



# JSP/Servlet

ch1. Servlet 개요/데이터 전송

1. Servlet의 사용 목적을 이해할 수 있다.
2. Request와 Response를 이해할 수 있다.

# 자바 웹 개발환경

- 자바 웹 개발을 위한 기본적인 도구
  - 자바 개발도구(JDK), 통합개발환경(IDE), 서블릿 컨테이너, 데이터베이스
- 구축 개발환경의 버전

항목	프로그래밍	버전
자바개발도구	JDK	JDK11
통합개발환경	이클립스	Eclipse IDE for Java Enterprise Developer
서블릿 컨테이너	아파치 톰캣	Apache Tomcat 9
데이터베이스	MySQL	8.0

- Network = Net + Work
- 구성: Node + Link

"그물처럼 서로 엮여서 일하는 것"

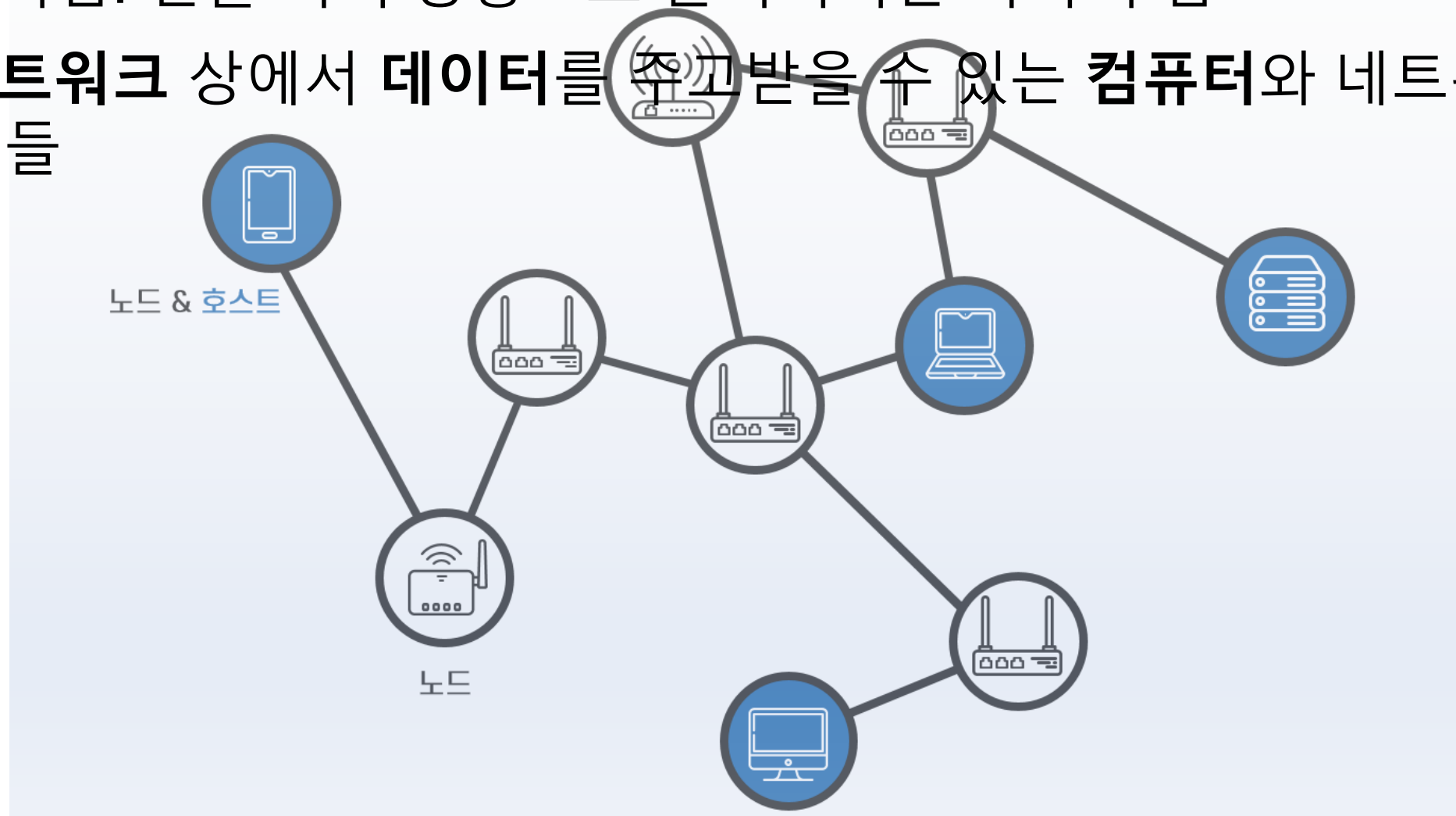
통신 장비들이 그물망처럼 연결되어  
데이터를 교환하는 형태



## 용어 익히기 >> 노드

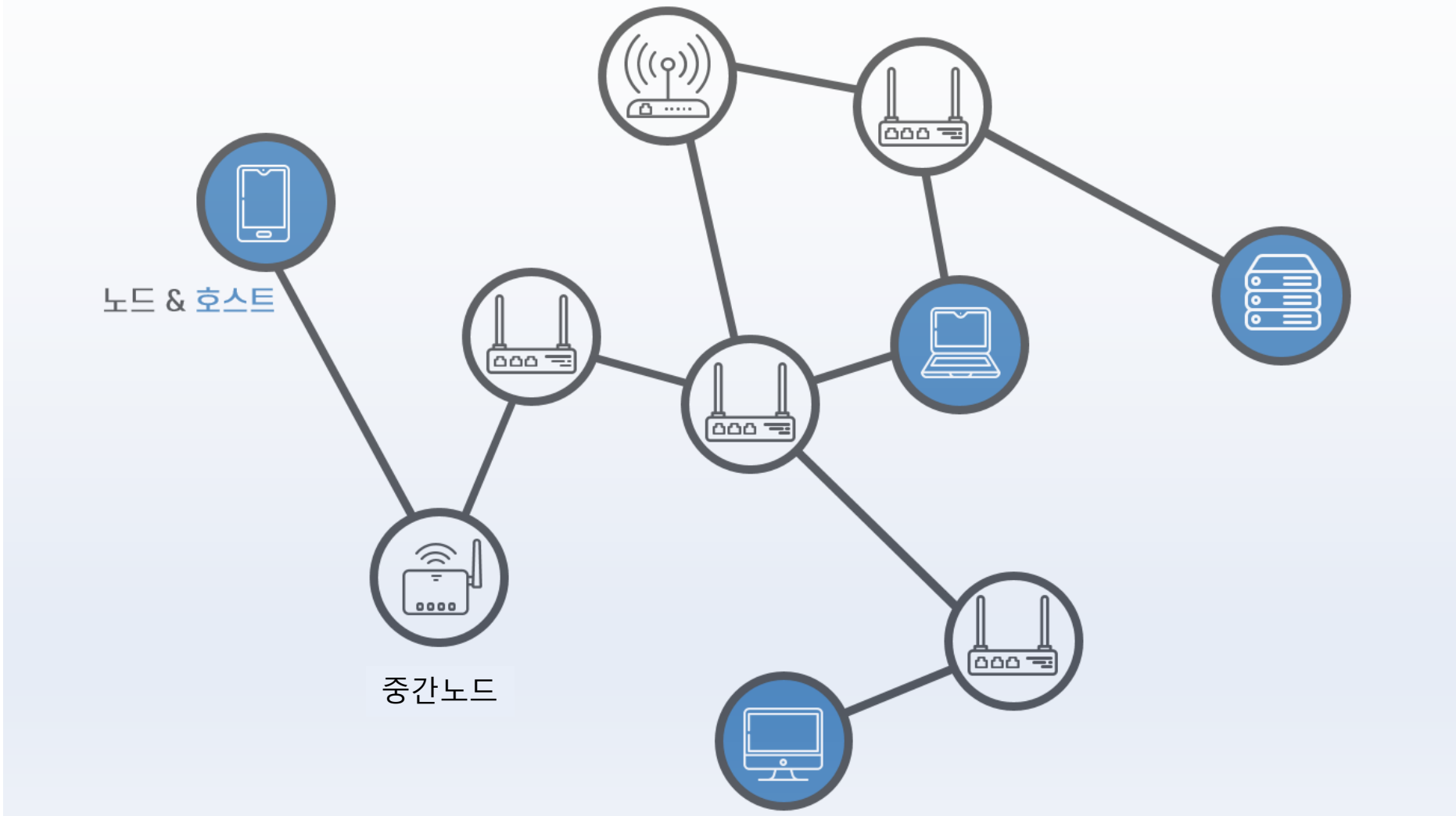
= 분기점: 선을 여러 방향으로 분기시키는 각각의 점

= 네트워크 상에서 데이터를 주고받을 수 있는 컴퓨터와 네트워크 장비들



## 용어 익히기 >> 호스트

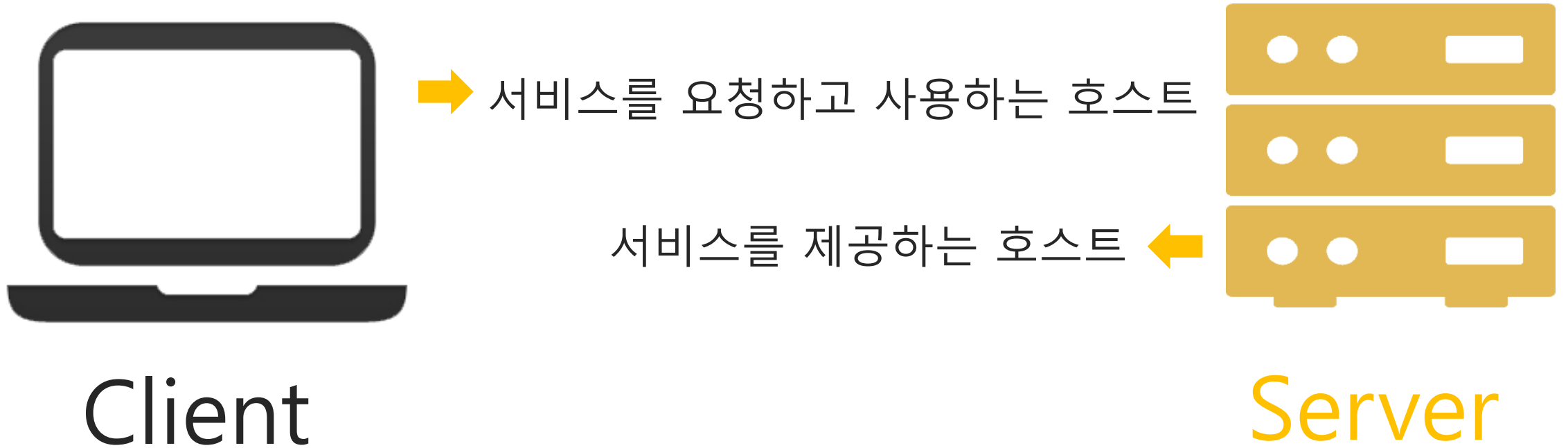
= 노드 중에서 애플리케이션을 실행할 수 있는 컴퓨팅 시스템을 갖춘 기기



