5. MVC 기법

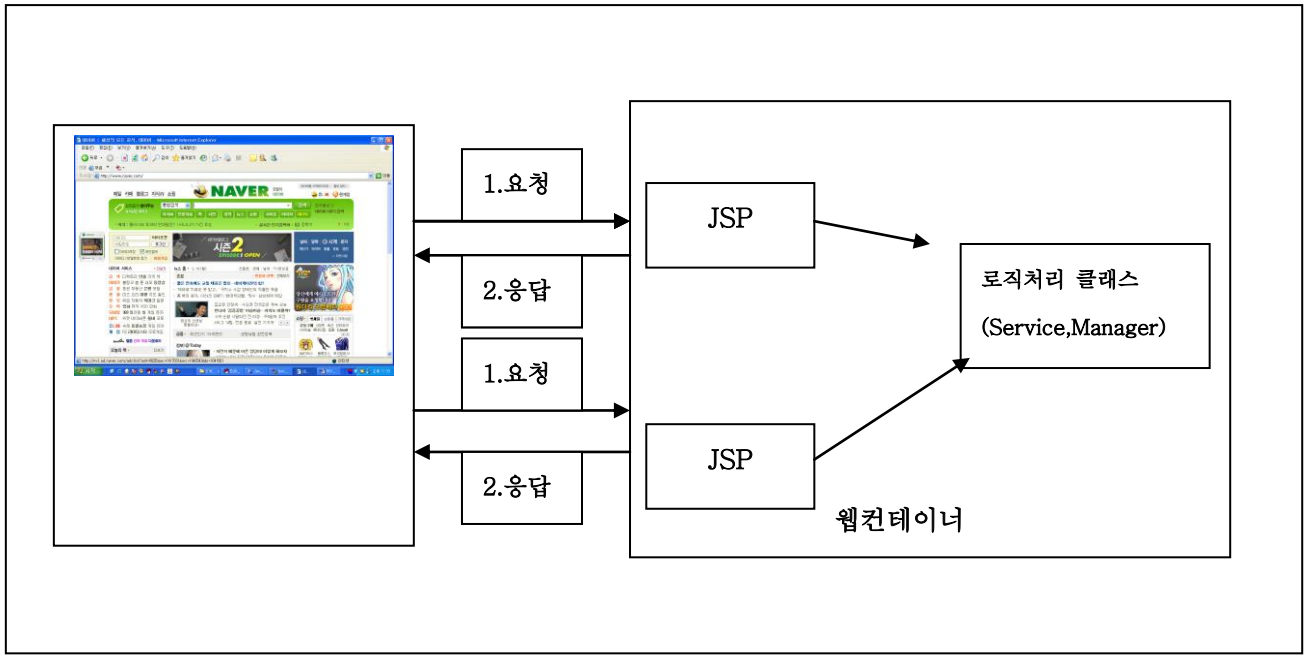
5-1. 모델2 구조와 MVC 패턴

JSP를 기반으로하는 웹애플리케이션을 구성하는 방법에는 크게 모델 1구조와 모델2 구조가 존재한다. JSP 에서 업무로직과 출력을 처리하느냐 아니면 JSP에서는 출력만 처리하느냐에따라서 모델1 구조와 모델2 구조로 구분된다.
MODEL1 과 MODEL2 의 차이점에대해살펴보고 MVC 패턴과 MODEL2 구조의 관계에 대해서 살펴보도록 하겠다.

5-1-1.모델 1구조

모델 1구조는 JSP 를 이용한 단순한 모델이다.
처리구조는 다음그림과같다..

