 서버에 보냄 인자에는 요청과 함께 서버로 보낼 내용을 지정 GET방식 = null POST방식 = 쿼리스트링
<code>String getAllResponseHeaders()</code>	응답을 받은 모든 헤더 정보를 문자열로 반환
<code>String getResponseHeader (String header)</code>	응답을 받은 경우 header 인자로 지정한 이름의 헤더 정보 값을 문자열로 반환
<code>void setResonseHeader (String header, Sring value)</code>	요청을 보내기 전에 header 헤더 정보의 값을 value로 설정
<code>void abort()</code>	<code>send()</code> 메소드로 보낸 현재의 요청을 중단

5. CSS

- CSS 목적
- CSS 규칙
- CSS 속성

CSS의 목적

- HTML에 스타일을 부여
- 문서의 외관을 보다 효과적으로 제어

- 1996년 12월 W3C에서 CSS1 발표
- 1998년 5월 CSS2 발표
- 하나의 문서에 여러 개의 스타일시트 지정 가능
- 스타일시트는 역순으로 적용

CSS 규칙

```
선택자 {속성: 값}  
h1 { color: blue }
```

- 선택자와 선언부로 구성
- 선택자(selector, tag) : 'h1'
- 모든 HTML tag가 선택자가 될 수 있음
-
- 선언부(declaration) : 'color: blue'
- 속성(property) : 'color'
- 값(value) : 'blue'

HTML 문서에서 CSS 연결 방법

1. `<link>` 요소 사용하여 외부 스타일시트 연결
2. `<style>` 요소와 `@import` 사용하여 외부 스타일시트 연결
3. `<style>` 요소를 사용하여 직접 스타일 지정(내부 스타일 시트)
4. `<body>` 요소의 하위 요소에 `'style'` 속성을 사용하여 지정(inline)

CSS 연결 예

```
<html>
  <head>
    <title> css 연결 </title>
    <link rel=stylesheet type="text/css" href="samp.css">
    <style type="text/css">
      @import url("basic.css");
      h3 { color: blue }
    </style>
  </head>
  <body>
    <h1> h1은 빨간색입니다. </h1>
    <h2> h2는 노란색입니다. </h2>
    <h3> h3는 파란색입니다. </h3>
    <p style="color: green"> 그러나 이 문장은 녹색입니다. </p>
  </body>
</html>
```

선택자의 그룹화

- 선택자의 요소들을 콤마(,)를 사용하여 그룹화

```
h1, h2, h3 { font-family: 굴림 }
```

선언부의 그룹화

- 선언부의 요소들을 세미콜론(;)을 사용하여 그룹화

```
h1 {  
  font-weight: bold;  
  font-size: 12pt;  
  line-height: 14pt;  
  font-family: helvetica;  
  font-variant: normal;  
  font-style: normal  
}
```


<div> 태그와 태그

- <div> 태그
태그가 끝난 후 줄 바꿈을 하는 태그
- 태그
태그가 끝나도 줄 바꿈을 하지 않는 태그

개별 요소에 대한 스타일 지정

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
<title>Insert title here</title>
</head>
<body>
  <div style="font-size: 20pt; font-weight: bold"> 이것은 크고 굵은 글
    씨가 나타나는 div입니다. </div>
  <p> 이것은 일반적인 문장인데, <span style="font-size: 24pt;
    font-style: italic"> span </span> 요소는 가운데 강조되게 됩니다.
  </p>
</body>
</html>
```

class 속성

```
<html>
  <head>
    <title> class 속성 설명 </title>
    <style type="text/css">
      h1.italic { font-style: italic }
    </style>
  </head>
  <body>
    <h1> 보통 머리글 </h1>
    <h1 class="italic"> 이탤릭체 머리글 </h1>
  </body>
</html>
```

- 모든 요소에서 CLASS 속성 사용시