그림참조: <https://better-together.tistory.com/74>

## 용어익히기 >> Client와 Server

- 호스트 사이에 제공되는 서비스를 기준으로 호스트를 세분화
- **이용하는 서비스의 종류에 따라 클라이언트가 될 수도 있고 서버가 될 수도 있음**



노드



호스트



클라이언트



① 서비스 요청



네트워크



② 서비스 제공



노드



호스트



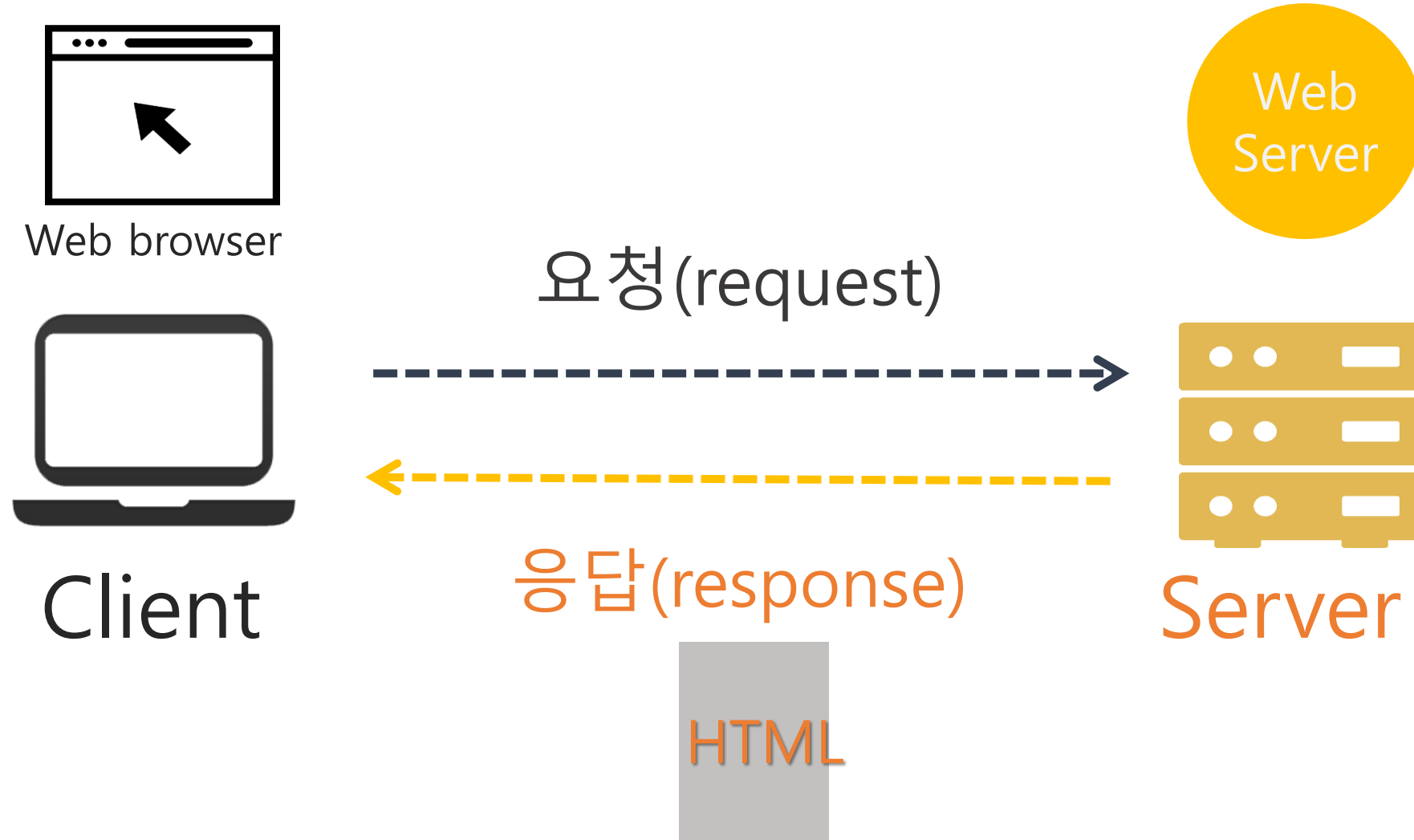
서버





# 용어익히기>> 웹 Client와 웹Server

- HTTP 프로토콜 기반의 호스트



# Web Server의 종류



1. 오픈소스
2. 다양한 모듈 제공
3. 강력한 커뮤니티, 다양한 자료
4. 확장성 좋음
5. 보안 수준 높음



# 웹 페이지의 종류

---

- 정적페이지(Static Web Page)

- 서버(웹 서버)에 미리 저장된 파일(HTML, 이미지, JavaScript 파일 등)이 그대로 전달되는 웹페이지
- 사용자는 서버에 저장된 데이터가 변경되지 않는 한 고정된 웹 페이지를 보게 됨

- 동적페이지(Dynamic Web Page)

- 서버(웹 서버)에 있는 데이터들을 가공 처리하여 생성된 결과를 전달하는 웹페이지
- 사용자는 상황, 시간, 요청 등에 따라 달라지는 웹 페이지를 보게 됨



정적 웹 서버

HTTP 서버(소프트웨어)가 있는  
컴퓨터(하드웨어)로 구성



Server

서버에 저장되어 있는 파일을  
클라이언트에 전송

“정적”

클라이언트의 요청에 맞는  
HTML 파일을  
만들어 주는 서버프로그램이 있다면?

