5. DOM 과 XML

DOM,즉 Document Object Model은 문서를 객체로 표현하기위한표준으로서
HTML 이나 XML등의 문서를 객체로 표현할 때 사용되는API이다.
자바스크립트 ,자바,C,C# 등다양한 언어에서 DOM API를 제공하고있다.
DOM은 문서를 트리구조로 표현하기 때문에 쉽게이해할수있다.
XMLHttpRequest 객체는 응답텍스트대신 XML 응답결과를 사용할수있는데,이때
DOM API를 사용해서 서버가생성한 XML로부터 데이터를 추출할수있다.

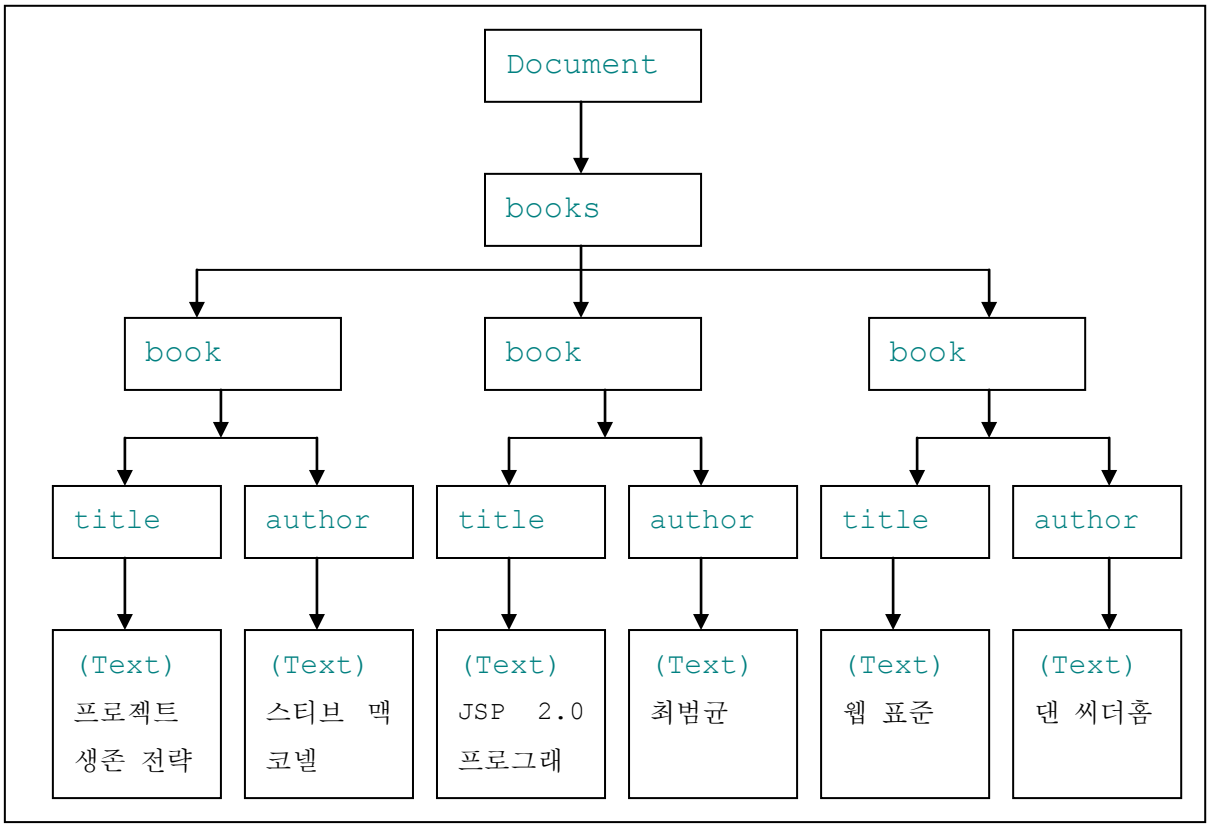
5-1.XML 문서와 DOM 트리구조

books.xml

```
<?xml version="1.0" encoding="euc-kr" ?>

<books>
      <book>
       <title>프로젝트 생존 전략</title>
       <author>스티브 맥코넬</author>
      </book>
      <book>
       <title>JSP 2.0 프로그래밍</title>
       <author>최범균</author>
      </book>
      <book>
       <title>웹 표준</title>
       <author>댄 씨더홈</author>
      </book>
</books>
```

