

JSP 4강

강사 | 최정호

INDEX

1. Servlet

Servlet

>> Servlet 이란?



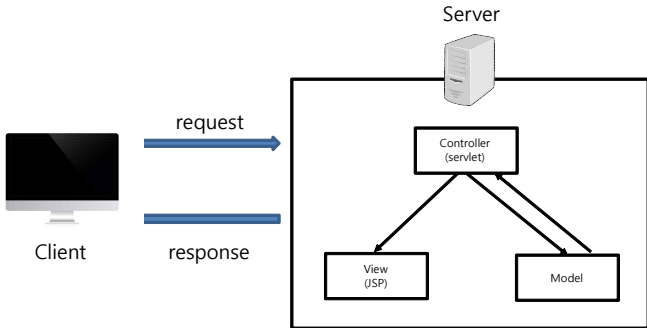
- JSP표준이 나오기 전에 만들어진 표준으로 자바 를 이용해 웹어플리케이션을 개발할 수 있도록 하는 게 목적

01

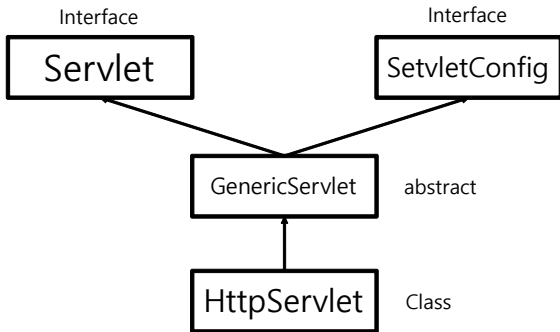
02

03

» JSP 페이지 처리과정



>> JSP 페이지 처리과정

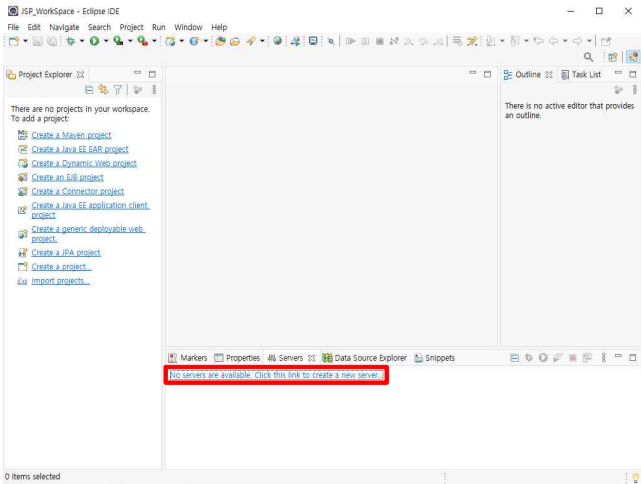


Servlet

01

02

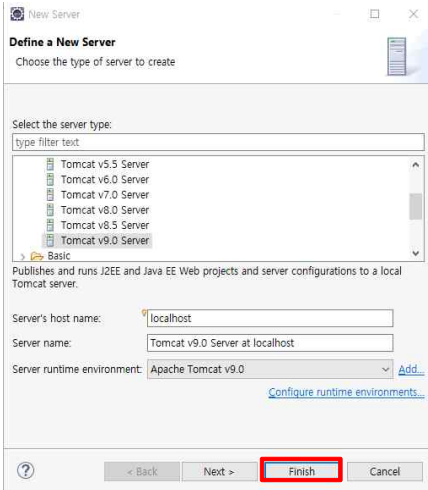
03



01

02

03



New Server

Define a New Server

Choose the type of server to create

Select the server type:

type filter text

- Tomcat v5.5 Server
- Tomcat v6.0 Server
- Tomcat v7.0 Server
- Tomcat v8.0 Server
- Tomcat v8.5 Server
- Tomcat v9.0 Server**

> Basic

Publishes and runs J2EE and Java EE Web projects and server configurations to a local Tomcat server.

