Ajax & jQuery

목차

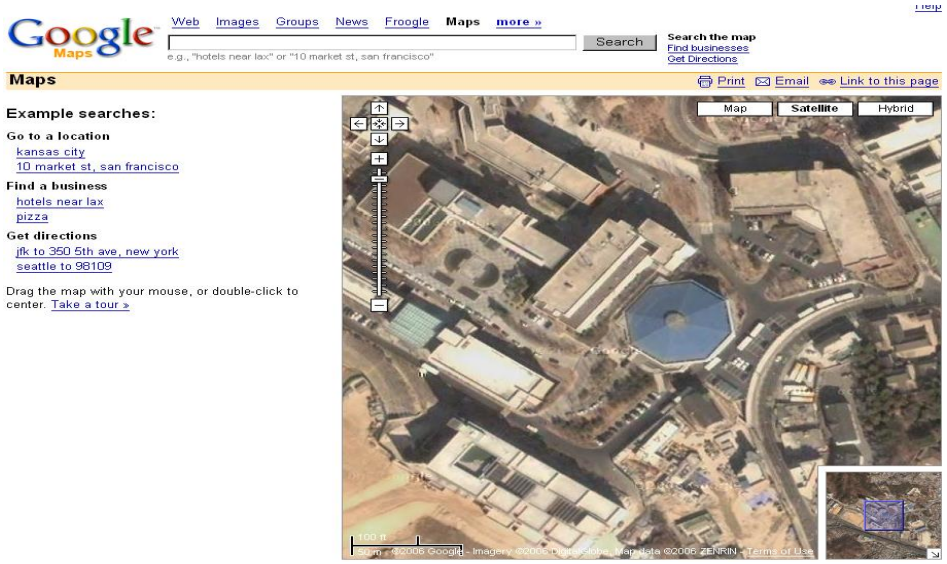
- Jesse James Garrett
- Ajax: A New Approach to Web Applications
- Feb 18, 2005

참고

- https://courses.cs.washington.edu/courses/cse490h/07sp/readings/ajax_adaptive_path.pdf

- AJAX 개요
  - 비동기적 Javascript 언어와 XML 등의 기타 웹 표준을 종합적으로 사용하는 기술로서 Web 2.0의 핵심 기술.
  - AJAX의 개별 기술은 신기술이 아니지만 이들을 종합적으로 활용하는 기술이 핵심임.(기존의 기술들을 효율적으로 조합해 사용)
  - 구글맵과 더불어 2005년 부터 AJAX라는 용어의 등장과 함께 웹 분야에서 주목을 받고 있는 최신 기술임
  - JavaScript, XML, HTML, DHTML, XmlHttpRequest, DOM, CSS등을 기반으로 작성됨

- AJAX 사용예(http://maps.google.com)
  - Ajax 로만 구현
  - 완벽한 멀티 플랫폼 서비스
  - 위성사진 제공
  - API공개로 각종 변종 서비스 제공
  - MSN Virtual Earth, A9 Maps 등 경쟁서비스 봇물



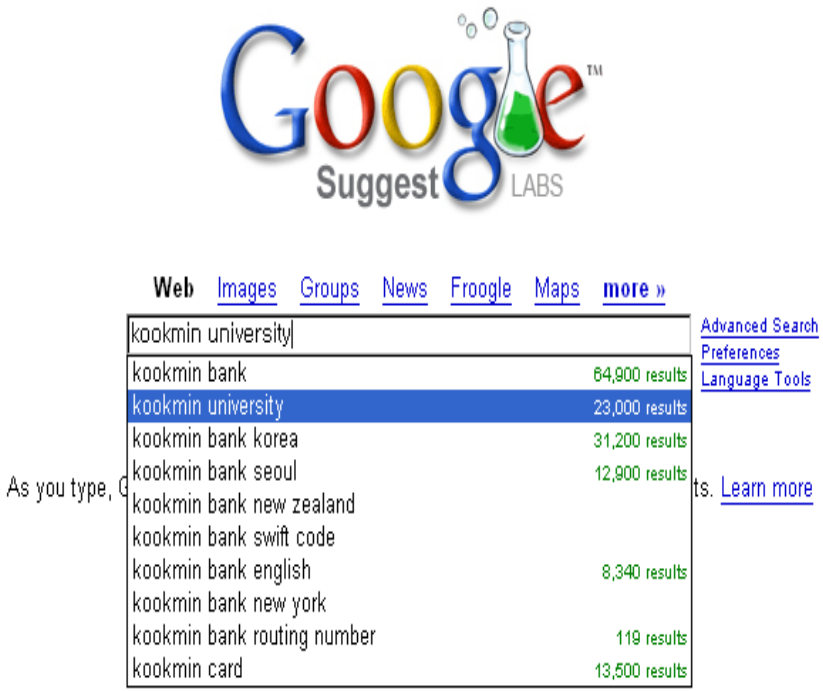
- AJAX 사용예(Google Suggest)

(http://www.google.com/webhp?complete=1&hl=en)

-검색어에 대한 추천단어 서비스

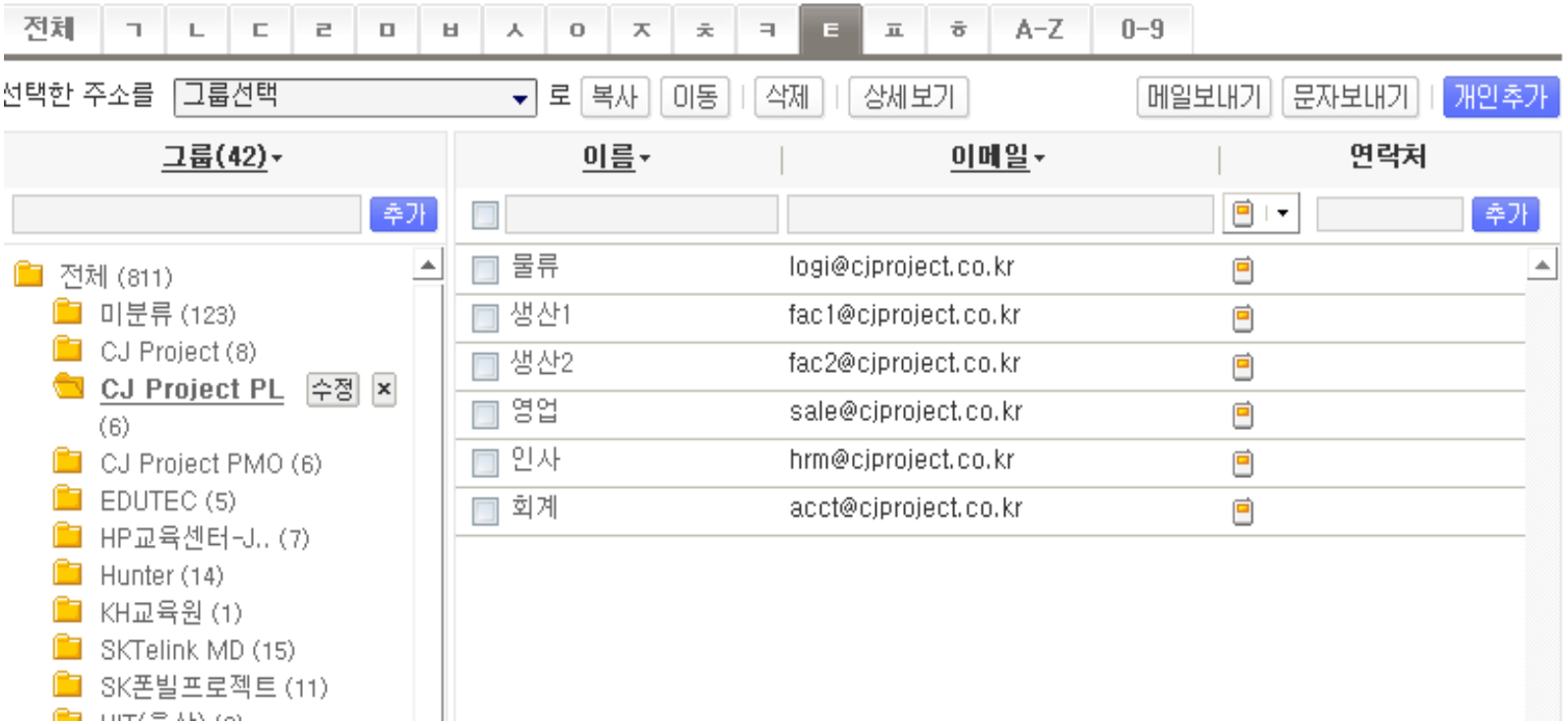
-Ajax를 이용한 대표적 서비

-국내 포탈도 동일서비스 제공



- AJAX 사용 예(DAUM 주소록)

- 주소정렬, 검색, 편집 시 DOM과 Ajax 사용
- 주소를 두 번 클릭하면 간단하게 정보를 수정.
- 메일 검색 시에 순차적으로 검색결과를 가져온다



- 장점

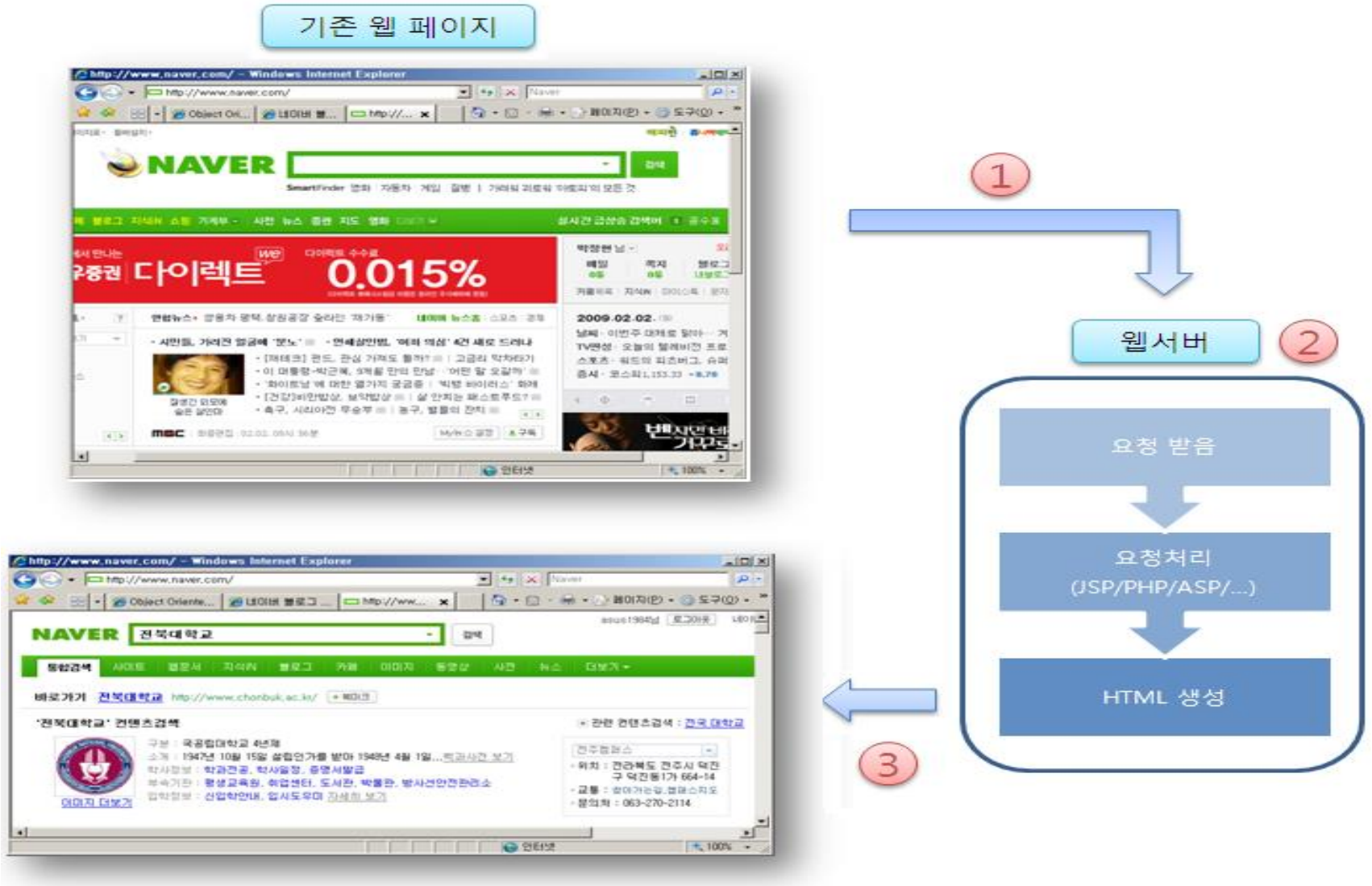
  - 페이지 이동 없이 빠르게 화면을 전환
  - 서버처리를 기다리지 않고 비동기 요청이 가능
  - 수신하는 데이터 양을 줄일 수 있음
  - ActiveX나 플래시 등의 플러그인 없이도Interactive한 웹 페이지 작성
  - Reload가 필요 없는 웹 페이지 작성 가능