books.xml 을 DOM 트리로 표현

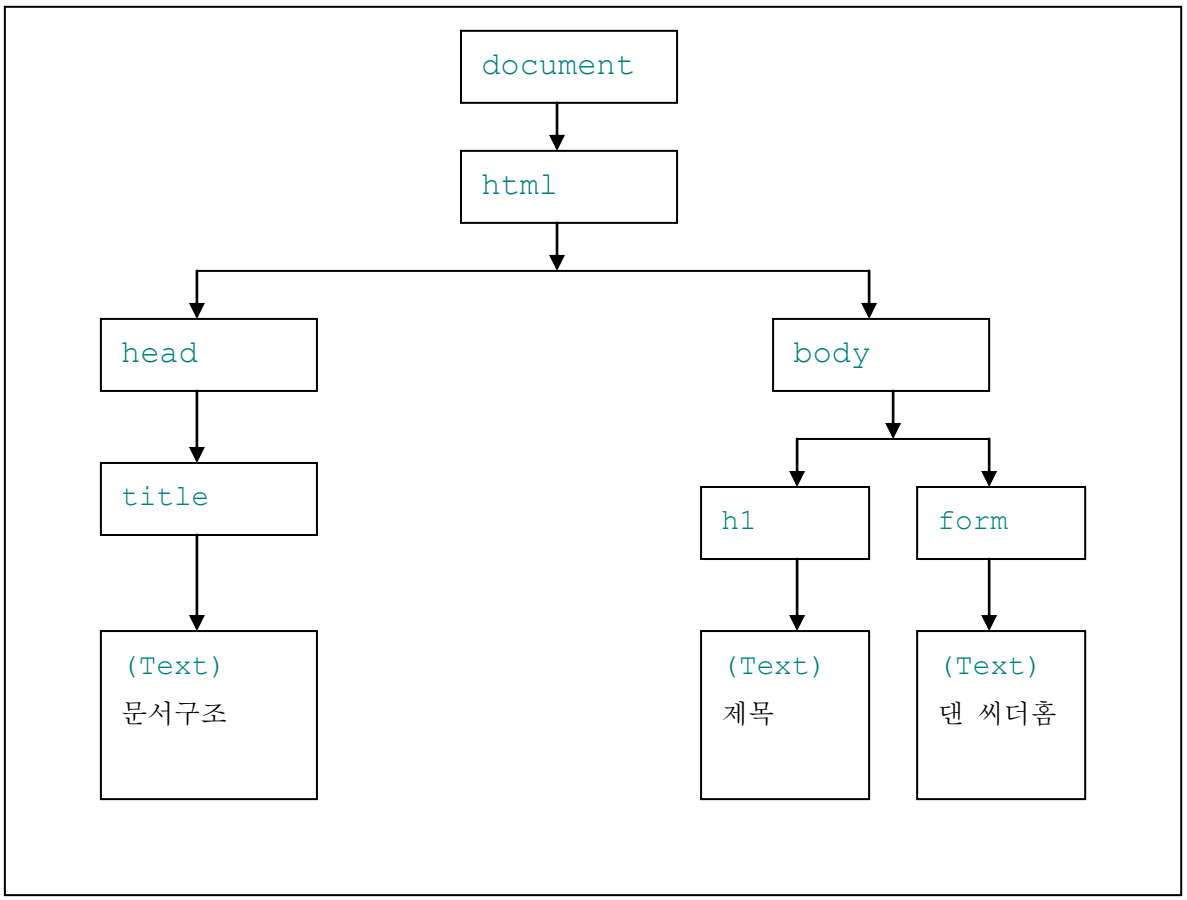


5-2.HTML 문서와 DOM 트리구조

simple.html

```
<html>
<head>
	<title>문서구조</title>
</head>
<body>
	<h1>제목</h1>
	<form> 댄 씨더홈 </form>
</body>
</html>
```

