

AJAX

Asynchronous Javascript And XML

AJAX?

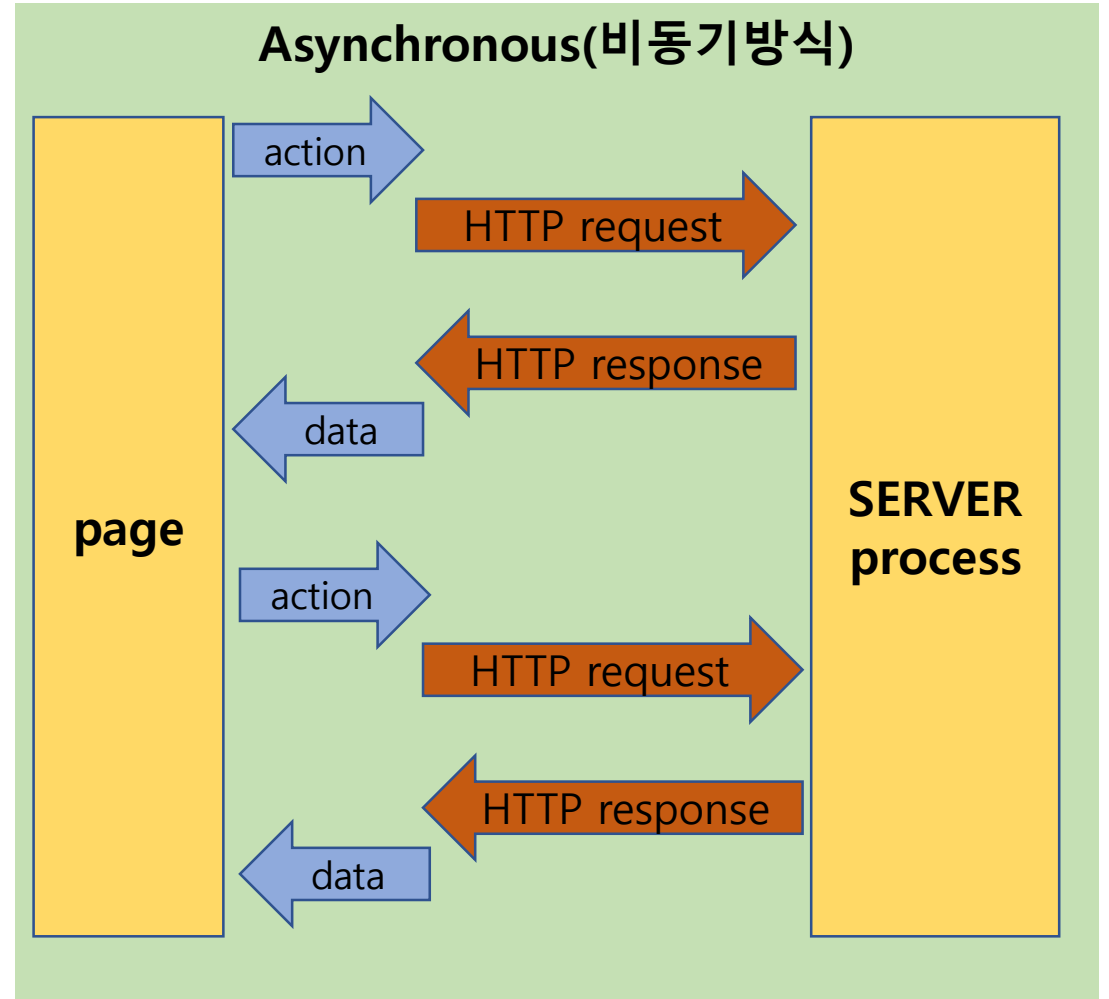
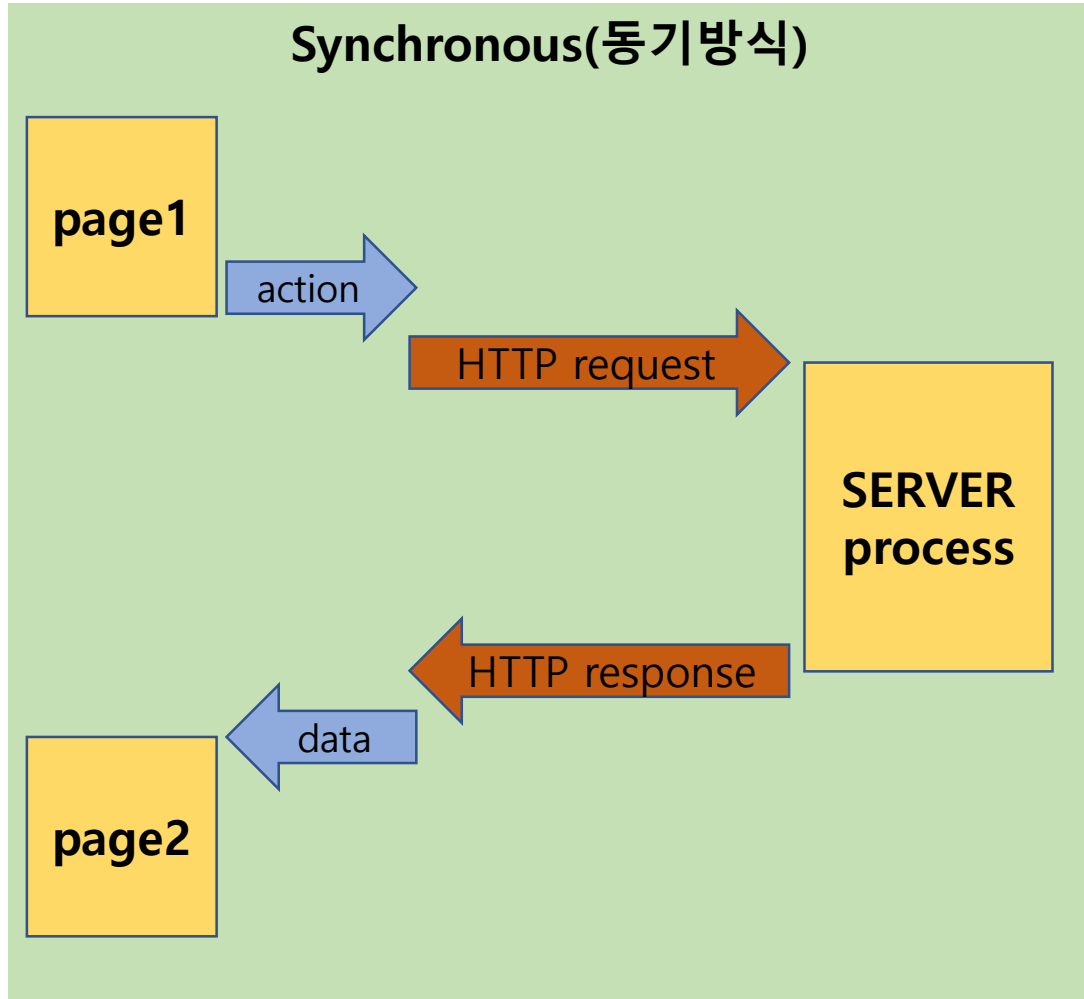
Asynchronous Javascript And Xml

