KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

요청 포워딩, 리다이렉트

서블릿에서 처리하는 기술

[KB] IT's Your Life

JSP기억나능가
상세히 설명하지 않았다.
왜냐하면 이는 model1에서할 경우만 해당하기 때문
실무에서는 모델1을 하지 않기 때문

모델2개발에 필요한 애플리케이션 기술이 있는데

모델2는 역할을 나누는 것 비즈니스로직은 서블릿에서

그럼다음에 그 결과를 jsp에서 출력을 맡은것.

어떻게 서블릿에서 처리한 결과를 jsp에게 넘길까

모델2에 필요한 기술 요청 포워딩과 리다이렉트.
둘은 서블릿에서 처리하는 기술

받은 데이터를 출력할때 jsp사용하는 el jstl기술을
배울것이다.

비즈니스 결과가나오고 2가지 경우의수
가 있다.

jsp로 넘겨서 출력하거나

다른 페이지로 요청으로 이동해야하는 수.

jsp로 넘겨서 결과 출력은 forwording기술

페이지를 이동시키는 경우는
(예: 등록,수정)결과출력보다페이지이동을함.
리다이렉트

일반적으로 post요청시 리다이렉트하는 경우가 많음.

KB 국민은행

모델 1

아까 설명한 모델1

- 기존의 JSP로만 구현한 웹 애플리케이션
- 웹 브라우저의 요청을 JSP 페이지가 받아서 처리하는 구조
- JSP 페이지에 비즈니스 로직 처리 코드와 웹 브라우저에 결과를 출력하는 코드가 섞이는 것

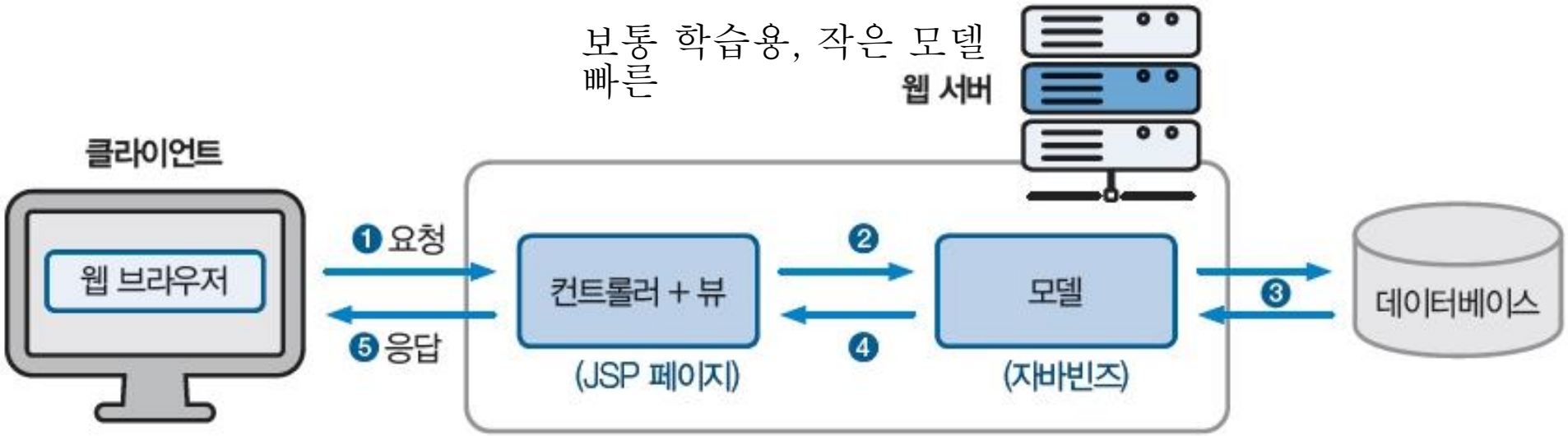
jsp가 모든 것을 다하는 것

- 모델 1에서는 JSP가 핵심 역할을 수행함

객체지향 SOLID 원칙 위배. => 유지보수 엄청 힘들어짐.