simple.html을 DOM 트리로 표현



5-3.DOM API를 이용해서 HTML 문서의정보를 탐색하는예제

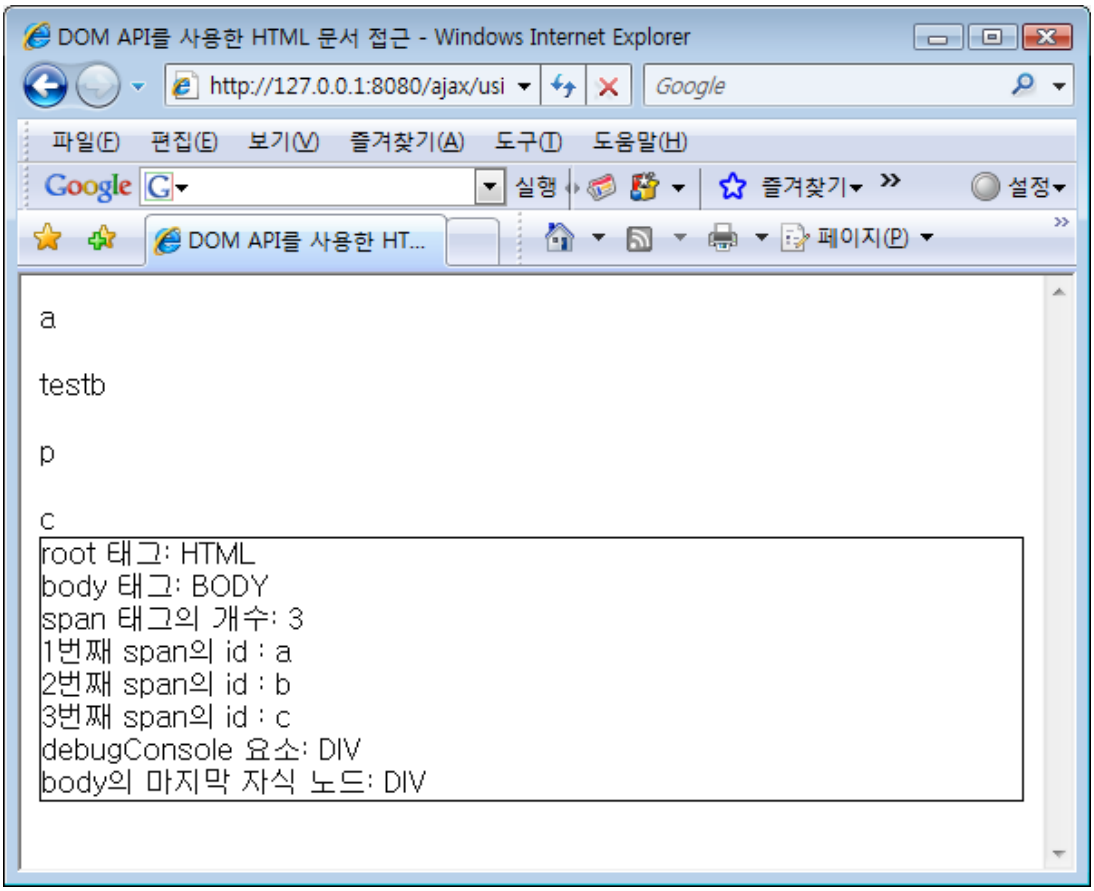
usingDOM.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko" lang="ko">
<head><meta http-equiv="content-type" content="text/html;charset=euc-kr" />
        <title>DOM API를 사용한 HTML 문서 접근</title>
        <script type="text/javascript" src="js/log.js"></script>
        <script type="text/javascript">
        window.onload = function() {
                var roodNode = document.documentElement;
                log("root 태그: "+roodNode.tagName);
                var bodyNode =
document.getElementsByTagName("body").item(0);
                log("body 태그: "+bodyNode.tagName);
                var spanList = document.getElementsByTagName("span");
                log("span 태그의 개수: "+spanList.length);
                for (var i = 0 ; i < spanList.length ; i++) {
                        var span = spanList.item(i);
                        log((i+1)+"번째 span의 id :
"+span.getAttribute("id"));
                }
                var debugConsoleDiv =
document.getElementById("debugConsole");
                log("debugConsole 요소: "+debugConsoleDiv.tagName);
                var bodyLastChild = bodyNode.lastChild;
                log("body의 마지막 자식 노드: "+bodyLastChild.nodeName);
        }
        </script>
</head>
<body>
        <SPAN id="a">a</SPAN>
        <p>test<span id="b">b</span></p>
        <div><p>p</p><span id="c">c</span></div>
        <div id="debugConsole" style="border: 1px solid #000"></div>
</body>
</html>
```

log.js