Fig>모델1구조의요청처리 응답

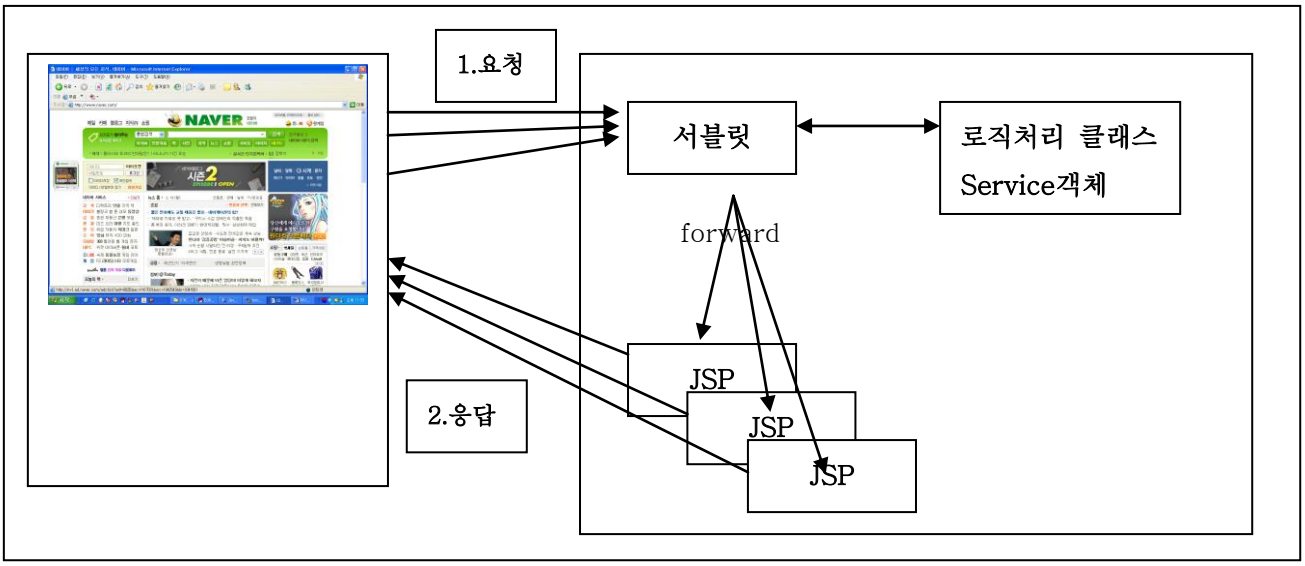


모델1구조는 그림 과같이 웹부라우져의 요청이 곧바로 JSP 에전달된다.
웹부라우져의 요청을받은 JSP는 서비스클래스와 통신하여 웹부라우져가 요청한 작업을 수행하고 그결과를 클라이언트에 출력해준다.

5-1-2.모델 2구조

모델2 구조는 모델1구조와 달리 웹부라우져의 요청을 **하나의 서블릿**이 받게된다.
서블릿은 웹부라우져의 요청을 알맞게처리한후 그결과를 보여줄 JSP로 포워딩한다.
포워딩을 통해서 요청흐름을 받은JSP 페이지는 결과화면을 클라이언트에 전송한다.
즉서블릿이 **비즈니스 로직부분을** 처리하게되는 것이다.
다음 그림은 모델2 구조의 요청처리순서를 보여주고있다.

Fig>모델2구조의요청처리 응답



모델 2구조의 특징은 **웹부라우져의 모든요청이 단일 진입점 ,즉 하나의 서블릿에서 처리**
하나의 서블릿이 웹부라우져의 모든요청을 받기 때문에 서블릿은
웹부라우져의 요청을 구분할 수 있는 방법을 필요로하며 ,서블릿은 웹부라우져의 요청을
처리한후 웹부라우져에 보여질 JSP 를 선택하게된다.