- 모델 1의 구조와 요청 처리 흐름

보통 학습용, 작은 모델
빠른



모델2

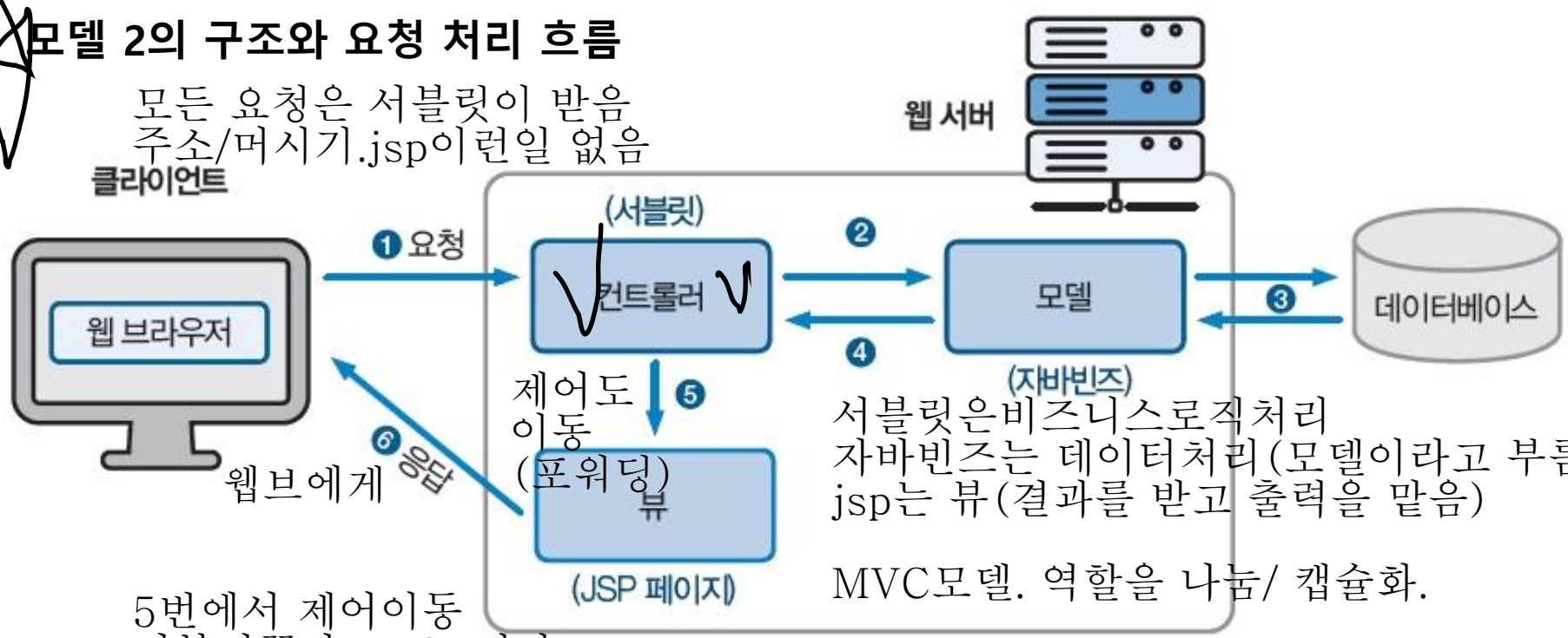
대표적인 웹 앱 모델

모델2아키텍처 == MVC패턴

- 클라이언트의 요청 처리, 응답 처리, 비즈니스 로직 처리 부분을 모듈화한 구조
- 요청에 대한 로직을 처리할 자바빈즈나 자바 클래스인 모델,
  요청 결과를 출력하는 JSP 페이지인 뷰,
  모든 흐름을 제어는 서블릿인 컨트롤러로 나뉘어 웹 브라우저가 요청한 작업을 처리함
- 모델 2에서는 서블릿이 중요한 역할을 함

모델 2의 구조와 요청 처리 흐름

모든 요청은 서블릿이 받음
주소/머시기.jsp이런일 없음



서블릿은비즈니스로직처리
자바빈즈는 데이터처리(모델이라고 부름)
jsp는 뷰(결과를 받고 출력을 맡음)

MVC모델. 역할을 나눔/ 캡슐화.