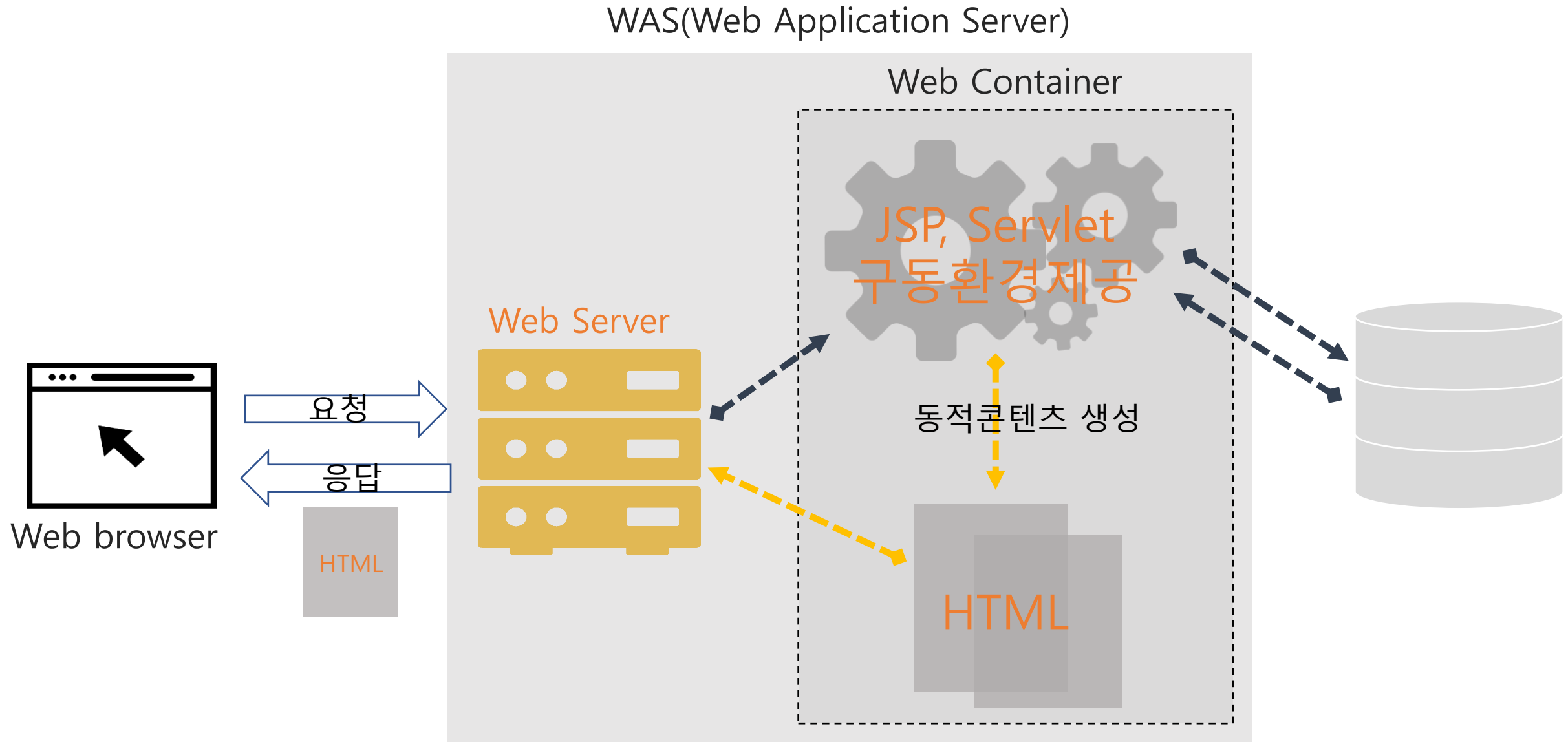
# 서버사이드 스크립트 언어



프로그램이 동작되기 위한  
작업을 서버에서 제공

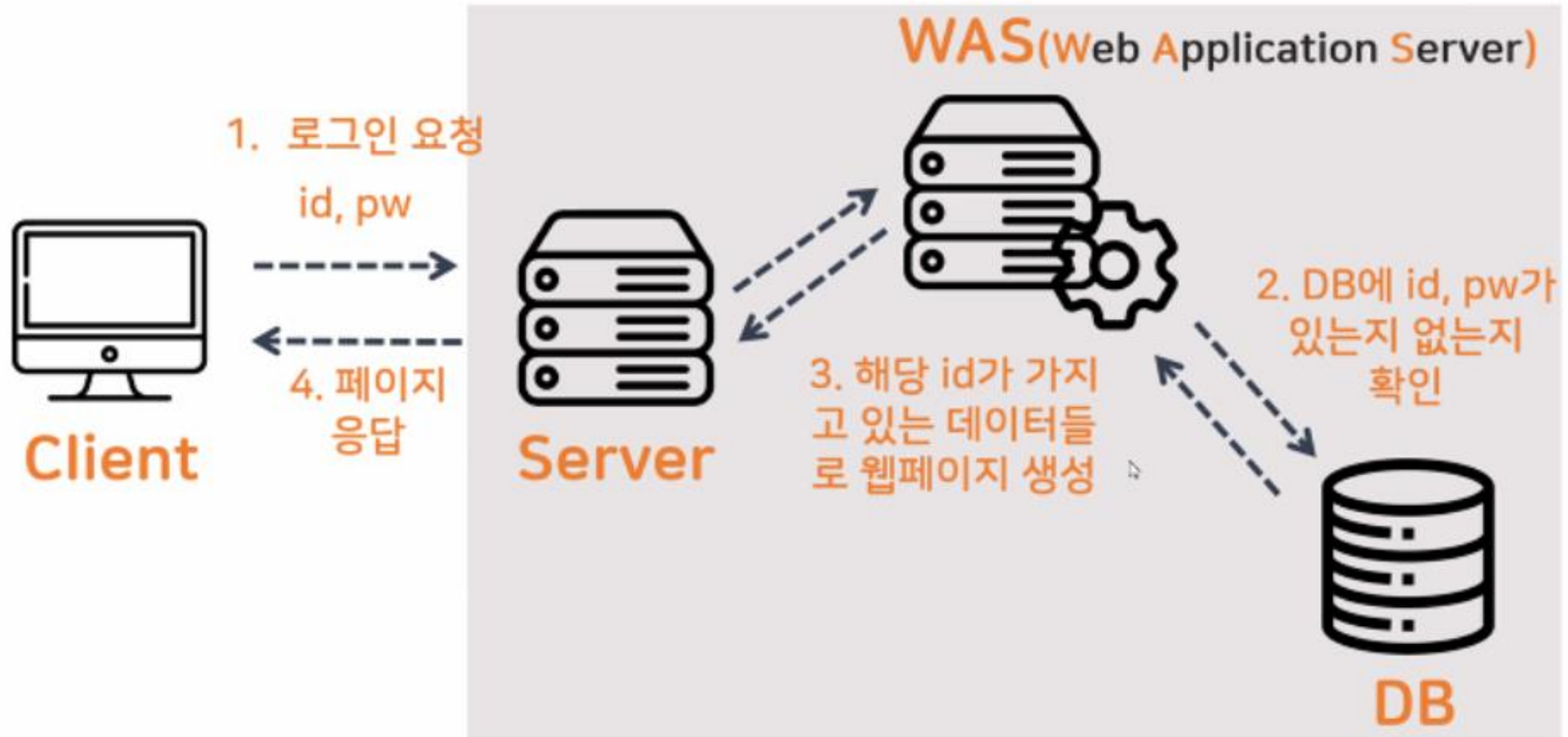
- Servlet=Server + Applet
  - Java를 기반으로 하는 Web Application Programming 기술
  - Client 요청에 동적으로 작동, 응답은 HTML 형식으로 제공
- 서블릿 컨테이너(=Jsp 컨테이너)
  - 서블릿을 실행하기 위한 서버 소프트웨어
  - JSP 나 서블릿을 실행하기 위한 환경

# Web Server와 WAS





# Servlet 동작 과정: 로그인 요청~응답

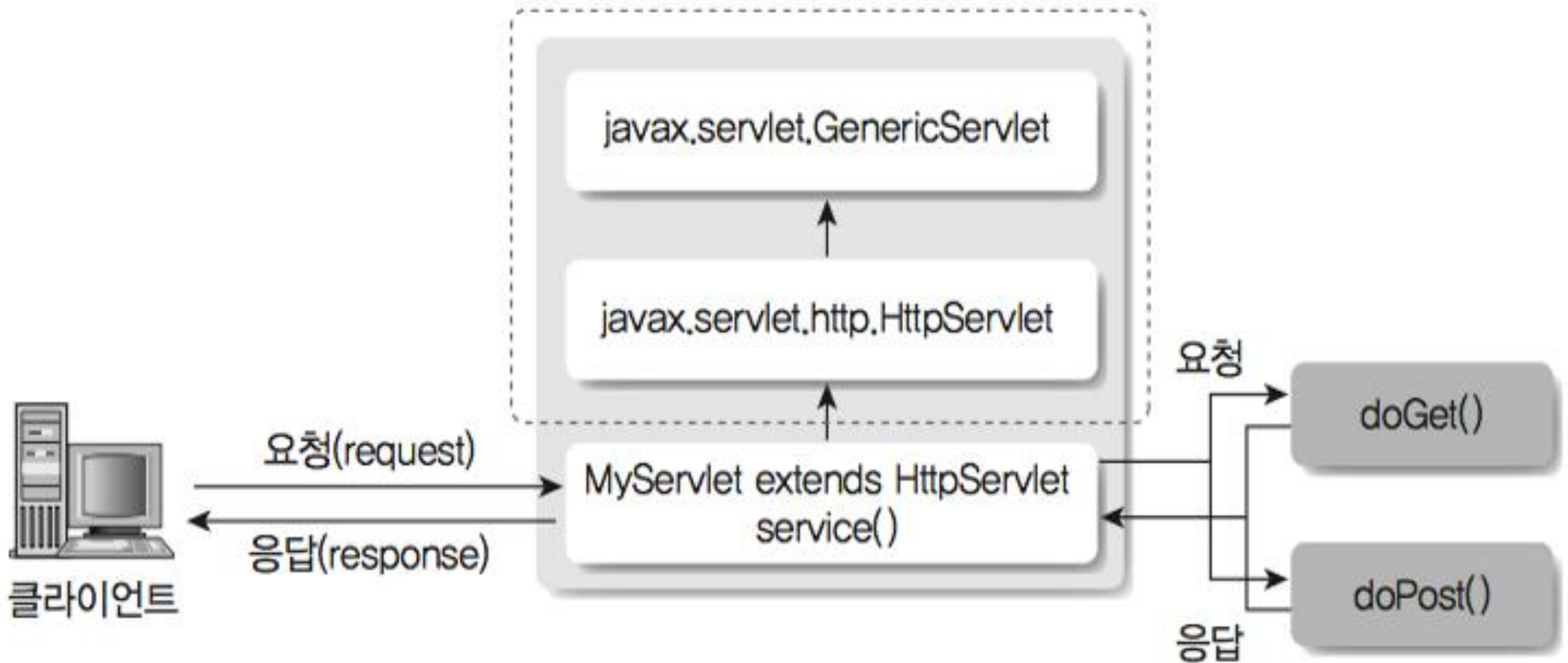


# Servlet 특징

---

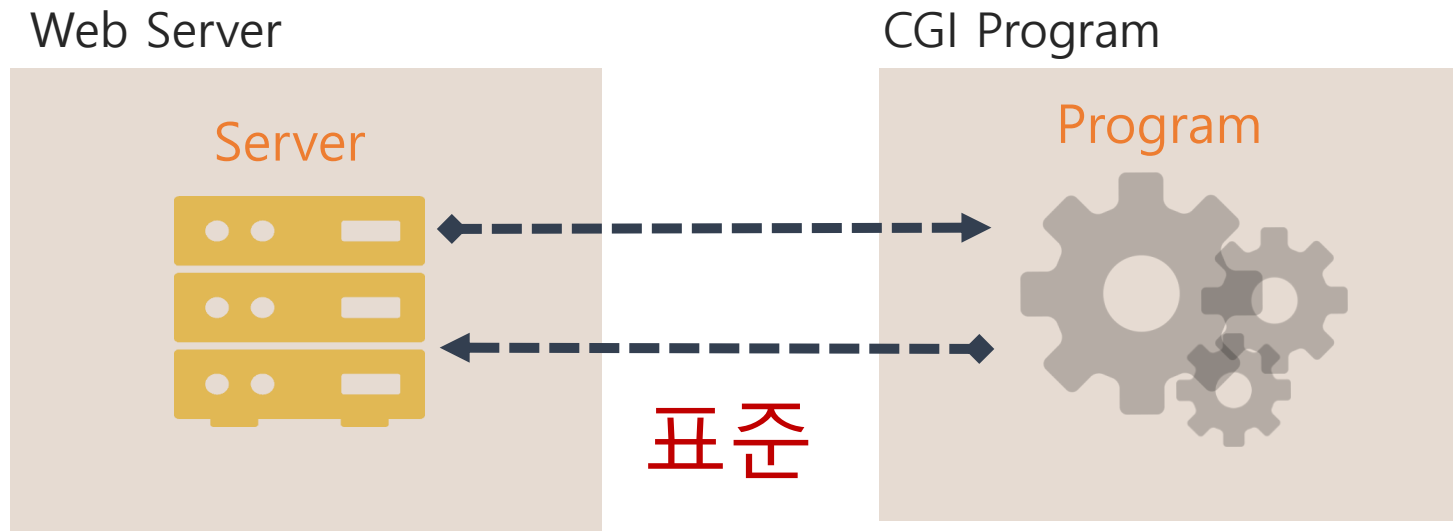
- .java 확장자를 가짐(컴파일 과정 필요)
- Java Multi Thread를 이용하여 동작함
  - 속도와 메모리 면에서 효율적임
- 객체지향적 -> 대규모 Application 개발에 적합함
- `java.servlet.http.HttpServlet` 클래스를 상속받음.
  - 사용자 요청에 따라 Get, Post 방식으로 구별해 처리함