5-1-3. MVC (MODEL-VIEW-CONTROLLER) 패턴

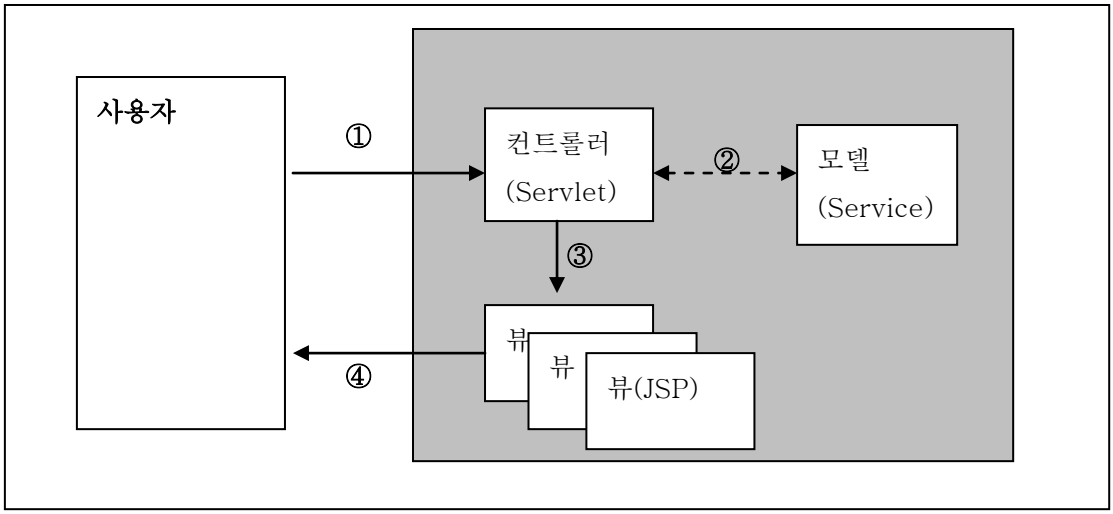
MVC 패턴은 크게 모델,뷰,컨트롤러의 세부분으로 구성되며 각각의요소가하는일은 다음과같다.

모델- 비즈니스영역의 상태정보를 처리한다.

뷰 - 비즈니스 영역에대한 프리젠테이션(즉 사용자가 보게될결과 화면)을 담당한다.

컨트롤러- 사용자의 입력 및 흐름제어를 담당한다.

fig>MVC 패턴의 구조



① 요청

② 비즈니스 로직처리

③ 뷰 선택

④ 응답

사용자가 원하는 기능을 처리하기위한 모든요청을 단일 컨트롤러에게 보낸다.
모델은 비즈니스와 관련된 상태정보 및 관련기능을 제공하는데 컨트롤러는 이 모델을 통해서 사용자의 요청을 처리한다. 모델을 사용하여 알맞은 비즈니스 로직을 수행한후 컨트롤러는 사용자에게 보여줄 뷰를 선택하며,선택된 뷰는 사용자에게알맞은 결과 화면을 보여준다. 뷰가 사용자에게 결과화면을 보여줄때는 결과정보가 필요한데 이정보는 컨트롤러가 뷰에 전달해준다.

MVC 패턴의 핵심은 다음과같다.

- 비즈니스로직을 처리하는 모델과 결과화면을 보여주는 뷰가 분리되어있다.
- 어플리케이션의 흐름제어나 사용자의 처리요청은 컨트롤러에 집중된다.

즉 MVC 패턴은 각각의 하는일이 분리되어있어서 유지보수 작업이간단해지고
애플리케이션의 확장이 손쉬워진다.