5번에서 제어이동
서블리끝나고 JSP시작

- **MVC** == 모델2 아키텍처 SOLID원칙 잘 지킴.
  - Model, View, Controller의 약자 데이터, 화면출력, 로직흐름제어(비즈니스로직)
  - 웹 애플리케이션을 비즈니스 로직, 프레젠테이션로직, 데이터로 분리하는 디자인 패턴
  - 웹 애플리케이션에서는 일반적으로 애플리케이션을 비즈니스 로직, 프레젠테이션, 요청 처리 데이터로 분류
    - 비즈니스 로직: 애플리케이션의 데이터, 즉 고객, 제품, 주문 정보의 조작에 사용됨
    - 프레젠테이션: 애플리케이션이 사용자에게 어떻게 표시되는지, 즉 위치, 폰트, 크기
    - 요청 처리 데이터: 비즈니스 로직과 프레젠테이션 파트를 함께 묶음

현대의 웹 앱 백엔드 패턴. == mvc 패턴

MVC 패턴의 구성 요소

모델(model)

- 애플리케이션의 데이터와 비즈니스 로직을 담는 객체

모델계층이라 부름=4<=데이터
서비스 계층이라 부름<=비즈니스로직

컨트롤러(controller) 판단내리고. 결과에 따라 포워딩할지 리다이렉트할지

- 모델과 뷰 사이에 어떤 동작이 있을 때 조정하는 역할
- 웹으로부터 받은 요청에 가장 적합한 모델을 생성하는 것을 처리하는 역할
- 사용자에게 응답하는 적절한 뷰를 선택하여 해당 모델을 전달하는 역할

요청분석후 판단

→ Servlet

뷰(view) 결과데이터를 출력

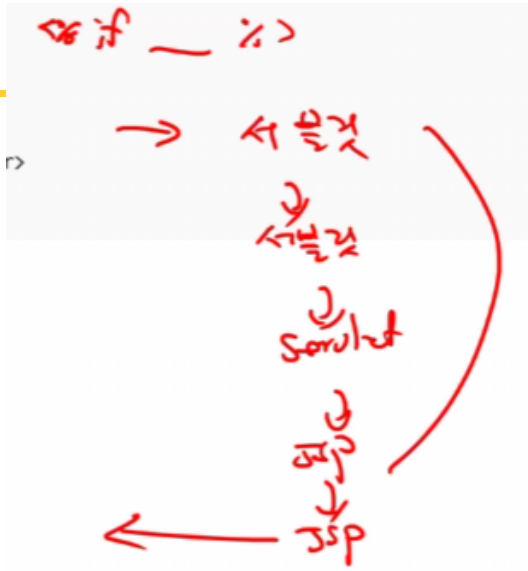
- 사용자에게 모델의 정보(데이터)를 보여주는 역할
- 비즈니스 로직을 포함하지 않으며, 하나의 모델을 다양한 뷰에서 사용

→ JSP

현재는 jsp에서 마치지만

최종적으로는 저어언에 배운 vue3로 바뀔거임.
우리의 최종 목적

2 요청 포워딩(Request Forwarding)



포워딩 갯수는
제한이 없다

요청 포워딩의 필요성

- 요청 처리 작업의 모듈화
- 모듈의 재사용성 증가 및 유지 보수의 편이
- MVC 모델(Model2)의 기본 기능

  - 요청 처리: FrontController
  - 응답 처리: JSP(View)

요청 포워딩 방법

- RequestDispatch 클래스를 이용한 forward 방법
- HttpServletResponse 클래스를 이용한 redirect 방법
  리다이렉트방법

리퀘스트를 전달한다라는 의미.
이때 전달에 해야하는 정보 : 어느 jsp로 이동할거냐
데이터 전달은? 서로 물리적으로 다른 컴포넌트인데?
req,res객체를 통해서!