# java.servlet.http.HttpServlet 처리 구조

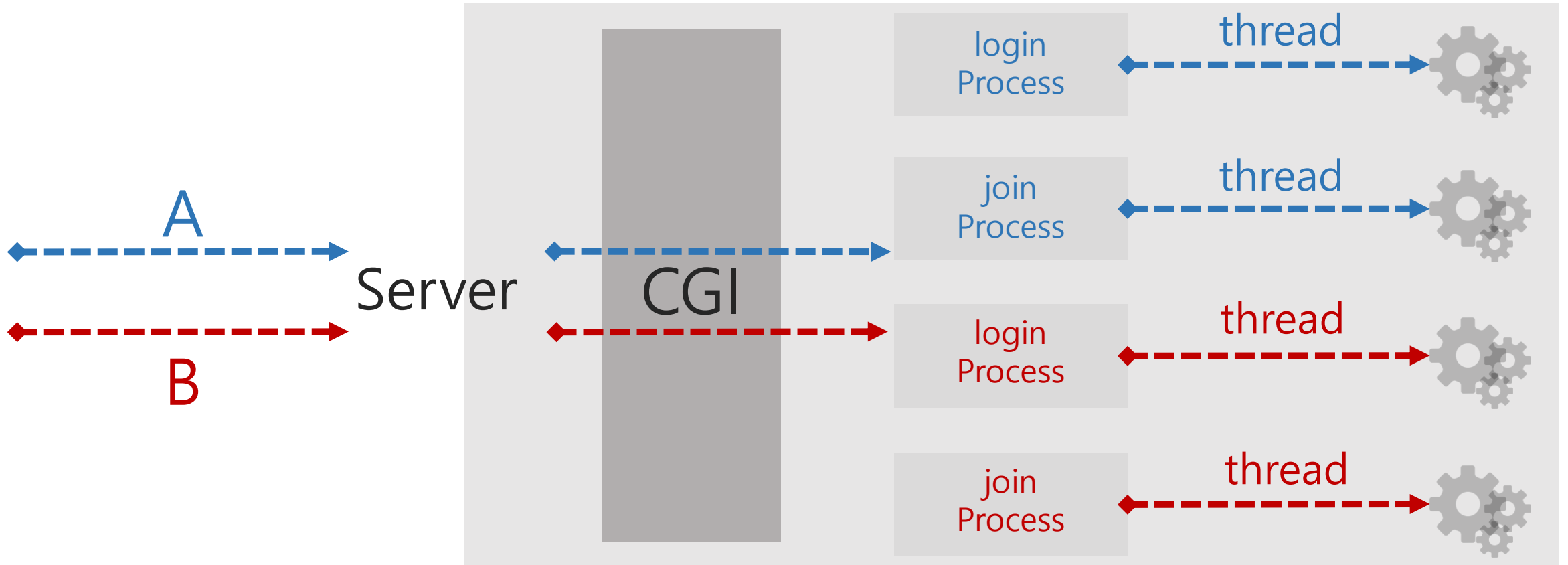


## CGI (Common Gateway Interface)

: 웹서버와 외부 프로그램 사이에서 정보를 주고받는 방법이나 규약들

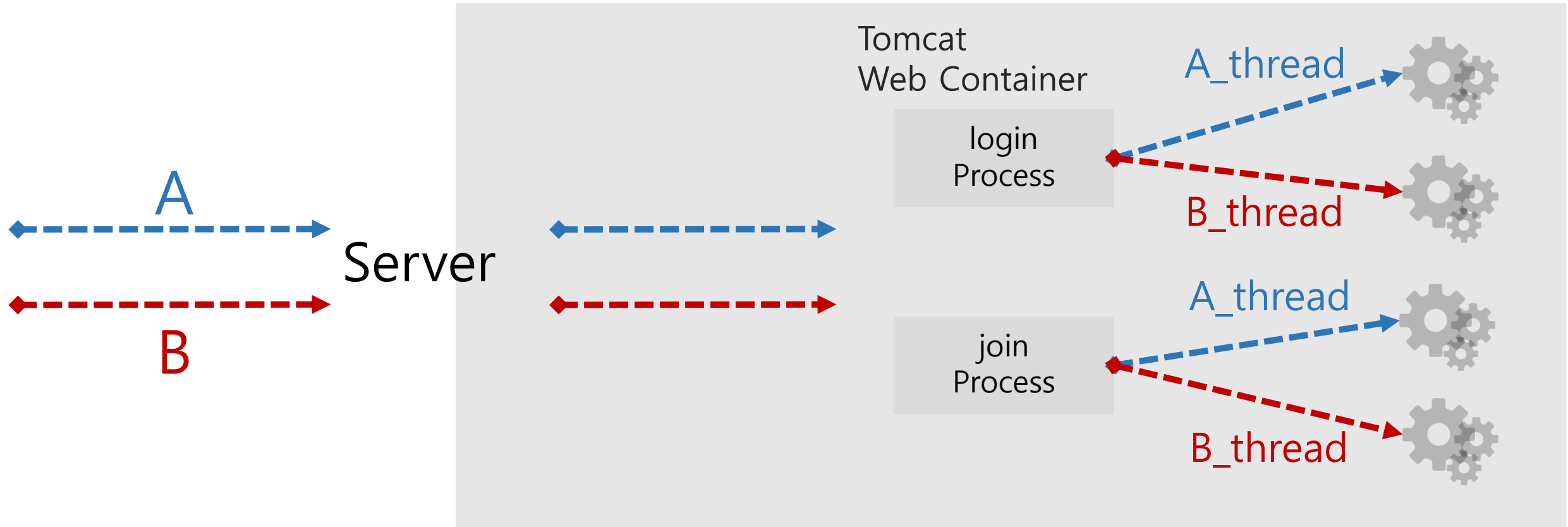


## 용어익히기 >> CGI



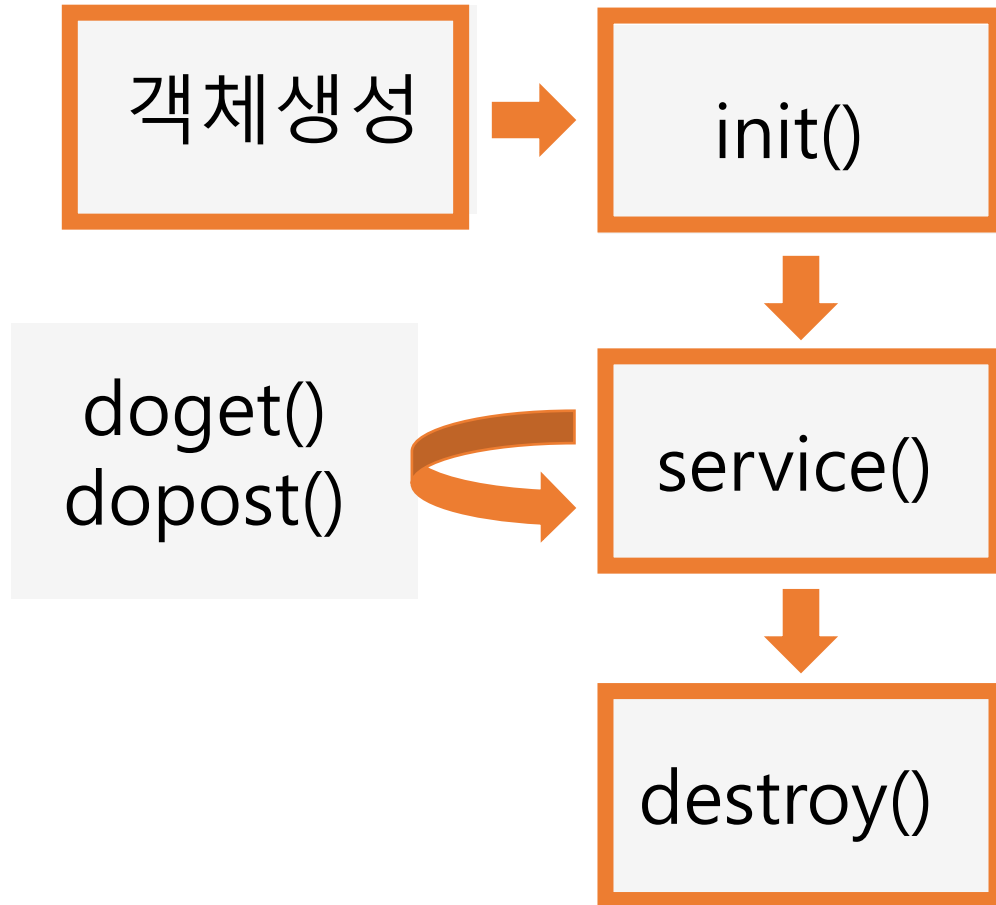
- CGI 프로그램은 **프로세스 단위**로 실행
- 사용자의 요청이 많을 때는 서버에 부하가 크게 감