5-1-3. MVC (MODEL-VIEW-CONTROLLER) 패턴과 모델구조의 맵핑

모델 2의구조와 MVC 패턴은 완벽하게 일치한다

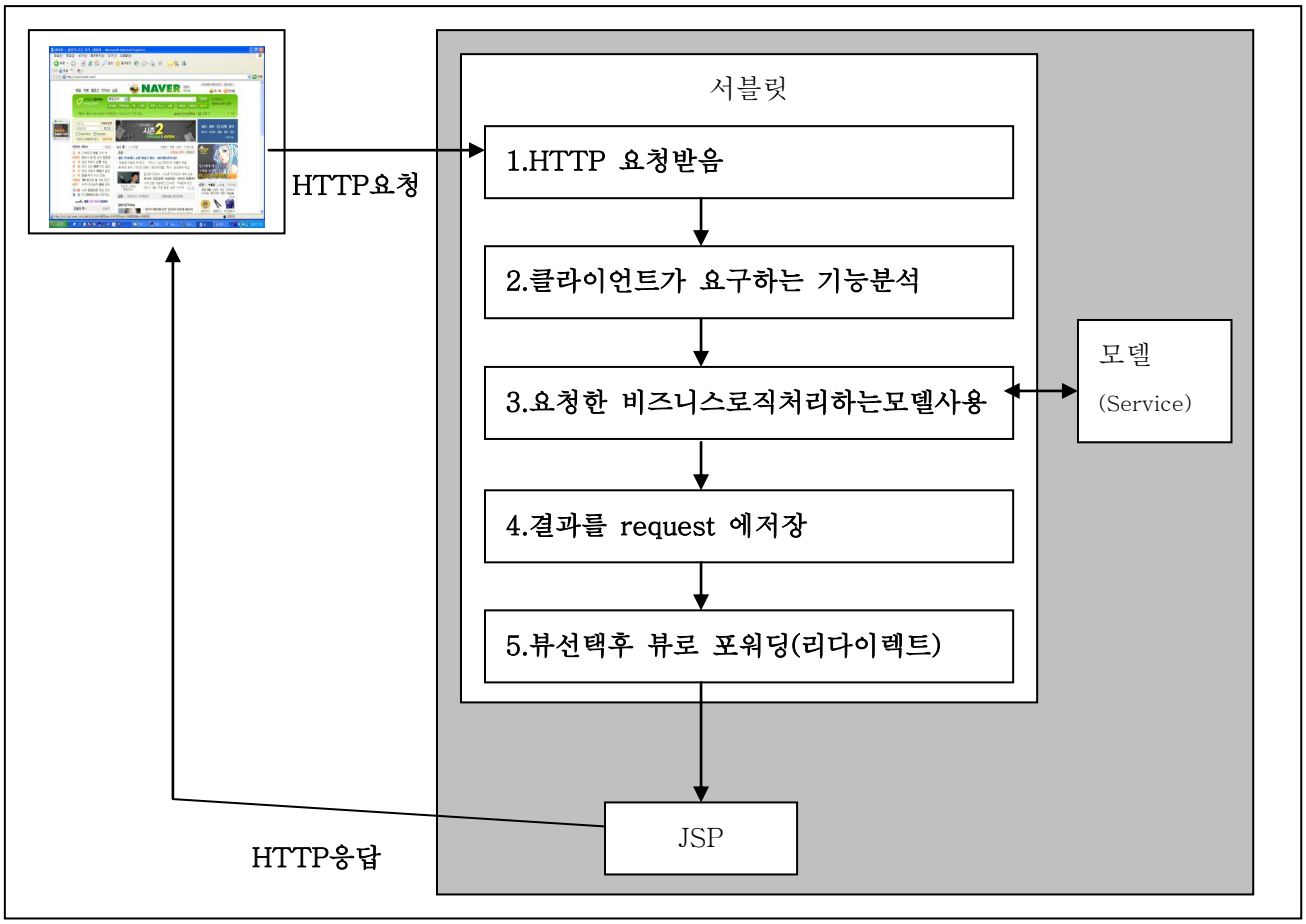
- 컨트롤러 = 서블릿
- 모델 = 비즈니스 로직 처리 클래스,자바빈
- 뷰 = JSP
- 사용자 = 웹부라우져내지 휴대폰과 같은 다양한기기

5-1-4. MVC 의 컨트롤러 :서블릿

모델2에서의 서블릿은 MVC 패턴의 컨트롤러 역할을 한다. 서블릿은 웹부라우져의 요청과 웹애플리케이션의 전체적인 흐름을 제어하게된다.

다음그림은 컨트롤러로서 서블릿이 어떤순서로 실행되는지를 보여주고있다.

fig>컨트롤러 서블릿의 내부동작방식



컨트롤러의 역할은 그림과같이 5단계의 과정을 거쳐서 웹부라우져의 요청을 처리하게된다.

과정1- 웹부라우져가 전송한 HTTP 요청을 받는다 서블릿의 doGet 메쏘드나 doPost 메쏘드 가 호출된다.

과정2 – 부라우져가 어떤기능을 요청했는지 분석한다 예를 들어 ,게시판목록을 요청했는지 글쓰기를 요청했는지 알아낸다.

과정3- 모델을 사용하여 요청한 기능을 수행한다.

과정4 – 모델로부터 전달받은 결과물을 알맞게 가공한후 request나 session 의 setAttribute() 메쏘드를 사용하여 결과값을 속성에 저장한다.이렇게 저장된결과값은 뷰인 JSP 에서 사용된다.