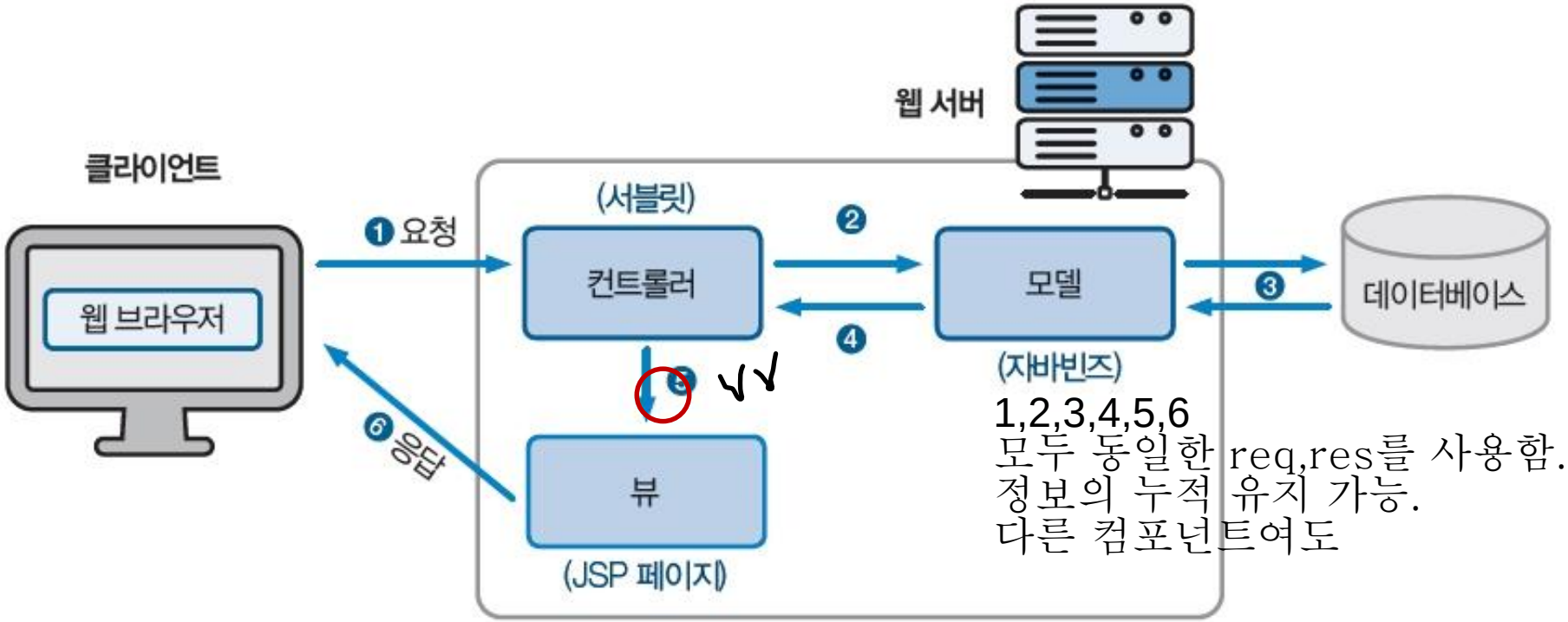
클라리언트에서 바라보는
요청 url은 변함이 없지만

RequestDispatcher 클래스를 이용한 forward 방법

```
RequestDispatcher dis = req.getRequestDispatcher(target);
dis.forward(req, res);
```

target: jsp경로

코드.
목적지(jsp)넘겨주며
리퀘스트 디스패처 얻고
req,res객체 넘겨주며 포워딩



1,2,3,4,5,6
모두 동일한 req,res를 사용함.
정보의 누적 유지 가능.
다른 컴포넌트여도

데이터를 전달하는 과정에서 중요한거 scope

1, 2번의 요청이 같은 HttpServletRequest를 사용
파라미터 값과 setAttribute로 저장한 정보를 공유 → Request scope

