

Class[AJAX & JSON]

▼ AJAX & JSON

JSP 는 사용자의 요청에 대한 결과 페이지를 생성하기 위해 대부분의 역할을 수행한다.

이런 방식은 서버 중심의 처리 방식으로 볼 수 있다.

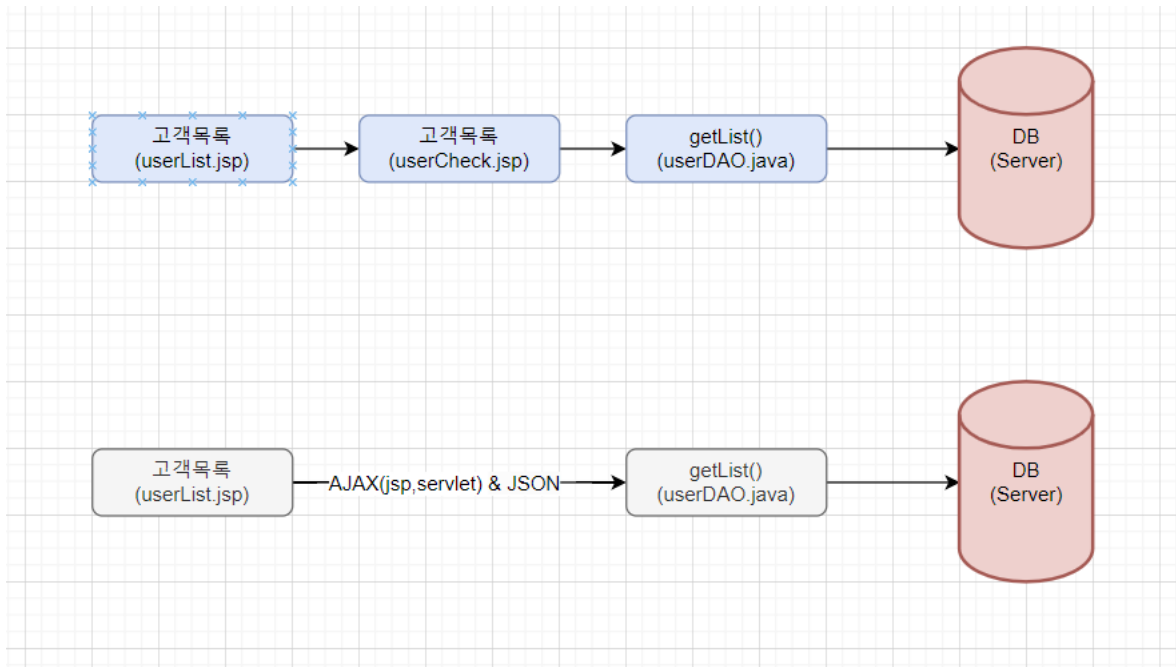
그러나 이런 경우에 사용자가 많아지게 되면 서버에 로드가 기하급수적으로 커지는 문제점이 있다.

이러한 문제점은 페이스북, 트위터 등의 SNS가 등장하면서 더욱 부각되었다.

이들테면 SNS 서버에는 동시 사용자가 10 ~ 30만명 등이 넘어가는 경우가 비일비재하다.

따라서 서버의 부하를 줄이기 위해 서버가 하던 작업을 클라이언트로 넘기는 다양한 기술들이 등장했다.

그중에 AJAX와 JSON이 핵심적인 역할을 수행하게 된다.



▼ AJAX

AJAX(Asynchronous JAVA and XML)

JAVA나 XML형식의 데이터를 **비동기식**으로 전송하기 위한 기술

AJAX는 URL을 동일하게 유지하면서 내부적으로 여러개의 HTTP 요청과 응답을 전송할수 있도록 지원한다. 이를 통해 웹 브라우저에서 페이지를 고치지 않고도 여러개의 HTTP 요청과 응답을 가능하게 한다.

jquery 설정

The screenshot shows the jQuery CDN website at `releases.jquery.com`. The page features a navigation bar with links to jQuery Core, jQuery UI, jQuery Mobile, jQuery Color, QUnit, and PEP. The main heading is "jQuery CDN - Latest Stable Versions", powered by STACKPATH. A modal titled "Code Integration" is displayed, showing the following script tag:

```
<script
  src="https://code.jquery.com/jquery-3.6.3.min.js"
  integrity="sha256-pvPw+upLPUjgMXy0G+800xUf+/Im1MZjXxxg0cBQBxU="
  crossorigin="anonymous"></script>
```

Below the code, a note explains that the `integrity` and `crossorigin` attributes are used for Subresource Integrity (SRI) checking, ensuring resources have not been tampered with. It recommends SRI as a best practice when loading libraries from third-party sources, with a link to srihash.org for more information.