```
function log(msg) {
	var console = document.getElementById("debugConsole");
	if (console != null) {
		console.innerHTML += msg +"<br/>";
	}
}
```

결과

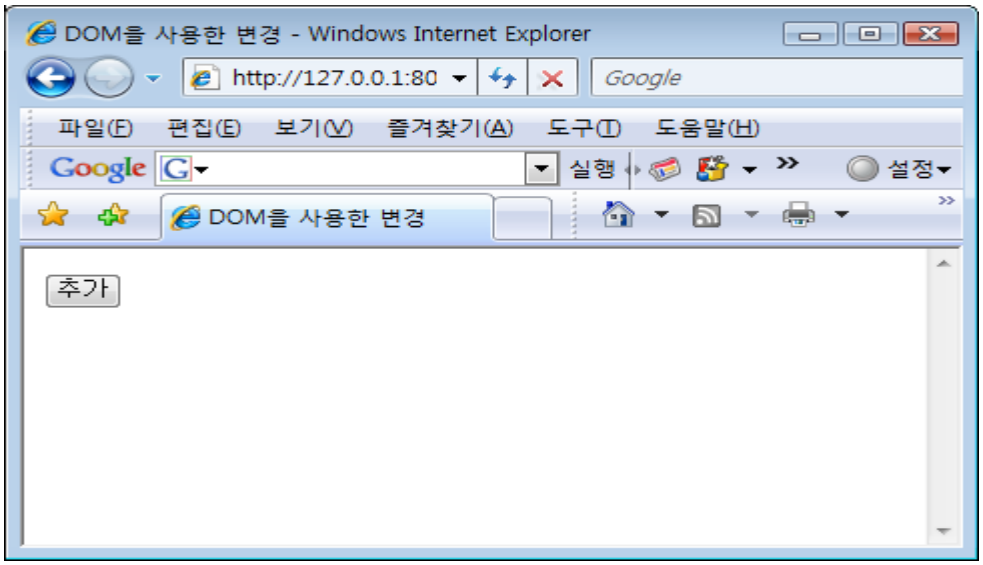


5-4.DOM API를 이용해서 HTML 화면변경 하는예제

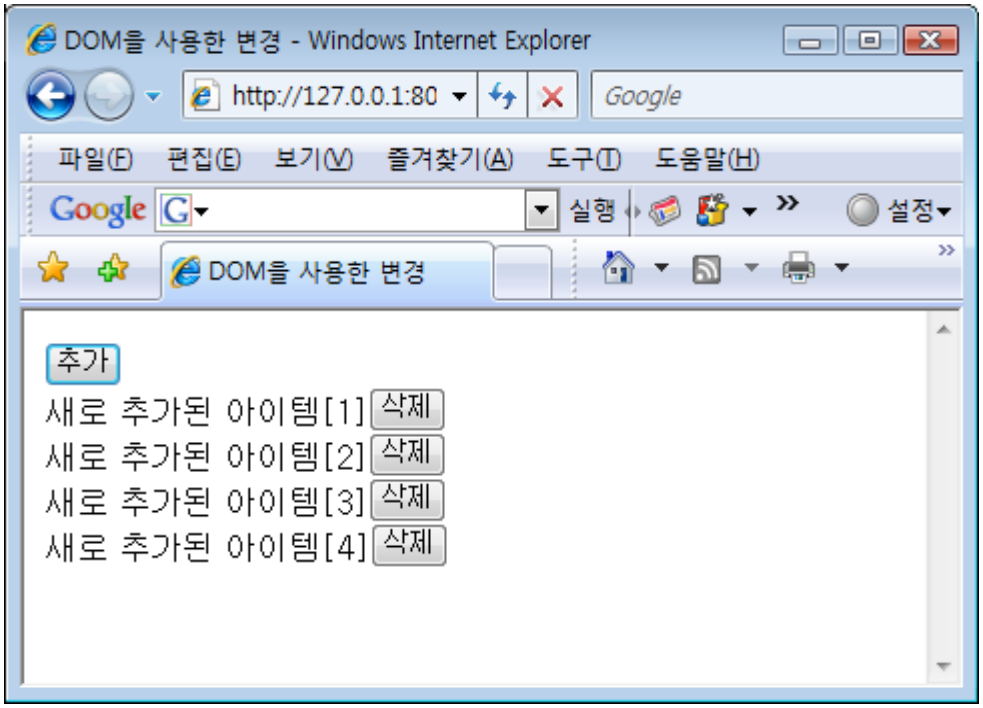
changeUsingDom.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko" lang="ko">
<head> <meta http-equiv="content-type" content="text/html;
charset=euc-kr" />
       <title>DOM을 사용한 변경</title>
       <script type="text/javascript">
       var count = 0;
       function appendItem() {
              count++;
              var newItem = document.createElement("div");
              newItem.setAttribute("id", "item_" + count);
              var html = '새로 추가된 아이템['+count+']'+
              '<input type="button" value="삭제"
onclick="removeItem(' + count + ')"/>';
              newItem.innerHTML = html;
              var itemListNode =
document.getElementById('itemList');
              itemListNode.appendChild(newItem);
       }
       function removeItem(idCount) {
              var item = document.getElementById("item_"+idCount);
              if (item != null) {
                     item.parentNode.removeChild(item);
              }
       }
       </script>
</head>
<body>
<input type='button' value='추가' onclick='appendItem()' />
<div id="itemList"></div>
</body>
</html>
```

