**10/29(화)**

<강의목차>

* AJAX
* 도서검색 프로그램 완성(JSON Parsing)
* Open API활용
* R에서 Open API

1. **JSON(JavaScript Object Notation ->표현법)**

: JSON(JavaScript Object Notation)은 속성-값 쌍(attribute–value pairs and array data types,

(or any other serializable value)) 또는 "키-값 쌍"으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷이다.

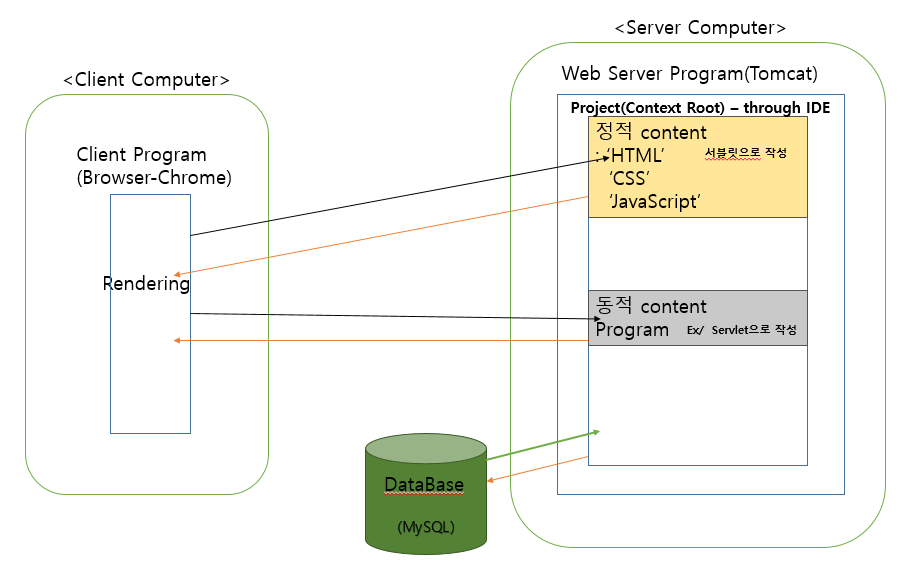
(도서검색시스템 이어서)

http://localhost :8080/bookSearch/search?keyword=java

? 이후의 문자열 → Query String 이라고 함

Query String 의 형식은 key = value 의 형식으로 데이터를 전달하기 위해서 사용.

Context root : 우리 프로젝트를 web client 가 지칭하기 위해서 사용하는 프로젝트에 대한 논리적인 이름 .(ex/ 수업 중 만든 프로젝트 ajax)



예제1 : eclipse를 통한 연습프로그램 작성 – 연산프로그램을 작성하자

(★dynamic web project 작성시

Html 파일은 web content폴더 하단에,

Servlet 은 Java Resource – src 하단에 생성하도록 한다. )

**(1). 정적 content작성**

**|index.html|**

<!**DOCTYPE** html>

<**html**>

<**head**>

<**meta** charset=*"UTF-8"*>

<**title**>Insert title here</**title**>

</**head**>

<**body**>

<**h1**>더하기 프로그램 입니다!!</**h1**>

<**input** type = *"text"*> + <**input** type = *"text"*>

<**br**><**br**>

<**a** href = *""*>서버프로그램 호출</**a**> <!—a : 앵커, 하이퍼 링크 설정 -->

</**body**>

</**html**> <!-- http://localhost:8080/ajax/index.html -->

**(2). 동적 content작성**

**|AddServlet.java|**

package com.test;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

\* Servlet implementation class AddServlet

\*/

*@WebServlet*("/add")

public class AddServlet extends HttpServlet {

private static final long ***serialVersionUID*** = 1L;

/\*\*

\* **@see** HttpServlet#HttpServlet()

\*/

public AddServlet() {

super();

// **TODO** Auto-generated constructor stub

}

/\*\*

\* **@see** HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

\*/

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// **TODO** Auto-generated method stub

//클라이언트가 프로그램을 호출하면 이 부분이 실행된다.