# Java Multi Thread



- Servlet은 스레드 단위로 실행
- 서버의 부하를 줄일 수 있음

# Servlet 생명주기



- 서블릿 실행시 호출되는 메서드로 초기에 한 번만 실행.
- 공통적으로 필요한 작업 등 수행

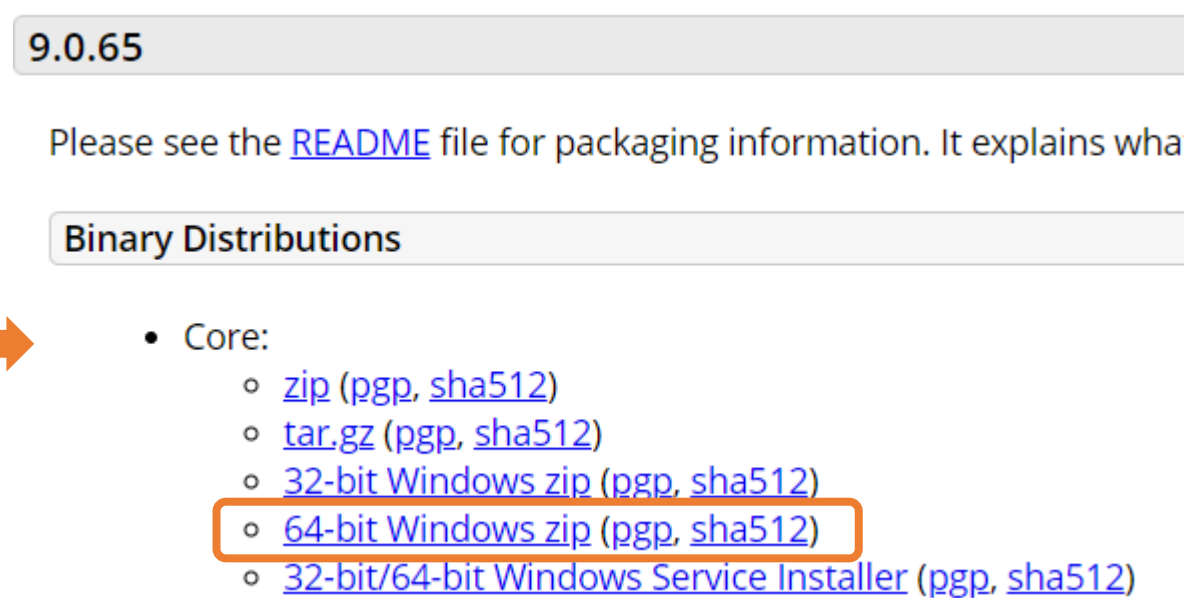
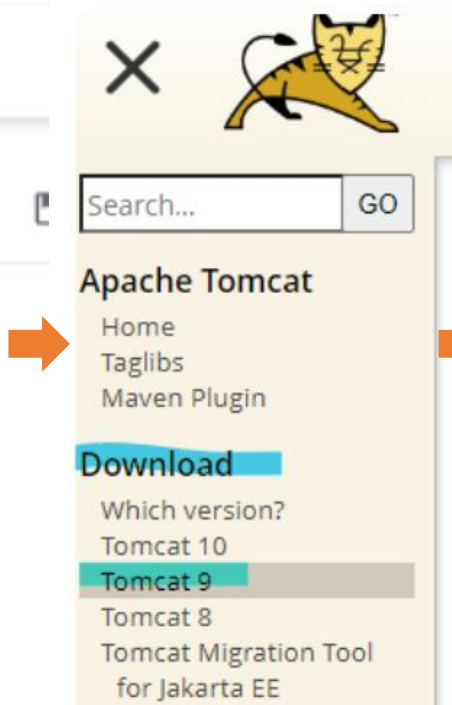
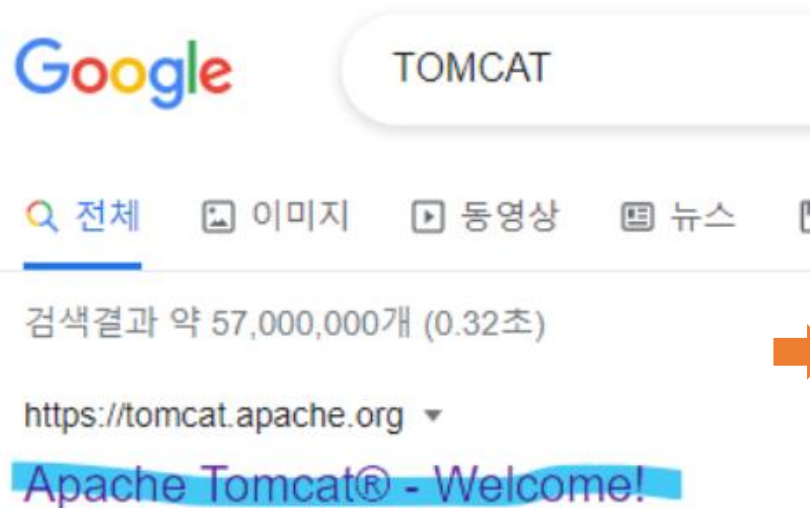
- 사용자 요청에 따라 스레드로 실행되는 메서드
- doGet() 혹은 doPost() 메서드 호출

- 컨테이너로부터 서블릿 종료 요청이 있을 때 호출되는 메서드
- init() 처럼 한 번만 실행.

# 아파치 톰캣 설치(<https://tomcat.apache.org/>)

## Apache Tomcat

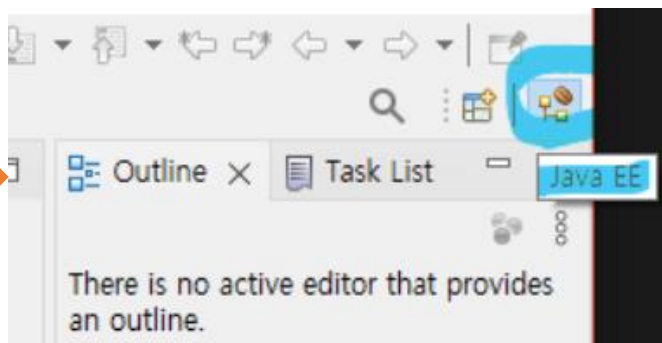
- 웹어플리케이션 서버의 한 종류
- JSP와 Servlet을 실행할 수 있는 환경 제공



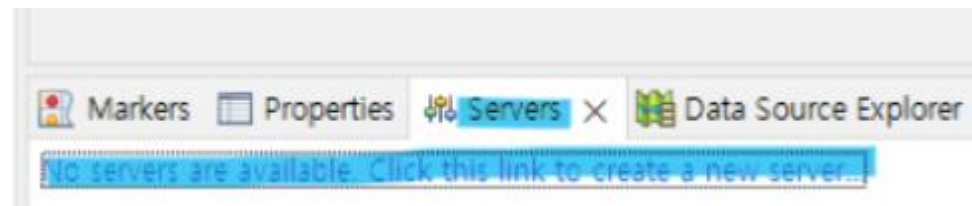
다운 받은 후 web 폴더(특정 폴더)에 압축 풀기



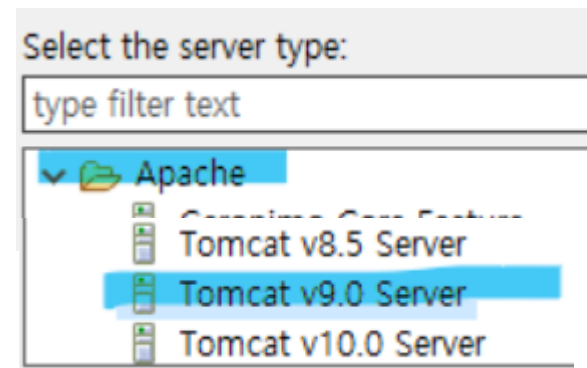
# 이클립스에서 아파치 톰캣 서버 연결하기



JavaEE 버전 이클립스 실행



서버 탭에서 new server 클릭



## Tomcat Server

Specify the installation directory

Name:

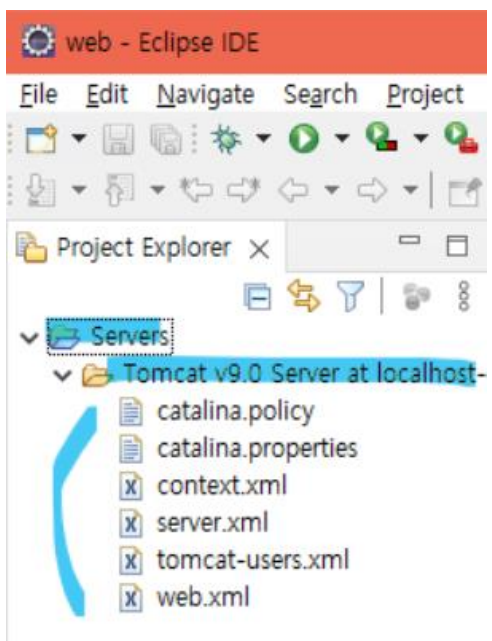
Apache Tomcat v9.0

Tomcat installation directory:

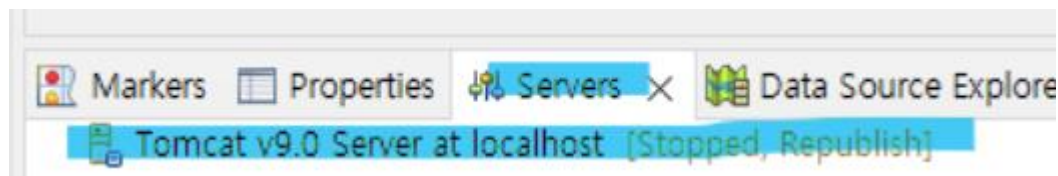
C:\Users\smhrd\Desktop\web\apache-tomcat-9.0.64

Browse...

클릭 후  
톰캣 설치 위치 지정

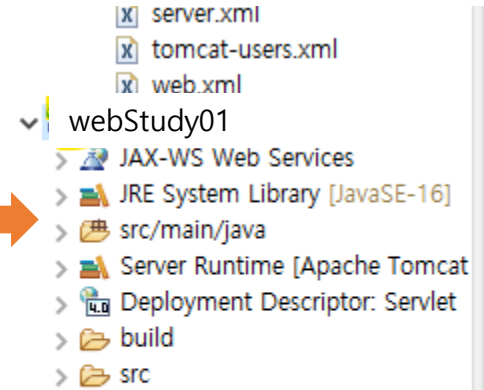
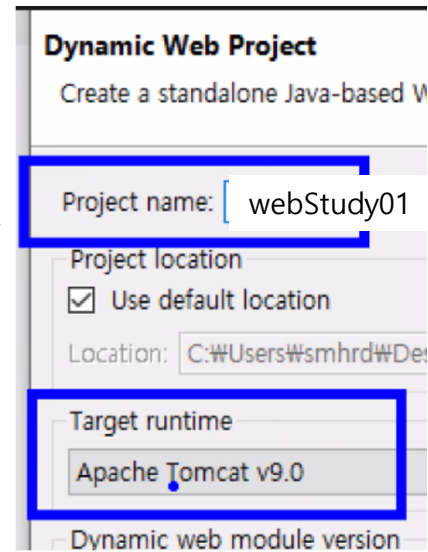
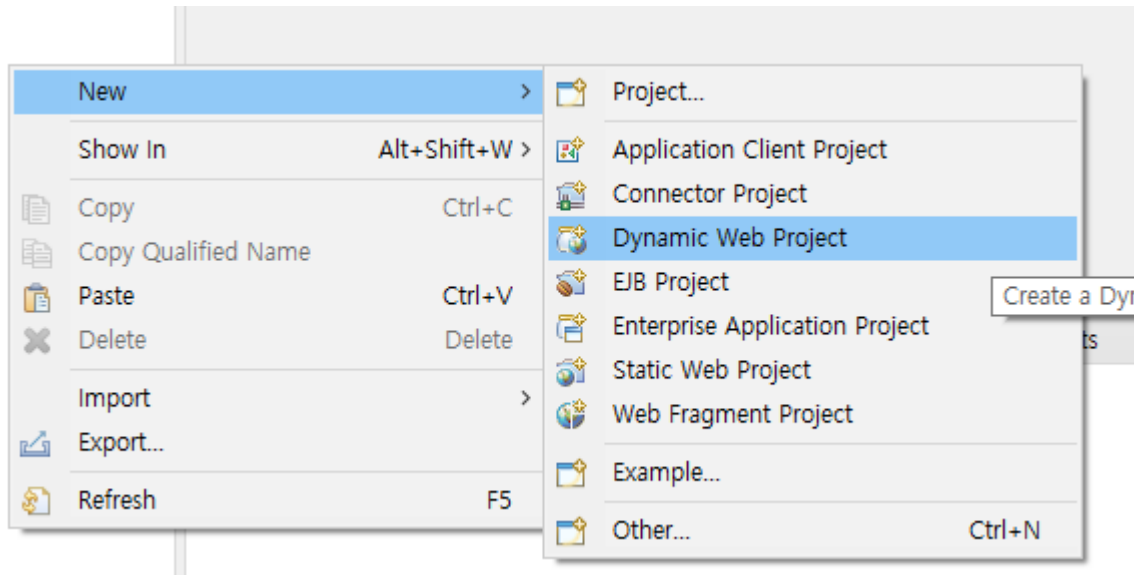