Server's host name: localhost

Server name: Tomcat v9.0 Server at localhost

Server runtime environment: Apache Tomcat v9.0 [Add...](#)

[Configure runtime environments...](#)

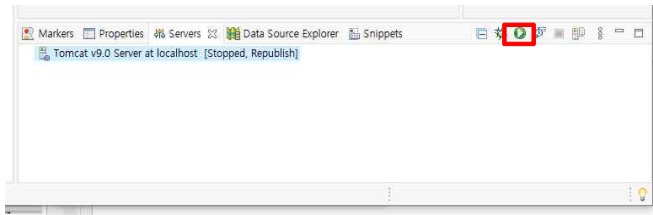
? < Back Next > **Finish** Cancel

Servlet

01

02

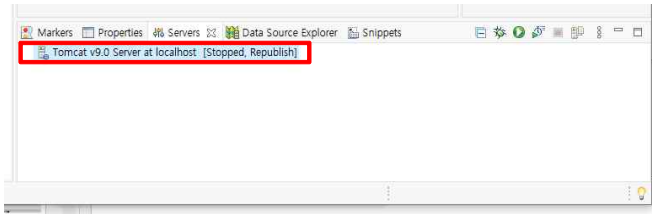
03



01

02

03



01

02

03

Overview

General Information
Specify the host name and other common settings.

Server name:

Tomcat v9.0 Server at localhost

Host name:

localhost

Runtime Environment:

Apache Tomcat v9.0

Configuration path:

/Servers/Tomcat v9.0 Server at loca

Browse...

[Open launch configuration](#)

Server Locations
Specify the server path (i.e. catalina.base) and deploy path. Server must be published with no modules present to make changes.

☐ Use workspace metadata (does not modify Tomcat installation)

☒ Use Tomcat installation (takes control of Tomcat installation)

☐ Use custom location (does not modify Tomcat installation)

Server path:

C:\Tomcat 9.0

Browse...

► Publish

► Time

▼ Ports
Modify t

Port M

► To

► HT

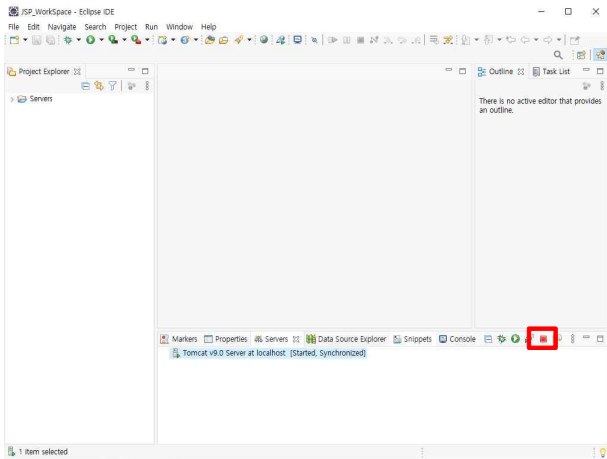
► MIMI

Servlet

01

02

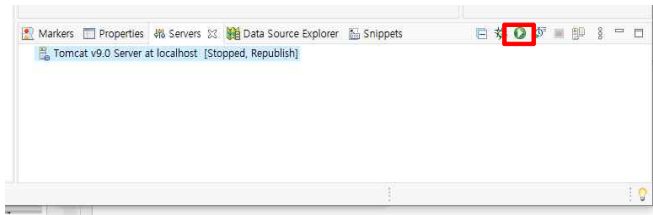
03



01

02

03



Apache Tomcat/9.0.37 Tomcat v9.0 Server at localhost


<http://localhost:8080/>

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.37

APACHE SOFTWARE FOUNDATION
<http://www.apache.org/>

If you're seeing this, you've successfully installed Tomcat. Congratulations!



Recommended Reading:

- [Security Considerations How-To](#)
- [Manager Application How-To](#)
- [Clustering/Session Replication How-To](#)

Server Status
Manager App
Host Manager

Developer Quick Start

- [Tomcat Setup](#)
- [Realms & AAA](#)
- [Examples](#)
- [Servlet Specifications](#)
- [First Web Application](#)
- [JDBC DataSources](#)
- [Tomcat Versions](#)

Managing Tomcat

For security, access to the [manager webapp](#) is restricted. Users are defined in:

```
SCATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager application is split between different users.
[Read more...](#)

Documentation

- [Tomcat 9.0 Documentation](#)
- [Tomcat 9.0 Configuration](#)
- [Tomcat Wiki](#)

Find additional important configuration information in:

```
SCATALINA_HOME/RUNNING.txt
```

Getting Help

FAQ and Mailing Lists

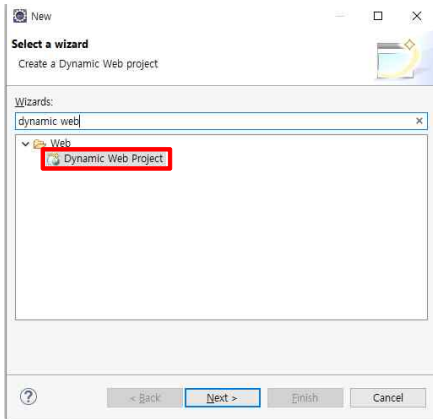
The following mailing lists are available:

- [tomcat-announce](#)
Important announcements, releases, security vulnerability notifications. (Low volume).
- [tomcat-users](#)
User support and discussion

01

02

03



01

02

03

New Dynamic Web Project

Dynamic Web Project
Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: JSP_Servlet_tutorial

Project location
☒ Use default location
Location: D:\JSP_WorkSpace\JSP_Servlet_tutorial [Browse...](#)

Target runtime
<None> [New Runtime...](#)

Dynamic web module version
3.0

Configuration
Default Configuration [Modify...](#)
The default configuration provides a good starting point. Additional facets can later be installed to add new functionality to the project.

EAR membership
☐ Add project to an EAR
EAR project name: EAR [New Project...](#)

Working sets
☐ Add project to working sets [New...](#)
Working sets: [Select...](#)

[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

Servlet

01

02

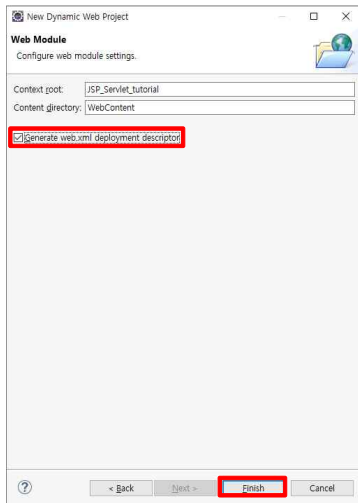
03



01

02

03

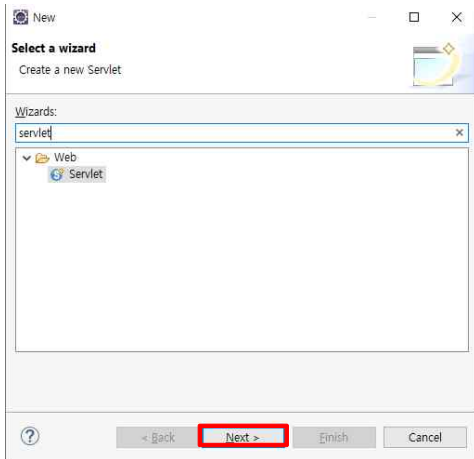


Servlet

01

02

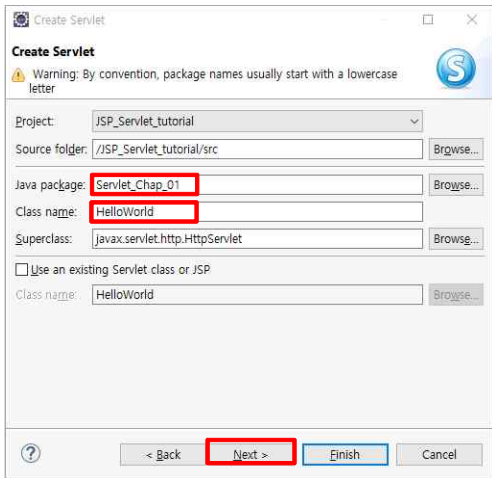
03



01

02

03



The image shows a 'Create Servlet' dialog box from an IDE. It contains several input fields and buttons. The 'Project' field is set to 'JSP_Servlet_tutorial'. The 'Source folder' is '/JSP_Servlet_tutorial/src'. The 'Java package' is 'Servlet_Chap_01' and the 'Class name' is 'HelloWorld', both of which are highlighted with red rectangles. The 'Superclass' is 'javax.servlet.http.HttpServlet'. There is a checkbox for 'Use an existing Servlet class or JSP' which is unchecked, and its 'Class name' is also 'HelloWorld'. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a red rectangle.

Create Servlet

Create Servlet

Warning: By convention, package names usually start with a lowercase letter

Project: JSP_Servlet_tutorial