//1. 입력(없음)

//2. 로직처리(없음)

//3. 처리된 결과를 출력

// → 클라이언트에게 결과로 전달된 내용이 어떤 내용인지 알려줘야 한다.

response.setContentType("text/plain; charset=UTF-8");

// (response.getWriter().append("Served at: ").append(request.getContextPath()); (주석처리)

// 클라이언트에게 결과를 전달하기 위한 데이터 전달 통로를 하나 생성

PrintWriter out = response.getWriter();

// 통로를 통해서 데이터를 클라이언트에게 전달한다.

out.println("소리없는 아우성!!");

// 데이터 전달이 끝났으면 통로를 닫는다.

out.close();

}

/\*\*

\* **@see** HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

\*/

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// **TODO** Auto-generated method stub

doGet(request, response);

}

}

1. **AJAX**

:Ajax(Asynchronous JavaScript and XML, 에이잭스)는 비동기적인 웹 애플리케이션의 제작을 위해 아래와 같은 조합을 이용하는 웹 개발 기법이다.

표현 정보를 위한 HTML (또는 XHTML) 과 CSS

동적인 화면 출력 및 표시 정보와의 상호작용을 위한 DOM, 자바스크립트

웹 서버와 비동기적으로 데이터를 교환하고 조작하기 위한 XML, XSLT, XMLHttpRequest (Ajax 애플리케이션은 XML/XSLT 대신 미리 정의된 HTML이나 일반 텍스트, JSON, JSON-RPC를 이용할 수 있다).

\* Ajax는 자체가 하나의 특정한 기술을 말하는 것이 아니며,

함께 사용하는 기술의 묶음을 지칭하는 용어이다.

-> AJAX라는 방식은 JavaScript로 서버 프로그램을 호출하는 방식

순수 AJAX는 코드가 너무 복잡하고 어려우나, jQuery library를 이용하면 AJAX도 쉽게 구현 가능하다. -> 직관적으로 구현 가능 (한 눈에 볼 수 있다.)

**예제3 : 도서 검색 프로그램 완성하기 (AJAX 활용)**

이전 연산 프로그램을 활용하여 도서 검색 프로그램을 완성해보자.

**|index.html|**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<!-- jQuery CDN -->

<script

src=*"https://code.jquery.com/jquery-2.2.4.min.js"*

integrity=*"sha256-BbhdlvQf/xTY9gja0Dq3HiwQF8LaCRTXxZKRutelT44="*

crossorigin=*"anonymous"*></script>

<script src=*"js/index.js"*></script>

</head>

<body>

<h1>도서검색 프로그램 입니다!!</h1>

<input type = *"text"*>

<br><br>

<!-- <a href = "http://localhost:8080/ajax/add"></a> --> <!-- a: 앵커, 하이퍼 링크 설정 -->

<input type=*"button"* value=*"서버프로그램 호출"* onclick="myFunc()">

<div>찾은 결과를 출력해요!! </div>

</body>

</html>

<!-- http://localhost:8080/ajax/index.html -->

**|index.js|**

**function** myFunc() {

// 리로딩시 ctrl + F5

// 이제 여기에서 AJAX를 이용해서 서버 프로그램을 호출해 보아요

// 순수 AJAX코드는 너무 복잡하다.

// JQuery library를 이용하면 AJAX도 쉽게 구현할 수 있다. → 직관적으로 구현이 가능(한눈에 볼 수 있다.)

// 아래의 함수들을 통해서 서버 프로그램을 호출하는데

// 알려줘야 하는 여러가지 정보들이 있다. (서버 URL, 호출방식...)

// 이런 정보를 JavaScript 객체화 시켜서 인자로 사용한다.

// ex/ { name : "홍길동", age : 20, address : "서울" }

// JSON( JavaScript Object Notation )

$.ajax({

url : "http://localhost:8080/bookSearch/search",

data : {

keyword : $("[type=text]").val() //서버에 데이터 전달하는 부분

//검색으로 들어오는 걸 value(.val())로서 받겠다.

},

success : **function**(result) {

//성공하면 서버가 데이터를 보내준다.

//이 데이터를 함수의 인자로 받을 수 있다.

//받은 다음 화면 제어!