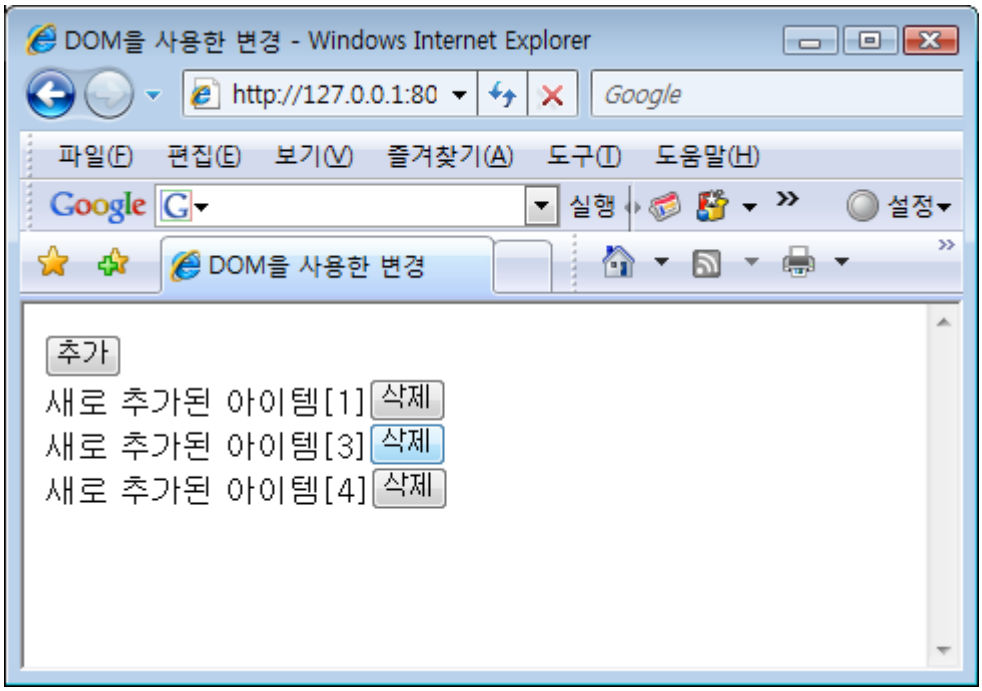
결과


DOM을 사용한 변경 - Windows Internet Explorer
http://127.0.0.1:80
Google
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
Google
실행
설정
DOM을 사용한 변경
추가