The screenshot shows a code editor with the file `*userList.jsp` open. The code is as follows:

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"
3   import="user.*, java.util.*"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title>userList</title>
9 </head>
10 <body>
11   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bc
12   <script src="https://code.jquery.com/jquery-3.6.3.min.js" integrity="sha256-
13
14
15
```

기존에 사용하던 form 태그는 더 이상 사용되지 않는다.

form 태그는 필연적으로 처리 페이지로 전달을 하기 위해서 화면 전환이 이루어지게 된다.

따라서 jquery 사용시에는 form 태그를 전혀 사용하지 않는다.

```
<form class="d-flex" role="search">
  <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
  <button class="btn btn-outline-primary" type="submit">Search</button>
</form>
```

```
<input class="form-control me-2" type="search" placeholder="Search" aria-l
<button class="btn btn-outline-primary" type="submit">Search</button>

</div>
```

```

</thead>
<tbody class="table-active" id="ajaxTable">
</tbody>
</table>

<!-- <div class="container text-center" id="ajaxTable"> -->
<!-- <div class="row row-cols-1 row-cols-sm-2 row-cols-md-4"> -->
<%-- <div class="col" style="padding: 10px; border: solid;"><%=user.getUserName() %></div> --%>
<%-- <div class="col" style="padding: 10px; border: solid;"><%=user.getUserAge() %></div> --%>
<%-- <div class="col" style="padding: 10px; border: solid;"><%=user.getUserGender() %></div> --%>
<%-- <div class="col" style="padding: 10px; border: solid;"><%=user.getUserEmail() %></div> --%>
<!-- </div> -->

<!-- </div> -->

<%-- <% } %> --%>

<br><br>

<div class="container themed-container text-center text-light bg-primary" style="padding: 10px; border: solid; border-color: black">

<div class="container themed-container text-center bg-warning" style="padding: 10px; border: solid;">

    <div class="row g-3">

        <div class="col-12">
            <label for="username" class="form-label">Name</label>
            <div class="input-group has-validation">
                <input type="text" class="form-control" id="registerName" placeholder="Name" required>
            </div>
        </div>

        <div class="col-12">
            <label for="address2" class="form-label">Email</label>
            <div class="input-group">
                <input type="text" class="form-control" id="registerEmail" placeholder="aaa@example.com">
                <button type="submit" class="btn btn-secondary">인증하기</button>
            </div>
            <div style="text-align: left;">re</div>
        </div>

        <div class="col-12">
            <label for="address" class="form-label">Gender</label>
            <div><label class="btn btn-primary active">
                <input type="radio" name="registerGender" autocomplete="off" value="male" checked="checked">남자
            </label>
            <label class="btn btn-primary">
                <input type="radio" name="registerGender" autocomplete="off" value="female">여자
            </label></div>
        </div>

        <div class="col-12">
            <label for="email" class="form-label">Age</label>
            <input type="text" class="form-control" id="registerAge" placeholder="20" required>
        </div>

    </div>
</div>
<div class="container themed-container text-center bg-primary" style="padding: 10px; border: solid;"><button class="btn btn-success">

<script type="text/javascript">

    var searchRequest = new XMLHttpRequest();
    var registerRequest = new XMLHttpRequest();

    function searchFunction() {
        searchRequest.open("POST", ".UserSearchServlet?userName=" + encodeURIComponent(document.getElementById("userName").value), true);
        searchRequest.onreadystatechange = searchProcess;
        searchRequest.send(null);
    }

    function searchProcess() {
        var table = document.getElementById("ajaxTable");
        table.innerHTML = "";
        if(searchRequest.readyState == 4 && searchRequest.status == 200) {
            var object = eval('(' + searchRequest.responseText + ')');
            var result = object.result;
            for (var i = 0; i < result.length; i++) { // 유저 데이터 한명 한명의 자료 추출

```

