52일차 강의

* <a href='./add'>신규 회원</a>
* <a> : anchor
* ./add : 상대경로 -> web.xml 을 보고 결정하라, 그리고 그에 해당하는 자바 클래스 파일이 있어야 한다
* add하면 doGet으로 오게 된다
* doGet 을 하게되면 무슨일이 벌어지나?
* <a>태그 수행시?

: doGet을 무조건 사용하도록 만들어 두었다. (불면의 진리)

* doGet(HttpServletRequest req, HttpServletResponse res)

: 이미 만들어져 있던 것, 그냥 사용만 하면됨

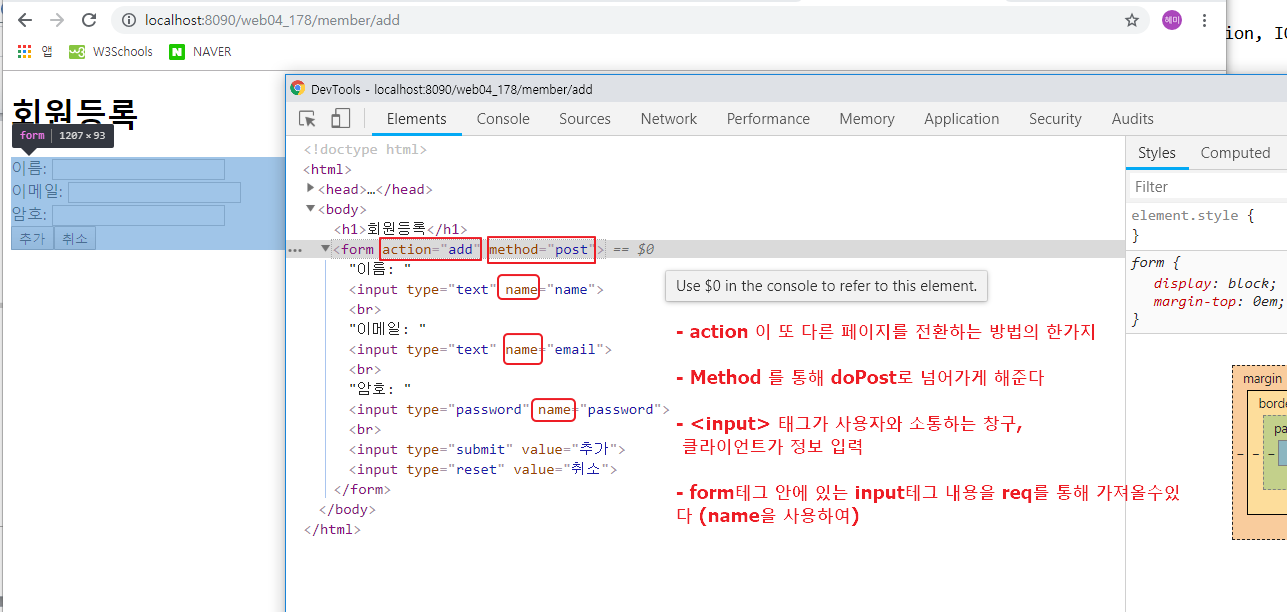
* req(요청) : 사용자가 입력할 수 있는 내용
* res(응답) : 응답에 대한 내용을 담는다 (페이지에 대한 설정 etc.)
* 사용자가 보게되는 화면을 만드는 것 (아래사진)



* <form> 태그 중요!!
* 폼에 메서드가 있어서 명시적으로 적었다
* <form action= >
* action 이 또 다른 페이지를 전환하는 방법의 한가지

: <form action='add' method='post'>

* Method 를 통해 doPost로 넘어가게 해준다 (submit 누르면 포스트방식으로 추가된다)
* <input> 태그가 사용자와 소통하는 창구, 클라이언트가 정보 입력