비동기식 통신기능

전체 페이지를 새로 변경하지 않고
자바스크립트를 이용하여
통신을 진행하고 그 결과를 페이지에
표현할 수 있는 방법

**전체 페이지를 새로 고치지 않고
데이터를 불러오는 방식**

Asynchronous(비동기)



전체 리소스를 다시 전송하는 것이 아니라
필요한 부분만 전송하므로 낭비를 막을 수 있다.

AJAX와 연관된 기술

AJAX는 단독으로 사용되지 않으며

HTML

DOM

JavaScript

HttpRequest

JSP

DataBase

등과 연결되어 이용된다.

DataBase와의 통신도 진행할 수 있다.

XMLHttpRequest

JSP에서는 request의 정보를 JAVA Object가 실행해 주지만
브라우저 만으로 request의 정보를
사용하기 위해서는 XMLHttpRequest가 필요하다.

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>XMLHttpRequest</title>
8   </head>
9   <body>
10    <script>
11      //XMLHttpRequest : 과거 인터넷 브라우저에서 ActiveX형식으로 제공된 내용
12      //모든 브라우저마다 작동방식이 조금씩 다른 문제가 존재하고 있다.
13      //W3C표준이 아니므로 IE에서는 ActiveX Component형식으로 구현되었고
14      //다른 브라우저에서는 JavaScript Object 방식으로 구현되었다.
15      //현재는 자바스크립트를 통해서 XHR에 접근할 수 있다.
16
17      var XHR;
18      function createXMLHttpRequest(){
19        if(window.ActiveXObject){
20          XHR=new ActiveXObject("Microsoft.XMLHTTP");
21        }else if(window.XMLHttpRequest){
22          XHR=new XMLHttpRequest();
23        }
24      }
25      createXMLHttpRequest();
26      console.log(XHR);
27    </script>
28  </body>
29 </html>
```