과정5 – 웹부라우져에 보여줄 JSP 를 선택한후,해당 JSP로 포워딩한다,경우에 따라서는 리다이렉트를 하기도한다.

5-1-5. MVC 의 뷰 : J S P

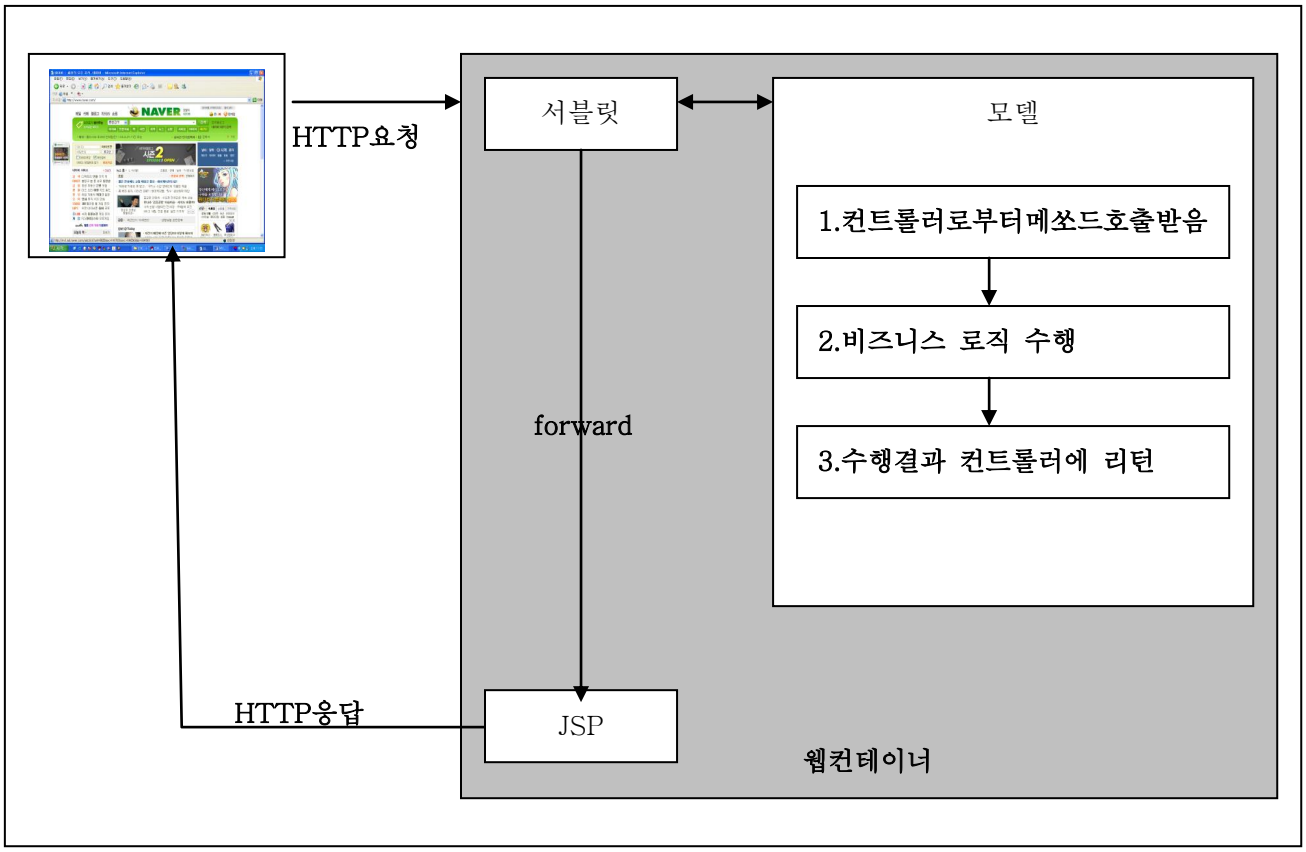
모델2 구조에서 JSP는 뷰의역할을 담당한다. **비즈니스 로직과 관련된 코드가 없는점**을 제외하면 **일반** JSP와 동일한 형태를 취한다. 차이점이있다면 ,뷰역할을 하는 JSP는 컨트롤러 부분에서 request 기본 객체나 session 기본객체에 저장한 데이터를 사용하여 웹부라우져에 알맞은 결과를 출력해준다는 점이다.