* 폼테그 안에 있는 인풋테그 내용을 req를 통해 가져올수있다 (name을 사용하여)
* 요청에 관련된 모든 내용은 폼태그안의 인풋태그에 존재한다
* Name 이라는 속성을 적어주면, 이후에 이것을 들고온다.
* prepareStatement 진화한 형태의 객체 (무조건 이것을 사용하라!)
* pstmt = conn.prepareStatement(sql);
* ? 물음표를 통해 분리시켜 : 보안에 대한 것을 나눌 수 있게 됨
* 띄어쓰기도 안해도 됨
* 이전 Statement 는 사용자가 sql문을 수정가능하여, 공격이 가능하며 보안이 취약하고 이것을 악용할 가능성이 있음
* rs = stmt.executeQuery(sql); // select문을 수행하는 것
* pstmt.executeUpdate(); // 수행 // insert, delete 같은 수정을 한다
* select 빼고는 executeQuery를 사용한다
* a태그 수행시

form 의 method가 get인 경우 또는 생략한 경우

doGet 메서드를 호출한다

* form 의 메서드가 post인경우 doPost 메서드를 호출한다
* name 이 존재하는 한에 req에 요청가능 하다.
* 그러므로 submit 이나 reset 같은 (name이 존재하지않는 것은) 그저 그 역할을 수행할 뿐이다.
* 디버깅을 배우자!!!!
* 디버깅을 할때는 어떤 상황인가?

: 수행이 잘됐는지 아닌지는 화면단을 보고 알수가 없다

1. Console 창부터 본다 – 오류를 체크한다
2. 브라우저 에서 - F12 로 오류를 체크한다
3. 1,2번 수행을 했는데도 오류가 없다면?
   * + 논리적 에러를 잡을려면 디버깅을 찍어본다!!

브라우저의 경로를 보러간다

* Web.xml 의 경로나 어떤 파일이 문제인지 찾아간다
* 브레이크 포인트를 설정하고, 서버 마우스 우클릭 -> debug 실행하면 runtime 시점의 오류를 잡을수 있다
* doGet으로 호출하면 input 태그안의 내용이 주소창에 표시된다
* **Get 방식으로 페이지 호출**
* 서비스 주소?매개변수명=값&매개변수명=값

(네입=벨류, 키=벨류 형태로 되어 있다)

* ? : 서비스와 데이터를 분리하는 구분자

-

<http://localhost:8090/web04_178/member/add?testName=newName&google=intel>

-https://www.google.com/search?sourceid=chrome&ie=UTF-8

* add? 뒤의 물음표를 기점으로
* 매개변수명 = name
* 값 = name value
* **Get을 선택할 조건** (언제 사용하나?)
* 자료검색, 게시글 상세 정보, 특정 상품의 정보 조회 등
* 데이터 조회의 경우 적합하다.
* 특징: 즐겨 찾기에 등록하고 싶은 경우 get을 사용해야 한다
* 장점 ?

: 결과화면에 대해 쉽게 공유할 수 있다. (상대방과 공유하고 싶을때)

: Post에 비해 속도가 빠르다

* 문제점?

: 보안에 좋지 않다.

(웹브라우저에 모든 사용자의 정보가 노출된다)

로그인, 개인정보 관리등 보안을 요구한다면 get을 사용하지 않는다.

: 전달값에 제한이 있다 (only String만 전달가능)

응용프로그램에 대한 글자수 제한이 있다(1000글자같은건 못보냄)

* doPost 방식으로 하면 /add 뒤의 주소가 보이지않는 형식으로됨 (보안)
* **Post 방식으로 페이지 호출**
* Get의 문제점을 해결하려고 만듬
* https://www.google.com/
* **장점?**

: url에 데이터가 포함되지 않는다 -> 외부 노출 방지

: 메시지 본문에 데이터 포함 -> 실행 결과 공유 불가

: 바이너리(2진법) 및 대용량 데이터 전송 가능

(문자열 뿐 아니라 숫자, 이미지, 사용자정의 객체든 무엇이든 전송가능)

* 문제점 ?

: get방식에 비해 무겁다. -> 속도가 느린 편이다.