Source folder: /JSP_Servlet_tutorial/src Browse...

Java package: Servlet_Chap_01 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

01

02

03

 Create Servlet



Create Servlet

Enter servlet deployment descriptor specific information.

Name:


Description:

Initialization parameters:

Name	Value	Description

URL mappings:

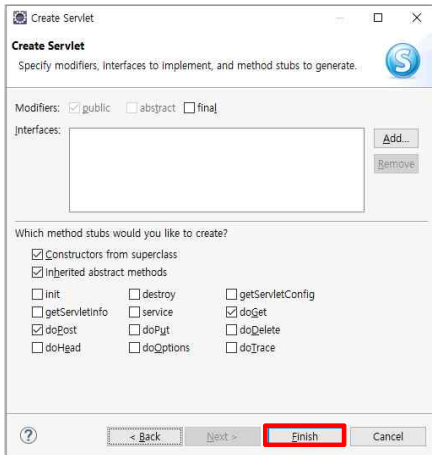
☐ Asynchronous Support



01

02

03



The image shows a 'Create Servlet' dialog box from an IDE. It has a title bar with a gear icon and the text 'Create Servlet'. Below the title bar is a section titled 'Create Servlet' with a description: 'Specify modifiers, interfaces to implement, and method stubs to generate.' To the right of this text is a blue circular icon with a white 'S'. The dialog is divided into three main sections. The first section is for 'Modifiers' with three checkboxes: 'public' (checked), 'abstract' (unchecked), and 'final' (unchecked). The second section is for 'Interfaces' with a large empty text box and two buttons: 'Add...' and 'Remove'. The third section is titled 'Which method stubs would you like to create?' and contains a grid of checkboxes. The first two rows have checkboxes for 'Constructors from superclass' and 'Inherited abstract methods', both of which are checked. The remaining rows have checkboxes for various methods: 'init', 'destroy', 'getServletConfig', 'getServletInfo', 'service', 'doGet' (checked), 'doPost' (checked), 'doPut', 'doDelete', 'doHead', 'doOptions', and 'doTrace'. At the bottom of the dialog is a footer bar with a question mark icon, a '< Back' button, a 'Next >' button, an 'Finish' button (which is highlighted with a red rectangle), and a 'Cancel' button.

Create Servlet

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?

☒ Constructors from superclass

☒ Inherited abstract methods

☐ init ☐ destroy ☐ getServletConfig

☐ getServletInfo ☐ service ☒ doGet

☒ doPost ☐ doPut ☐ doDelete

☐ doHead ☐ doOptions ☐ doTrace

? < Back Next > Finish Cancel

Servlet

