

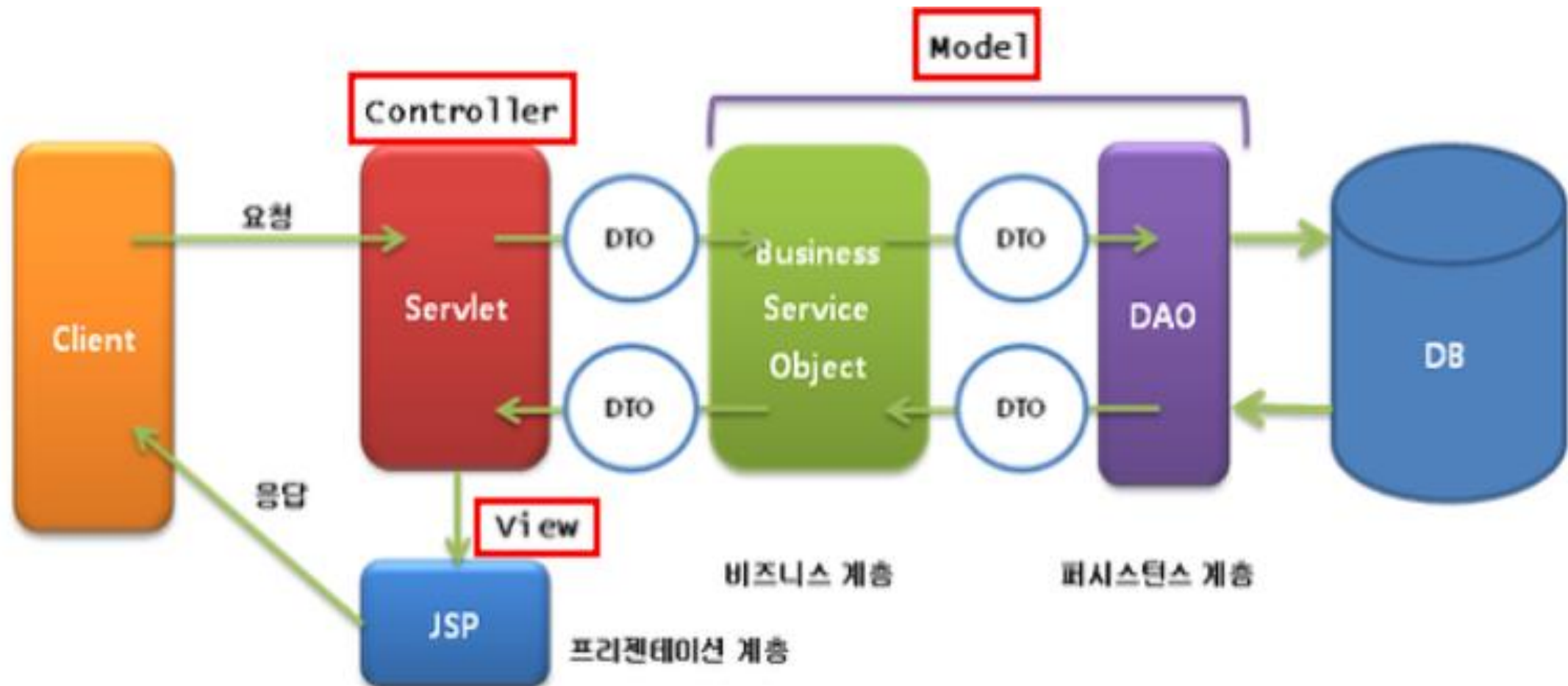
The background features a light blue gradient with three horizontal wavy bands in shades of blue. Three spheres are positioned on these bands: a large light blue sphere on the left, a small green sphere in the upper right, and a medium light blue sphere on the right.

# Java Servlet

안화수

# 모델2 설계방식

## ❖ MVC Architecture



# 모델2 설계방식

## ❖ MVC 패턴

MVC는 Model / View / Controller 의 약자로 애플리케이션을 세 역할로 나누어서 개발하는 개발 방법론이다.