//result가 문자열이에요.. 문자열처릴르 해서 제목만 뽑아내면 되겠따

// → 근데 생각보다 어려움 (프로그램이 거지같음, 찾아서 짜르고 추출 ...)

// 문자열을 직접 처리하는 것은 상당히 좋지 않은 방식

$("div").text(result);

},

error : **function**() {

alert("뭔가 이상해요!!")

} // ajax콜이 잘 안됨을 인지시켜줌

});

}

* **JSON parsing**

: json parsing이란 검색의 대상의 되는 json을 받아서 객체화 시키는 것을 통칭

**예제4 : 도서 검색 프로그램 완성하기 (JSON parsing 활용)**

**|index.html|**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

<!-- jQuery CDN -->

<script

src=*"https://code.jquery.com/jquery-2.2.4.min.js"*

integrity=*"sha256-BbhdlvQf/xTY9gja0Dq3HiwQF8LaCRTXxZKRutelT44="*

crossorigin=*"anonymous"*></script>

<script src=*"js/index.js"*></script>

</head>

<body>

<h1>도서검색 프로그램 입니다!!</h1>

<input type = *"text"*>

<br><br>

<!-- <a href = "http://localhost:8080/ajax/add"></a> --> <!-- a: 앵커, 하이퍼 링크 설정 -->

<input type=*"button"* value=*"서버프로그램 호출"* onclick="myFunc()">

<!-- 출력결과 목록으로 작성하기 -->

<ul id=*"myList"*> </ul>

</body>

</html>

<!-- http://localhost:8080/ajax/index.html -->

<!—상기 url로 접근 -->

**|index.js|**

**function** myFunc() {

// 리로딩시 ctrl + F5

// 이제 여기에서 AJAX를 이용해서 서버 프로그램을 호출해 보아요

// 순수 AJAX코드는 너무 복잡하다.

// JQuery library를 이용하면 AJAX도 쉽게 구현할 수 있다. → 직관적으로 구현이 가능(한눈에 볼 수 있다.)

// 아래의 함수들을 통해서 서버 프로그램을 호출하는데

// 알려줘야 하는 여러가지 정보들이 있다. (서버 URL, 호출방식...)

// 이런 정보를 JavaScript 객체화 시켜서 인자로 사용한다.

// ex/ { name : "홍길동", age : 20, address : "서울" }

// JSON( JavaScript Object Notation )

$.ajax({

url : "http://localhost:8080/bookSearch/search",

dataType : "json", //서버가 보내준 JSON문자열을 자바스크립트 객체형태(자료구조)

//형태로 변환

data : {

keyword : $("[type=text]").val() //서버에 데이터 전달하는 부분

//검색으로 들어오는 걸 value(.val())로서 받겠다.

},

success : **function**(result) {

//성공하면 서버가 데이터를 보내준다.

//이 데이터를 함수의 인자로 받을 수 있다.

//받은 다음 화면 제어!

//result가 문자열이에요.. 문자열처릴르 해서 제목만 뽑아내면 되겠따

// → 근데 생각보다 어려움 (프로그램이 거지같음, 찾아서 짜르고 추출 ...)

// 문자열을 직접 처리하는 것은 상당히 좋지 않은 방식

//(주석처리) $("div").text(result); (

// json parsing : 서버가 보내준 JSON문자열을 JavaScript 객체로 변환

// $("div").text(result[0].title) //js는 첨자가 0부터 시작, 즉 [1]은 두번째

// 결과검색 중 첫번째 타이틀만 빼라

// 찾아온 책 전부 제목만 추출해서 목록으로 출력하라

//<ul>

// <li>여행가요~</li>

// <li>자바와 떠나는 여향</li>

//</ul>

// i가 0부터 1씩 증가하면서 result의 길이보다 작을 동안 반복

$("#myList").empty(); //기존 목록을 전부 삭제, 검색할때마다 새로운 목록이 형성되게

**for**(i=0; i<result.length; i++){

**let** title = result[i].title;

**let** li = $("<li></li>").text(title)

// <li> 자바와 떠나는 여행 </li>

$("#myList").append(li);

//또는, $("#myList").append($("<li></li>").text(result[i].title));

}

},

error : **function**() {

alert("뭔가 이상해요!!")