프로젝트 탐색기에서 Server 확인

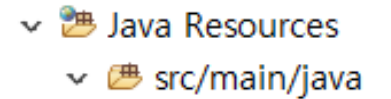
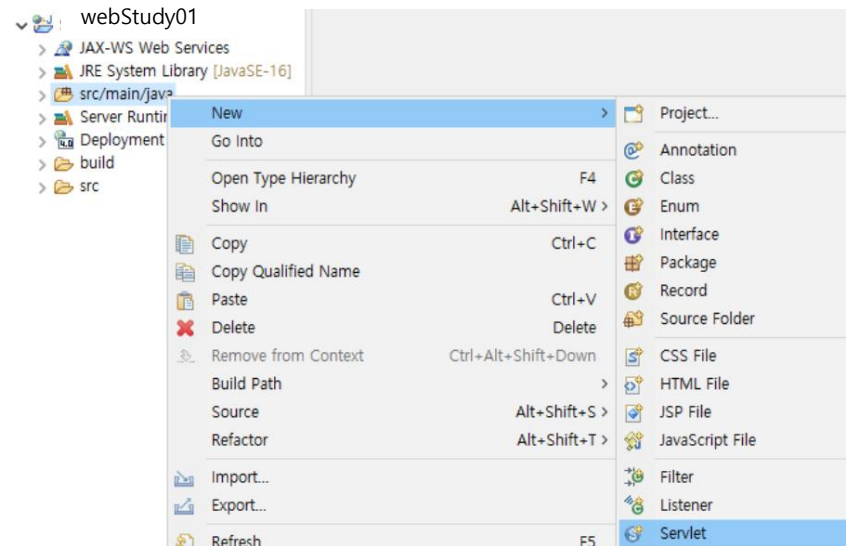
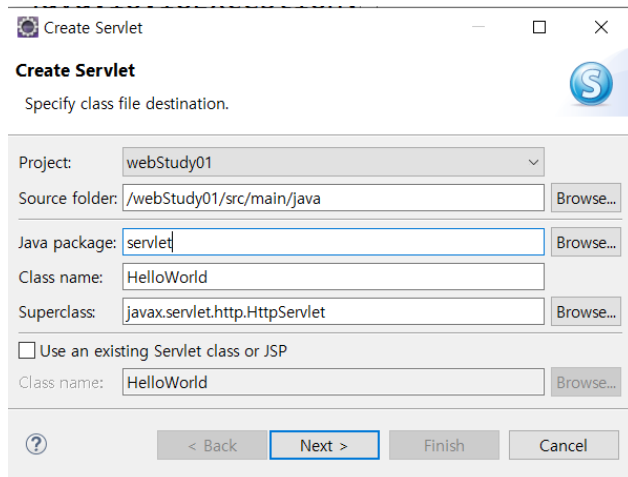


Finish 후 서버 탭에서 Tomcat v9.0 확인

# 이클립스에서 Web Project & servlet 만들기

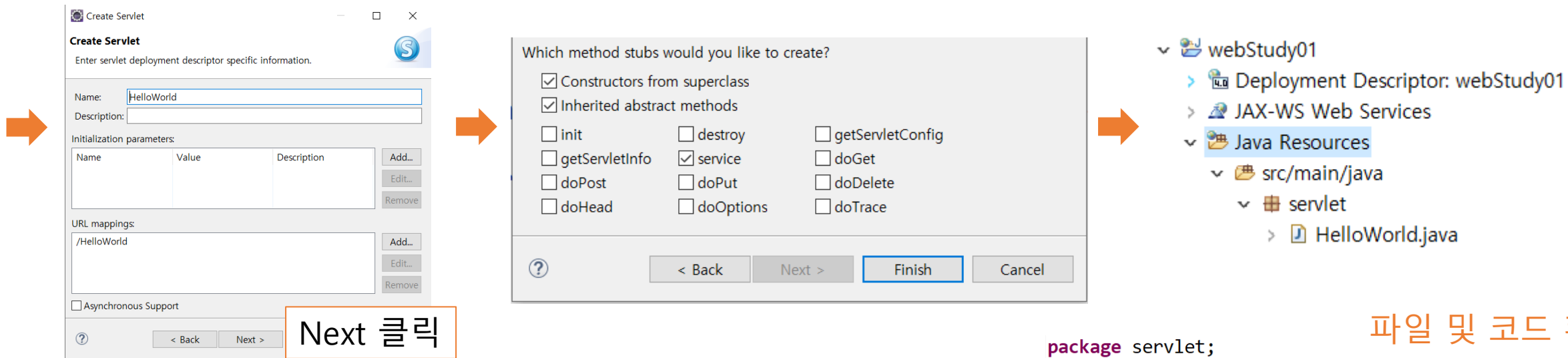


자동



프로젝트의  
src/main/java  
위치에서  
Servlet 생성

# 이클립스에서 Web Project & servlet 만들기



파일 및 코드 확인

## 실습1. 내 servlet에 요청이 들어온 ip주소를 확인하기

```
String client_id = request.getRemoteHost();  
System.out.println("들어온 사람 : " + client_id);
```

↓ 실행결과

```
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jdk-17.0.2\bin\jav  
INFO: 서버가 [1581] 포트에 시작되었습니다  
들어온 사람 : 0:0:0:0:0:0:0:1
```

주석제거후  
코드 입력

```
package servlet;  
  
import java.io.IOException;  
  
@WebServlet("/HelloWorld")  
public class HelloWorld extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    public HelloWorld() {  
        super();  
        // TODO Auto-generated constructor stub  
    }  
  
    protected void service(HttpServletRequest request,  
        HttpServletResponse response)  
        throws ServletException, IOException {  
        // TODO Auto-generated method stub  
    }  
}
```

# 이클립스에서 Web Project & servlet 만들기

## 실습2. response객체 : 사용자에게 응답할 때 사용하는 객체

```
response.setContentType("text/html; charset=utf-8");
```

```
// 웹에 출력하기 -> PrintWriter라는 객체 사용
```

```
PrintWriter out = response.getWriter();
```

```
out.print("안녕하세요~!");
```

```
out.print("<h1> Hello Web World! </h1>");
```

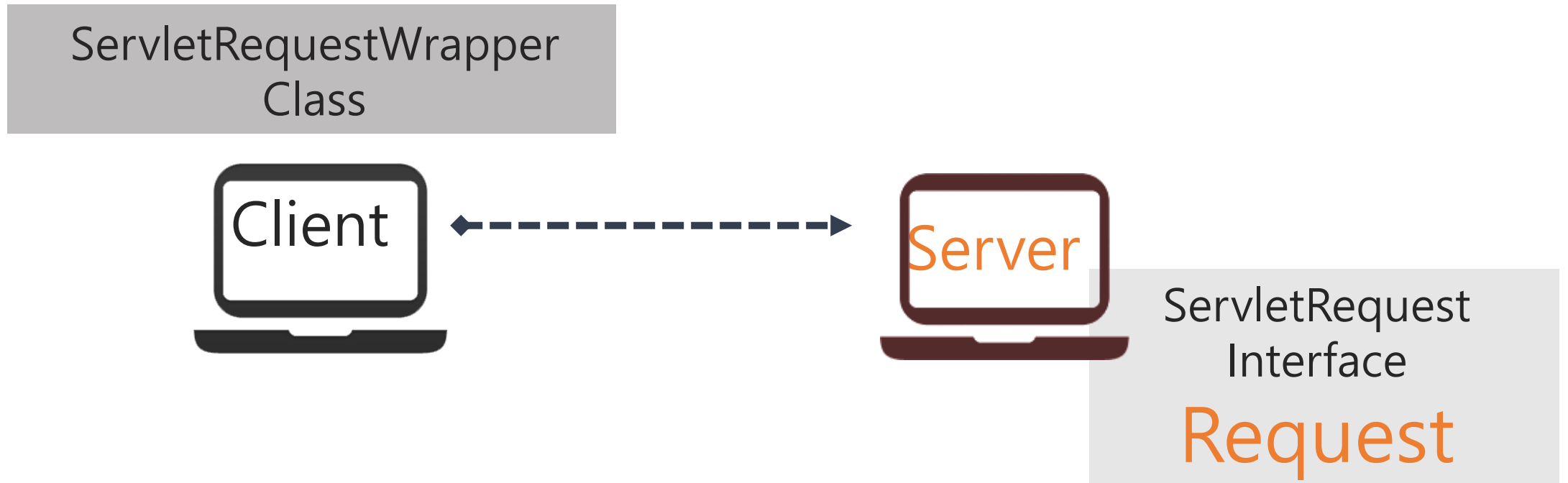
```
//이미지 출력하기
```

```
//servlet이 실행되는 위치 : webapp/images폴더
```

```
out.print("<img src='images/Elvis.png'>");
```