- 일반 Web Application 의 특징
  - Click, Wait, and Refresh
  - 동기식 통신 모델
    - request / response
  - Page 단위로 갱신 ➔ Page-driven

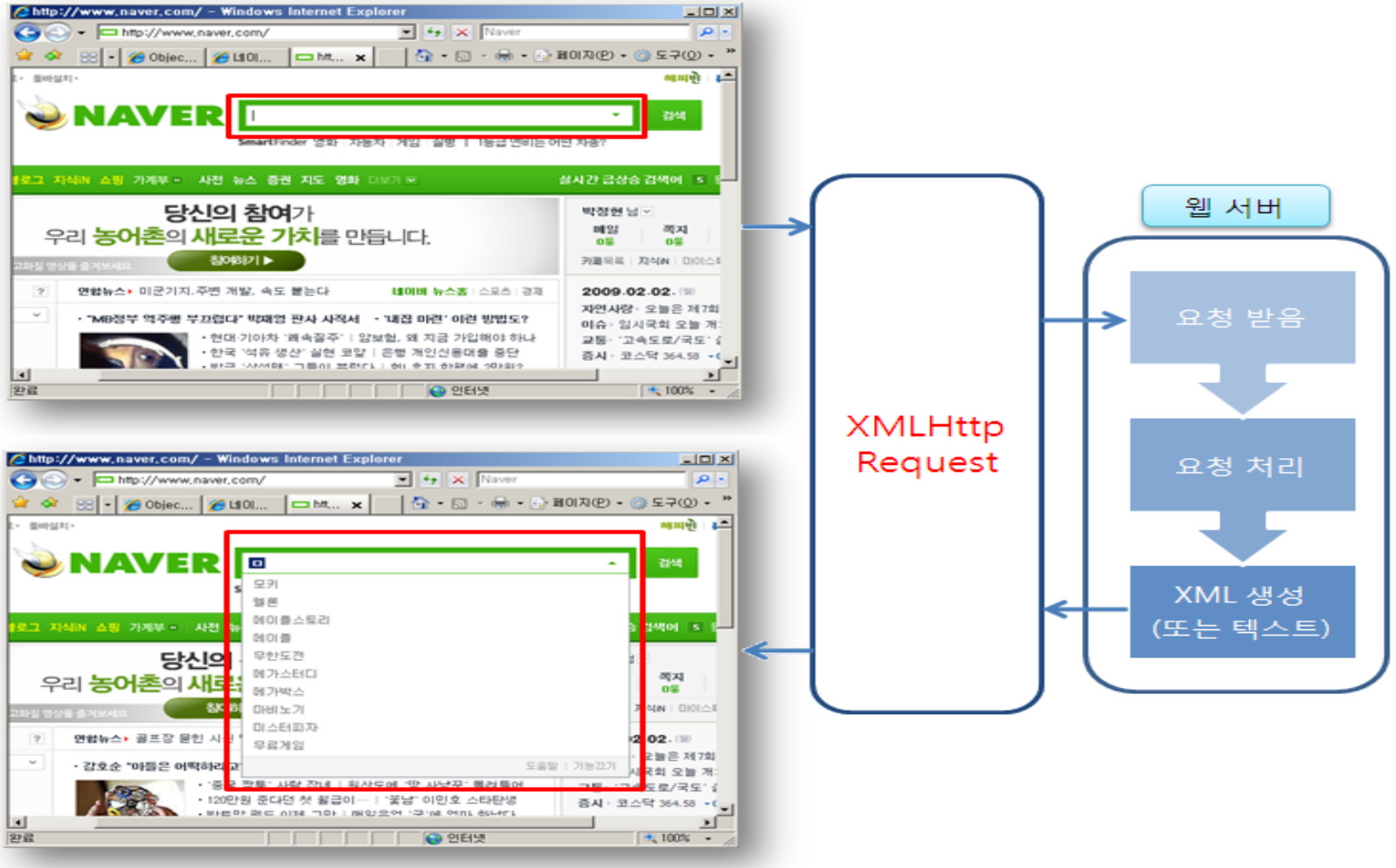
- 일반 Web Application의 문제점
  - 느린 응답시간과 화면 refresh
  - page 갱신 동안 이전 내용의 분실
  - 과도한 서버의 load로 인한 bandwidth의 소비

    ➔ Ajax가 등장하게 된 동기

- 일반 Web의 통신방법


기존 웹 페이지
NAVER
다이렉트
0.015%
1
웹서버
2
요청 받음
요청처리
(JSP/PHP/ASP/...)
HTML 생성
3
NAVER
전북대학교

- Ajax 의 통신방법


NAVER
당신의 참여가
우리 농어촌의 새로운 가치를 만듭니다.
XMLHttp
Request
웹 서버
요청 받음
요청 처리
XML 생성
(또는 텍스트)

XMLHttpRequest 객체

- XMLHttpRequest의 개요
  - MS(마이크로 소프트)는 Internet Explorer(IE)가 자바스크립트로부터 XML을 불러올 수 있도록 XMLHttpRequest를 설계.
  - XMLHttpRequest객체는 현재 대부분의 브라우저에 내장되어 있는 객체이다.
  - XMLHttpRequest는 실제로 자바스크립트에 대한 일반적인 HTTP 클라이언트 이기 때문에 GET과 POST HTTP 요청을 서버에 할 수 있다.
    - **클라이언트와 서버간의 통신을 담당하는 객체.**

XMLHttpRequest 객체

- XMLHttpRequest의 개요
  - XMLHttpRequest는 간단한 API와 몇 가지 메서드와 속성들을 제공한다. 그러나 브라우저간의 차이점이 존재하기 때문에 “Cross-Brower”(브라우저의 종류에 관계없이 동작) 부분을 생각할 필요가 있다.
  - 내용의 변경 없이 새로운 컨텐츠를 불러올 수 있다.
  - JavaScript로 동기 방식의 호출을 할 수 있다.

XMLHttpRequest 객체

- Cross-Browsing
  - 일반적으로 많이 사용되는 브라우저는 IE이다. 그러나 IE 브라우저를 많이 사용하는 것이지 브라우저가 IE만 있는 것은 아니다.
  - Firefox, Netscape, Safari, Opera, Konqueror
  - 브라우저마다 XMLHttpRequest 객체의 명칭이 다르다.
  - 각기 다른 객체를 통합하여 하나로 만들어야 사용자가 어떤 브라우저를 사용하더라도 이를 대응할 수 있다.
  - 이렇게 브라우저에 상관없이 동작하는 것을 크로스 브라우징(Cross-Browsing)이라고 한다.

XMLHttpRequest 객체

- XHR(XMLHttpRequest) 메소드
  - void open(string method, string url, boolean async, string username, string password)
    - 요청을 초기화한다.
    - 파라미터중에서 method, url 두개만 필수항목이고 나머지는 선택항목이다.
    - method 는 POST, GET, PUT 중 하나를 사용하면 되며, url 은 요청하고자 하는 서버의 url 이다. asynch 는 요청이 비동기인지 여부를 판단하는 항목이다. 입력하지 않으면 디폴트로 true 가 설정되어 요청은 비동기로 처리된다. false 로 설정하면 요청은 동기로 처리되기 때문에 서버에서 응답을 받을 때까지 프로세스는 기다리게 된다.
    - 사실 XHR 을 사용하는 가장 큰 이점중의 하나인 비동기 처리를 위해서는 asynch 항목을 true 로 설정해서 사용해야 한다.

- XHR(XMLHttpRequest) 메소드
  - void send(content)

    -실질적으로 요청을 서버로 보낸다.

    -요청이 비동기이면 이 메소드는 바로 리턴되지만 요청이 동기이면 서버에서 응답을 받을때까지 계속 대기한다.

    - content 는 선택사항이며, DOM 객체(XML 객체)이거나 input stream, string 값으로 설정할 수 있으며 HttpRequest body 의 한 부분으로 서버로 전달된다.

    - content 에 값을 넘기려면 **open() 메소드는 반드시 POST 로 설정해야 하며,** GET 방식으로 요청하려면 null 을 설정하면 된다.

XMLHttpRequest 객체

- XHR(XMLHttpRequest) 메소드

- void setRequestHeader(string header, string value)
  - header 에 해당하는 value 값으로 HttpRequest 헤더에 값을 설정하는 메소드로써, **반드시 open() 메소드 다음에 위치해야 한다.**
- void abort() : 요청을 중지한다.
- string getAllResponseHeaders()
  - 요청에 대응되는 응답의 헤더정보를 리턴한다. 즉, Content-Length, Date, URI 등을 포함하는 헤더정보를 string 형식으로 반환한다.
- string getResponseHeader(string header)
  - 응답의 헤더정보중에서 header 에 대응되는 값을 string 형식으로 반환한다.

- XHR(XMLHttpRequest) 속성
  - **Onreadystatechange 이벤트**
    - 자바스크립트 콜백함수(funtion pointer)를 지정한다.
    - 콜백함수는 readyState 값이 변할 때 마다 호출된다.
    - 요청이 서버로 보내지면 readyState 는 5가지 숫자 값으로 계속 변화가 일어나게 된다.
  - **readyState**
    - 요청의 상태를 의미한다.