```
XMLHttpRequest {onreadystatechange: null, readyState: 0, timeout: 0, withCredentials: false, upload: XMLHttpRequestUpload, ...}
  s: false, upload: XMLHttpRequestUpload, ...
  onabort: null
  onerror: null
  onload: null
  onloadend: null
  onloadstart: null
  onprogress: null
  onreadystatechange: null
  ontimeout: null
  readyState: 0
  response: ""
  responseText: ""
  responseType: ""
  responseURL: ""
  responseXML: null
  status: 0
  statusText: ""
  timeout: 0
  upload: XMLHttpRequestUpload {onloadstart: null, onprogress: null, onabort: null, ...}
  withCredentials: false
  [[Prototype]]: XMLHttpRequest
```

자바스크립트이므로 F12를 이용하여 확인

XMLHttpRequest 패러미터

XMLHttpRequest 정보를 확인하기 위한 변수들

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>브라우저 에서의 request 처리</title>
8   </head>
9   <body>
10    <form action="#">
11      <input type="button" value="테스트!" onclick="startRequest()">
12    </form>
13    <script>
60 </body>
61 </html>
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 이런 데이터가 들어 있습니다.
```

```
14 //XMLHttpRequest 객체를 가져옴
15 var XHR;
16 function createXMLHttpRequest(){
17   if(window.ActiveXObject){
18     XHR=new ActiveXObject("Microsoft.XMLHTTP");
19   }else if(window.XMLHttpRequest){
20     XHR=new XMLHttpRequest();
21   }
22 }
23 //XMLHttpRequest.open(method, url, asynch, user, pw);
24 //해당 경로로 request를 전달하는 기능
25 //XMLHttpRequest.send();
26 //실질적으로 데이터를 서버로 전송하는 부분, get방식이라면 null을 보내고
27 //post방식이라면 데이터를 전송한다.
28 //XMLHttpRequest 속성값
29 //XMLHttpRequest.onreadystatechange; 함수를 저장하는 부분.
30 //XMLHttpRequest.readyState; 요청의 상태를 확인
31 // 0 : uninitialized, 초기화 되지 않은 상태
32 // 1 : loading, 읽고있는 상태
33 // 2 : loaded, 모든 정보를 읽은 상태
34 // 3 : interactive, 읽은 정보를 가져오는 상태
35 // 4 : complete, 모든 요청이 완료된 상태
36 //XMLHttpRequest.responseText; 응답받은 데이터를 String으로 표현
37 //XMLHttpRequest.responseXML; 응답받은 데이터를 XML로 표현
38
39 function startRequest(){
40   createXMLHttpRequest();
41   XHR.onreadystatechange=handleStateChange;
42   console.log("request 전 : "+XHR.readyState);
43   XHR.open("GET", "data.xml", true);
44   console.log("request 후 : "+XHR.readyState);
45   XHR.send(null);
46 }
47
48 function handleStateChange(){
49   if(XHR.readyState==4){
50     if(XHR.status==200){
51       alert("다음의 정보 전달 : "+XHR.responseText);
52     }
53   }
54 }
```

<script>내부에 작성할 것

responseText의 사용

```
1<table>
2  <tr>
3    <th>이름</th>
4    <th>과목</th>
5    <th>시간</th>
6  </tr>
7  <tr>
8    <td>이영준</td>
9    <td>jsp</td>
10   <td>09:00</td>
11 </tr>
12 <tr>
13   <td>고희선</td>
14   <td>android</td>
15   <td>14:00</td>
16 </tr>
17 <tr>
18   <td>이진선</td>
19   <td>python</td>
20   <td>16:30</td>
21 </tr>
22 </table>
```

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>response</title>
8   </head>
9   <body>
10    <form action="#">
11      <input type="button" value="테스트!" onclick="startRequest()">
12    </form>
13    <div id="results"></div>
14    <script>
39   </body>
40 </html>
```

```
<script>
  var XHR;
  function createXMLHttpRequest(){
    if(window.ActiveXObject){
      XHR=new ActiveXObject("Microsoft.XMLHTTP");
    }else if(window.XMLHttpRequest){
      XHR=new XMLHttpRequest();
    }
  }

  function startRequest(){
    createXMLHttpRequest();
    XHR.onreadystatechange=handleStateChange;
    XHR.open("GET", "data.xml", true);
    XHR.send(null);
  }

  function handleStateChange(){
    if(XHR.readyState==4){
      if(XHR.status==200){
        document.getElementById("results").innerHTML=XHR.responseText;
      }
    }
  }
</script>
```

테스트!

테스트!

이름	과목	시간
이영준	jsp	09:00
고희선	android	14:00
이진선	python	16:30

<script>내부에 작성할 것

XML DOM 구조의 연결

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <class>
3   <iot>
4     <subject>iot 입문</subject>
5     <subject>aduno</subject>
6     <subject>python</subject>
7   </iot>
8   <web>
9     <subject>HTML</subject>
10    <subject>JavaScript</subject>
11    <subject>JSP</subject>
12  </web>
13  <app>
14    <subject>java</subject>
15    <subject>android</subject>
16    <subject>mysql</subject>
17  </app>
18 </class>
```