```
package Tutorial_Chap01;
```

```
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 Hello
 */
```

```
@WebServlet("/Hello")
```

```
public class Hello extends HttpServlet {
    private static final long serialVersionUID = 1L;
```

```
    /**
     * @see HttpServlet#HttpServlet()
     */
    public Hello() {
        super();
        // TODO Auto-generated constructor stub
    }
```

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

```
}
```

01

02

03

01

02

03

```
@WebServlet("/Now")
public class NowTime extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public NowTime() {
        super();
        // TODO Auto-generated constructor stub
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        response.setCharacterEncoding("utf-8");//문자 인코딩
        response.setContentType("text/html; charset=utf-8");//응답할 내용 부분 설정

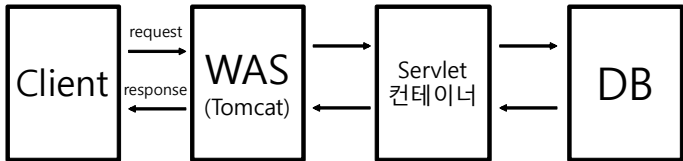
        //웹 화면으로 출력을 하게 해줄 출력 스트림 얻음
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title>현재 시간 </title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>지금 시간은 </h1>");
        out.println(new Date());
        out.println("<h1>입니다. </h1>");
        out.println("</body>");
        out.println("</head>");
        out.println("</html>");
        out.close();
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    }
}
```


01

02

03



Client 의 요청이 들어오면 서버에서 받아서 처리한다.
여기서 중요한 것은 요청처리객체 및 응답처리객체인
request 와 **response** 를 Tomcat 서버가 직접 생성해 준다.
WAS(Wap Application Server)

01

02

03

- Doget 방식
 - url값으로 정보가 전송되어 **보안에 약함**(아이디,비밀번호)
 - url 경로 뒤 물음표(?)와 함께 파라미터를 붙여 전송함.
 - 각각의 파라미터를 & 으로 구분함
- Dopost 방식
 - heade를 이용해 정보가 전송되어 **보안에 강함**

01

02

03

- DoGetorDoPost 활용 예제 작성

01

02

03

• 로그인 활용 예제 코드

01

02

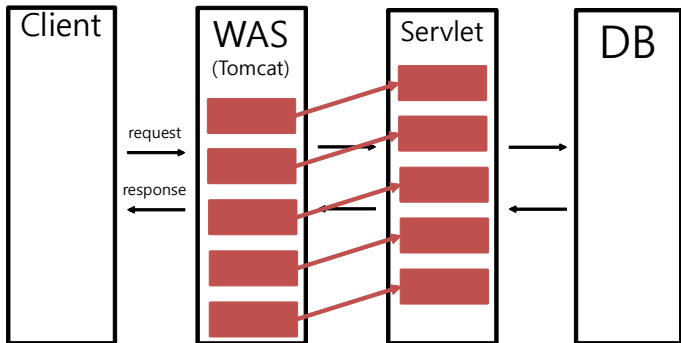
03

• 이미지 활용 코드 예제

01

02

03



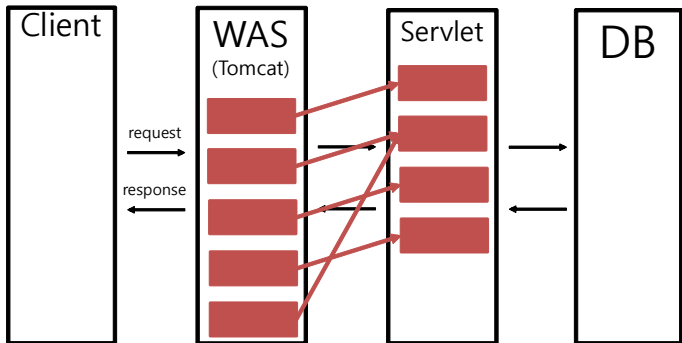
쓰레드 풀(Thread pool)

기존에 요청이 들어올 시 쓰레드를 **필요할 때 마다** 만들기 때문에
연산 및 속도 저하가 심하다
그래서 쓰레드 수를 **제한시켜 재사용성**을 올리는게 목적

01

02

03



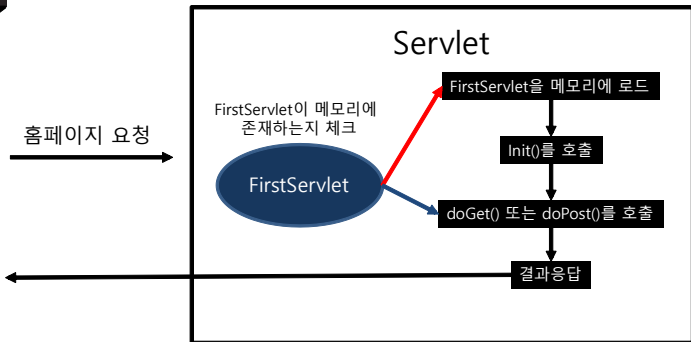
쓰레드 풀(Thread pool)

기존에 요청이 들어올 시 쓰레드를 **필요할 때 마다** 만들기 때문에
연산 및 속도 저하가 심하다
그래서 쓰레드 수를 **제한시켜 재사용성**을 올리는게 목적

01

02

03



쓰레드 풀(Thread pool)

기존에 요청이 들어올 시 쓰레드를 **필요할 때 마다** 만들기 때문에
연산 및 속도 저하가 심하다
그래서 쓰레드 수를 **제한시켜 재사용성**을 올리는게 목적

01

02

03

Servlet 생명주기(Life Cycle)

Servlet 객체 생성

선처리
@PostConstruct

Init() 호출

service(),doGet(),doPost()를 호출