값	의 미	설 명
0	uninitialized	객체만 생성되고 아직 호출되지 않음
1	loading	open 메서드가 호출되고 아직 send메서드가 호출되지 않음
2	loaded	send 메서드 호출, status와 헤더는 도착하지 않음
3	interactive	데이터의 일부를 받은 상태
4	complete	데이터를 전부 받음

XMLHttpRequest 객체

- **responseText(=response)**
  - 서버의 응답을 string 형식으로 나타낸다. 단순 text 를 innerHTML 속성으로 표현하기에는 알맞지만 논리적으로 파싱하거나 동적으로 페이지 컨텐츠를 생성하기는 힘들다.
- **responseXML**
  - 서버의 응답을 XML 로 나타낸다. 이 속성은 DOM 객체로 파싱할 수 있다.
- **status**
  - 서버로부터의 HTTP 상태코드이다.(예 200(OK), 404(NOT Found), 202(결과 값이 없을 때)등등)

- XHR(XMLHttpRequest) 이벤트
  - **onreadystatechange** 가장 '오래된 ' 메서드:(readyState)가 변경될 때마다 발생
  - **onloadstart** open 메서드가 불러질 때 호출
  - **onprogress** 데이터를 수신중일 때 반복해서 발생
  - **onloadend** 데이터 수신이 끝났을 때 발생
  - **onload** 수신이 성공했을 때 발생
  - **onerror** 수신이 실패했을 때 발생
  - **onabort** 수신중 abort() 메서드가 호출되었을 때
  - **ontimeout** 수신중 타임아웃 되었을 때 발생

- simpleRequest.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>simple xhr request</title>
<script>
var xhr;
function startRequest() {
xhr = new XMLHttpRequest();
xhr.open("GET", "simpleResponse.xml");
xhr.onload = onload;
xhr.send();
}
function onload() {
alert(xhr.response);
}
</script>
</head>
<body>
<form>
<input type="button" value="async test" onclick="startRequest()"/>
</form>
</body></html>
```

- simpleResponse.xml

```
<고객>
<이름>홍길동</이름>
<나이>30</나이>
</고객>
```

첫째, 사용자가 Start Basic Asynchronous Request 버튼을 클릭하면 이벤트가 발생해서 startRequest() 메소드가 실행된다.
둘째, XHR 객체가 생성되고 handleStateChange 콜백함수가 XHR 객체의 onreadystatechange 속성에 저장된다.
셋째, GET 방식의 비동기로 서버에 요청을 보낸는데, 이때 서버의 simpleResponse.xml 파일을 요구한다.
넷째, 서버는 Ajax 클라이언트의 요청 url 인 simpleResponse.xml 을 찾아서 읽은 후에 string 형식으로 XHR 객체로 보낸다.
다섯째, 콜백메소드는 XHR 의 state 가 변할때 실행되므로 readyState=4 이고 stat=200 일때 결과값을 브라우저에 보낸다.

innerHTML

- innerHTML이란?

- 특정 html 태그 안에 들어갈 html을 말한다.
- document.getElementById()함수를 이용해서 원하는 html 태그에 구조적으로 접근하여 html 태그를 삽입 또는 추출 할 수 있다.
- 간단하고 단순한 문자열을 처리하는데 적합한 형태이다.

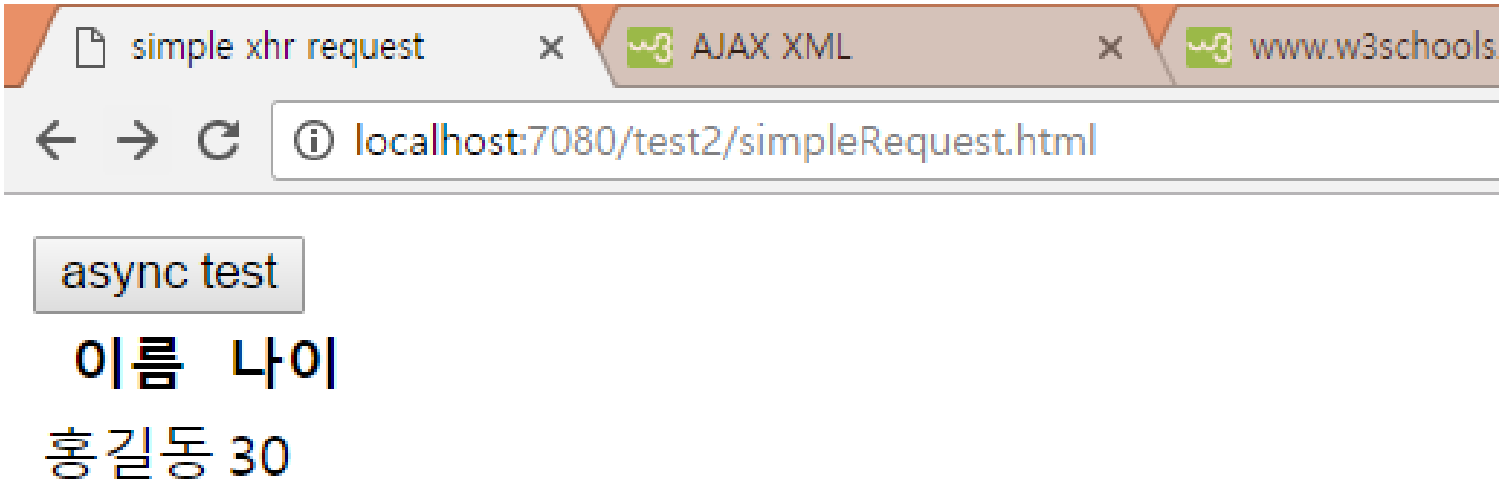
- simpleRequest.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>simple xhr request</title>
<script>
var xhr;
function startRequest() {
xhr = new XMLHttpRequest();
xhr.open("GET", "simpleResponse.xml");
xhr.onload = onload;
xhr.send();
}
function onload() {
                var xmlDoc = xhr.responseXML;
                var table = "<table><tr><th>이름</th><th>나이</th></tr>";
                var n = xmlDoc.getElementsByTagName("이름");
                var a = xmlDoc.getElementsByTagName("나이");
                table += "<tr><td>" + n[0].childNodes[0].nodeValue;
                table += "</td><td>" + a[0].childNodes[0].nodeValue;
                table += "</td></tr></table>";
                document.getElementById("result").innerHTML = table;
}
```

- simpleRequest.html

```
</script>
</head>
<body>
	<form>
		<input type="button" value="async test"
onclick="startRequest()"/>
	</form>
	<div id="result"></div>
</body>
</html>
```

- simpleRequest.html



Simple Application

- Auto Complete(Struts2)
- Ajax Validation Example
- Auto Refresh 구현하기
- ProgressBar

AutoComplete(Struts2)

- 먼저 struts2_ajax라는 Tomcat 프로젝트 작성
- Commons-logging, freemarker, ognl, struts2-core, xwork, log4j 라이브러리를 적절히 추가
- [web.xml]

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_9" version="2.4"
        xmlns="http://java.sun.com/xml/ns/j2ee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
                        http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">

   <display-name>Struts2 HelloWorld</display-name>

   <filter>
      <filter-name>struts2</filter-name>
      <filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-class>
   </filter>

   <filter-mapping>
   <filter-name>struts2</filter-name>
   <url-pattern>/*</url-pattern>
   </filter-mapping>
</web-app>
```

AutoComplete(Struts2)

- [/WEB-INF/src/struts.xml]

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
    "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
        <include file="ajax.xml" />
</struts>
```

- [/WEB-INF/src/ajax.xml]

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
        "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
        "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

        <package name="ajax" extends="struts-default">

                <global-results>
                        <result name="Exception">/exception.jsp</result>
                </global-results>

                <global-exception-mappings>
                        <exception-mapping exception="java.lang.Exception"
                                result="Exception" />
                </global-exception-mappings>


                <action name="autoComplete" method="autoComplete" class="web.ajax.AjaxAction">
                        <interceptor-ref name="servlet-config" />
                        <interceptor-ref name="exception" />
                        <result name="searchTitleList_success">/AutoComplete.jsp</result>
                </action>

                <action name="defaultAction">
                        <result>/WrongAction.jsp</result>
                </action>

        </package>
</struts>
```

[/ajax.js]

```
var AutoComplete = {
        xmlHttp : null,
        /* XMLHttp객체생성/
        getXMLHttpRequest: function() {
if(window.ActiveXObject){//인터넷 익스플로러일 경우
                        try{  return new ActiveXObject("Msxml2.XMLHTTP");
                        }catch(e){
                                try{
                                        return new ActiveXObject("Microsoft.XMLHTTP");
                                 }catch(e1){
                                        return null;
                                }
                        }
                }
                 //다른 브라우저일 경우
                else if(window.XMLHttpRequest){    return new XMLHttpRequest();          }
                //브라우저 식별 실패
                 else{                          return null;                 }
        },
```

AutoComplete(Struts2)

```
 /* XMLHttp Request를 서버로 */
ContentLoader: function(url, execFunction) {
        xmlHttp = this.getXMLHttpRequest();
        xmlHttp.onreadystatechange = execFunction;
        xmlHttp.open("POST", url, true); // url의 주소를 GET방식으로 열 준비를 한다.
       xmlHttp.send(); //서버에 전송한다.
},
/* XMLHttp Response가 서버로 부터 온전한 상태로 왔는지 확인   */
getState: function() {
        if(xmlHttp.readyState == 4){ //데이터의 전부를 받은 상태
                if(xmlHttp.status == 200){//요청 성공
                        return true;
                }else{
                        return false;
                }
        }
}
}
```

- [/WEB-INF/src/log4j.xml]

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration>
<configuration xmlns='http://logging.apache.org/'>
   <appender name="CONSOLE" class="org.apache.log4j.ConsoleAppender">
      <layout class="org.apache.log4j.PatternLayout">
         <param name="ConversionPattern"
            value="%p - %C{1}.%M(%L) | %m%n"/>
      </layout>
   </appender>

   <logger name="org.apache">
      <level value="WARN"/>
   </logger>

   <logger name="org.springframework">
      <level value="DEBUG"/>
   </logger>

   <logger name="springJDBC">
      <level value="DEBUG"/>
   </logger>

   <root>
      <level value="WARN"/>
<appender-ref ref="CONSOLE"/>
   </root>
</configuration>
```

AutoComplete(Struts2)

- [/WEB-INF/src/web/ajax/AjaxAction.java]

```
package web.ajax;
import javax.servlet.http.*;
import org.apache.log4j.Logger;
import org.apache.struts2.interceptor.*;
import com.opensymphony.xwork2.ActionSupport;

public class AjaxAction extends ActionSupport implements ServletRequestAware,
ServletResponseAware {

    private HttpServletRequest request;
    private HttpServletResponse response;
  private Logger logger = Logger.getLogger(this.getClass());

  public String autoComplete() throws Exception {

      try{

          logger.info("autoComplete start");
```

- [/WEB-INF/src/web/ajax/AjaxAction.java]

```
String[] wordArr = {"acrobat“,                "adobe“,                "bmw",
                    "brave“,                    "cbs“,                  "cnn",
                    "heaven“,                   "html“,                 "kbs",
                    "key“,                    "net“,                  "news",
                    "queen“,                    "quick“,                "soup",
                    "super“,                    "ufo“,                  "usb",
                    "work“,                     "wow“,                  "zebra",
                    "zenith"};
        StringBuffer xmlStr = new StringBuffer("");
        boolean isTitleExist = false;

        //실제업무에서는 비지니스 레이어에서 XML String을 구현한다.
        //String xmlStr = ajax.getAutoCompleteTitleList(boardVO);

        xmlStr.append("<?xml version=\"1.0\" encoding=\"EUC-KR\"?>");
        xmlStr.append("<words>");
for(int i = 0;  i < wordArr.length ; i++){

            if(wordArr[i].indexOf(request.getParameter("word")) != -1){

                xmlStr.append("<word>");
                xmlStr.append(wordArr[i]);
                xmlStr.append("</word>");
                isTitleExist = true;
            }

        }
```