### ➤ Model

- 애플리케이션의 데이터 처리를 담당함
- Service 클래스 + DAO 클래스로 구현함

### ➤ View

- 사용자 인터페이스를 처리함
- JSP 를 이용해서 구현함 : EL(Expression Language, 표현언어) + JSTL(JSP Standard Tag Library)

### ➤ Controller

- 클라이언트의 요청을 받아 Model과 View사이에서 흐름을 조정한다.
- Java Servlet으로 구현함

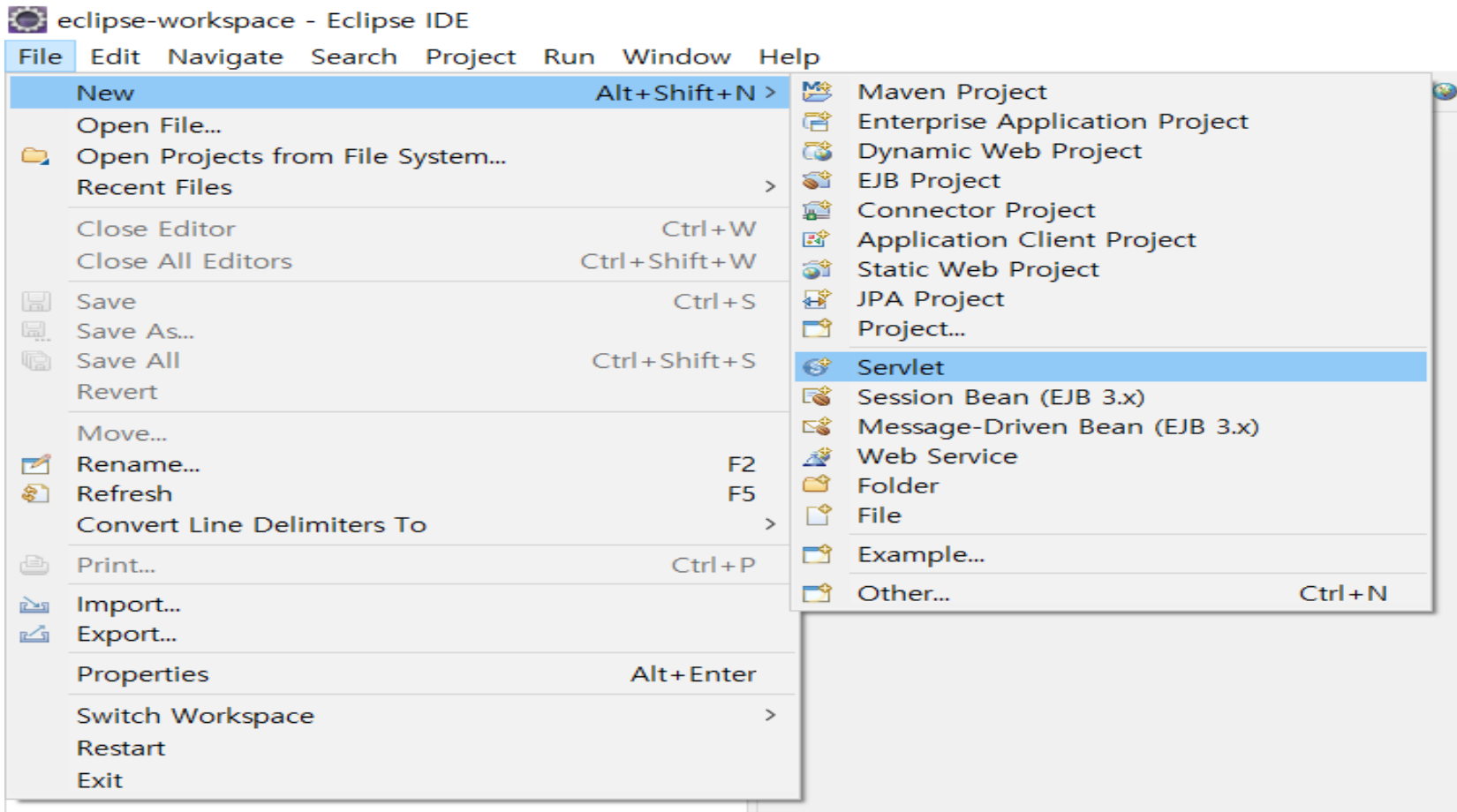
# Java Servlet

## ❖ Java Servlet 클래스

- Java로 작성된 웹프로그램을 의미함
- Java Servlet 클래스에는 HTML, JavaScript 코드를 포함할 수 있다.
- Java Servlet 클래스는 웹브라우저로 실행 결과를 출력할 수 있다.

# Java Servlet

## ❖ Java Servlet 클래스 생성



# Java Servlet

## ❖ Java Servlet 클래스 생성

Create Servlet

**Create Servlet**

Specify class file destination.

Project: jspproject2

Source folder: /jspproject2/src/main/java Browse...

Java package: send Browse...

Class name: HelloWorld

Superclass: javax.servlet.http.HttpServlet Browse...

☐ Use an existing Servlet class or JSP

Class name: HelloWorld Browse...

? < Back **Next >** Finish Cancel

# Java Servlet

## ❖ Java Servlet 클래스 생성

Create Servlet

**Create Servlet**

Enter servlet deployment descriptor specific information.

Name: HelloWorld

Description: 처음 작성하는 자바서블릿

Initialization parameters:

Name	Value	Description
------	-------	-------------

URL mappings:

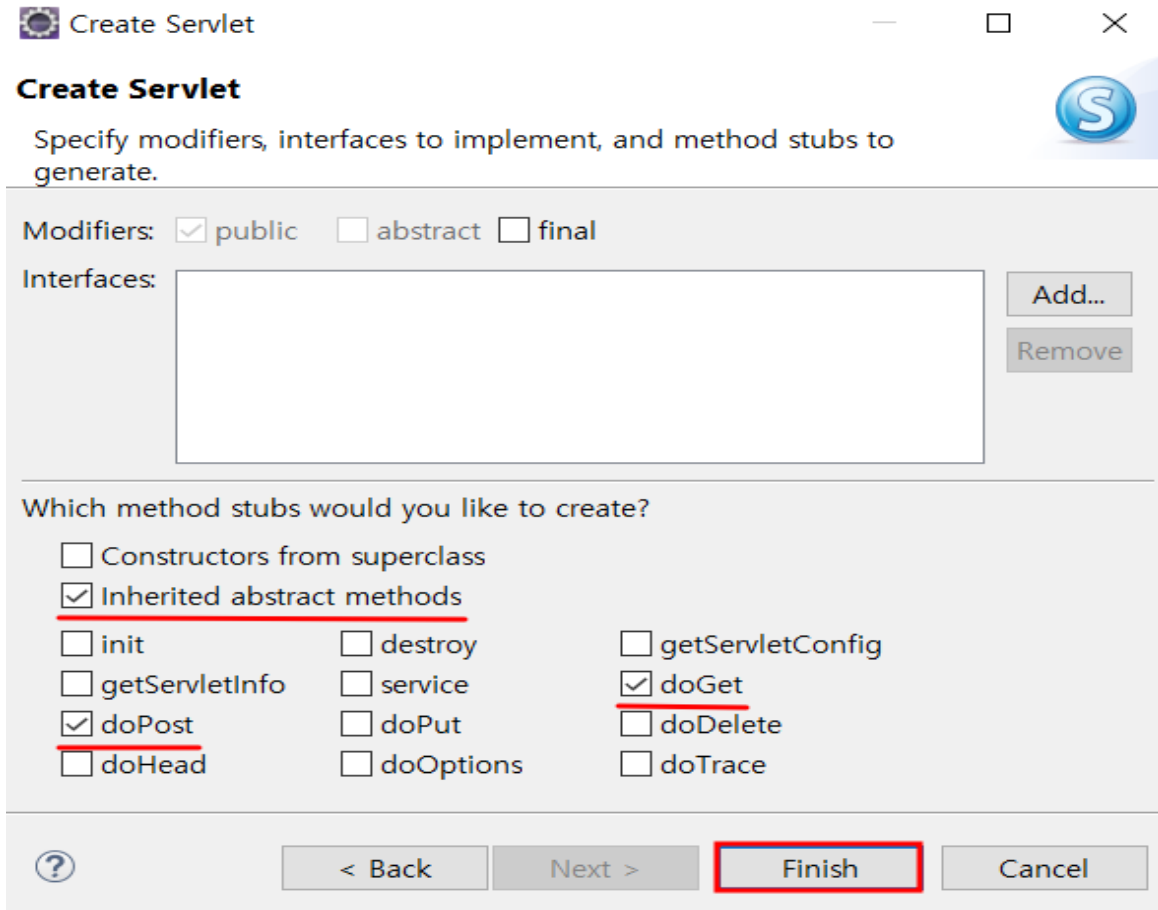
/HelloWorld
-------------

☐ Asynchronous Support

< Back **Next >** Finish Cancel

# Java Servlet

## ❖ Java Servlet 클래스 생성



**Create Servlet**

Specify modifiers, interfaces to implement, and method stubs to generate.

Modifiers: ☒ public ☐ abstract ☐ final

Interfaces:  Add... Remove

Which method stubs would you like to create?















<input type="checkbox"/> Constructors from superclass		
<input checked="" type="checkbox"/> <u>Inherited abstract methods</u>		
<input type="checkbox"/> init	<input type="checkbox"/> destroy	<input type="checkbox"/> getServletConfig
<input type="checkbox"/> getServletInfo	<input type="checkbox"/> service	<input checked="" type="checkbox"/> <u>doGet</u>
<input checked="" type="checkbox"/> <u>doPost</u>	<input type="checkbox"/> doPut	<input type="checkbox"/> doDelete
<input type="checkbox"/> doHead	<input type="checkbox"/> doOptions	<input type="checkbox"/> doTrace

? < Back Next > **Finish** Cancel



# Java Servlet

## ❖ Java Servlet 클래스

-  jspproject2
  - >  Deployment Descriptor: jspproject2
  - >  JAX-WS Web Services
  - >  Java Resources
  - >  build
  - ▼  src
    - ▼  main
      - ▼  java
        - ▼  send
          -  HelloWorld.java
      - ▼  webapp
        - >  META-INF
        - >  servlet
        - >  WEB-INF