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>response</title>
8   </head>
9   <body>
10    <form action="#">
11      <input type="button" value="iot 과목확인" onclick="startRequest('iot')">
12      <input type="button" value="web 과목확인" onclick="startRequest('web')">
13      <input type="button" value="app 과목확인" onclick="startRequest('app')">
14    </form>
15    <div id="results"></div>
16    <script>
84
85 </html>
```

```
var XHR;
var type=""; //내용을 변경하기 위한 변수
function createXMLHttpRequest(){
  if(window.ActiveXObject){
    XHR=new ActiveXObject("Microsoft.XMLHTTP");
  }else if(window.XMLHttpRequest){
    XHR=new XMLHttpRequest();
  }
}

function startRequest(requestType){
  type=requestType; //해당 타입을 변수에 저장
  createXMLHttpRequest();
  XHR.onreadystatechange=handleStateChange;
  XHR.open("GET", "parse.xml", true);
  XHR.send(null);
}

function handleStateChange(){
  if(XHR.readyState==4){
    if(XHR.status==200){
      if(type=="iot"){
        listIOT();
      }else if(type=="web"){
        listWEB();
      }else if(type=="app"){
        listAPP();
      }
    }
  }
}

function listIOT(){
  var XMLDOC=XHR.responseXML;
  var iot=XMLDOC.getElementsByTagName("iot")[0];
  var out="iot과목";
  var subject=iot.getElementsByTagName("subject");

  output(out, subject);
}

function listWEB(){
  var XMLDOC=XHR.responseXML;
  var iot=XMLDOC.getElementsByTagName("web")[0];
  var out="web과목";
  var subject=iot.getElementsByTagName("subject");

  output(out, subject);
}

function listAPP(){
  var XMLDOC=XHR.responseXML;
  var iot=XMLDOC.getElementsByTagName("app")[0];
  var out="app과목";
  var subject=iot.getElementsByTagName("subject");

  output(out, subject);
}

function output(title, subject){
  var out=title;
  var name=null;
  for(var i=0; i<subject.length; i++){
    name=subject[i];
    out=out+"\n"+name.childNodes[0].nodeValue;
  }
  alert(out);
}
```

localhost:8080 내용:

iot과목
iot 입문
aduno
python

확인

localhost:8080 내용:

web과목
HTML
JavaScript
JSP

확인

localhost:8080 내용:

app과목
java
android
mysql

확인

<script>내부에 작성할 것

XML DOM 구조의 연결-전체 조회

```
<form action="#">
  <input type="button" value="iot 과목확인" onclick="startRequest('iot')">
  <input type="button" value="web 과목확인" onclick="startRequest('web')">
  <input type="button" value="app 과목확인" onclick="startRequest('app')"><hr>
  <input type="button" value="전체조회" onclick="startRequest('all')">
</form>
```

iot 과목확인

web 과목확인

app 과목확인

전체조회

```
function handleStateChange(){
  if(XHR.readyState==4){
    if(XHR.status==200){
      if(type=="iot"){
        listIOT();
      }else if(type=="web"){
        listWEB();
      }else if(type=="app"){
        listAPP();
      }else if(type=="all"){
        listALL();
      }
    }
  }
}
```

```
function listALL(){
  var XMLDOC=XHR.responseXML;
  var cla=XMLDOC.getElementsByTagName("class")[0];
  var out="전체과목";
  var subject=cla.getElementsByTagName("subject");

  output(out, subject);
}
```

iot 과목확인

전체조회

localhost:8080 내용:

전체과목
iot 입문
aduno
python
HTML
JavaScript
JSP
java
android
mural

확인

이전의 내용을 조금만 수정

Dynamic XML parse

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <products>
3   <product>
4     <name>노트북</name>
5     <price>1000000</price>
6     <comments>윈도우10, i7, 1tb</comments>
7   </product>
8   <product>
9     <name>스마트폰</name>
10    <price>2000000</price>
11    <comments>갤럭시폴드3, SD888, 512gb</comments>
12  </product>
13  <product>
14    <name>냉장고</name>
15    <price>3000000</price>
16    <comments>비스포크 양문형, 800L</comments>
17  </product>
18 </products>
```