추가


DOM을 사용한 변경 - Windows Internet Explorer
http://127.0.0.1:80
Google
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
Google
실행
설정
DOM을 사용한 변경
추가
새로 추가된 아이템[1] 삭제
새로 추가된 아이템[2] 삭제
새로 추가된 아이템[3] 삭제
새로 추가된 아이템[4] 삭제

삭제


DOM을 사용한 변경 - Windows Internet Explorer
http://127.0.0.1:80
Google
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
Google
실행
설정
DOM을 사용한 변경
추가
새로 추가된 아이템[1] 삭제
새로 추가된 아이템[3] 삭제
새로 추가된 아이템[4] 삭제

5-7.XML 응답사용하기

XMLHttpRequest 객체는서버로부터 XML 문서를 응답으로 읽어올수있으며 서버에서 읽어온 XML 문서는 DOM 트리로 변환되어서 저장된다. DOM API를 사용하면 서버가생성한 XML 문서로부터 원하는 정보를 읽어올수있다.

① 서버에서 XML 응답 생성하기

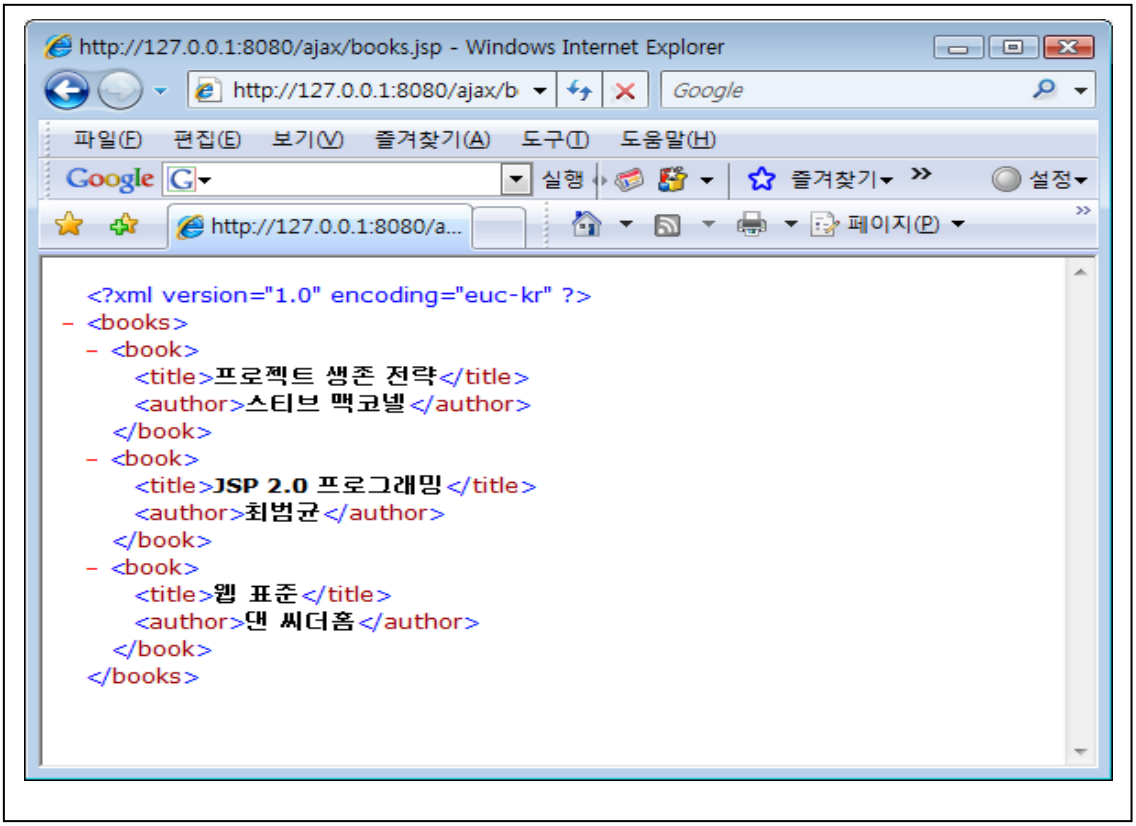
XMLHttpRequest 객체가 XML 문서를 응답으로 받을려면 서버에서 응답데이타를 XML 으로 생성해야할것이다.