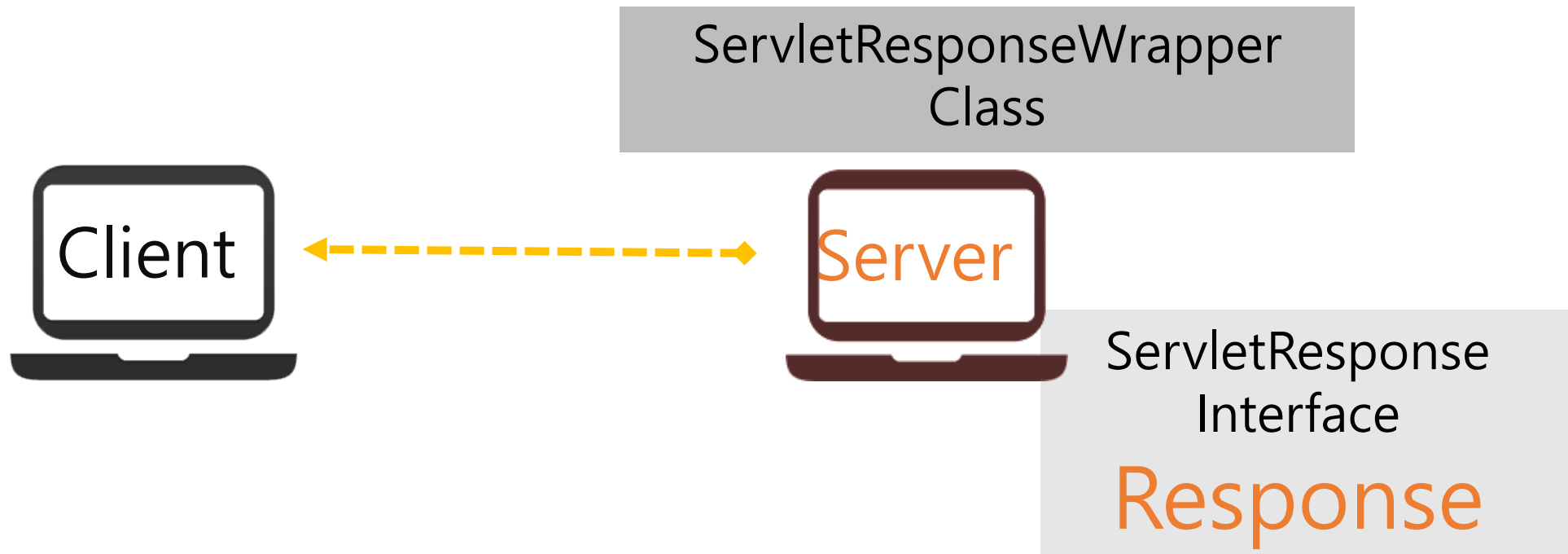
# Request

클라이언트의 요청 정보를 서블릿으로 넘겨주기 위한 객체  
(요청에 대한 정보를 가진 객체)



# Response

서블릿이 클라이언트에 응답을 보내는데 필요한 객체  
(응답에 대한 정보를 가진 객체)



# URL Mapping

- URL Mapping

- Web browser에서 Servlet을 동작시키기 위해 실제 Java 클래스의 이름 대신, Servlet을 요청하기 위한 문자열을 Servlet 클래스와 Mapping(맵핑)시키는 것
- 중복되면 안됨

<원래 주소>

http://localhost:8081/FirstProject/**Servlet/HelloWeb**

길다 → 불편함

경로가 드러남 → 보안상의 문제

<Mapping된 주소>

http://localhost:8081/FirstProject/**HWeb**



# URL Mapping

## URL Mapping 방법 : annotation 사용

`@WebServlet("/맵핑할 이름")`

- 유일해야 함
- Servlet 대체하는 문자열이므로 삭제하면 안됨

```
@WebServlet("/HelloWorld")
public class HelloWorld extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public HelloWorld() {
        super();
        // TODO Auto-generated constructor stub
    }
}
```



localhost:8090/webStudy01/HelloWorld



# URL Mapping

Name: HelloWorld

Description:

Initialization parameters:

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

URL mappings:

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

☐ Asynchronous Support

? < Back Next > Finish Cancel

**URL Mappings (Modal Dialog)**

Pattern: /Hello

OK Cancel

※ 서블릿 만들 때 수정 가능 Edit -> 변경

# Annotation

---

@WebServlet("/맵핑할 이름")



Annotation (주석)

- 컴파일이나 배포, 실행할 때 참조할 수 있는 주석
- 클래스나 필드, 메서드에 대해 부가정보를 등록할 수 있음
- 프로그램의 의미적인 부분에 직접 영향을 주지 않음

# URL Pattern

protocol    host(domain)    port    queryString

http://localhost:9000/Servlet/ex01Request?num=123

contextPath    servletPath

The diagram shows a URL with color-coded parts and labels above them. 'http' is purple and labeled 'protocol'. 'localhost' is blue and labeled 'host(domain)'. '9000' is black and labeled 'port'. '/Servlet/' is red and labeled 'contextPath'. 'ex01Request' is blue and labeled 'servletPath'. '?num=123' is green and labeled 'queryString'.

http://localhost:9000/Servlet/ex01Request?num=123

A horizontal line is drawn under the URL. A yellow box labeled 'requestURL' is positioned under the first part of the URL, and an orange box labeled 'requestURI' is positioned under the second part.

requestURL

requestURI

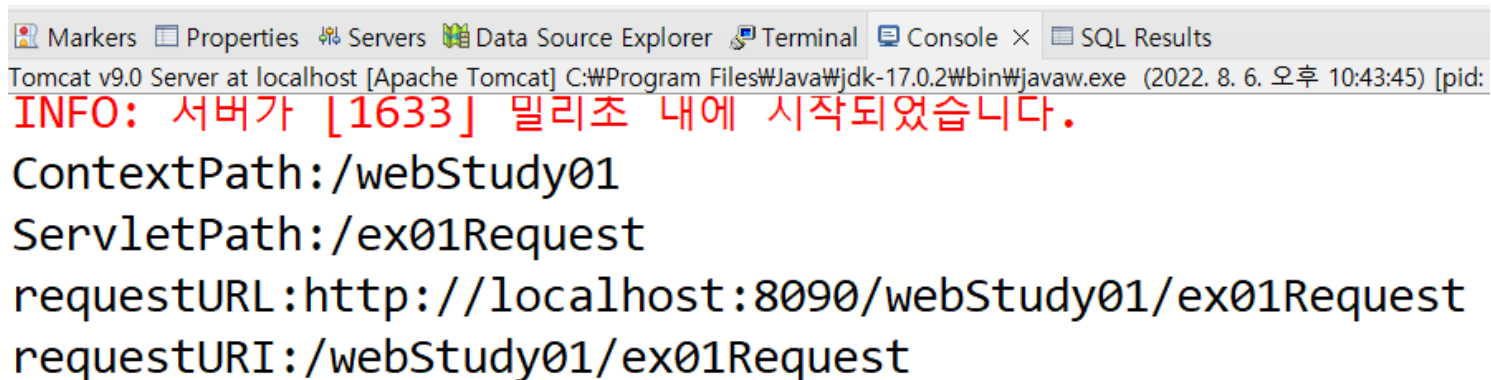
# URL Pattern

```
protected void service(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    System.out.println("ContextPath:"+request.getContextPath());

    System.out.println("ServletPath:"+request.getServletPath());

    System.out.println("requestURL:"+request.getRequestURL());

    System.out.println("requestURI:"+request.getRequestURI());
}
```



The screenshot shows an IDE interface with a console window. The console displays the following text:

```
Tomcat v9.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jdk-17.0.2\bin\javaw.exe (2022. 8. 6. 오후 10:43:45) [pid:
INFO: 서버가 [1633] 밀리초 내에 시작되었습니다.
ContextPath:/webStudy01
ServletPath:/ex01Request
requestURL:http://localhost:8090/webStudy01/ex01Request
requestURI:/webStudy01/ex01Request
```

실습1-1. 화면에 "Hello Servlet!" 문자열 출력하기

조건: 서블릿 이름은 "ex1" 로 매핑

# Hello Servlet!

```
PrintWriter out = response.getWriter();
```

텍스트 출력 스트림 생성

```
out.print("<html>");  
out.print("<body>");  
out.print("Hellow Servlet!");  
out.print("</body>");  
out.print("</html>");
```

Ex01HelloServlet.java

실습1-2. 화면에 "헬로우 서블릿" 문자열 출력하기

조건: 서블릿 이름을 "ex2"로 매핑

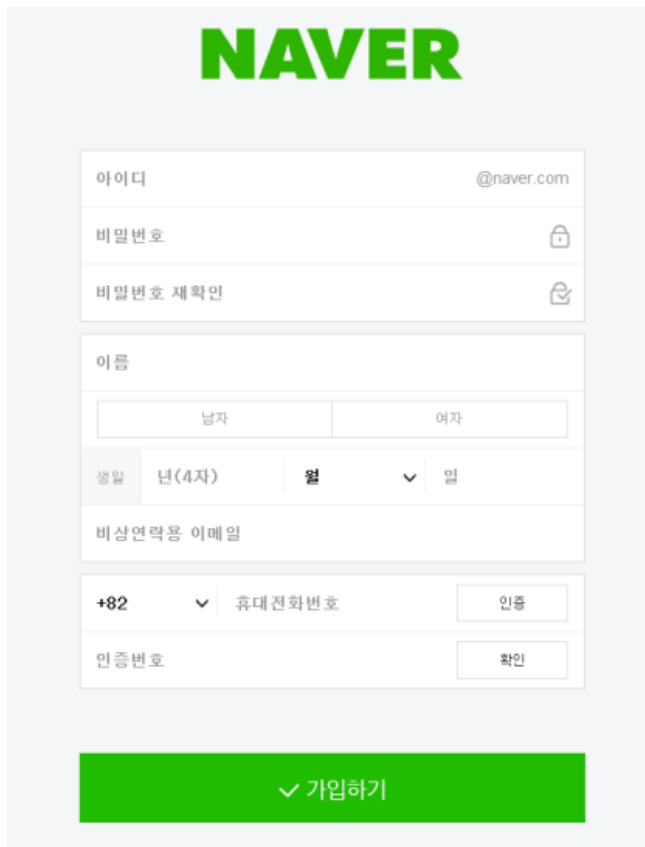
## 헬로우 서블릿

결과 출력 내용이 한글포함 될  
경우 인코딩 방식 반드시 지정

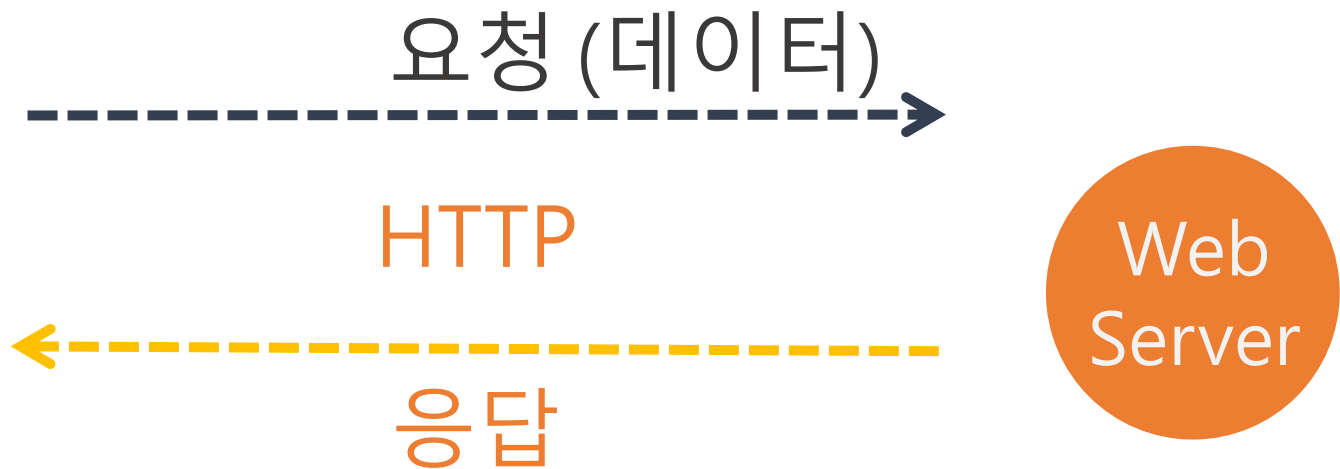
```
response.setContentType("text/html;charset=utf-8");  
PrintWriter out = response.getWriter();  
out.println("<html>");  
out.println("<body>");  
out.println("헬로우 서블릿");  
out.println("</body>");  
out.println("</html>");
```