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>동적 데이터통신</title>
8   </head>
9   <body>
10    <h1>동적 웹 개발 구현</h1>
11    <form action="#">
12      가격대를 선택해 주세요
13      <select>
14        <option value="500000">500000</option>
15        <option value="1000000">1000000</option>
16        <option value="1500000">1500000</option>
17      </select>
18      예서
19      <select>
20        <option value="1000000">1000000</option>
21        <option value="2000000">2000000</option>
22        <option value="3000000">3000000</option>
23      </select>
24      까지
25      <input type="button" value="조회" onclick="Search()">
26    </form>
27    <span id="header"></span>
28    <table id="resultsTable">
29      <tbody id="resultsBody"></tbody>
30    </table>
31    <script>
32
33  </body>
34 </html>
```

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

function Search(){
  createXMLHttpRequest();
  XHR.onreadystatechange=handleStateChange;
  XHR.open("GET", "dynamic.xml", true);
  XHR.send(null);
}

function handleStateChange(){
  if(XHR.readyState==4){
    if(XHR.status==200){
      clearResult();
      parseResult();
    }
  }
}
```

```
function clearResult(){
  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]);
  }
}

function parseResult(){
  var results=XHR.responseXML;
  var product=null;
  var name="";
  var price="";
  var comments="";
  var products=results.getElementsByTagName("product");
  for(var i=0; i<products.length; i++){
    product=products[i];
    name=product.getElementsByTagName("name")[0].firstChild.nodeValue;
    price=product.getElementsByTagName("price")[0].firstChild.nodeValue;
    comments=product.getElementsByTagName("comments")[0].firstChild.nodeValue;

    addTableRow(name, price, comments);
  }
  var header=document.createElement("h2");
  var headerText=document.createTextNode("검색결과 : ");
  header.appendChild(headerText);
  document.getElementById("header").appendChild(header);
  document.getElementById("resultsTable").setAttribute("border", "1");
}
```

```
function addTableRow(address, price, comments){
  var min=document.getElementsByTagName("select")[0].value;
  var max=document.getElementsByTagName("select")[1].value;
  var int_price=parseInt(price);

  console.log("최소값 : "+min+", "+(min<=int_price));
  console.log("최대값 : "+max+", "+(int_price<=max));
  console.log("검색가격 : "+int_price);
  if(min<=int_price && int_price<=max){
    var row=document.createElement("tr");
    var cell=createCellWithText(address);
    row.appendChild(cell);

    cell=createCellWithText(price);
    row.appendChild(cell);

    cell=createCellWithText(comments);
    row.appendChild(cell);

    document.getElementById("resultsBody").appendChild(row);
  }
}

function createCellWithText(text){
  var cell=document.createElement("td");
  var textNode=document.createTextNode(text);
  cell.appendChild(textNode);

  return cell;
}
```

화면의 갱신이 없이 데이터를 전송받을 수 있다.

XMLHttpRequest 통신

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>패러미터의 전송</title>
8   </head>
9   <body>
10    <h1>패러미터를 입력하여 전송</h1>
11    첫째 패러미터 : <input type="text" id="first"><br>
12    두번째 패러미터 : <input type="text" id="second"><br>
13    마지막 패러미터 : <input type="text" id="last"><br>
14    <form action="#">
15      <input type="button" value="GET 방식 전송" onclick="request_doGet()"><br>
16      <input type="button" value="POST 방식 전송" onclick="request_doPost()">
17    </form>
18    <hr>
19    <h2>서버 response</h2>
20    <div id="serverResponse"></div>
21    <script>
22
23
24
25
26  </body>
27 </html>
```

```
var XHR;
var type=""; //내용을 변경하기 위한 변수
function createXMLHttpRequest(){
  if(window.ActiveXObject){
    XHR=new ActiveXObject("Microsoft.XMLHTTP");
  }else if(window.XMLHttpRequest){
    XHR=new XMLHttpRequest();
  }
}

function createString(){
  var first=document.getElementById("first").value;
  var second=document.getElementById("second").value;
  var last=document.getElementById("last").value;

  var dataString="first="+first+"&second="+second+"&last="+last;
  return dataString
}

function request_doGet(){
  createXMLHttpRequest();
  var dataString="Test?";
  dataString+=createString();
  XHR.onreadystatechange=handleStateChange;
  XHR.open("GET", dataString, true);
  XHR.send(null);
}

function request_doPost(){
  createXMLHttpRequest();
  var url="Test";
  var dataString=createString();

  XHR.onreadystatechange=handleStateChange;
  XHR.open("POST", url, true);
  XHR.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
  XHR.send(dataString);
}
```