books.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<books>
    <book>
        <title>프로젝트 생존 전략</title>
        <author>스티브 맥코넬</author>
    </book>
    <book>
        <title>JSP 2.0 프로그래밍</title>
        <author>최범균</author>
    </book>
    <book>
        <title>웹 표준</title>
        <author>댄 씨더홈</author>
    </book>
</books>
```

서버의 XML 응답결과(books.xml)

웹브라우져가 응답문서를 XML 로인식해서 출력하는 것을 알수있다.



② responseXML로 응답읽어오기

XMLHttpRequest 객체는 XML 문서를 응답데이타로 사용할수있다.

XML 응답데이타를 사용할때에는 responseXML 프로퍼티를 이용하면 된다.

```
sendRequest("books.xml", null, loadedBooks, "GET");

function loadedBooks() {
      if (httpRequest.readyState == 4) {
            if (httpRequest.status == 200) {
                  var xmlDoc = httpRequest.responseXML;
                        .. ..
            }
      }
}
```

viewBooks.html(books.xml을 alert 경고창에 띄우기)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko" lang="ko">
<head><meta http-equiv="content-type" content="text/html;charset=euc-kr" />
        <title>책정보 보기</title>
<script type="text/javascript" src="js/HttpRequest.js"></script>
<script type="text/javascript">
function loadBooks() {
        sendRequest("books.xml", null, loadedBooks, "GET");
}
function loadedBooks() {
        if (httpRequest.readyState == 4) {
                if (httpRequest.status == 200) {
                        var xmlDoc = httpRequest.responseXML;
                        var bookList = xmlDoc.getElementsByTagName("book");
                        var message = "책 개수 : "+bookList.length+"권\n";
                        for (var i = 0 ; i < bookList.length ; i++) {
                                var book = bookList.item(i);
var titleValue = book.getElementsByTagName("title").item(0)
                           .firstChild.nodeValue;
var authorValue = book.getElementsByTagName("author").item(0)
                                                        .firstChild.nodeValue;
                        message += titleValue + "(" + authorValue + ")\n";
                                }
                        alert(message);
                }
        }
}
window.onload = function() {
        loadBooks();
}
</script></head>
<body>책 정보를 alert 으로 출력</body>
</html>
```

5-7.XSLT를 사용하여 XML을HTML로 변환하기

XSLT는 XML 문서를 XSL을 사용해서 원하는문서로 변환시킬때사용한다.

books.xsl

```
<?xml version="1.0" encoding="utf-8" ?>
  <xsl:stylesheet
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
       <xsl:output method = "html" indent="yes" encoding="utf-8" />
       <xsl:template match="books"><ul>
              <xsl:for-each select="book">
              <li><b><xsl:value-of select="title" /></b>
              (<i><xsl:value-of select="author" /></i>)
       </li>
              </xsl:for-each>     </ul>
       </xsl:template>
</xsl:stylesheet>
```

books.xml

```
<?xml version="1.0" encoding="euc-kr" ?>
<books>
    <book>
       <title>프로젝트 생존 전략</title>
       <author>스티브 맥코넬</author>
    </book>
    <book>
       <title>JSP 2.0 프로그래밍</title>
       <author>최범균</author>
    </book>
    <book>
       <title>웹 표준</title>
       <author>댄 씨더홈</author>
    </book>
</books>
```