- [/WEB-INF/src/web/ajax/AjaxAction.java]

```
xmlStr.append("</words>");
                //logger.info("request.getParameter(\"title\") : " + request.getParameter("title"));
                logger.info("xmlStr.toString() : " + xmlStr.toString());
                if(isTitleExist){
                    response.setContentType("text/xml");
                   response.setHeader("Cache-Control", "no-cache");
                   response.getWriter().write(xmlStr.toString());
                }else{
                    response.setStatus(HttpServletResponse.SC_NO_CONTENT);
                }
                logger.info("forward :" + request.getParameter("forward"));
                logger.info("autoComplete end");
                return null;
            }catch(Exception e){
                   logger.info("Error : " + e.toString());
                   e.printStackTrace();
                   throw e;
              }    }

   public void setServletRequest(HttpServletRequest request) {
      this.request = request;
   }
      public void setServletResponse(HttpServletResponse response) {
            this.response = response;
      }
}
```

- [/AutoComplete.jsp]

```
<%@ page contentType="text/html; charset=EUC-KR"%>
<html>
<head>
<title>autoComplete</title>
<script type="text/javascript" src="ajax.js"></script>
<script>

	var div;
	var table;
	var text;

	/*
	 * 변수값 초기화
	 */
	function onLoad(){

		div =
document.getElementById("autoCompleteDiv");
		table =
document.getElementById("autoCompleteTable");
		text =
document.getElementById("autoCompleteText");

	}

	/*
	 * 조회
	 */
	function listAutoComplete(){

		if(text.value != ""){

			if(event.keyCode != 229){


				var url = "autoComplete.action?word=" +
text.value;

				AutoComplete.ContentLoader(url,
viewTitleList);

			}

		}else{

			clearTitleList();

		}

	}
```

AutoComplete(Struts2)

- [/AutoComplete.jsp]

```
/*
 * 테이블을 DHTML로 작성
 */
function viewTitleList(){
    clearTitleList();
    if(AutoComplete.getState()){ //데이터의 전부를 받은 상태
        var tr, td;
        var words =
xmlHttp.responseXML.getElementsByTagName("word");
        var size = words.length;
        for(var i = 0 ; i < size ; i++){
            tr = table.insertRow();
            tr.style.width = "100%“;
            td = tr.insertCell();
        td.style.width = "100%";
        td.style.cursor = "hand";

        td.innerText = words[i].firstChild.nodeValue;
            td.onclick = function() { wordClick(this);};
        td.onmouseover = function() { mouseOver(this);};
            td.onmouseout = function() { mouseout(this);};

        }
    }
}
```

```
/*
 * 테이블의 목록을 초기화
 */
function clearTitleList(){

    var ind = table.rows.length;
for (var i = ind - 1; i >= 0 ; i--) {

  table.deleteRow(i);

}
}
/*
 * 클릭시
 */
function wordClick(oTd){

    text.value = oTd.innerText;

}
/*
 * 마우스오버
 */
function mouseOver(td){
    td.style.background= "#CBE7BA“;
}
```

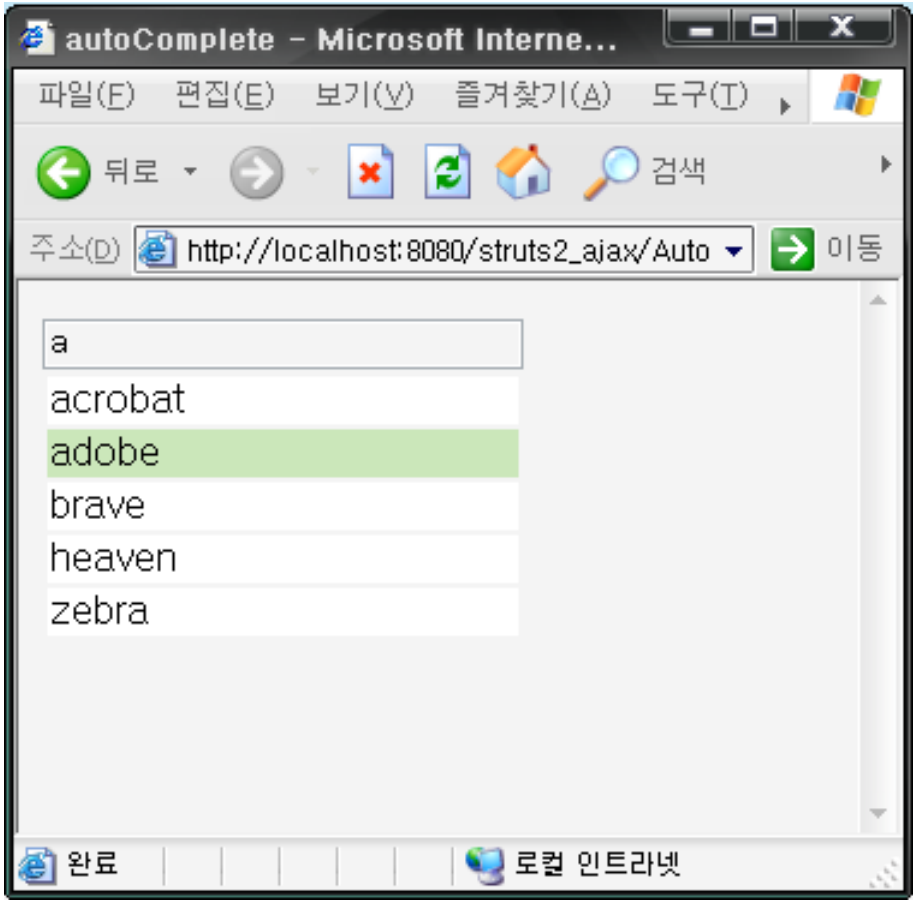
- [/AutoComplete.jsp]

```
/*
	* 마우스아웃
	*/
	function mouseout(td){
		td.style.background= "#FFFFFF“;
	}

</script>
</head>
<body onLoad="onLoad()">
<input type="text" id="autoCompleteText" style="width:200px" autocomplete="off"
onkeyup="javascript:listAutoComplete()"><br>
<div id="autoCompleteDiv" style="position:absolute; width:200px; height:180px; scrolling:yes;
overflow-y:auto;">
	<table id="autoCompleteTable" style="width:100%"></table>
</div>
</body>
</html>
```

AutoComplete(Struts2)

- [/exception.jsp]
- 오류발생!!

- [/WrongAction.jsp]
- 잘못된 Action 입니다!



Ajax Validation Example

[form_test.html]

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Using Ajax for validation</title>
    <script type="text/javascript">
        var xmlHttp;
        function createXMLHttpRequest() {
            if (window.ActiveXObject) {
                xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
            }
            else if (window.XMLHttpRequest) {
                xmlHttp = new XMLHttpRequest();
            }
        }
        function validate() {
            createXMLHttpRequest();
            var date = document.getElementById("birthDate");
            var url = "validation?birthDate=" + escape(date.value);
            xmlHttp.open("GET", url, true);
            xmlHttp.onreadystatechange = callback;
            xmlHttp.send(null);
        }
```

Ajax Validation Example

```
function callback() {
        if (xmlHttp.readyState == 4) {
           if (xmlHttp.status == 200) {
              //var mes = xmlHttp.responseXML.getElementsByTagName("message")[0].firstChild.data;
    var mes = xmlHttp.responseXML.getElementsByTagName("message")[0].firstChild.data;
              var val = xmlHttp.responseXML.getElementsByTagName("passed")[0].firstChild.data;
              setMessage(mes, val);
           }
        }
      }
      function setMessage(message, isValid) {
        var messageArea = document.getElementById("dateMessage");
        var fontColor = "red“;
        if (isValid == "true") {
           fontColor = "green";
        }
        messageArea.innerHTML = "<font color=" + fontColor + ">" + message + " </font>";
      }
   </script>
 </head>  <body>    <h1>Ajax Validation Example</h1>
   Birth date: <input type="text" size="10" id="birthDate" onchange="validate();"/>
   <div id="dateMessage"></div>
 </body>
</html>
```

Ajax Validation Example

[/Web-INF/web.xml]

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
   PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
   "http://java.sun.com/dtd/web-app_2_3.dtd">
<web-app>
   <servlet>
      <servlet-name>ValidationServlet</servlet-name>
      <servlet-class>validation.ValidationServlet</servlet-class>
   </servlet>
   <servlet-mapping>
      <servlet-name>ValidationServlet</servlet-name>
      <url-pattern>/validation</url-pattern>
   </servlet-mapping>
</web-app>
```

Ajax Validation Example

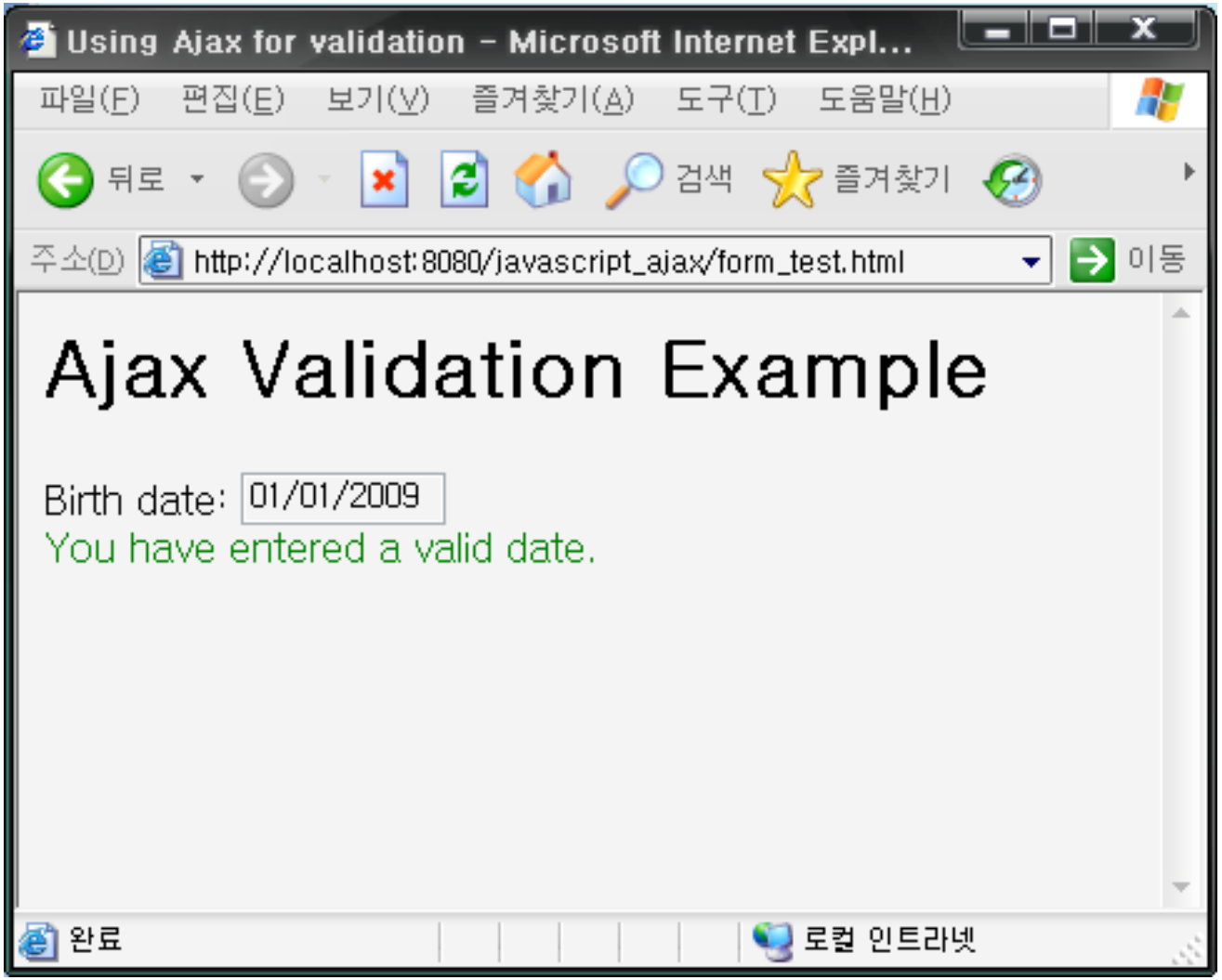
[/Web-INF/src/validation/ ValidationServlet .java]

```
package validation;

import java.io.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;

import javax.servlet.*;
import javax.servlet.http.*;

public class ValidationServlet extends HttpServlet {
   protected void doGet(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {
      PrintWriter out = response.getWriter();

      boolean passed = validateDate(request.getParameter("birthDate"));
      response.setContentType("text/xml");
      response.setHeader("Cache-Control", "no-cache");
      String message = "You have entered an invalid date.";
```