뷰역할을하는 JSP는 웹부라우져가 요청한 결과를 보여주는 프리젠테이션의 역할을 할뿐만아니라 웹부라우져의 요청을 컨트롤러에 전달해주는 매게체가 되기도한다. (링크,action,src..)

5-1-5. MVC 의 모델

컨트롤러는 서블릿을 통해서 구현되고 뷰는 JSP를 통해서 구현되는 반면에 모델은 명확하게 어떤 것을 통해서 구현된다는 규칙은 없다. 비즈니스 로직을 처리해주면 모델이 될수있다.

fig>컨트롤러 역할을하는 서블릿과 모델간의 통신



위그림은 모델의 일반적인 업무 흐름을 보여주고있다.컨트롤러 서블릿이 웹부라우져의 요청을 분석하여 알맞은 모델을 호출하면서부터 모델의 기능이 시작된다.
모델은 컨트롤러가 요청한 작업을 처리한후 알맞은결과를 컨트롤러에게 전달해주는데 이때 처리한 결과값을 저장하는 객체로 보통자바빈(DTO,VO)을 사용한다.모델은 데이터베이스와 같은
시스템을 사용하여 비즈니스 로직에 필요한 데이터를 처리한다.
위그림에서는 모델을 웹콘테이너 영역에 포함시켰는데 ,모델이 반드시 웹콘테이너영역에 포함되는것은 아니다.(EJB 컴포넌트는 웹컨테이너밖에 존재한다..)

5-1. 모델2 구조를 이용한 MVC 패턴 구현

여기서 살펴볼 모델2 구조의 예제는 다음과같이 간단한 기능을 제공하는예제이다.

1. 서블릿은 화면에 출력할메세지를 생성해서 JSP에전달한다.
2. JSP는 서블릿이 전달할메세지를 화면에 출력한다.

5-1-1.서블릿클래스(Controller) 작성

SimpleController.java

```java
package mvc.controller;
import java.io.IOException;
import javax.servlet.*;
import javax.servlet.http.*;
public class SimpleController extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }
    public void doPost(HttpServletRequest request,
                       HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }
    private void processRequest(HttpServletRequest request,
                            HttpServletResponse response)
    throws IOException, ServletException {
        // 2단계, 요청 파악
        // request 객체로부터 사용자의 요청을 파악하는 코드
        String type = request.getParameter("type");
        // 3단계, 요청한 기능을 수행한다.
        // 사용자에 요청에 따라 알맞은 코드
        Object resultObject = null;
        if (type == null || type.equals("greeting")) {
            resultObject = "안녕하세요.";
        } else if (type.equals("date")) {
            resultObject = new java.util.Date();
        } else {
            resultObject = "Invalid Type";
        }
         // 4단계, request나 session에 처리 결과를 저장
        request.setAttribute("result", resultObject);
         // 5단계, RequestDispatcher를 사용하여 알맞은 뷰로 포워딩
        RequestDispatcher dispatcher =
                  request.getRequestDispatcher("/simpleView.jsp");
        dispatcher.forward(request, response);
    }
}
```

5-1-2.JSP(View) 작성

simpleView.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>

<html>
<head><title>뷰</title></head>
<body>

결과: <%= request.getAttribute("result") %>

</body>
</html>
```

5-1-3. SimpleController 서블릿을 초기진입점으로하기위한 web.xml 설정

```
<?xml version="1.0" encoding="euc-kr"?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
            http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">
    <display-name>simple MVC</display-name>
    <description>
        간단한 MVC 패턴연습
    </description>
     <servlet>
        <servlet-name>controller</servlet-name>
        <servlet-class>mvc.controller.SimpleController</servlet-class>
   </servlet>
    <servlet-mapping>
        <servlet-name>controller</servlet-name>
        <url-pattern>*.do</url-pattern>
    </servlet-mapping>
</web-app>
```

5-1-4 부라우져에서 요청

http://localhost:8080/controller.do?type=greeting

http://localhost:8080/controller.do?type=date

http://localhost:8080/controller.do