bookList.html (books.xml을 HTML 웹으로 띄우기)

```
<html><head><title>책 목록</title>
	<script type="text/javascript" src="js/HttpRequest.js"></script>
	<script type="text/javascript">
	var xmlDoc = null;var xslDoc = null;
	function loadBooks() {
		sendRequest("books.xml", null, loadedBooksXML, "GET");
	}
	function loadedBooksXML() {
		if (httpRequest.readyState == 4) {
			if (httpRequest.status == 200) {
				xmlDoc = httpRequest.responseXML;
	sendRequest("books.xsl", null, loadedBooksXSL, "GET");}}}
	function loadedBooksXSL() {
		if (httpRequest.readyState == 4) {
			if (httpRequest.status == 200) {
				xslDoc = httpRequest.responseXML;
				doXSLT();}}}
	function doXSLT() {
		if (xmlDoc == null || xslDoc == null) return;
		var bookList = document.getElementById("bookList");
		if (window.ActiveXObject) {
			bookList.innerHTML = xmlDoc.transformNode(xslDoc);
		} else {
			var xsltProc = new XSLTProcessor();
			xsltProc.importStylesheet(xslDoc);
		var fragment = xsltProc.transformToFragment(xmlDoc, document);
			bookList.appendChild(fragment);
		}}
	window.onload = function() {
		loadBooks();
	}
	</script></head><body><h1>신규 책 목록</h1>
<div id="bookList"></div>
</body>
</html>
```

6 . 자바스트립트객체

자바스크립트는 개체지향 프로그램을 완벽하게 지원하고있지는 않지만
자바스크립트가 지원하는 몇가지 기능만으로 충분하게 객체지향적으로
프로그래밍을 할수있다.

6-1. prototype을 이용한 자바스크립트 클래스만들기

자바스크립트는 Date,RegEx,String 등의 클래스가 있고 이들클래스를
사용하여 새로운객체를 생성할수있다.

ex>

```
var now = new Date();
var year=now.getFullYear();
```

자바 스크립트가 제공하는 기본적인 클래스외에 추가적으로 개발자가 직접새로운
클래스를 정의 할수있으며 ,이를 통해 함수기반이아닌 객체기반으로 자바스크립트
프로그래밍이 가능하다

①새로운 클래스정의

```
클래스 이름 =  function (파라메타){
.. ..
}
```

②객체생성시 id,name 파라메타 값을 전달받는 Member 라는 클래스정의

정의

```
Member  =  function (id,name){
        this.id=id;
        this.name=name;
}
```

생성

```
var mem=new Member('a001','천송이');
```

③ 클래스내에 함수 정의

```
# 함수정의
클래스이름.prototype.함수이름 = function(파라메타){
        …
}

#.정의예 Member 클래스에 setValue라는함수를 추가.
Member.prototype.setValue=function(newId,newname){
        this.id= newId;
        this.name= newname;
}
#.호출

var mem=new Member("a001","천송이");
mem.setValue('a002',"전지현");
```

member.js

```
Member = function(name, id, securityNo) {
	this.name = name;
	this.id = id;
	this.securityNo = securityNo;
}
Member.prototype.setValue = function(newName, newId, newSecurityNo)
{
	this.name = newName;
	this.id = newId;
	this.securityNo = newSecurityNo;
}
Member.prototype.getAge = function() {
	var birthYear = parseInt(this.securityNo.substring(0, 2));
	var code = this.securityNo.substring(6,7);
	if (code == '1' || code == '2') {
		birthYear += 1900;
	} else if (code == '3' || code == '4') {
		birthYear += 2000;
	}
	var today = new Date();
	return today.getFullYear() - birthYear;
}
Member.prototype.toString = function() {
	return this.name + "[" + this.id + "]";
}
```

useMember.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ko" lang="ko">
<head>
       <meta http-equiv="content-type" content="text/html;
charset=euc-kr" />
       <title>Member 클래스 사용</title>
       <script type="text/javascript" src="js/log.js"></script>
       <script type="text/javascript" src="js/member.js"></script>
       <script type="text/javascript">
       window.onload = function() {
              var mem = new Member("guard", "김경호",
"7700001000000");
              log('변경전');
              log('회원 : ' + mem.toString());
              log('나이 : ' + mem.getAge());

              mem.setValue("tomato", "김은희", "8000001000000");
              log('변경후');
              log('회원 : ' + mem.toString());
              log('나이 : ' + mem.getAge());
       }
       </script>
</head>
<body>
<div id="debugConsole"></div>
</body>
</html>
```