Ajax Validation Example

```
if (passed) {        message = "You have entered a valid date.“;       }
      out.println("<response>");
      out.println("<passed>" + Boolean.toString(passed) + "</passed>");
      out.println("<message>" + message + "</message>");
      out.println("</response>");
      out.close();
    }

   private boolean validateDate(String date) {
      boolean isValid = true;
      if(date != null) {
         SimpleDateFormat formatter= new SimpleDateFormat("MM/dd/yyyy");
         try {
            formatter.parse(date);
         } catch (ParseException pe) {
            System.out.println(pe.toString());
            isValid = false;
         }
      } else {          isValid = false;       }
      return isValid;
   }
}
```

Ajax Validation Example



Auto refresh 구현하기

- auto refresh 기능, 즉 자동으로 페이지의 일정 부분을 지정한 시간마다 정보를 업데이트 해 주는 기능.
- **일정시간을 설정하여 지속적으로 해당 부분만 정보가 수정되는 것을 확인 할 수 있다.**

- dynamicUpdate.html

```
 1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
 2 <html>
 3 <head>
 4 <title>Ajax Dynamic Update</title>
 5 <script language="javascript" type="text/javascript" src="createXMLHttpRequest.js"></script>
 6 <script type="text/javascript">
 7 var xmlHttp;
 8
 9 function doStart() {
10     xmlHttp = createXMLHttpRequest();
11     var url = "DynamicUpdateServlet?task=reset";
12     xmlHttp.open("GET", url, true);
13     xmlHttp.onreadystatechange = startCallback;
14     xmlHttp.send(null);
15 }
16
17 function startCallback() {
18     if (xmlHttp.readyState == 4) {
19         if (xmlHttp.status == 200) {
20             setTimeout("pollServer()", 5000);
21             refreshTime();
22         }
23     }
24 }
25
26 function pollServer() {
27     createXMLHttpRequest();
28     var url = "DynamicUpdateServlet?task=foo";
29     xmlHttp.open("GET", url, true);
30     xmlHttp.onreadystatechange = pollCallback;
31     xmlHttp.send(null);
32 }
33
34 function refreshTime(){
35     var time_span = document.getElementById("time");
36     var time_val = time_span.innerHTML;
```

```
38      var int_val = parseInt(time_val);
39      var new_int_val = int_val - 1;
40
41      if (new_int_val > -1) {
42          setTimeout("refreshTime()", 1000);
43          time_span.innerHTML = new_int_val;
44      } else {
45          time_span.innerHTML = 5;
46      }
47  }
48
49  function pollCallback() {
50      if (xmlHttp.readyState == 4) {
51          if (xmlHttp.status == 200) {
52              var message = xmlHttp.responseXML.getElementsByTagName("message")[0].firstChild.data;
53
54              if (message != "done") {
55                  var new_row = createRow(message);
56                  var table = document.getElementById("dynamicUpdateArea");
57                  var table_body = table.getElementsByTagName("tbody").item(0);
58                  var first_row = table_body.getElementsByTagName("tr").item(1);
59                  table_body.insertBefore(new_row, first_row);
60                  setTimeout("pollServer()", 5000);
61                  refreshTime();
62              }
63          }
64      }
65  }
66
67  function createRow(message) {
68     var row = document.createElement("tr");
69      var cell = document.createElement("td");
70      var cell_data = document.createTextNode(message);
71      cell.appendChild(cell_data);
72      row.appendChild(cell);
73      return row;
74  }
75  </script>
76  </head>
```

```
77 <body>
78 <h1>Ajax Dynamic Update Example</h1>
79 This page will automatically update itself:
80 <input type="button" value="Launch" id="go" onclick="doStart();" />
81 <p>Page will refresh in <span id="time">5</span> seconds.
82 <p>
83 <table id="dynamicUpdateArea" align="left">
84    <tbody>
85       <tr id="row0">
86          <td></td>
87       </tr>
88    </tbody>
89 </table>
90 </body>
91 </html>
```

• DynamicUpdateServlet.java

```
1  package com.gugun.ajax;
2
3  import java.io.*;
4  import java.net.*;
5  import javax.servlet.*;
6  import javax.servlet.http.*;
7
8  public class DynamicUpdateServlet extends HttpServlet {
9      private int counter = 1;
10
11     protected void doGet(HttpServletRequest request, HttpServletResponse response)
12     throws ServletException, IOException {
13         String res = "";
14         String task = request.getParameter("task");
15         String message = "";
16
17         if (task.equals("reset")) {
18             counter = 1;
19         } else {
20             switch (counter) {
21                 case 1: message = "Steve walks on stage"; break;
22                 case 2: message = "iPods rock"; break;
23                 case 3: message = "Steve says Macs rule"; break;
24                 case 4: message = "Change is coming"; break;
25                 case 5: message = "Yes, OS X runs on Intel - has for years"; break;
26                 case 6: message = "Macs will soon have Intel chips"; break;
27                 case 7: message = "done"; break;
28             }
29             counter++;
30         }
31
32         res = "<message>" + message + "</message>";
33
```

```
34          PrintWriter out = response.getWriter();
35          response.setContentType("text/xml");
36          response.setHeader("Cache-Control", "no-cache");
37          out.println("<response>");
38          out.println(res);
39          out.println("</response>");
40          out.close();
41      }
42
43      protected void doPost(HttpServletRequest request, HttpServletResponse response)
44      throws ServletException, IOException {
45          doGet(request, response);
46      }
47
48      public String getServletInfo() {
49          return "Short description";
50      }
51  }
52
```

- progressBar.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Ajax Progress Bar</title>
<script language="javascript" type="text/javascript" src="createXMLHttpRequest.js"></script>
<script type="text/javascript">
var xmlHttp;
var key;
var bar_color = 'gray';
var span_id = "block";
var clear = "   "

function go() {
    xmlHttp = createXMLHttpRequest();
    checkDiv();
    var url = "ProgressBarServlet?task=create";
    var button = document.getElementById("go");
    button.disabled = true;
    xmlHttp.open("GET", url, true);
    xmlHttp.onreadystatechange = goCallback;
    xmlHttp.send(null);
}

function goCallback() {
    if (xmlHttp.readyState == 4) {
        if (xmlHttp.status == 200) {
            setTimeout("pollServer()", 1000);
        }
    }
}

function pollServer() {
    xmlHttp= createXMLHttpRequest();
    var url = "ProgressBarServlet?task=poll";
    xmlHttp.open("GET", url, true);
    xmlHttp.onreadystatechange = pollCallback;
    xmlHttp.send(null);
}

```

```
40 function pollCallback() {
41     if (xmlHttp.readyState == 4) {
42         if (xmlHttp.status == 200) {
43             var percent_complete
44             = xmlHttp.responseXML.getElementsByTagName("percent")[0].firstChild.data;
45
46             var index = processResult(percent_complete);
47             for (var i = 1; i <= index; i++) {
48                 var elem = document.getElementById("block" + i);
49                 elem.innerHTML = clear;
50
51                 elem.style.backgroundColor = bar_color;
52                 var next_cell = i + 1;
53                 if (next_cell > index && next_cell <= 9) {
54                     document.getElementById("block" + next_cell).innerHTML
55                     = percent_complete + "%";
56                 }
57             }
58             if (index < 9) {
59                 setTimeout("pollServer()", 1000);
60             } else {
61                 document.getElementById("complete").innerHTML = "Complete!";
62                 document.getElementById("go").disabled = false;
63             }
64         }
65     }
66 }
67
68 function processResult(percent_complete) {
69     var ind;
70     if (percent_complete.length == 1) {
71         ind = 1;
72     } else if (percent_complete.length == 2) {
73         ind = percent_complete.substring(0, 1);
74     } else {
75         ind = 9;
76     }
77     return ind;
78 }
79
```

```
function checkDiv() {
    var progress_bar = document.getElementById("progressBar");
    if (progress_bar.style.visibility == "visible") {
        clearBar();
        document.getElementById("complete").innerHTML = "";
    } else {
        progress_bar.style.visibility = "visible"
    }
}

function clearBar() {
    for (var i = 1; i < 10; i++) {
        var elem = document.getElementById("block" + i);
        elem.innerHTML = clear;
        elem.style.backgroundColor = "white";
    }
}
</script>
</head>
<body>
<h1>Ajax Progress Bar Example</h1>
Launch long-running process:
<input type="button" value="Launch" id="go" onclick="go();" />
<p>
<table align="center">
   <tbody>
      <tr>
         <td>
         <div id="progressBar"
            style="padding:2px;border:solid black 2px;visibility:hidden">
         <span id="block1">   </span> <span id="block2">   </span>
         <span id="block3">   </span> <span id="block4">   </span>
         <span id="block5">   </span> <span id="block6">   </span>
         <span id="block7">   </span> <span id="block8">   </span>
         <span id="block9">   </span></div>
         </td>
      </tr>
      <tr>
         <td align="center" id="complete"></td>
      </tr>
   </tbody>
</table>
</body>
</html>
```

- ProgressBarServlet.java

```
1  package com.gugun.ajax;
2
3  import java.io.*;
4
5  import javax.servlet.*;
6  import javax.servlet.http.*;
7
8  public class ProgressBarServlet extends HttpServlet {
9      private int counter = 1;
10
11     protected void doGet(HttpServletRequest request, HttpServletResponse response)
12     throws ServletException, IOException {
13         String task = request.getParameter("task");
14         String res = "";
15
16         if (task.equals("create")) {
17             res = "<key>1</key>";
18             counter = 1;
19         }
20         else {
21             String percent = "";
22             switch (counter) {
23                 case 1: percent = "10"; break;
24                 case 2: percent = "23"; break;
25                 case 3: percent = "35"; break;
26                 case 4: percent = "51"; break;
27                 case 5: percent = "64"; break;
28                 case 6: percent = "73"; break;
29                 case 7: percent = "89"; break;
30                 case 8: percent = "100"; break;
31             }
32             counter++;
33
34             res = "<percent>" + percent + "</percent>";
35         }
36
```

Auto refresh 구현하기