전에 아주 간단하게 말한
리퀘스트스코프에
set속성하여 넘겨줄 정보
저장하여 req,res 넘기기

2 요청 포워딩(Request Forwarding)

- **forward 방법 실습**
  - 프로젝트명: ex05
  - 패키지: org.scoula.ex05
  - 클래스: RequestServlet
  - URL 맵핑: /request

scope에 대하여

application scope 모든 서블릿이 공유되는 공간
session scope 브라우저별로 (사용자별X) 공유 공간
request scope 요청과응답과정에서 공유되는 공간
page scope 지역변수와비슷한 한페이지의
컨트롤러,뷰 내에서만 공유되는 공간

사용법은 setAttribute(key,valueobj)
getAttribute(key)->obj이므로 캐스팅필수
로 공통되다.
없는 경우 null이 나오므로
항상 ㅜnull 안정성검사해라.

RequestServlet.java

URL/request 시 요청. 서블릿으로 온거임

```
@WebServlet("/request")
public class RequestServlet extends HttpServlet {

  @Override
  protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

    //속성 설정
    req.setAttribute("username", "홍길동");
    req.setAttribute("useraddress", "서울");

    //forward
    RequestDispatcher dis = req.getRequestDispatcher("/res.jsp");
    dis.forward(req, res);

  }
}
```

리퀘스트 스코프에 저장하는 코드

리퀘스트 디패처 획득후
/res.jsp로 포워딩.

정보를 넘기며

경로앞에
뭐 다른 폴더
없다면
webapp/,,,.jsp임

다르ㄴ페이지도 비슷한 흐름.

BL처리하고

req scope결과 저장. 데이터 특성에 따라 세션스코프에도 다른 스코프에도.

포워딩

=>자동화 가능한데?
타켓만 바꾸면 되잖아.
->프렘웤에 녹아듬.

