```
package com.fastcampus.ch2;

import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;
import java.lang.reflect.Array;
import java.lang.reflect.Method;
import java.lang.reflect.Parameter;
import java.util.Arrays;
import java.util.Iterator;
import java.util.Map;
import java.util.Scanner;

import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.ui.Model;
import org.springframework.validation.support.BindingAwareModelMap;

@WebServlet("/myDispatcherServlet")  // http://localhost/ch2/myDispatcherServlet?
year=2021&month=10&day=1
public class MyDispatcherServlet extends HttpServlet {
    @Override
    public void service(HttpServletRequest request, HttpServletResponse response)
throws IOException {
        Map    map = request.getParameterMap();
        Model  model = null;
        String viewName = "";

        try {
            Class clazz = Class.forName("com.fastcampus.ch2.YoilTellerMVC");
            Object obj = clazz.newInstance();

                // 1. main메서드의 정보를 얻는다.
            Method main = clazz.getDeclaredMethod("main", int.class, int.class,
int.class, Model.class);

                // 2. main메서드의 매개변수 목록(paramArr)을 읽어서 메서드 호출에 사용
할 인자 목록(argArr)을 만든다.
            Parameter[] paramArr = main.getParameters();
            Object[] argArr = new Object[main.getParameterCount()];

            for(int i=0;i<paramArr.length;i++) {
                String paramName = paramArr[i].getName();
                Class  paramType = paramArr[i].getType();
                Object value = map.get(paramName);

                // paramType중에 Model이 있으면, 생성 & 저장
                if(paramType==Model.class) {
                    argArr[i] = model = new BindingAwareModelMap();
```

```
                } else if(paramType==HttpServletRequest.class) {
                    argArr[i] = request;
                } else if(paramType==HttpServletResponse.class) {
                    argArr[i] = response;
                } else if(value != null) {  // map에 paramName이 있으면,
                    // value와 parameter의 타입을 비교해서, 다르면 변환해서 저장
                    String strValue = ((String[])value)[0]; // getParameterMap()에
서 꺼낸 value는 String배열이므로 변환 필요
                    argArr[i] = convertTo(strValue, paramType);
                }
            }

            // 3. Controller의 main()을 호출 - YoilTellerMVC.main(int year, int
month, int day, Model model)
            viewName = (String)main.invoke(obj, argArr);
        } catch(Exception e) {
            e.printStackTrace();
        }

        // 4. 텍스트 파일을 이용한 rendering
        render(model, viewName, response);
    } // main

    private Object convertTo(Object value, Class type) {
        if(type==null || value==null || type.isInstance(value)) // 타입이 같으면 그
대로 반환
            return value;

        // 타입이 다르면, 변환해서 반환
        if(String.class.isInstance(value) && type==int.class) { // String -> int
            return Integer.valueOf((String)value);
        } else if(String.class.isInstance(value) && type==double.class) { //
String -> double
            return Double.valueOf((String)value);
        }

        return value;
    }

    private String getResolvedViewName(String viewName) {
        return getServletContext().getRealPath("/WEB-INF/views")
+"/"+viewName+".jsp";
    }

    private void render(Model model, String viewName, HttpServletResponse
response) throws IOException {
        String result = "";

        response.setContentType("text/html");
        response.setCharacterEncoding("utf-8");
        PrintWriter out = response.getWriter();

        // 1. 뷰의 내용을 한줄씩 읽어서 하나의 문자열로 만든다.
        Scanner sc = new Scanner(new File(getResolvedViewName(viewName)), "utf-
```

```
8");

        while(sc.hasNextLine())
            result += sc.nextLine()+ System.lineSeparator();

        // 2. model을 map으로 변환
        Map map = model.asMap();

        // 3.key를 하나씩 읽어서 template의 ${key}를 value바꾼다.
        Iterator it = map.keySet().iterator();

        while(it.hasNext()) {
            String key = (String)it.next();

            // 4. replace()로 key를 value 치환한다.
            result = result.replace("${"+key+"}", map.get(key)+"");
        }

        // 5.렌더링 결과를 출력한다.
        out.println(result);
    }
}
```