```
function handleStateChange(){
  if(XHR.readyState==4){
    if(XHR.status==200){
      parseResult();
    }
  }
}

function parseResult(){
  var div=document.getElementById("serverResponse");
  if(div.hasChildNodes()){
    div.removeChild(div.childNodes[0]);
  }
  var text=document.createTextNode(XHR.responseText);
  div.appendChild(text);
}
```

패러미터를 화면의 갱신없이 서버로 전송

XMLHttpRequest 통신 서블릿

```
1 package servlet;
2
3 import java.io.*;
4 import javax.servlet.*;
5 import javax.servlet.annotation.*;
6 import javax.servlet.http.*;
7
8 @WebServlet("/Test")
9 public class testServlet extends HttpServlet {
10     private static final long serialVersionUID = 1L;
11
12     protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
13         process(request, response, "GET");
14     }
15
16     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
17         process(request, response, "POST");
18     }
19     protected void process(HttpServletRequest request, HttpServletResponse response, String method) throws ServletException, IOException{
20         request.setCharacterEncoding("utf-8");
21         response.setCharacterEncoding("utf-8");
22         response.setContentType("text/xml");
23         String first=request.getParameter("first");
24         String second=request.getParameter("second");
25         String last=request.getParameter("last");
26         String responseText="첫번째 데이터:"+first+", 두번째 데이터:"+second+", 마지막 데이터:"+last+", 방식:"+method;
27         PrintWriter out=response.getWriter();
28         out.println(responseText);
29         out.close();
30     }
31 }
```

서블릿을 통하여 결과를 돌려받아 표현함

XMLHttpRequest 통신 결과

패러미터를 입력하여 전송

첫번째 패러미터 :

두번째 패러미터 :

마지막 패러미터 :

서버 response

첫번째 데이터:GET방식으로, 두번째 데이터:데이터를 , 마지막 데이터:전송합니다, 방식:GET

패러미터를 입력하여 전송

첫번째 패러미터 :

두번째 패러미터 :

마지막 패러미터 :

서버 response

첫번째 데이터:POST방식은, 두번째 데이터:다음처럼, 마지막 데이터:동작합니다, 방식:POST

패러미터를 입력하여 전송

첫번째 패러미터 :

두번째 패러미터 :

마지막 패러미터 :

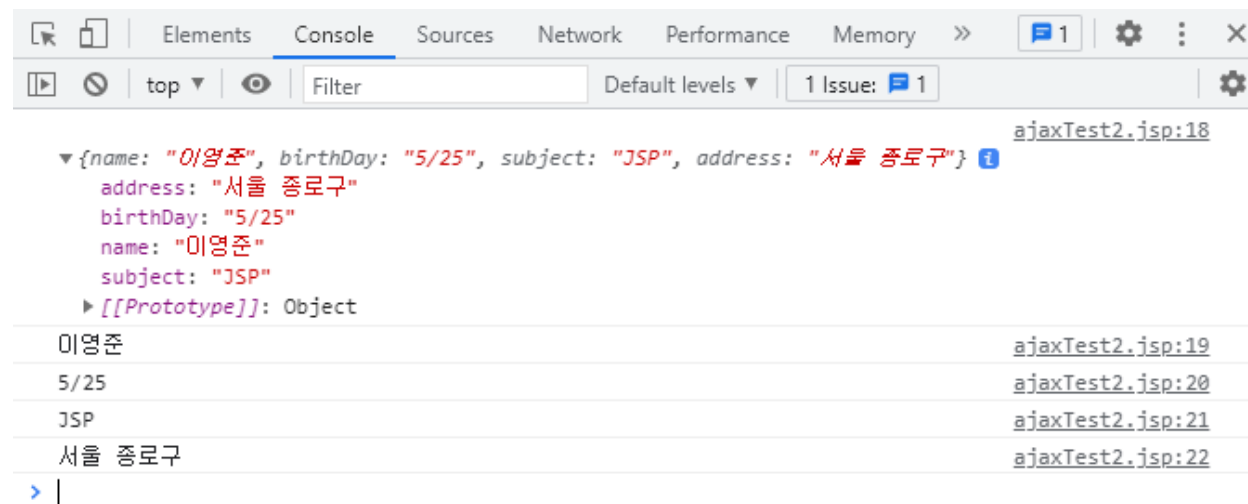
서버 response

서블릿을 통하여 결과를 돌려받아 표현함

JSON의 사용

JavaScript Notation

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>XML 대신 JSON의 사용</title>
8   </head>
9   <body>
10    <script>
11      //JSON : JavaScript Object Notation
12      //언어 독립적인 데이터결합으로 Map타입의 데이터를 담은 배열같은 형식으로 데이터를 보존
13      var student={
14        "name":"이영준",
15        "birthDay":"5/25",
16        "subject":"JSP",
17        "address":"서울 종로구"
18      }
19      console.log(student);
20      console.log(student.name);
21      console.log(student.birthDay);
22      console.log(student.subject);
23      console.log(student.address);
24    </script>
25  </body>
26 </html>
```