```
.italic { font-style: italic }
```

id 속성

```
<html>
  <head>
    <title> id 속성 설명 </title>
    <style type="text/css">
      #space { letter-spacing: 0.3em }
      h1#space { letter-spacing: 2.0em }
    </style>
  </head>
  <body>
    <h1> 보통 머리글 </h1>
    <h1 id="space"> id가 적용된 머리글</h1>
    <p> 보통 문장 </p>
    <p id="space"> id가 적용된 문장 </p>
  </body>
</html>
```

class와 id 속성 사용 예

```
<html>
  <head>
    <title> class와 id를 사용한 스타일 지정 </title>
    <style type="text/css">
      div.bold { font-size: 20pt; font-weight: bold }
      div.italic { font-size: 20pt; font-style: italic }
      span.bold { font-weight: bold }
      span#pastoral {color: green; font-size: 50pt; font-style: italic }
    </style>
  </head>
  <body>
    <div class="bold"> 이것은 크고 굵은 글씨체의 div입니다. </div>
    <div class="italic"> 이것은 크고 이탤릭체의 div입니다.</div>
    <p> 이것은 일반적인 문장인데, <span id="pastoral"> span </span>
    요소때문에 가운데가 <span class="bold">강조</span>되게 됩니다. </p>
  </body>
</html>
```

문맥 선택자

- 상위 요소의 스타일을 계승을 통해 하위 요소에 적용시키고, 스타일시트에는 예외로 발생하는 스타일만 지정

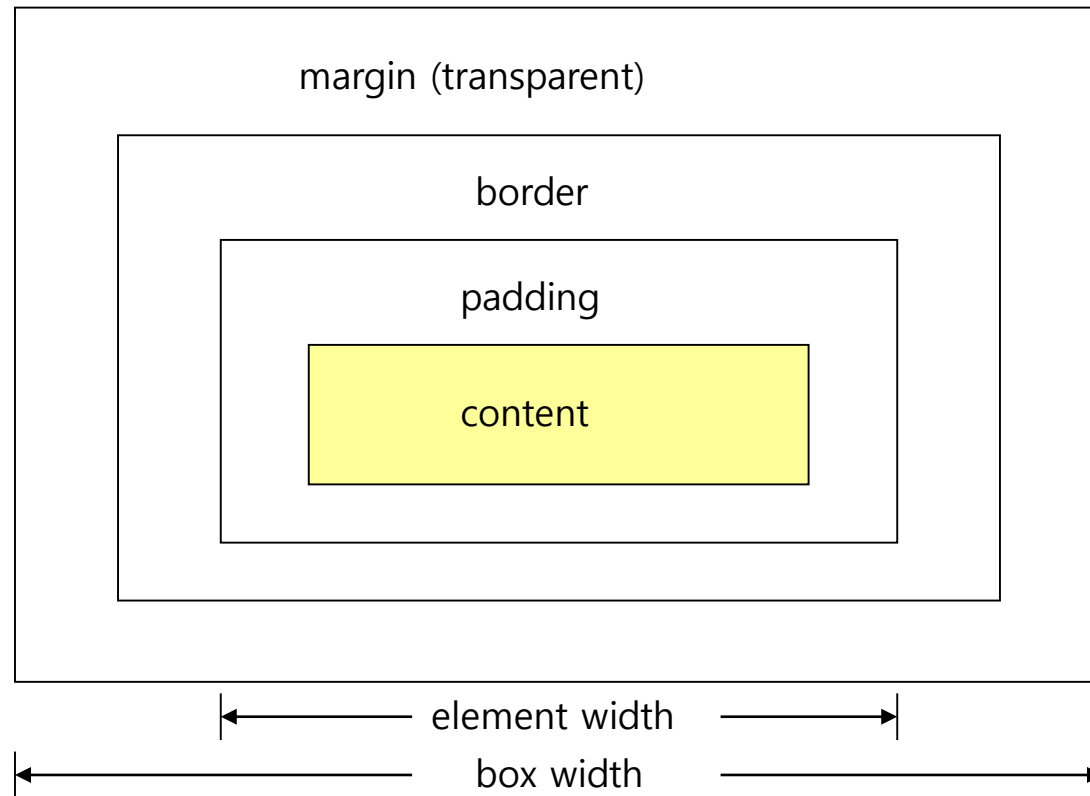
```
h1 { color: blue }  
em { color: red }
```



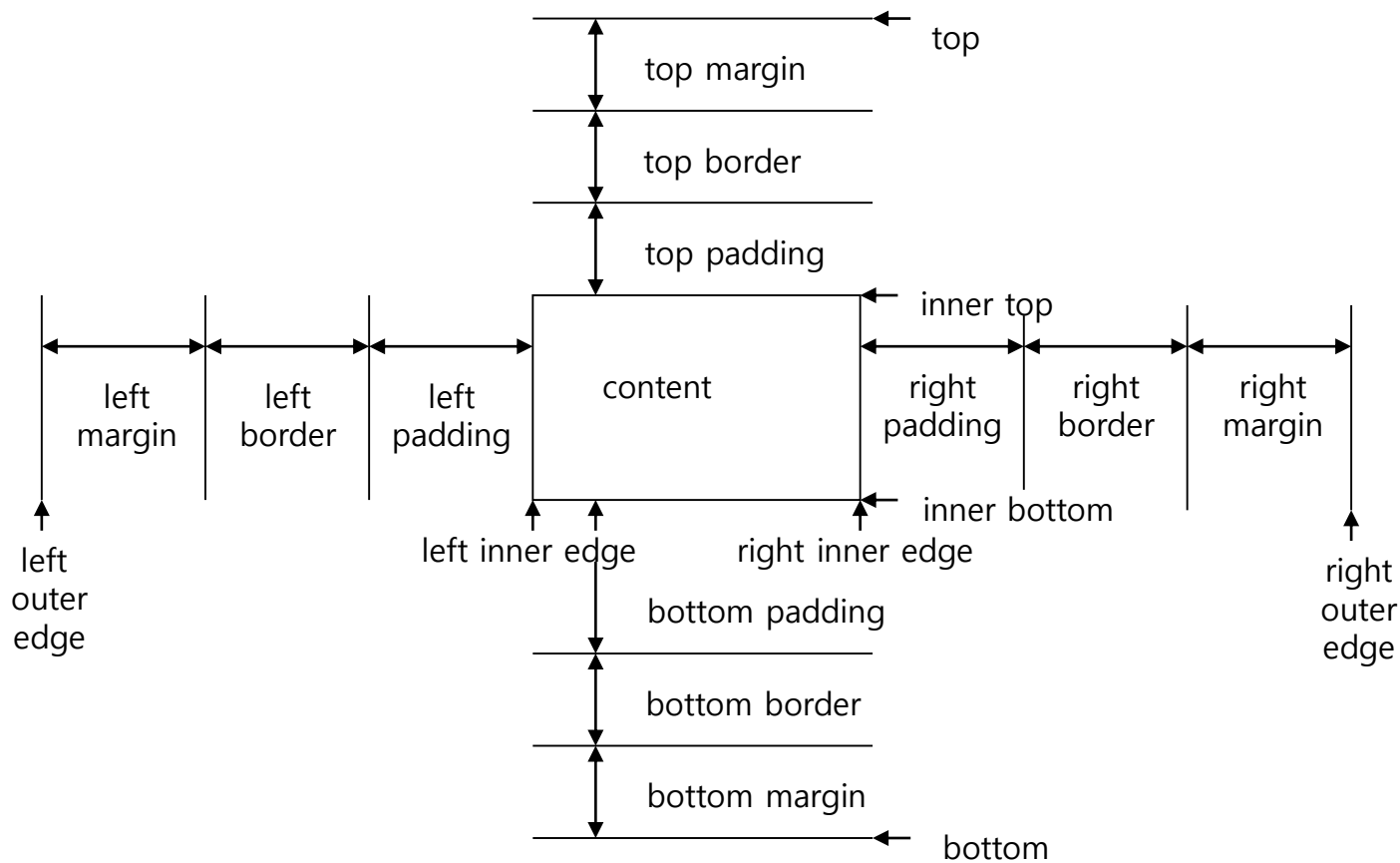
h1 요소의 하위로 나오는 em 요소
의 글자색만 변경