Ex02HelloServlet.java

# 데이터 전송 : 3가지 조건



The image shows a Naver registration form. At the top is the Naver logo. Below it are input fields for '아이디' (ID) with a placeholder '@naver.com', '비밀번호' (Password) with a lock icon, and '비밀번호 재확인' (Confirm Password) with a checkmark icon. There is a section for '이름' (Name) with '남자' (Male) and '여자' (Female) options. Below that is a date selection section with '생일' (Birthday), '년(4자)' (Year), '월' (Month), and '일' (Day). There is also a field for '비상연락용 이메일' (Emergency contact email). At the bottom, there are fields for '+82' (Country code), '휴대전화번호' (Mobile phone number), and '인증번호' (Verification number), each with a corresponding '인증' (Verify) button. A large green button at the bottom says '✓ 가입하기' (Sign up).



데이터를 전송하기 위해 사용하는 태그?     `<form>` 태그

## 데이터 전송 : 3가지 조건

값을 어디로 보낼지

```
<form action="url">
```

어떤 값을 보내는지

```
  ID : <input type="text" name="param">
  PW : <input type="text">
  <input type = submit value="login">
</form>
```

값을 보내는 시점

**\*action, name, submit\***



실습2-1. 1개 값을 입력받는 form 태그 작성

DATA :


```
<form action="ex01datasend">  
  DATA : <input type="text" name = "data">  
  <input type="submit" value="SEND">  
</form>
```

Ex01DataSend.html

## 실습2-2. 입력한 값 서버로 전송하여 console창에 출력

Ex01DataSend.html

```
<form action="ex01datasend">  
    DATA : <input type="text" name = "data">  
    <input type="submit" value="SEND">  
</form>
```



ex01DataSend.java

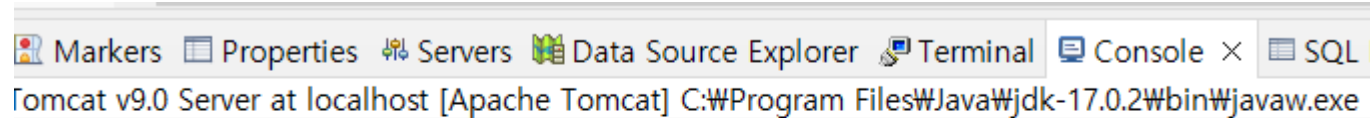
```
String data = request.getParameter("data");
```

HTTP 요청의 파라미터 값을 얻는 메소드

# Servlet 실습

```
@WebServlet("/ex01DataSend")
public class ex01DataSend extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void service(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        String data = request.getParameter("data");
        System.out.println("사용자가 입력한 값 : "+data);
    }
}
```



The screenshot shows the bottom portion of an IDE window. The top bar contains tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', 'Terminal', 'Console', and 'SQL'. The 'Console' tab is active, displaying the output of the application. The text in the console is: 'INFO: 서버가 [1625] 밀리초 내에 시작되었습니다.' followed by a vertical line, and then '사용자가 입력한 값 : 데이터 간다'.

INFO: 서버가 [1625] 밀리초 내에 시작되었습니다.  
사용자가 입력한 값 : 데이터 간다

실습2-3. 입력한 값 서버로 전송하여 html문서로 출력

입력한 데이터: 123

```
response.setContentType("text/html; charset=UTF-8");

String data = request.getParameter("data");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body>");
out.println("입력한 데이터: " + data);
out.println("</body>");
out.println("</html>");
```

Ex02DataSend.java

실습3. 2개의 정수를 입력 받아 덧셈식 출력

+

Ex03Plus.html

12 + 12 = 24

Ex03Plus.java

```
int num1 = Integer.parseInt(request.getParameter("num1"));
```

정수형 Type으로 변환

실습4. 2개의 정수와 연산기호를 입력 받아 연산식 출력

<input type="text"/>	<input type="text" value="+"/>	<input type="text"/>	<input type="button" value="계산"/>
----------------------	--------------------------------	----------------------	-----------------------------------

Ex04Operation.html

$$5 + 5 = 10$$

$$5 - 5 = 0$$

$$5 * 5 = 25$$

$$5 / 5 = 1$$

Ex04Operation.java

실습5. 아래와 같은 방이 6개인 테이블을 만드시오.

1	2	3	4	5	6
---	---	---	---	---	---

Ex05Table.java

실습6. 정수 1개를 입력받아 입력한 숫자에 따라 방을 생성하시오.

몇 개의 방을 만들까요?

Ex06MakeTable.html

1	2	3	4	5
---	---	---	---	---

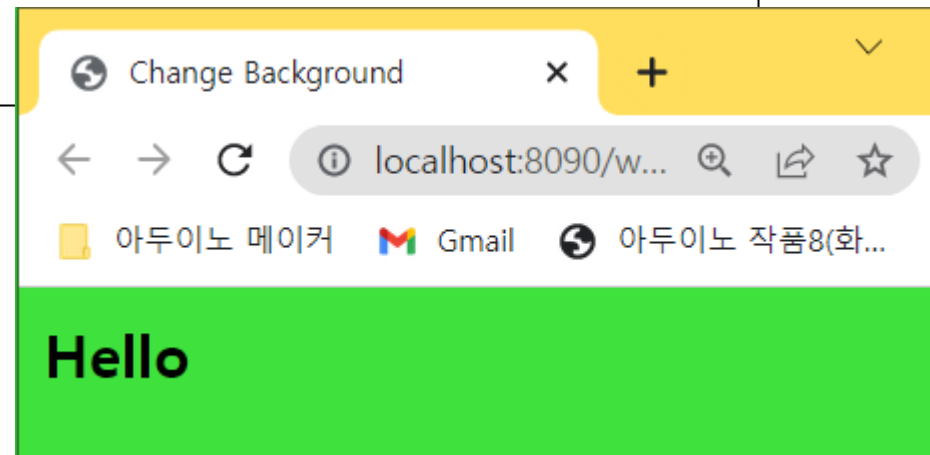
Ex06MakeTable.java



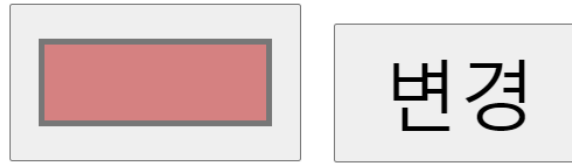
## 실습7. 배경색을 변경하는 서블릿 작성

```
PrintWriter out = response.getWriter();  
out.print("<html><head><title>Change Background</title>");  
out.print("<style> body { background: #3fe23c; }</style>");  
out.print("<body>");  
out.print("<h3>Hello</h3>");  
out.print("</body></html>");
```

ChangeBg.java



실습7. 지정한 색깔로 배경색을 변경하시오.



Ex07ChangeBg.html



Ex07ChangeBg.java

# Servlet 실습

실습8. 정수 2개와 색깔을 입력받아 테이블에 구구단을 출력하고  
지정한 색깔로 테이블 배경색을 변경하시오.

색상 선택 :

에서  까지의 구구단을 출력

Ex08MakeGugu.html

3*1=3	3*2=6	3*3=9	3*4=12	3*5=15	3*6=18	3*7=21	3*8=24	3*9=27
4*1=4	4*2=8	4*3=12	4*4=16	4*5=20	4*6=24	4*7=28	4*8=32	4*9=36
5*1=5	5*2=10	5*3=15	5*4=20	5*5=25	5*6=30	5*7=35	5*8=40	5*9=45
6*1=6	6*2=12	6*3=18	6*4=24	6*5=30	6*6=36	6*7=42	6*8=48	6*9=54

Ex08MakeGugu.java

```
out.print("<table border>");
//바깥쪽 for 시작
out.print("<tr>");
for(int i = 1; i <= 9; i++) {
    //방 만들기 <td>~</td>
    out.print("<td>" + 2 + "*" + i + "=" + 2*i
    + "</td>");
}

out.print("</tr>");
//바깥쪽 for 끝
out.print("</table>");
```

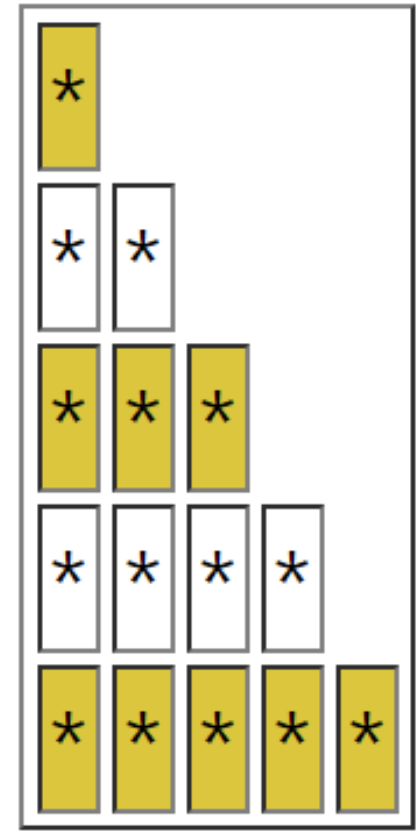
# Servlet 과제

과제. 정수 1개와 색깔을 입력 받아 다음과 같이 출력하시오.

색상 선택 :

줄 수 입력 :

A01Star.html



A01Star.java