XML과 더불어 가장 광범위하게 사용되는 형식

JavaScript에서의 객체사용

JavaScript Object

```
function Dog(race, name, year, color){
    this.race=race;
    this.name=name;
    this.year=year;
    this.color=color;
    this.show=function() {
        console.log(this.name);
    };
}
var obj=new Dog("말티즈", "폴이", "15", "흰색");
console.log(obj);
console.log(obj.name);
var json_obj=JSON.stringify(obj);
console.log(json_obj);
console.log(json_obj.name); //문자열로 변경되어 undefined가 발생
console.log(typeof(obj));
console.log(typeof(json_obj));
var normal_obj=JSON.parse(json_obj);
console.log(normal_obj); //원상복구 된 모양
console.log(normal_obj.race); //원상복구 된 모양
```

```
var arr=[1,9,15,18,22];
console.log(arr);
console.log(arr[2]); //배열이기 때문에 배열의 번호로 인식된다.
var json_arr=JSON.stringify(arr);
console.log(json_arr);
console.log(json_arr[2]); //문자열로 변경되어 버려 문자의 번호로 인식된다.
```

▶ Dog {race: "말티즈", name: "폴이", year: "15", color: "흰색"}	ajaxTest2.jsp:31
폴이	ajaxTest2.jsp:32
{"race":"말티즈","name":"폴이","year":"15","color":"흰색"}	ajaxTest2.jsp:34
undefined	ajaxTest2.jsp:35
object	ajaxTest2.jsp:36
string	ajaxTest2.jsp:37
▶ {race: "말티즈", name: "폴이", year: "15", color: "흰색"}	ajaxTest2.jsp:39
말티즈	ajaxTest2.jsp:40
▶ (5) [1, 9, 15, 18, 22]	ajaxTest2.jsp:44
15	ajaxTest2.jsp:45
[1,9,15,18,22]	ajaxTest2.jsp:47
,	ajaxTest2.jsp:48
>	

Class와 같은 방식으로 사용 가능하다.

JSON데이터의 서버전송

```
1 <%@ page language="java" contentType="text/html; charset=utf-8"
2   pageEncoding="utf-8"%>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta charset="utf-8">
7     <title>JSON방식의 데이터 전송</title>
8   </head>
9   <body>
10    <form action="#">
11      <input type="button" value="눌러서 데이터를 서버로 전송" onclick="doJSON()">
12    </form>
13    <h2>server response : </h2>
14    <div id="serverResponse"></div>
15    <script>
61   </body>
62 </html>
```

```
<script>
var XHR;
function createXMLHttpRequest(){
  if(window.ActiveXObject){
    XHR=new ActiveXObject("Microsoft.XMLHTTP");
  }else if(window.XMLHttpRequest){
    XHR=new XMLHttpRequest();
  }
}
function Dog(race, name, year, color){
  this.race=race;
  this.name=name;
  this.year=year;
  this.color=color;
}
function getDogObject(){
  return new Dog("리트리버", "폴이", "11", "금색");
}
function doJSON(){
  var dog=getDogObject();

  var json_dog=JSON.stringify(dog);
  alert("stringify된 dog데이터 : "+json_dog);

  var url="JSONExample?timeStamp="+new Date().getTime();
  createXMLHttpRequest();
  XHR.open("POST", url, true);
  XHR.onreadystatechange=handleStateChange;
  XHR.send(json_dog);
}


function parseResults(){
  var responseDiv=document.getElementById("serverResponse");
  if(responseDiv.hasChildNodes()){
    responseDiv.removeChild(responseDiv.childNodes[0]);
  }
  var responseText=document.createTextNode(XHR.responseText);
  responseDiv.appendChild(responseText);
}

function handleStateChange(){
  if(XHR.readyState==4){
    if(XHR.status==200){
      parseResults();
    }
  }
}
}
</script>
```