```
h1 { color: blue }  
h1 em { color: red }
```

CSS의 포매팅 모델(box model)



포매팅 지정에 사용되는 용어



CSS에서 길이의 단위

- 상대적인 단위
 - em : 현재 사용중인 폰트 크기를 1로 함
 - ex : 문자 'x'의 크기를 1로 함
 - px : 모니터의 화소(pixel)에 대한 상대적인 크기
- 절대적인 단위
 - in : inches, 1in = 2.54cm
 - cm : centimeters
 - mm : millimeters
 - pt : points, 1pt = 1/72 in
 - pc : picas, 1pc = 12pt

CSS에서 여백 설정

```
<html>
  <head>
    <title> 여백 설정 </title>
    <style type="text/css">
      h1 {
        margin-top: 2em;
        margin-right: 2em;
        margin-bottom: 3px;
        margin-left: 12.3%;
      }
    </style>
  </head>
  <body>
    <h1> 여백을 설정하는 예입니다. </h1>
  </body>
</html>
```

CSS에서 패딩 설정

```
<html> <head> <title> 패딩 설정 </title>
  <style type="text/css">
    body {
      padding-top: 0.3em;
      padding-right: 10px;
      padding-bottom: 2em;
      padding-left: 20%;
    }
  </style>
</head>
<body>
  <p><em> 계단형 스타일시트</em>는 html을 보다 구조적으로 만들기 위해
  제정되었다.</p>1996년 늦게 w3c는 css 제1수준(level 1) 명세를 내 놓았는데,
  마이크로 소프트는 익스플로러 3.0에 몇 가지 css의 기능을 구현하고, 4.0에
  기능을 보강하였다.
</body>
</html>
```

여러 종류의 경계선 지정

```
<html>
  <head>
    <title> 경계선 설정 </title>
  </head>
  <body>
    <h1 style="border-style: dotted"> dotted 경계선 </h1>
    <h1 style="border-style: dashed"> dashed 경계선 </h1>
    <h1 style="border-style: solid"> solid 경계선 </h1>
    <h1 style="border-style: double"> double 경계선 </h1>
    <h1 style="border-style: groove"> groove 경계선 </h1>
    <h1 style="border-style: ridge"> ridge 경계선 </h1>
    <h1 style="border-style: inset"> inset 경계선 </h1>
    <h1 style="border-style: outset"> outset 경계선 </h1>
  </body>
</html>
```

텍스트, 배경, 경계선의 색 지정

```
<html> <head>
  <title> 경계선 설정 </title>
  <style type="text/css">
    p {
      color: white;
      background: green;
      border-style: solid;
      border-color: red;
    }
  </style>
</head>
<body>
  <h1> css 소개 </h1>
  <p> 1996년 늦게 w3c는 css 제1수준(level 1) 명세를 내 놓았는데,
  마이크로 소프트는 익스플로러 3.0에 몇 가지 css의 기능을 구현하고, 4.0에
  기능을 보강하였다. </p>
</body>
</html>
```

문단 속성에 대한 스타일 지정