```
37          PrintWriter out = response.getWriter();
38          response.setContentType("text/xml");
39          response.setHeader("Cache-Control", "no-cache");
40          out.println("<response>");
41          out.println(res);
42          out.println("</response>");
43          out.close();
44      }
45
46      protected void doPost(HttpServletRequest request, HttpServletResponse response)
47      throws ServletException, IOException {
48          doGet(request, response);
49      }
50
51      public String getServletInfo() {
52          return "Short description";
53      }
54  }
```

DOM
(Document Object Model)

DOM(Document Object Model)

- DOM은 문서(HTML, XML)를 객체로 표현하기 위한 방법
- XHR 객체의 resonseText 속성 단순한 문자열 처리에 적합한 형태이다.
- 복잡한 응답데이터의 경우는 단순한 문자열로 처리할 수 없으며 XML 형식으로 처리하는 것이 훨씬 논리적이고 효율적일 것이다.
- 브라우저는 서버로부터 전달받은 XML 문서를 W3C 의 DOM 을 이용해서 처리한다.
- DOM 은 HTML 과 XML 을 다루는 API 를 제공
- XML 문서를 트리 구조로 해석하고 트리 구조의 노드를 조작하여 문서를 다루기 위한 방법
- XML 파서를 이용하여 문자열 형태의 XML 문서가 DOM 구조를 가진 객체들로 생성 된다

DOM(Document Object Model)

- Java 언어에서는 클래스 라이브러리의 형태로 XML 파서를 사용 할 수 있으며 이것을 이용하면 메소드를 통해 XML 문서의 조작이 가능
- DOM에 대한 표준 명세를 제공하는 곳은 http://www.w3c.org/DOM
- DOM은 XML 문서를 트리 구조로 취급 하므로 다루기 용이하다. 그러나 XML 문서 전체를 읽어 트리로 조립하기에 커다란 XML 문서를 다루는기에는 용이 하지 못한 부분도 있다. 그래서 XML 문서를 다루는 방법으로 DOM과 함께 SAX라고 불리는 방법이 주로 사용 된다.
- SAX의 경우 앞부분 부터 순서대로 읽어 들여 처리하므로 요소나 텍스트를 단순히 추출하는것은 간단하나 문서의 일부분을 조작하는 것은 DOM에 비해 복잡하다.

DOM(Document Object Model)

- XML 문서를 처리하기 위한 DOM 요소의 속성
  - childNodes : 현재 요소의 자식을 배열로 표현한다.
  - firstChild : 현재 요소의 첫번째 자식이다.
  - lastChild : 현재 요소의 마지막 자식이다.
  - nextSibling : 현재 요소와 바로 다음의 요소를 의미한다.
  - nodeValue : 해당 요소의 값을 읽고 쓸 수 있는 속성을 정의한다.(=data)
  - parentNode : 해당 요소의 부모노드이다.
  - previousSibling : 현재 요소와 바로 이전의 요소를 의미한다.

DOM(Document Object Model)

- XML 다큐먼트를 다루는 유용한 DOM 요소의 메소드
  - getElementById(id) : 다큐먼트에서 특정한 id 속성값을 가지고 있는 요소를 반환한다.
  - getElementsByTagName(name) : 특정한 태그 이름을 가지고 있는 자식 요소로 구성된 배열을 리턴 한다.
  - hasChildNodes() : 해당 요소가 자식 요소를 포함하고 있는지를 나타내는 Boolean 값을 리턴 한다.
  - getAttribute(name) : 특정한 name 에 해당하는 요소의 속성값을 리턴 한다.

Dynamic DOM 다루기

- **컨텐츠를 동적으로 생성할 수 있게 해주는 W3C DOM 의 속성과 메소드.**
  - document.createElement(tagName) : tagName 으로 된 엘리먼트를 생성한다. div 를 메소드 파라미터로 입력하면 div 엘리먼트가 생성된다.
  - document.createTextNode(text) : 정적 텍스트를 담고 있는 노드를 생성한다.
  - <element>.appendChild(childNode) : 특정 노드를 현재 엘리먼트의 자식 노드에 추가시킨다. (예를들어 select 엘리먼트에 option 엘리먼트 추가)
  - <element>.getAttribute(name) : 속성명이 name 인 속성값을 반환한다.

Dynamic DOM 다루기

- <element>.setAttribute(name, value) : 속성값 value 를 속성명이 name 인 곳에 저장한다.
- <element>.insertBefore(newNode, tartgetNode) : newNode 를 tartgetNode 전에 삽입한다.
- <element>.removeAttribute(name) : 엘리먼트에서 name 속성을 제거한다.
- <element>.removeChild(childNode) : 자식 엘리먼트를 제거한다.
- <element>.replaceChild(newNode, oldNode) : oldNode 를 newNode 로 치환한다.
- <element>.hasChildNodes() : 자식 노드가 존재하는지 여부를 판단한다. 리턴형식은 Boolean 이다.

Dynamic DOM 다루기 예제

[dynamicContent.html]

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhmtl">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">
<title>페이지 내용을 동적으로 바꾸는 예제</title>
<script language="JavaScript">
<!--
var xmlHttp;
var requestType;

// XHR 객체를 생성한다.
function createXMLHttpRequest() {
    if(window.ActiveXObject) {
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    } else if(window.XMLHttpRequest) {
        xmlHttp = new XMLHttpRequest();
    }
}
```

Dynamic DOM 다루기 예제

```
// 조회를 한다.
function doSearch() {
    requestType = document.getElementById("부서").value;
    createXMLHttpRequest();
    xmlHttp.onreadystatechange = handleStateChange;
    xmlHttp.open("GET", "dynamicContent.xml", true);
    xmlHttp.send(null);
}
// 요청 처리 콜백 함수
function handleStateChange() {
    if(xmlHttp.readyState == 4) {
        if(xmlHttp.status == 200) {
            clearPreviousResults();
            parseResults();
        }
    }
}
```

Dynamic DOM 다루기 예제

```
// 이전 결과를 지운다.
function clearPreviousResults() {
    var header = document.getElementById("header");
    if(header.hasChildNodes()) {
        header.removeChild(header.childNodes[0]);
    }
    var tableBody = document.getElementById("resultsBody");
    while(tableBody.childNodes.length > 0) {
        tableBody.removeChild(tableBody.childNodes[0]);
    }
}

// 요청의 결과를 테이블 형태로 동적으로 출력한다.
function parseResults() {
    var results = xmlHttp.responseXML;
    var code = "";
    var departmentName = "";
    var position = "";
    var name = "";
```

Dynamic DOM 다루기 예제

```
// 모든 사원노드를 배열로 가져옵니다.
    var staffs = results.getElementsByTagName("사원");
    for(var i = 0; i < staffs.length; i++) {
        // 부모노드의 부서 코드를 가져옵니다.
        code = staffs[i].parentNode.getAttribute("code");
        // 부서를 선택했을 경우 코드가 다르면 건너뜁니다.
        if(requestType != "all" && code != requestType) continue;
        // 부서명, 직위, 이름 정보를 가져옵니다.
        departmentName = staffs[i].parentNode.getAttribute("title");
        position = staffs[i].getAttribute("position");
        name = staffs[i].childNodes[0].nodeValue;
        addTableRow(departmentName, position, name);
    }
    var header = document.createElement("h2");
    var headerText = document.createTextNode("조회결과:");
    header.appendChild(headerText);
    document.getElementById("header").appendChild(header);
    document.getElementById("resultsTable").setAttribute("border", "1");
}
```

Dynamic DOM 다루기 예제

```
// 테이블의 TR 엘리먼트를 주어진 내용으로 생성한다.
function addTableRow(departmentName, position, name) {
    // 테이블 행 엘리먼트를 생성합니다.
    var row = document.createElement("tr");
    // 테이블 데이터 엘리먼트를 생성해서 추가합니다.
    var cell = createCellWithText(departmentName);
    row.appendChild(cell);
    cell = createCellWithText(position);
    row.appendChild(cell);
    cell = createCellWithText(name);
    row.appendChild(cell);
    // 테이블에 행을 추가합니다.
    document.getElementById("resultsBody").appendChild(row);
}
// 테이블의 TD 엘리먼트를 주어진 text 를 내용으로 하여 생성한다.
function createCellWithText(text) {
   // 테이블 데이터 엘리먼트를 생성합니다.
    var cell = document.createElement("td");
    var textNode = document.createTextNode(text);
    cell.appendChild(textNode);
    return cell;
}
```

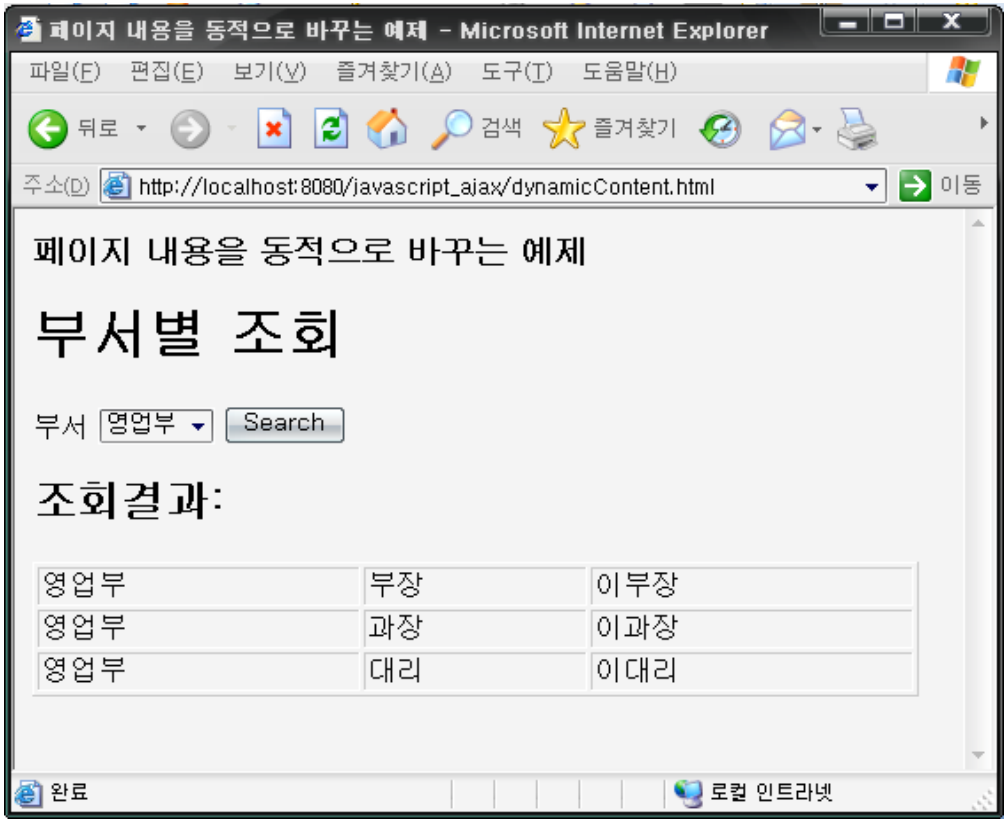
Dynamic DOM 다루기 예제

```
//-->
</script>
</head><body>
<h3>페이지 내용을 동적으로 바꾸는 예제</h3>
<h1>부서별 조회</h1>
<form action="#">
부서
<select id="부서">
  <option value="all">전체</option>
  <option value="sale">영업부</option>
  <option value="insa">인사부</option>
</select>
<input type="button" value="Search" onclick="doSearch();"/>
</form>
<span id="header"></span>
<table id="resultsTable" width="500" border="0">
  <tbody id="resultsBody">
  </tbody>
</table></body></html>
```