```

        var row = table.insertRow(0);
        for (var j = 0; j < result[i].length; j++) { // 유저 한명의 4개 필드를 하나씩 추출
            var cell = row.insertCell(j);
            cell.innerHTML = result[i][j].value;
        }
    }
}

function registerFunction() {
    registerRequest.open("POST", ". /UserRegisterServlet?userName=" + encodeURIComponent(document.getElementById("registerName").value) +
        "&userAge=" + encodeURIComponent(document.getElementById("registerAge").value) +
        "&userGender=" + encodeURIComponent($('input[name=registerGender]:checked').val()) +
        "&userEmail=" + encodeURIComponent(document.getElementById("registerEmail").value)
        , true);
    registerRequest.onreadystatechange = registerProcess;
    registerRequest.send(null);
}

function registerProcess(){
    if(registerRequest.readyState == 4 && registerRequest.status == 200){
        var result = registerRequest.responseText;
        if(result!=1){ //잘못되었다면
            alert('등록에 실패했습니다. ');
        }
        else{
            var userName=document.getElementById("userName");
            var registerName=document.getElementById("registerName");
            var registerAge=document.getElementById("registerAge");
            var registerEmail=document.getElementById("registerEmail");
            userName.value = "";
            registerName.value = "";
            registerAge.value = "";
            registerEmail.value = "";
            searchFunction();
        }
    }
}

window.onload = function() {
    searchFunction();
}

</script>

</body>
</html>

```

UserSearchServlet.java

```

package jdbc;

import java.io.IOException;
import java.sql.SQLException;
import java.util.ArrayList;

import javax.naming.NamingException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet("/UserSearchServlet")
public class UserSearchServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response){

        try {
            request.setCharacterEncoding("UTF-8");
            response.setContentType("text/html; charset=UTF-8");
            String userName = request.getParameter("userName");

```

```

        response.getWriter().write(getJSON(userName));

    } catch (Exception e) {
        e.printStackTrace();
    }
}

public String getJSON(String userName) {
    if (userName == null) userName = "";

    StringBuffer result = new StringBuffer("");

    result.append("{\"result\":[");
    userDao users = new userDao();
    ArrayList<user> userList = users.search(userName);
    //조회해온 결과 값을 JSON으로 변경, 추후 쉽게 만들어주는 라이브러리 이용 예정

    for (int i = 0; i < userList.size(); i++) {
        result.append("[{\"value\": \"" + userList.get(i).getUserName() + "\",");
        result.append("{\"value\": \"" + userList.get(i).getUserAge() + "\",");
        result.append("{\"value\": \"" + userList.get(i).getUserGender() + "\",");
        result.append("{\"value\": \"" + userList.get(i).getUserEmail() + "\"}],");
    }

    result.append("]");
    return result.toString();
}
}

```

UserDAO.java

```

package jdbc;

import java.sql.*;
import java.util.*;

import util.ConnectionPool;

public class userDao {

    public static ArrayList<user> search(String userName){
        //비슷한 것을 모두 찾기 sql
        String sql = "SELECT * FROM user WHERE userName LIKE ? ";

        ArrayList<user> users = new ArrayList<user>();

        try {

            Connection conn = ConnectionPool.get();

            PreparedStatement pstmt = conn.prepareStatement(sql);

            pstmt.setString(1, "%" + userName + "%");

            ResultSet rs = pstmt.executeQuery();

            while (rs.next()) {
                users.add(new user(rs.getString(1),
                                    rs.getString(2),
                                    rs.getString(3),
                                    rs.getString(4)));
            }

            conn.close();
        } catch (Exception e) {
            e.printStackTrace();
        }

        return users;
    }
}

```

▼ JSON

JSON(JavaScript Object Notation)

자바스크립트에서 객체를 표현하기 위한 형식.

XML과 아주 유사하지만 xml에 비해 쉬운 문법을 사용하고 처리속도도 빠르다는 장점이 있다.

XML	JSON
<pre><empinfo> <employees> <employee> <name>James Kirk</name> <age>40</age> </employee> <employee> <name>Jean-Luc Picard</name> <age>45</age> </employee> <employee> <name>Wesley Crusher</name> <age>27</age> </employee> </employees> </empinfo></pre>	<pre>{ "empinfo" : { "employees" : [{ "name" : "James Kirk", "age" : 40, }, { "name" : "Jean-Luc Picard", "age" : 45, }, { "name" : "Wesley Crusher", "age" : 27, }] } }</pre>

따라서 모바일 앱 등의 구현에 있어서 json이 점점 더 많이 사용되고 있다.

자바 스크립트에서는 객체를 중괄호로 정의한다.

객체는 이름 - 값의 쌍 형태로 정의된 속성을 하나 이상 포함 할 수 있고 각각의 속성은 쉼표로 구분된다. 이때 이름은 스트링 형식으로 표현되고 값은 임의의 자료형으로 정의 될 수 있다.

json Example

json

```
{
  id: "aaa@naver.com",
  pass: "0000",
  name: "kim"
}
```

배열 형식으로도 표현 가능하다.