```
<html>
  <head>
    <title> 문단 속성에 대한 스타일 지정 </title>
  </head>
  <body>
    <p> 이것은 보통 글자 스타일입니다.</p>
    <p style="letter-spacing:10px">
      글자의 간격이 넓어졌습니다.</p>
    <p style="text-decoration: underline">
      글자에 밑줄이 들어갔군요.</p>
    <p style="text-decoration: line-through">
      글자에 줄이 들어갔습니다.</p>
    <p style="text-transform: capitalize">
      this line is something difference.</p>
    <p style="text-transform: uppercase">
      this line is something difference too.</p>
  </body>
</html>
```

6. JQuery

- JQuery 개요
- JQuery Syntax
- JQuery Selectors
- JQuery Events

JQuery 개요

- JavaScript Library
- JavaScript를 이용한 웹 페이지 개발을 보다 더 쉽게 도와주는 framework
- 편리한 기능의 함수가 제공됨
- 다양한 플러그인 제공

JQuery 개요-CDN

Google CDN:

```
<head>  
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>  
</head>
```

Microsoft CDN:

```
<head>  
<script src="http://ajax.aspnetcdn.com/ajax/jquery/jquery-1.11.2.min.js"></script>  
</head>
```

JQuery Syntax

`$(selector).action()`

- \$: jquery sign
- select: html elements
- action(): element에 대해 실행할 동작
- `$("p").hide();`

JQuery selector

- HTML element를 선택하고 조작하기 위해 필요
 1. Element selector
 2. #id selector
 3. .class selector

Examples of selector

<code>\$("p")</code>	All <code><p></code> elements
<code>\$("h1,div,p")</code>	All <code><h1></code> , <code><div></code> and <code><p></code> elements
<code>\$("*")</code>	All elements
<code>\$("#lastname")</code>	The element with <code>id="lastname"</code>
<code>\$(".intro")</code>	All elements with <code>class="intro"</code>
<code>\$(".intro,.demo")</code>	All elements with the class <code>"intro"</code> or <code>"demo"</code>

Examples of selector

<code>\$("p:first")</code>	The first <code><p></code> element
<code>\$("p:last")</code>	The last <code><p></code> element
<code> \$("tr:even")</code>	All even <code><tr></code> elements
<code> \$("tr:odd")</code>	All odd <code><tr></code> elements
<code> \$("p:first-child")</code>	All <code><p></code> elements that are the first child of their parent
<code> \$("p:first-of-type")</code>	All <code><p></code> elements that are the first <code><p></code> element of their parent
<code> \$("p:last-child")</code>	All <code><p></code> elements that are the last child of their parent
<code> \$("p:last-of-type")</code>	All <code><p></code> elements that are the last <code><p></code> element of their parent
<code> \$("p:nth-child(2))"</code>	All <code><p></code> elements that are the 2nd child of their parent

Examples of selectors

<u>[attribute]</u>	<code>\$("[href]")</code>	All elements with a href attribute
<u>[attribute=value]</u>	<code>\$("[href='default.htm']")</code>	All elements with a href attribute value equal to "default.htm"
<u>:text</u>	<code>\$(":text")</code>	All input elements with type="text"
<u>:password</u>	<code>\$(":password")</code>	All input elements with type="password"
<u>:radio</u>	<code>\$(":radio")</code>	All input elements with type="radio"
<u>:checkbox</u>	<code>\$(":checkbox")</code>	All input elements with type="checkbox"
<u>:submit</u>	<code>\$(":submit")</code>	All input elements with type="submit"
<u>:reset</u>	<code>\$(":reset")</code>	All input elements with type="reset"
<u>:button</u>	<code>\$(":button")</code>	All input elements with type="button"
<u>:image</u>	<code>\$(":image")</code>	All input elements with type="image"
<u>:file</u>	<code>\$(":file")</code>	All input elements with type="file"

JQuery Event

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

JQuery Event - Document ready event

HTML 문서의 로딩이 끝난 후 jquery가 실행되도록 해주는 이벤트.

(HTML문서 로딩 전 jquery가 실행되는 것을 막아 줌)

```
$(document).ready(function(){  
  
    // jQuery methods go here...  
  
});
```