Dynamic DOM 다루기 예제

[dynamicContent.xml]

```
<?xml version="1.0" encoding="utf-8"?>
<부서들>
  <부서 code="insa" title="인사부">
    <사원 position="과장">김과정</사원>
    <사원 position="대리">김대리</사원>
    <사원 position="사원">김사원</사원>
  </부서>
  <부서 code="sale" title="영업부">
    <사원 position="부장">이부장</사원>
    <사원 position="과장">이과장</사원>
    <사원 position="대리">이대리</사원>
  </부서>
</부서들>
```

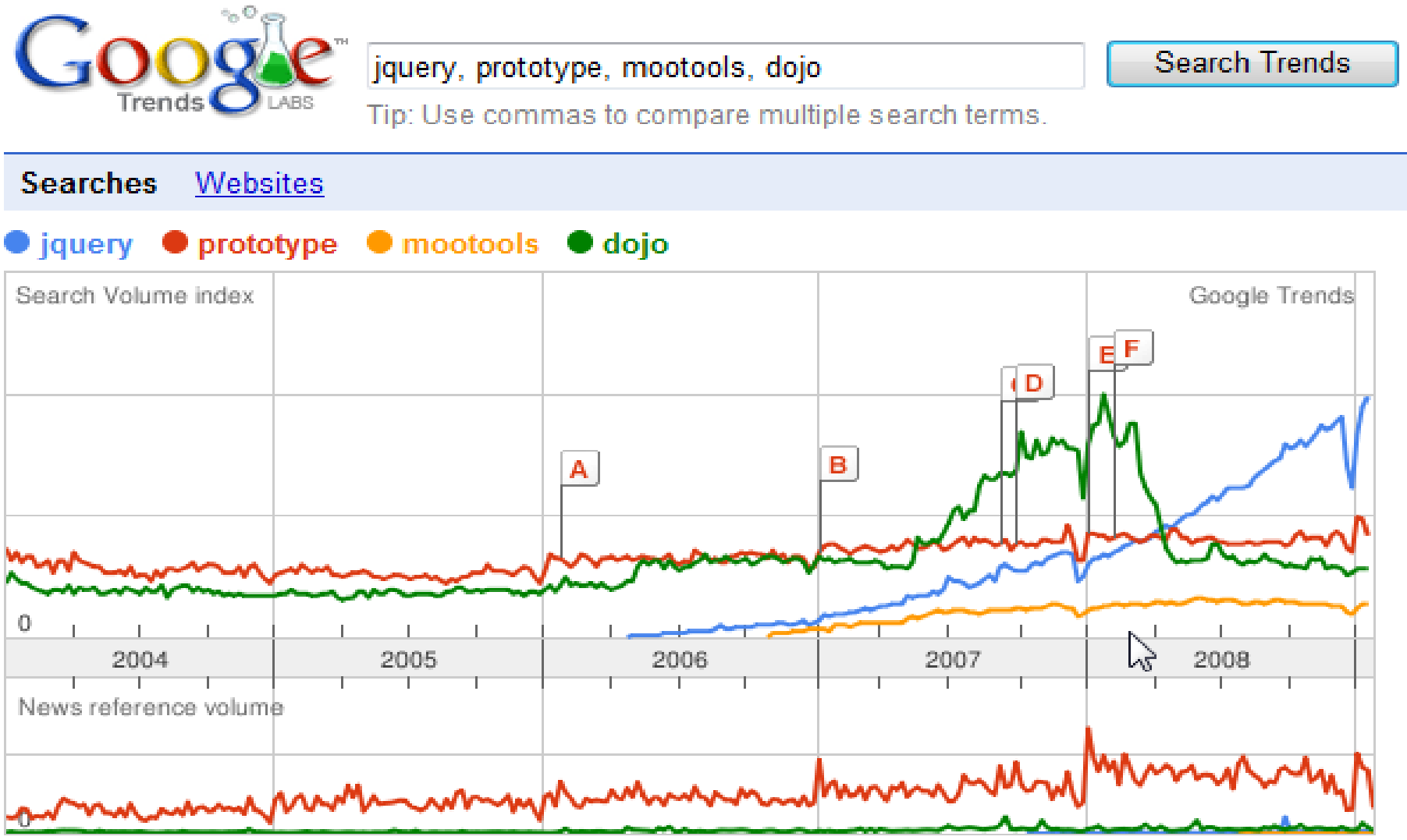


JQuery

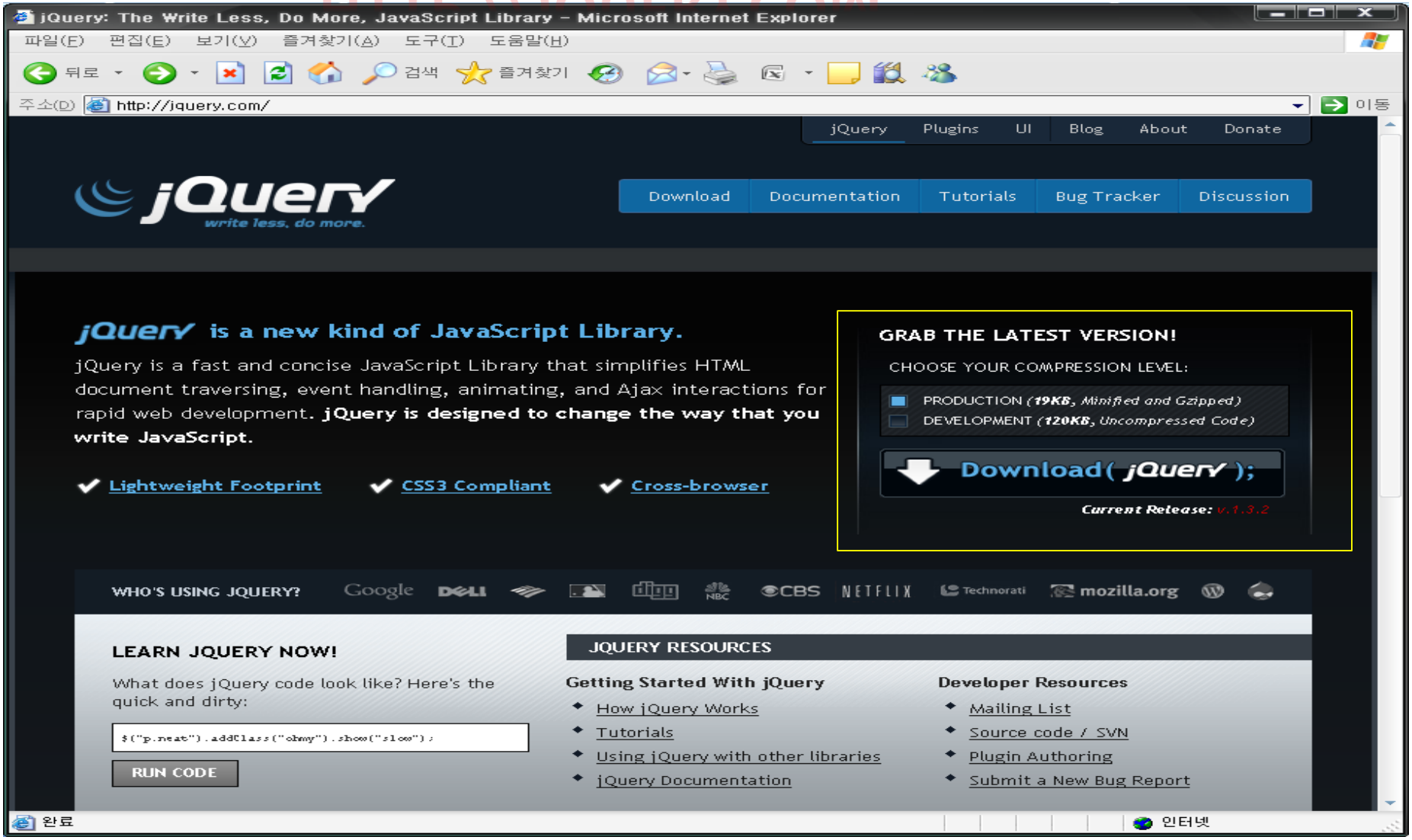
- 개요

- 2006년 초, John Resig가 만든 JavaScript Library
- JavaScript™와 Asynchronous JavaScript + XML (Ajax) 프로그래밍을 단순화 함
- DOM 스크립팅과 Ajax의 반복성을 단순하게
- 코드를 단순하고 간결하게 유지한다.
- 많은 반복 루프와 DOM 스크립팅 라이브러리 호출을 작성할 필요가 없다.

- JavaScript Library 종류



- 다운로드 HTTP://JQUERY.COM



- 특징

CSS Selector

도큐먼트의 구조를 명료하면서도 읽기 쉬운 형태로 표현 가능

Plug-in Architecture

feature creep를 피하고 창의적 산출물을 공유

Method Chaining

여러 동작을 한 줄에 나열이 가능하며, 임시 변수 사용을 최소화 하거나 불필요한 반복을 피할 수 있음

- jQuery를 사용하지 않았을 때

```
var external_links = document.getElementById('external_links');
var links = external_links.getElementsByTagName('a');
 for (var i=0;i < links.length;i++) {
        var link = links.item(i); link.onclick = function() {
            return confirm('You are going to visit: ' + this.href);
        };
 }
```

- jQuery를 사용했을 때

```
$('#external_links a').click(function() {
    return confirm('You are going to visit: ' + this.href); }
  );
```

- 예제분석

  - $() 함수는 jQuery에서 가장 강력한 함수로 이 함수를 사용하여 문서에서 엘리먼트를 선택한다. 이앞 에서, 이 함수는 Cascading Style Sheets (CSS) 신택스를 포함하고 있는 스트링으로 전달되고, jQuery는 효율적으로 이 엘리먼트를 찾는 것이다.

  - #external_links는 external_links의 id를 가진 엘리먼트를 찾는다. a 앞에 있는 공간은 jQuery에게 external_links 엘리먼트 내의 **모든 <a> 엘리먼트를** 찾도록 명령한다.

  - click 함수를 호출함으로써 클릭 핸들러 함수를 jQuery 객체의 각 엘리먼트에 할당할 수 있다.

- ${} 함수의 다른 사용법

  - 엘리먼트나 엘리먼트의 어레이를 $() 함수로 전달하면 window 객체 같은 것에 jQuery 함수를 적용하는 것이 가능하다. 일반적으로 이 함수를 다음과 같이 로드 이벤트에 할당한다.

```
window.onload = function() {
        // do this stuff when the page is done loading
};
```

jQuery를 사용하면, 같은 코드도 다음과 같이 된다.

```
$(window).load(function() {
        // run this when the whole page has been downloaded
});
```

페이지 로드 시점에 작업 수행하기

onload : 관련된 모든 도큐먼트가 브라우저로 다운로드 된 후에 발생한다.
이미지 다운 및 링크 설정 등이 전부 완료된 후 실행된다.

$(document).ready() : DOM이 로드되어 사용할 준비가 끝나는 시점에 바로 호출된다. 스크립트에서 모든 요소들에 접근할 수 있다는 것은 onload와 동일하나 관련된 모든 파일(이미지 등)들이 다운로드된 후를 의미하는 것은 아니다.