: post방식도 데이터를 전달 할 때 이름=값&이름=값 형태를 사용한다. 문자 데이터를 보낼 때는 문제없지만, 이미지나 동영상과 같은 바이너리 데이터를 보낼 때는 문제가 발생할 수 있기 때문이다.

이런 문제를 해결하기 위해 바이너리 데이터를 보낼 때는 특별한 형식으로 작성하여 보낸다.

* Statement 는 sql 공격을 받기 쉽다.
* PreparedStatement 타입으로 수정하라
* 셀렉 전체 조회중인데

임의로 한 사람의 내용만 조회

member/one 형태로 경로를 설정

< 요청해서 원하는 결과를 먼저 구현 >

1. 데이터베이스 sql문
2. sql문이 수행될 서블릿 클래스 구현

2-1. 화면의 모습 구현

1. web.xml에 서블릿 클래스 등록

* 전체 프로젝트를 복붙하는건 위험한데

그럼 이렇게 한다면 무엇을 바꾸어야하나?

Ex> web04\_178

1. 무조건 톰캣을 끄고 프로젝트를 복붙한다
2. 이런식으로 복붙하면 파일 형상 관리가 안된다
3. Web04\_194의 index로 가서 ctrl + F11

* 복붙했으므로 복붙한 프로젝트가 괄호로 붙어있음

1. 수정을 하려면 봐야할 몇가지가 있다 (영상을 돌려봐라 4시)
2. 네비게이터로 간다

* Navigator 가 설정에 대한 공통단을 다룬다

물리 영역이므로 진짜 조심해야한다

* 실행시 브라우저에서 한글이 깨지는 것을 어떻게 해결할 것인가?

req.setCharacterEncoding("UTF-8");

* 이전에 입력한 한글이 깨진 데이터는 바꿀수 있을까?
* 한번 입력한 데이터는 데이터베이스에서 보면 깨져있다
* <http://localhost:8090/web04_194/member/one?email=%EC%9D%BC%EC%A7%80%EB%A7%A4%40&name=%EC%9D%BC%EC%A7%80%EB%A7%A4>
* 한글부분이 인코딩으로 인해 다 깨져있음을 확인됨
* Request 할때 get parameter 할떄

res.setCharacterEncoding("UTF-8"); 해준다(? 확인필요)

ㅇ

ㅇ

* 유관순을 추가하고 브라우저에서 F5를 누르면 오류가 뜬다
* ORA-00001: unique constraint (JSP.MEMBERS\_EMAIL\_UK)
* 이전에 남아있던 양식(즉 유관순 가입정보)를 또다시 입력하려고 했으므로
  + 그러므로 새로고침 했을 때 다른 페이지로 전환하도록 만들어 줘야한다.

(ex, 은행에 입금했다- 근데 브라우저 F5 누르면 또 입금된다는 것, 큰일남!!!!)

* + 어떻게 해결 할것인가?