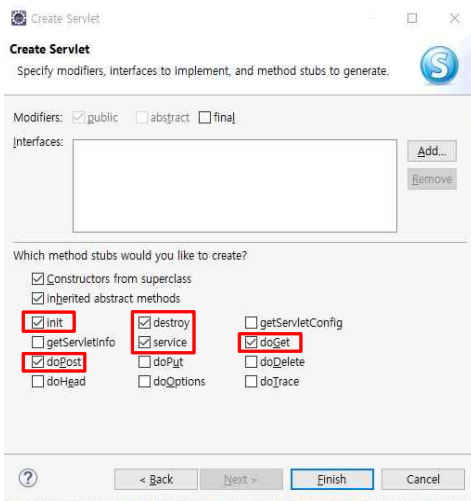
destroy() 호출

후처리
@PreDestroy

01

02

03



The image shows a 'Create Servlet' dialog box from an IDE. It has a title bar with a gear icon and the text 'Create Servlet'. Below the title bar, the text 'Create Servlet' is followed by a description: 'Specify modifiers, interfaces to implement, and method stubs to generate.' To the right of this text is a blue circular icon with a white 'S'. The dialog is divided into several sections. The 'Modifiers' section has three checkboxes: 'public' (checked), 'abstract' (unchecked), and 'final' (unchecked). The 'Interfaces' section has a text box and two buttons: 'Add...' and 'Remove...'. The 'Which method stubs would you like to create?' section has a list of checkboxes. The checked items are: 'Constructors from superclass', 'Inherited abstract methods', 'init', 'destroy', 'service', 'doGet', 'doPost', and 'doHead'. The unchecked items are: 'getServletInfo', 'doPut', 'doOptions', 'doDelete', 'doTrace', and 'getServletConfig'. At the bottom, there are four buttons: a help button (question mark), '< Back', 'Next >', and 'Finish' (highlighted with a blue border), and a 'Cancel' button.

Create Servlet

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?

☒ Constructors from superclass

☒ Inherited abstract methods

☒ init ☒ destroy ☐ getServletConfig

☐ getServletInfo ☒ service ☒ doGet

☒ doPost ☐ doPut ☐ doDelete

☐ doHead ☐ doOptions ☐ doTrace

? < Back Next > Finish Cancel

01

02

03

• Servlet LifeCycle 예제코드

01

02

03

• 회원가입 예제코드 작성

01

02

03

Quiz

GuGuDan 출력 페이지 만들기

← → ↻ ⓘ localhost:8080/Servlet_tutorial/gugudan.jsp ☆

출력할 구구단의 수를 입력해 주세요.

구구단 시작 값 입력 : 구구단 끝 값 입력 :

← → ↻ ⓘ localhost:8080/Servlet_tutorial/gugudan?start=2&end=5 ☆

2단	3단	4단	5단
2 X 1 = 2	3 X 1 = 3	4 X 1 = 4	5 X 1 = 5
2 X 2 = 4	3 X 2 = 6	4 X 2 = 8	5 X 2 = 10
2 X 3 = 6	3 X 3 = 9	4 X 3 = 12	5 X 3 = 15
2 X 4 = 8	3 X 4 = 12	4 X 4 = 16	5 X 4 = 20
2 X 5 = 10	3 X 5 = 15	4 X 5 = 20	5 X 5 = 25
2 X 6 = 12	3 X 6 = 18	4 X 6 = 24	5 X 6 = 30
2 X 7 = 14	3 X 7 = 21	4 X 7 = 28	5 X 7 = 35
2 X 8 = 16	3 X 8 = 24	4 X 8 = 32	5 X 8 = 40
2 X 9 = 18	3 X 9 = 27	4 X 9 = 36	5 X 9 = 45

01

02

03

>> JavaScript 란?



JavaScript

- JSP에서 사용되는 JavaScript는 기본적으로 **유효성 검사**를 하는데 주로 사용한다.

01

02

03

• 자바 스크립트 예제 작성

01

02

03

Quiz

Login2.jsp에서 보낸 값
Servlet 파일 만들어 웹에 출력하기

Thank you

강사 | 최정호