● 일반적인 자바스크립트를 사용하는 방법
function doStuff() { .... }를 정의한 후 HTML에서 아래와 같이 사용하는 방법
<body onload="doStuff();">

또는
function doOtherStuff() { .... }를 정의한 후
window.onload = doOtherStuff; 를 호출하는 방법이 있다.

단, onload 함수는 하나의 함수만 지정할 수 있다.

페이지 로드 시점에 작업 수행하기

● jQuery에서 간결하게 단축된 코드

다음은 모두 같은 의미로 사용된다.

```
$(document).ready( function() {
   ....
});

$().ready( function() {
 ....
});

$( function() {
 ....
});
```

- 셀렉터(Selector)

✓ **jQuery(selector) or $(selector)**

Selector / css 셀렉터를 사용한다.	
#ID	문서 내 ID와 일치하는 엘리먼트를 가져온다.
ElementName	문서 내 동일한 이름을 가진 엘리먼트를 가져온다.
ClassName	문서 내 동일한 클래스를 가진 엘리먼트를 가져온다.
*	문서 내 모든 엘리먼트를 가져온다.
Parent > Child	문서 내 부모 엘리먼트 아래에 있는 자식 엘리먼트를 가져온다.
Complex	ID,Element,ClassName을 복합적 사용

- 셀렉터(Selector)

#ID → $("#ID")

문서 내에 **ID**와 일치하는 요소를 가져온다.

```
<script type="text/javascript" src="lib/jquery.js"></script>
<script type="text/javascript">
//<![CDATA[
$(document).ready(function() {
    $("#Header").css("border", "#ff0000 1px solid");
});
//]]>
</script>

//HTML
<div id="Header">
jQuery | Write Less, Do More
</div>
```

- 셀렉터(Selector)

ElementName → $("DIV")

문서 내에 동일한 **Element**를 모두 가져 온다.

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("li").css("border", "#ff0000 1px solid");
 });
 //]]>
 </script>
//HTML
 <UL>
     <li>ASP.NET MVC 공식홈페이지</li>
     <li>다운로드</li>
     <li>ASP.NET MVC 튜토리얼</li>
 </UL>
```

- 셀렉터(Selector)

ClassName(Css) → $(".className")

문서 내에 동일한 **Class**를 모두 가져 온다.

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $(".layout3Left").css("border", "#ff0000 1px solid");
 });
 //]]>
 </script>
 //HTML
 <ul>
     <li>OracleJava.co.kr</li>
     <li class="layout3Left">다운로드1</li>
     <li class="layout3Left">다운로드2</li>
 </ul>
```

- 셀렉터(Selector)

Child Selector → $("p > a")

<p>엘리먼트를 부모로 가지고 있는 <a>를 선택 가져온다.

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
//<![CDATA[
$(document).ready(function() {
    $("div > a").css("border", "#ff0000 1px solid");
});
//]]>
</script>
//HTML
<div>
    <a href="http://www.oraclejava.co.kr">OracleJava</a>
    <span><a href="http://www.daum.net">DAUM</a></span>
    <a href="http://www.javaservice.net">자바서비스넷</a>
</div>
```

- 셀렉터(Selector)

Complex Selector → $("span.favLink")

엘리먼트와 **ClassName**이 일치하는 엘리먼트를 가져온다.

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("span.favLink").css("border", "#ff0000 1px solid");
 });
 //]]>
 </script>
 //HTML
 <div>
     <span><a href="http://www.oraclejava.co.kr">OracleJava</a></span>
     <span class="favLink">Naver.com</span>
     <a href="http://www.paran.com">My Paran</a>
 </div>
```

- 속성(attribute)

일치하는 집합에서 첫 번째 엘리먼트에 있는 속성 값을 가져오거나 일치하는 모든 엘리먼트에 있는 속성 값을 설정하는데 사용

$("selector").attr()

❖ **attr(name)**

선택한 엘리먼트의 name에 해당하는 어트리뷰트의 값을 가져온다.

❖ **attr(name, value)**

선택한 엘리먼트의 name에 해당하는 어트리뷰트의 값을 설정한다.

❖ **attr(properties)**

선택한 엘리먼트의 속성을 properties의 내용으로 설정합니다.

❖ **attr(name, single object function)**

선택한 엘리먼트의 속성을 function의 내용으로 설정합니다.

- 속성(attribute)

$("selector").attr(name)

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("#img1").click(function() {
         var title = $("#img1").attr("title");
         alert(title);
     });
 });
 //]]>
 </script>
 //HTML
 <div>
     <img id="img1" src="http://www.oraclejava.co.kr/oraclejavanew/img/logo.gif"  title="오라클자바" />
 </div>
```

- 속성(attribute)

$("selector").attr(name, value)

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("#img1").click(function() {
         $("#img1").attr("title", "오라클자바2");
         var title = $("#img1").attr("title");
         alert(title);
     });
 });
 //]]>
 </script>
 //HTML
 <div>
     <img id="img1" src=http://www.oraclejava.co.kr/oraclejavanew/img/logo.gif
title="오라클자바1" />
 </div>
```

- 속성(attribute)

$("selector").attr(properties)

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("#img1").click(function() {
         $("#img1").attr({ title: "오라클자바2", alt: "오라클자바2" });
         var title = $("#img1").attr("title");
         alert(title);
     });
 });
 //]]>
 </script>
 //HTML
 <img id="img1" src="http://www.oraclejava.co.kr/oraclejavanew/img/logo.gif"  title="
오라클자바1" />
 </div>
```

- 속성(attribute)

$("selector").attr(name, function)

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
     $("#img1").click(function() {
         $("#img1").attr("title", function() {
             return $("#img1").attr("src");
         });
         var title = $("#img1").attr("title");
         alert(title);
     });
 });
 //]]>
 </script>
 //HTML
 <div><img id="img1" src="http://www.oraclejava.co.kr/oraclejavanew/img/logo.gif"
title="오라클자바" /></div>
```

- 효과(effect)

선택한 **Element**의 효과 설정

Effect
Basics :
show(), hide(), toogle()
Sliding :
slideDown(), slideUp(), slideToggle()
Fading :
fadeIn(), fadeOut(), fadeTo()
Custom :
animate()

- 효과(Effect)

$().show()

show();
show(speed);
show(speed, callback);

$().hide()

hide();
hide(speed);
hide(speed, callback);

$().toggle()

toggle();
toggle(switch);
toggle(speed, callback);

speed(string or number)	
string	slow, normal, fast
number	milliseconds

switch (true, false)	
true	Display : block or visible
false	Display : none

callback (function) / optional
효과 후 호출할 함수명

- 효과(Effect)

$().slideDown()

slideDown();
slideDown(speed);
slideDown(speed, callback);

$().slideUp()

slideUp();
slideUp(speed);
slideUp(speed, callback);

$().slideToggle()

slideToggle();
slideToggle(speed, callback);

Down | Up	
Down	Element Display
Up	Element Hide

speed(string or number)	
string	slow, normal, fast
number	milliseconds

callback (function) / optional
효과 후 호출할 함수명

- 효과(Effect)

$().fadeIn()

fadeIn();
fadeIn(speed);
fadeIn(speed, callback);

$().fadeOut()

fadeOut();
fadeOut(speed);
fadeOut(speed, callback);

$().fadeTo()

fadeTo();
fadeTo(speed, callback);
fadeTo(speed, opacity, callback);

In | Out | To	
In	Element Display
Out	Element Hide
To	Opacity

speed(string or number)	
string	slow, normal, fast
number	milliseconds

callback (function) / optional
효과 후 호출할 함수명

- 효과(Effect)

$().animate()

animate(params, [duration], [easing], [callback]);
animate(params, options);

params
style sheet attribute

duration(optional)	
number	milliseconds

easing(optional) / need plug-in	
string	linear , swing

callback (function) / optional
executed whenever the animation c ompletes

- animate

```
<script type="text/javascript" src="jquery.js"></script>
 <script type="text/javascript">
 //<![CDATA[
 $(document).ready(function() {
      $("#img1").animate({
           width: "50%",
           opacity: 0.3
      }, 5000);
 });
 //]]>
 </script>
 //HTML
 <div><img id="img1" src="http://www.oraclejava.co.kr/oraclejavanew/img/logo.gif"
title="오라클자바" /></div>
```

● jQuery(html) Returns: jQuery
jQuery(html), 실행 후 jQuery객체를 반환

주어진 html을 가지고 빠르게 문서 원소를 생성한다.그리고 jQuery객체로서 그 것을 반환한다. 이는 곧 그것을 이어서 jQuery의 다른 함수와 함께 사용가능하다는 뜻이다.

```
<html>
<head>
  <script src=“jquery.js”></script>
  <script>
  $(document).ready(function(){
   $("<div><p>Hello</p></div>").appendTo("body");
  });
  </script>
</head>
<body>
</body>
</html>
```

● jQuery(elements) Returns: jQuery

jQuery(원소) jQuery(원소.원소.원소), 실행후 jQuery객체를 반환

하나 또는 다단계의 문서원소로서 사용할 수 있다.

```
<html>
<head>
  <script src=“jquery.js"></script>
  <script>
  $(document).ready(function(){
   $(document.body).css( "background", "black" );
  });
  </script>
</head>
<body>
</body>
</html>
```

● each(callback) Returns: jQuery
each(콜백함수), 실행후 jQuery객체를 반환

매치 되어진 모든 원소에 대해 주어진 콜백 함수를 실행한다. 루프의 의미이다.

```
<html><head>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script>
  $(document).ready(function(){
   $(document.body).click(function () {
     $("div").each(function (i) {
       if (this.style.color != "blue") {
         this.style.color = "blue";
       } else {
         this.style.color = "";
       }
     });
   });
  });
  </script>
```

● each(callback) Returns: jQuery(계속)

```
 <style>
  div { color:red; text-align:center; cursor:pointer;
      font-weight:bolder; width:300px; }
  </style>
</head>
<body>
  <div>Click here</div>
  <div>to iterate through</div>
  <div>these divs.</div>
</body>
</html>
```

● **size( ) Returns: Number**

size( ), 실행후 숫자를 반환

매치되어진 원소들의 갯수를 반환한다.

```
<html>
<head>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script>
  $(document).ready(function(){
   $(document.body).click(function () {
     $(document.body).append($("<div>"));
     var n = $("div").size();
     $("span").text("There are " + n + " divs." +
                    "Click to add more.");
   }).click(); // trigger the click to start
```

● size( ) Returns: Number(계속)

```
 });
  </script>
  <style>
  body { cursor:pointer; }
  div { width:50px; height:30px; margin:5px; float:left;
       background:blue; }
  span { color:red; }
  </style>
</head>
<body>
  <span></span>
  <div></div>
</body>
</html>
```

● **get( ) Returns: Array<Element>**

get( ), 실행 후 원소 배열 반환

매치되는 모든 문서 원소들을 배열로 반환한다.

```
<html>
<head>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
  <script>
  $(document).ready(function(){
   function disp(divs) {
     var a = [];
     for (var i = 0; i < divs.length; i++) {
       a.push(divs[i].innerHTML);
     }
     $("span").text(a.join(" "));
   }
```

● get( ) Returns: Array<Element> (계속)

```
disp( $("div").get().reverse() );
  });
  </script>
  <style>
  span { color:red; }
  </style>
</head>
<body>
  Reversed - <span></span>
  <div>One</div>
  <div>Two</div>
  <div>Three</div>
</body>
</html>
```