} // ajax콜이 잘 안됨을 인지시켜줌

});

}

* **OPEN API 활용하기**

: 영화진흥위원회 -> Open API 제공

날짜 별 박스 오피스 순위 등등을 알아낼 수 있다.

1. 링크접속

<http://www.kobis.or.kr/kobisopenapi/homepg/main/main.do>

(2) 화면 하단 응답예시 JSON 해당 양식

[http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.**json**?key=430156241533f1d058c603178cc3ca0e&targetDt=20120101](http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json?key=430156241533f1d058c603178cc3ca0e&targetDt=20120101)

(3) 키 발급 관리

회원가입을 통해서 발급 (내꺼 : [180bd775952d53adb92d8dcf2c6fd2d4](http://www.kobis.or.kr/kobisopenapi/homepg/apikey/ckUser/saveApikeyForm.do?apiKey=180bd775952d53adb92d8dcf2c6fd2d4) )

(4) 이전 링크에 내 키를 적어 넣는다

**http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json?key=**[**180bd775952d53adb92d8dcf2c6fd2d4**](http://www.kobis.or.kr/kobisopenapi/homepg/apikey/ckUser/saveApikeyForm.do?apiKey=180bd775952d53adb92d8dcf2c6fd2d4)**&targetDt=20120101** -> 복사 후 사용

(맨 뒤 날짜는 조사 일시, 필요여하에 따라 임의로 바꿀 것)

**예제5 : 영화진흥위원회 날짜 별 상영영화 검색 (Open API활용)**

(\*IDE : webstorm 활용하기)

(시작 전 힌트 : 소스가 되는 JSON의 데이터 형식 확인 -> {중괄호} [대괄호]냐

|  |  |
| --- | --- |
|  |  |

**<도서정보table 생성 스크립트> <박스오피스 영화정보 JSON>**

**☞** 중괄호일때는 ‘이름’ , 즉 key값 문자열로만 JSON에 접근해야 하지만

대괄호일때는 ‘숫자’, JSON 목록의 번호값으로도 접근이 가능하다

**(영화정보검색 이어서)**

**|boxOffice.html|**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
 <script  
 src="http://code.jquery.com/jquery-2.2.4.min.js"  
 integrity="sha256-BbhdlvQf/xTY9gja0Dq3HiwQF8LaCRTXxZKRutelT44="  
 crossorigin="anonymous"></script>  
 <script src = "js/boxOffice.js"></script>  
 <  
</head>  
<body>  
 <h1>날짜별 박스오핏 순위 검색 </h1>  
 입력날짜 : <input id = "searchDate" type="text"> <!--ID 하나 부여해서 jQuery로 검색하기 쉽게-->  
 <br><br>  
 <input type="button" value="영화검색" onclick="myFunc()">  
 <ul id="movieList">  
  
 </ul>  
</body>  
</html>

**|boxOffice.js|**

function myFunc() {  
 // Open API에 대한 Ajax 호출, 결과를 가져와서 화면에 적절하게 목록으로 출력이 목적  
 // sample url  
 // url : http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json  
 // 발급받은 키값: ?key=180bd775952d53adb92d8dcf2c6fd2d4  
 // 입력받는 날짜: &targetDt=20120101  
  
 ***$***.ajax( {  
 url : "http://www.kobis.or.kr/kobisopenapi/webservice/rest/boxoffice/searchDailyBoxOfficeList.json",  
 data : {  
 key :"180bd775952d53adb92d8dcf2c6fd2d4",  
 targetDt : ***$***("#searchDate").val()  
 //서버에 데이터 전달하는 부분  
  
 },  
 dataType : "json",

success : function (result) {  
 //alert("성공성공!!")  
 let movie = result.boxOfficeResult.dailyBoxOfficeList;  
 ***$***("#movieList").empty();  
 for(i=0; i<movie.length; i++){  
 let title = movie[i].movieNm;  
 let li = ***$***("<li></li>").text(title);  
 ***$***("#movieList").append(li);  
 //또는, $("#myList").append($("<li></li>").text(result[i].title));  
 }  
 },  
 error : function () {  
 alert("뭔가 이상해요!!")  
  
 }  
 }  
  
  
 )  
}