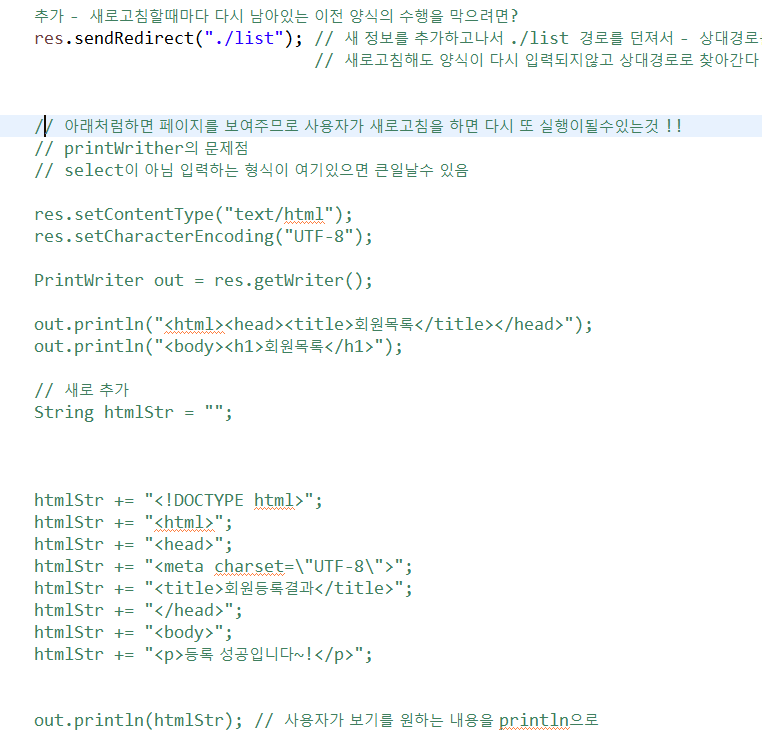
Add 를 추가했을떄 post로 갔을 때 문제가 생기므로

addServlet을 변경해야한다고 생각된다

* + 추가 - 새로고침할때마다 다시 남아있는 이전 양식의 수행을 막으려면?
* res.sendRedirect("./list"); 를 적어준다

// 새 정보를 추가하고나서 ./list 경로를 던져서 - 상대경로를 찾아간다

// 새로고침해도 양식이 다시 입력되지않고 상대경로로 찾아간다



* 이제는 경로가 중요하다

아래 세가지의 차이점을 알아야 한다

<a href=*'./member/one'* >연습 페이지로 이동</a>

* <http://localhost:8090/web04_194/member/one>
* 이처럼 상대경로를 명시적으로 표시하는 것이 좋다

<a href=*'member/one'* >연습 페이지로 이동</a>

* <http://localhost:8090/web04_194/member/one>
* 아무것도 안적으면 상대경로로 인식

<a href=*'/member/one'* >연습 페이지로 이동</a

* <http://localhost:8090/member/one>
* 절대경로
* [ 어떤 식으로 작동되는가? ]

1. 웹 브라우저에서 회원 등록 입력폼을 요청 (doGet 호출)

* 폼태그를 인식해야 한다.
* (사용자가 입력할 화면이 있을것이다)
* a태그를 ???

1. MemberAddServlet 은 회원 등록폼을 클라이언트로 보냄



1. 웹 브라우저는 회원 등록폼을 화면에 출력 ( html 내용 출력)
2. 사용자가 회원 정보를 입력하고 나서 추가 버튼을 누르면, 서버에 등록을 요청

* Action , submit 등을 만들어 뒀으니깐

1. 회원 정보를 db에 저장
2. MemberAddServlet은 (더 이상 자신의 페이지를 보여주는 것이 아니라)

이동할 페이지의 URL 정보(/member/list)를 보냄

* sendRedirect 를 하려고
* 다른 페이지로 넘어갔으므로 F5해서 중복내용의 수행을 방지함. 사용자를 다른 곳으로 보내버림.

1. 웹 브라우저는 서버로부터 응답을 받는 즉시 이동페이지의 url로 다시 요청

* 웹브라우저가 그 정보를 기억하여
* MemberListServlet 으로 다시 던짐

1. MemberListServlet은 회원 목록을 클라이언트로 보냄

* PrintWriter로 만들고
* **res.sendRedirect("./list"); 아주 중요!!!**

사용하면 이전 데이터를 다 소멸시키므로,

브라우저에서 뒤로가기를 눌러도 소용없다

Delete , insert, update (select 이외의 모든 문장) 에서 이를 사용해야한다.

* Web.xml 에서 매번 등록하고 관리하기 미칠 노릇

어떻게 할 것인가?

* Web.xml 에 있던 것을 삭제하고
* Annotation 을 통해 관리한다
* @WebServlet(value="/memeber/list") 를 MemberListServlet의 클래스에 추가해주면 된다

(import 아래에)

(큰 회사들은 이런 것이 만들어져 있어서 사용하면 그만이다)

* 하나의 페이지는 총 4가지 정도 기능을 수행해야한다

1. 회원조회
2. 회원등록
3. 회원변경

* 회원변경을 할 수 있도록 만들자
* 이메일, 이름, 비밀번호 – 가 수정가능