포워딩만 자동화녹아들고
나머지는 직접해야지

res.jsp //webapp/res.jsp임. '/'가 웹 루트죠

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="utf-8">
    <title>Insert title here</title>
</head>
<body>
username 값: <%=request.getAttribute("username") %><br>
useraddress 값: <%=request.getAttribute("useraddress") %><br>
</body>
</html>
```

지금은 일단 디렉티브로
꺼내서 사용하고 있는 중

스코프에 결과가 없다면 null

null검사도 해야하는 코드까지 들어가면
너무 복잡함.=>EL이 보완함.

http://localhost:8080/request

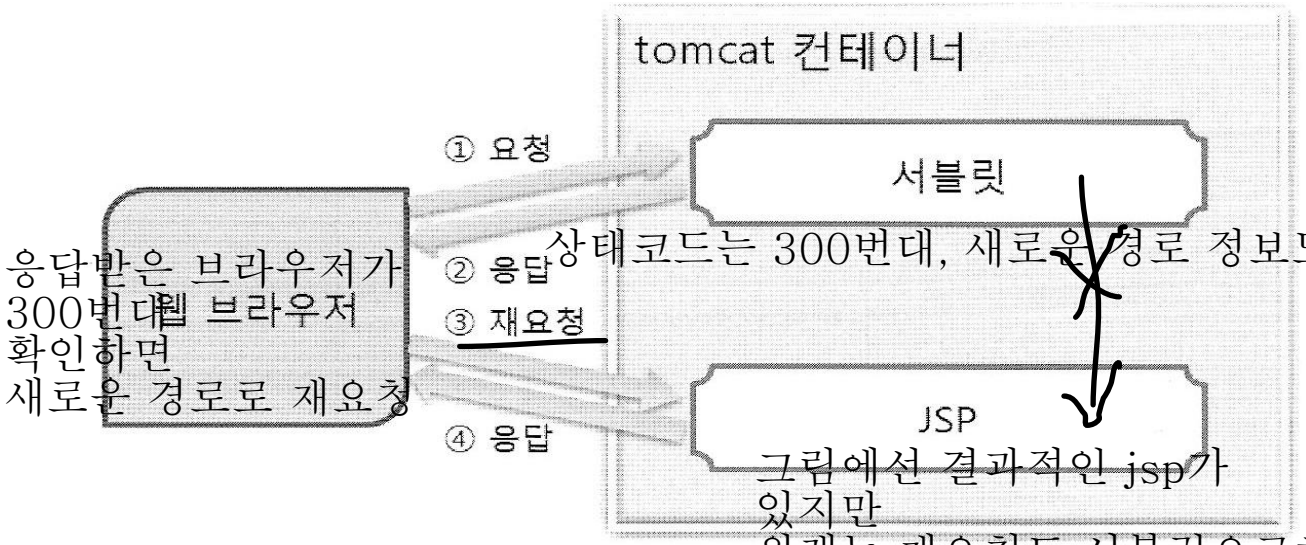
username 값: 홍길동
useraddress 값: 서울

HttpServletResponse 클래스를 이용한 redirect 방법

res.sendRedirect(target);

- target: 이동할 페이지

리다이렉트 과정



상태코드는 300번대, 새로운 경로 정보도 담겨있음

그림에선 결과적인 jsp가 있지만
원래는 재요청도 서블릿으로하고
서블릿에서 포워딩해서 jsp가 나온 결과임

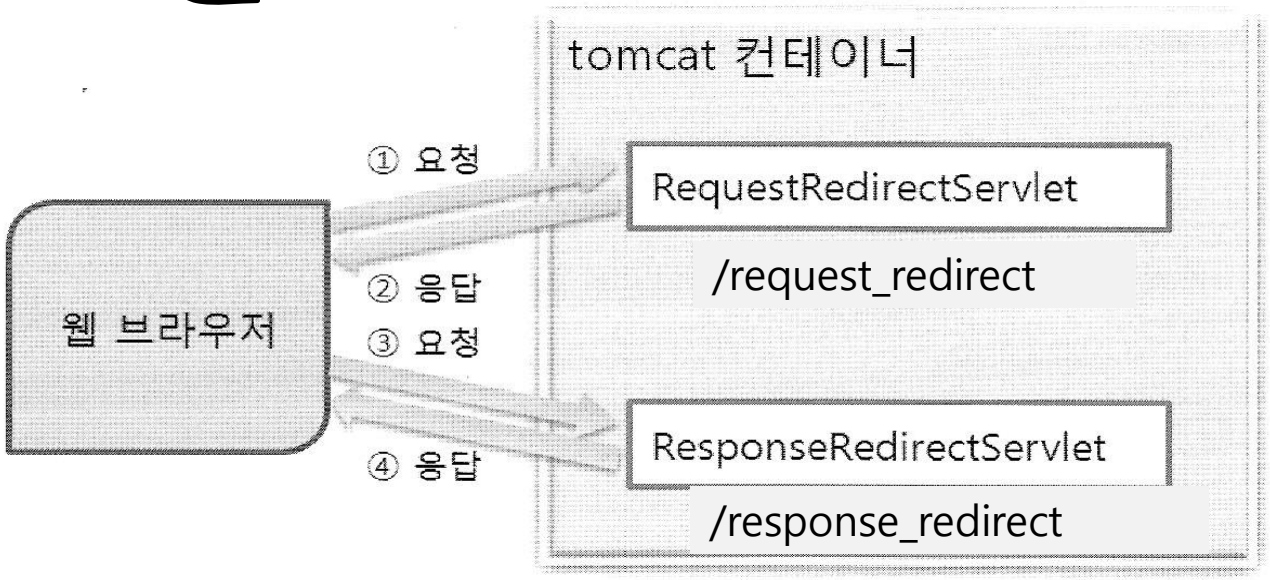
해당 과정은
사용자한테 숨겨져있는데
url은 변경되어잇음
a경로로 요청했지만
결과를 보면 b경로로 바뀌어있:음

- 첫번째 요청에 대해 응답하고 브라우저가 바로 redirect
- 동일 HttpServletRequest가 아닌 새로운 request가 사용됨
  → Request scope가 다름

redirect 방법 실습

- 패키지: org.scoula.ex05
- 클래스: RequestRedirectServlet
- URL 맵핑: /request_redirect

- 클래스: ResponseRedirectServlet
- URL 맵핑: /response_redirect



RequestRedirectServlet.java