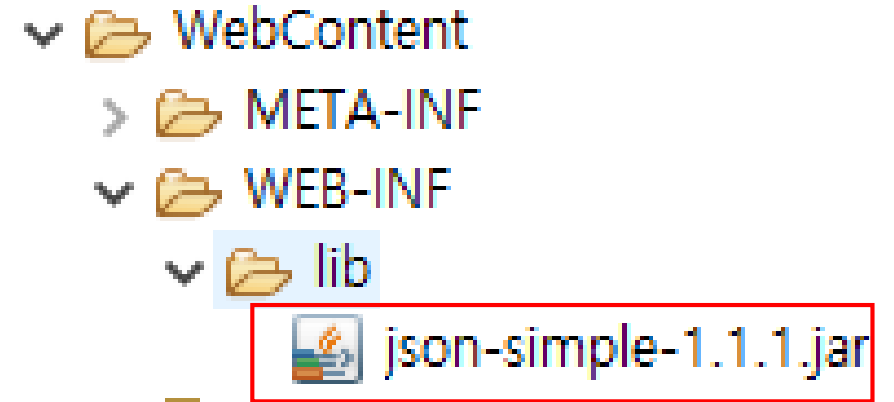
이후 동작해줄 서블릿을 제작해야 한다.

라이브러리 사용



Project	 json-simple
Source	
Issues	
Wikis	
Downloads	

File	Summary + Labels	Uploaded	Size
json-simple-1.1.1.jar	1.1.1 binary, with maven and OSGi support Featured OpSys-All Type-Archive	Feb 19, 2012	23.18KB
json-simple-1.1-bundle.jar	1.1 maven bundle Type-Package OpSys-All	Jul 29, 2009	30.19KB
json_simple.jar	Initial version of json_simple.jar Type-Package OpSys-All	Feb 16, 2009	14.1KB
json_simple-1.1-all.zip	1.1 source, document and jar binary Featured Type-Source OpSys-All	Feb 1, 2009	43.76KB
json_simple-1.1.jar	1.1 jar binary Featured Type-Package OpSys-All	Feb 1, 2009	15.67KB
json_simple-1.0.2-all.zip	source, document and jar binary Type-Source OpSys-All	Jan 10, 2009	31.14KB
json_simple-1.0.2.jar	jar binary Type-Package OpSys-All	Jan 10, 2009	9.51KB
json_simple.zip	Initial version of JSON.simple source and binary	Jan 7, 2009	49.71KB
json_simple-1.0.1-all.zip	source, document and jar binary Type-Source OpSys-All Deprecated	Nov 25, 2008	33.43KB
json_simple-1.0.1.jar	jar binary Type-Package OpSys-All Deprecated	Nov 25, 2008	10.52KB



구글사의 json-simple 라이브러리를 사용
<https://code.google.com/archive/p/json-simple/>

서블릿 제작

```
1 package servlet;
2
3 import java.io.*;
4 import javax.servlet.*;
5 import javax.servlet.annotation.*;
6 import javax.servlet.http.*;
7
8 import java.util.*;
9 import org.json.*;
10 import org.json.simple.*;
11 import org.json.simple.parser.*;
12
13 @WebServlet("/JSONExample")
14 public class JSONExample extends HttpServlet {
15     private static final long serialVersionUID = 1L;
16     private String readJSON(HttpServletRequest request) {
17         StringBuffer json=new StringBuffer();
18         String line=null;
19         try {
20             BufferedReader reader=request.getReader();
21             while((line=reader.readLine())!=null) {
22                 json.append(line);
23             }
24         } catch (Exception e) {
25             System.out.println("json 파일을 읽던 중 오류발생 : "+e);
26         }
27         return json.toString();
28     }
29     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
30         String json=readJSON(request);
31         System.out.println(json);//String으로 데이터가 전송되어 들어오는 것을 확인
32         Object obj=null;
33         JSONObject jobj=null;
34         try {
35             JSONParser parser=new JSONParser();
36             obj=parser.parse(json);
37             jobj=(JSONObject)obj;
38             System.out.println(jobj);//JSON방식으로 변경이 완료된 데이터확인
39         } catch (Exception e) {
40             System.out.println("JSON변환 중 오류 발생 : "+e);
41         }
42         String responseText="나이 : "+jobj.get("year")+", 종류 : "+jobj.get("race")+", 이름 : "+jobj.get("name");
43         response.setContentType("text/xml");
44         response.setCharacterEncoding("utf-8");
45         response.getWriter().print(responseText);
46     }
47 }
48 }
```

```
{"race": "리트리버", "name": "똥이", "year": "11", "color": "금색"}
{"race": "리트리버", "color": "금색", "year": "11", "name": "똥이"}
```

눌러서 데이터를 서버로 전송

server response :

localhost:8080 내용:

stringify된 dog데이터 : {"race": "리트리버", "name": "똥이", "year": "11", "color": "금색"}

확인

눌러서 데이터를 서버로 전송

server response :

나이 : 11, 종류 : 리트리버, 이름 : 똥이

한 개의 jsp 파일을 이용하여 request사용