```
{
  0: "aaa@naver.com",
}
```

```
1: "0000",  
2: "kim"  
}
```

배열 형태

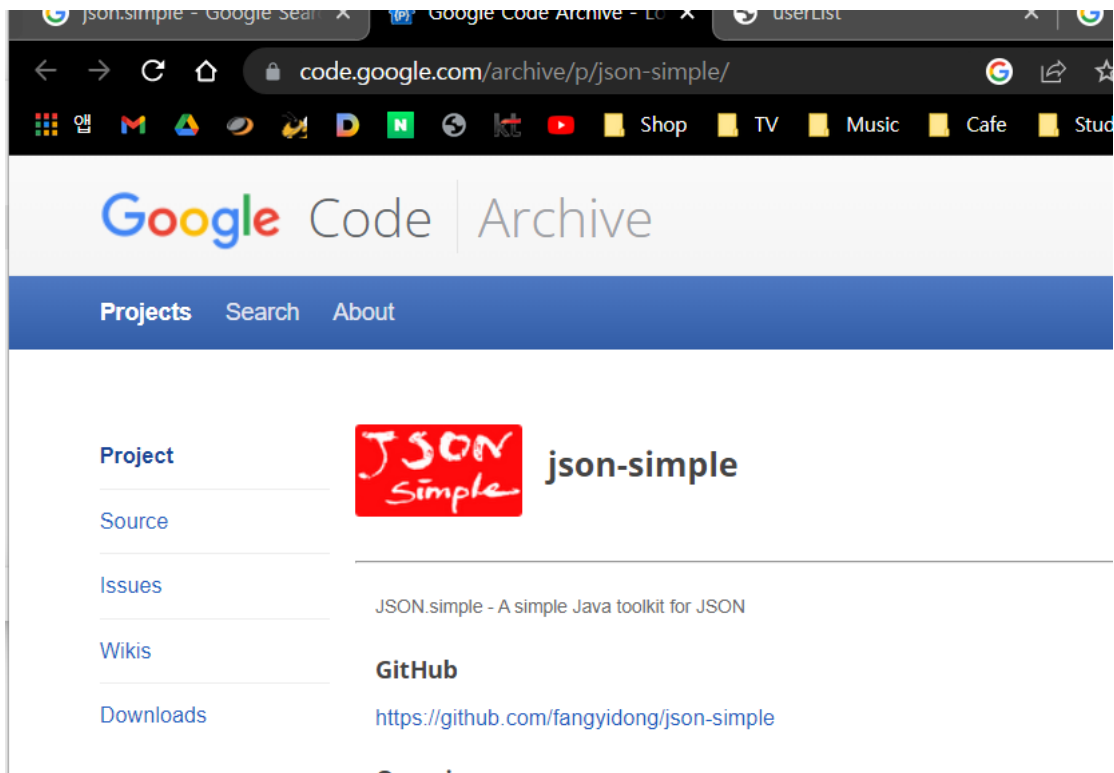
```
{"aaa@naver.com", "0000", "kim"}
```

배열 형태2

```
[  
  {id: "aaa@naver.com", pass: "0000", name: "kim"},  
  {id: "bbb@naver.com", pass: "1111", name: "han"},  
  {id: "abc@naver.com", pass: "1234", name: "lee"}  
]
```

이러한 json 배열을 클라이언트로 전송하여 html로 출력하게 된다.


JSON 형식으로 간단하게 바꾸어주는 라이브러리



code.google.com/archive/p/json-simple/downloads

Google Code Archive

Projects Search About

Project  json-simple

Source

Issues

Wikis

Downloads

File	Summary + Labels
json-simple-1.1.1.jar	1.1.1 binary, with maven and OSGi support Type-Archive

WEB-INF

lib

json-simple-1.1.1.jar

mysql-connector-j-8.0.31.jar