# Java Servlet

```
package send;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

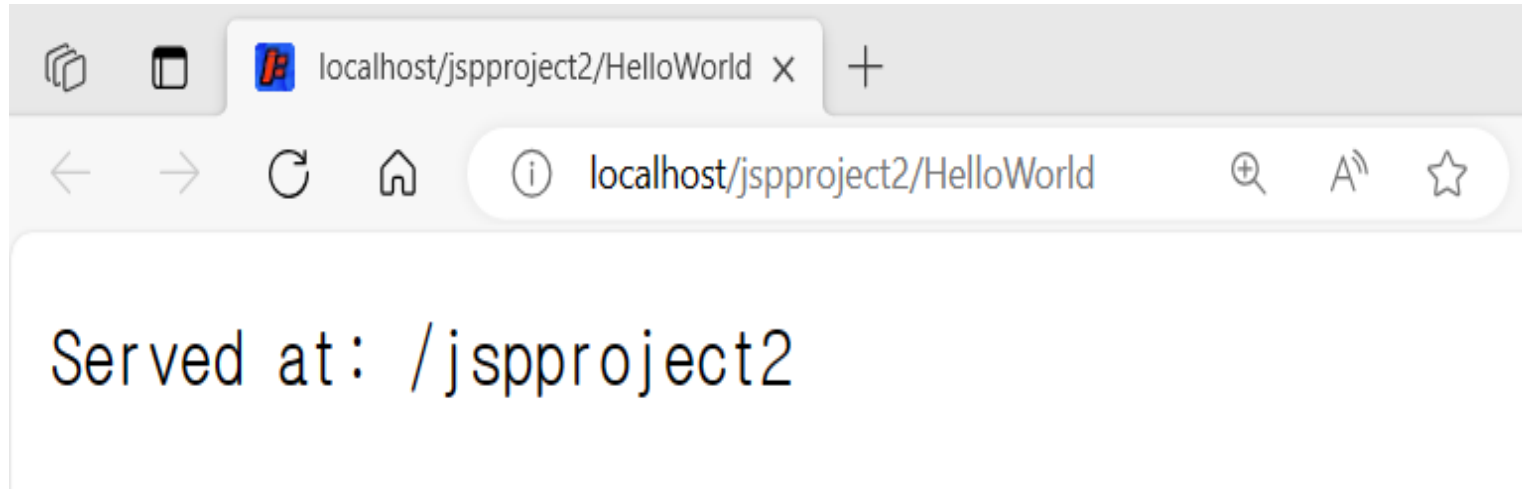
/**
 * Servlet implementation class HelloWorld
 */
@WebServlet(description = "처음 작성하는 자바서블릿", urlPatterns = { "/HelloWorld" })
public class HelloWorld extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
     * @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);
    }
}
```

# Java Servlet

## ❖ Java Servlet 클래스 실행 결과



# Java Servlet

## ❖ Java Servlet 클래스로 찾아 가는 방법

방법1. @WebServlet("/HelloWorld") 어노테이션

방법2. web.xml 파일의 서블릿 매핑

```
<servlet>
    <description>값전달연습</description>
    <display-name>HelloWorld</display-name>
    <servlet-name>HelloWorld</servlet-name>
    <servlet-class>send.HelloWorld</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>HelloWorld</servlet-name>
    <url-pattern>/HelloWorld</url-pattern>
</servlet-mapping>
```

# Java Servlet

## ❖ Java Servlet 클래스에서 JSP페이지로 포워딩 방법

### 1. Dispatcher 방식

```
RequestDispatcher dispatcher = request.getRequestDispatcher("dispatcher.jsp");  
dispatcher.forward(request, response);
```

### 2. Redirect 방식

```
response.sendRedirect("redirect.jsp");
```

# Java Servlet

## ❖ Java Servlet 클래스에서 JSP페이지로 포워딩 방법 : Dispatcher 방식

### DispatcherServlet.java

```
request.setAttribute("request","requestValue");           // 공유 설정  
RequestDispatcher dispatcher = request.getRequestDispatcher("dispatcher.jsp");  
dispatcher.forward(request, response);
```

### dispatcher.jsp

```
request 속성 값 : <%=request.getAttribute("request") %> // 공유한 값 구해오기
```

1. 자바 서블릿 클래스에서 request객체로 공유 설정후 dispatcher 방식으로 JSP파일로 포워딩하면 JSP파일에서 request객체로 공유한 곳에 접근할 수 있다. ( request 영역 )
2. dispatcher 방식으로 JSP파일로 포워딩하면 URL주소가 바뀌지 않는다.

# Java Servlet

## ❖ Java Servlet 클래스에서 JSP페이지로 포워딩 방법 : Redirect 방식

### RedirectServlet.java

```
request.setAttribute("request","requestValue");           // 공유 설정  
response.sendRedirect("redirect.jsp");
```

### redirect.jsp

request 속성 값 : <%=request.getAttribute("request") %> // 공유한 값 구해오기

1. 자바 서블릿 클래스에서 request객체로 공유 설정후 redirect 방식으로 JSP파일로 포워딩하면 JSP파일에서 request객체로 공유한 곳에 접근할 수 없다. ( request 영역을 벗어남 )
2. redirect 방식으로 JSP파일로 포워딩하면 URL주소가 바뀐다.