```
@WebServlet("/request_redirect")
public class RequestRedirectServlet extends HttpServlet {
  protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

    //속성 설정
    req.setAttribute("username", "홍길동");
    req.setAttribute("useraddress", "서울");

    //redirect
    res.sendRedirect("response_redirect");

  }
}
```

리퀘스트 스코프에 저장? 어차피 리다이렉트되면 새로운 요청이고
리퀘스트 스코프도 서로 다른거라매??

이때 리퀘스트 스코프는 응답하면 생명주기가 끝나느데

리다이렉트하면 새로운 리퀘스트 스코프가 만들어져 활요될텐데

정보 유지는? 리다이렉트인 경우 스코프를 통한 데이터전달(정보유지) XXX

어덯게 전달할까 2가지 기법이 있다.
querystring으로 넘기는 것. 첫번재 서블릿에서 응답을 보낼때 쿼리스트링정보를 담아서 보내면
다음 요청에서 쿼리스트링을 적용하여 정보를 보낸다. 겟파라미터를 통해서 정보를 꺼낼수잇다.
이는 아주 짧은 데이터만.

리퀘스트보다 한단계 높은 스코프를 사용하는 것. 세션스코프를 사용. 세션스코프으ㅔ 담아서 전달.
하지만 리퀘스트 스코프에 들어가는 정보는 한번만 필요한 정보인데 세션스코프에서는 계속 남아있을 텐데
꺼낼댸 세션스코프에서 지워야한다. 이를 플레쉬메모리라고도한다. 2기법은 지금 실습에선 안하고 스프링프레임웤배울때
도움받아서 확인해보자.

ResponseRedirectServlet.java

```
@WebServlet("/response_redirect")
public class ResponseRedirectServlet extends HttpServlet {

  protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

    String username = (String) req.getAttribute("username");
    String useraddress = (String) req.getAttribute("useraddress");

    //속성 설정
    req.setAttribute("username", username);
    req.setAttribute("useraddress", useraddress);

    //forward
    RequestDispatcher dis = req.getRequestDispatcher("/redirect_res.jsp");
    dis.forward(req, res);
  }

}
```

결과는 아무것도 안 담김.

새로운 리퀘스트에는 username과useraddress정보가 없거등

redirect_res.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="utf-8">
    <title>Insert title here</title>
</head>
<body>

username 값: <%=request.getAttribute("username") %><br>
useraddress 값: <%=request.getAttribute("useraddress") %><br>

</body>
</html>
```

언제 리다이렉트를 사용하는가

앞에서 말했듯이

결과출력이 필요하지 않을때

다른페이지로 이동으로 동작시키고 싶을때

생성수정삭제등 post동작이 대부분

포워딩은 결과에대한것을 보여줘야할때
검색, 상세보기 등등

http://localhost:8080/request_redirect → http://localhost:8080/response_redirect

username 값: null
useraddress 값: null

예상한 결과

리다이렉트를 반드시 사용해야하는 시나리오도 있다.
리다이렉트 안하면 안되는 경우.
게시글 등록하는 경우.
생성하여 db등록하고 포워딩하고 jsp로 출력함 이때 포워딩했을때는 요청 url바뀌지 않아. 그상황에서 새로고침하면?
새로고침==전에 보냈던 요청을 다시 보내겠다는 뜻. 그럼 또다시 생성하고 db등록하고 포워딩함. 또다시. 수량이 넘어감.
삭제도 마찬가지로 두번됐을때 문제. 그래서데이터를 수정하는 결과처리를 할때는 포워딩말고 redirect를 사용해야한다.