JSP(Java Server Pages)

서버환경구축

HelloServlet.java

```
package view; // 패키지(폴더)이름 — src/main/java 아래 view 폴더에 존재

   /*
   # 서블릿 이해하기
   - 정적 웹의 문제점을 보완 => 동적 웹을 구현하기위해 나온 서블릿,JSP가 있다.
     초기 동적 웹 페이지 =  서블릿(Servlet, 자바로 만든 CGI 프로그램)을 이용해 구현
     서블릿의 어려움 등의 문제점을 보완한것이 => JSP 이다.
     JSP는 서블릿의 기능을 따라가며, 서블릿을 이해하면 JSP를 쉽게 이해하고 익힐 수 있다.
     실제 웹 어플 개발시 JSP와 서블릿의 각각 고유 역할을 나누어 기능을 구현한다.
    ================================================================================
   # 아파치 톰캣
   - 자바 기반 오픈소스 WAS(웹 어플리케이션 서버)인 서블릿/JSP 컨테이너로
     서블릿은 톰캣안에 포함되어 있어 톰캣 안에서만 작동한다. => 서블릿의 컨테이너. 톰캣

   # 서블릿 동작 과정
   - 클라이언트(브라우저) 요청 → 웹서버(Apache)→ WAS(Tomcat) → 서블릿 (요청/Request)
   - 역순) 서블릿 요청 → WAS(Tomcat) → 웹서버(Apache) → 브라우저 (응답/Response)

   # 서블릿 특징
     a. 서버측 실행: 브라우저가 아니라 서버 안에서 자바 코드가 동작.
     b. 스레드 방식: 한 서블릿 객체가 여러 요청을 동시에 스레드로 처리.
     c. 자바 특성 활용: 자바 문법·상속·예외처리 가능, 단 톰캣 안에서만 실행.
     d. 보안 용이: 세션·쿠키·필터 등 톰캣이 제공하는 보안 기능 쉽게 적용.
     e. 요청 기반 실행: URL 요청이 들어올 때만 서블릿이 실행됨.
```

서블릿 API 계층 구조

1.Servlet 인터페이스 //서블릿의 최상위 기본 틀, 모든 서블릿이 구현해야 하는 규칙 제공.
|
2.GenericServlet 추상 클래스 // Servlet 인터페이스를 부분 구현, 공통 기능 제공
|
3.HttpServlet 클래스 -> 상속 // GenericServlet을 상속받아 HTTP 전용 서블릿 기능
(doGet, doPost 등) 추가.

HttpServlet 클래스란?
- HTTP 방식(웹 요청·응답)을 처리할 수 있게 해주는 서블릿의 부모 클래스이다.
클라이언트 요청시 service()가 먼저 호출되고,
=> 데이터 요청 방식에 따라 doGet(), doPost()호출됨(핵심)

javax.servlet.http 패키지란?
- 서블릿(웹 프로그램)에서 HTTP 요청과 응답을 처리하는 클래스들이 모여 있는 패키지이다.
HttpServlet, HttpServletRequest, HttpServletResponse, Cookie, Session 등
웹 통신에 필요한 핵심 클래스들이 포함되어 있다.

서블릿 생명주기 메서드
- 초기화 과정 -> 메모리에 인스턴스 생성 -> 서비스 수행 -> 소멸 과정을 거침 (자바와 동일)

서블릿은 각 요청 실행 단계마다 호출되어 기능을 수행하는 콜백 메서드들이 있다.

1. init() – 서블릿이 처음 메모리에 올라올 때 1회 실행 (초기화)
2. service() – 요청이 들어올 때마다 호출되어 요청 방식을 구분(GET/POST 판단)
3. doGet() / doPost() – service() 내부에서 호출되어 실제 요청 처리 수행
4. destroy() – 서버 종료 또는 서블릿이 메모리에서 해제될 때 1회 실행 (자원 정리)

요약: init(준비) → service(요청받기) → doGet/doPost(처리하기) → destroy

(정리하기)

서블릿 동작 과정 = 코딩 순서
a. 사용자 정의 서블릿 클래스 만들기
b. 서블릿 생명주기 메서드 = init(), doGet(),doPost(), destroy()
c. 서블릿 매핑 작업 = web.xml 또는 어노테이션(=주로 사용)
d. 클라이언트(브라우저)에서 서블릿 매핑 이름 요청

요약 : "서블릿 만들고 → 생명주기 메서드 구현 → 주소 매핑 → 브라우저 요청으로 실행."

서블릿 Thread 작동 과정
- 여러 클라이언트가 같은 서블릿을 요청했을때 서블릿 처리 과정

1. 클라이언트가 DoServlet이라는 서블릿을 요청했다고 가정
=> 서버는(아파치) 톰캣(동적담당)의 서블릿(컨트롤러)이 메모리에 로드되어 있는지 확인

2-1. DoServlet 서블릿이 만약 없다면?
=> DoServlet 서블릿 메모리에 로드
=> init()호출 => doGet(),doPost()호출
=> 클라이언트에 결과 응답

2-2. DoServlet 서블릿이 있으면 ?
=>doGet(),doPost()호출
=> 클라이언트에 결과 응답

(결론) 위의 방식으로 서블릿을 재사용하여 훨씬 빠르고 효율적으로 동작 됨
아파치(서버), 톰캣(자바동적담당), 서블릿(컨트롤러)
*/

```
import java.io.IOException; //입출력 예외 처리용 클래스
import javax.servlet.ServletConfig;  //서블릿의 설정정보(초기화용)
import javax.servlet.ServletException; //서블릿 실행 중 생기는 일반적인 예외
import javax.servlet.http.HttpServlet; //서블릿의 부모 클래스 (HTTP 전용 서블릿)
import javax.servlet.http.HttpServletRequest; //요청(request) 객체
import javax.servlet.http.HttpServletResponse; //응답(response) 객체
```

```
// 서블릿에서는 표기로 어노테이션 @매핑하는데 ( web → server 연동)
//  web.xml에서 매핑할 것이므로 @WebServlet 은 주석 처리
// @WebServlet("/HelloServlet")
public class HelloServlet extends HttpServlet {

    private static final long serialVersionUID = 1L; // 서블릿 고유 식별 (직렬 - 자동)

    @Override // 생명주기 메서드 = 초기화
  //서블릿이 생성(init) → 요청 처리(doGet/doPost) → 소멸(destroy)되는 모든 과정 : 생명주기

    public void init(ServletConfig config) throws ServletException {
        super.init(config); // 부모 초기화 메서드 호출 (필수)
        System.out.println("init 메서드 호출");
    }

    public HelloServlet() {
        super(); // 부모(HttpServlet) 생성자 호출
        // 서블릿이 처음 실행될 때 한 번만 메모리에 로드됨
    }

    @Override
    // ✅ 브라우저에서 GET 방식으로 요청했을 때 요청 처리 함수 (URL 입력, 링크 클릭 등)
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        // 출력 인코딩 설정 (한글 깨짐 방지)
        response.setContentType("text/html; charset=UTF-8");

        // 브라우저에 문장 출력
        response.getWriter().append("Served at: ").append(request.getContextPath());
        // 결과 예시: Served at: /JChap01ServletBasic
        System.out.println("doGet 메서드 호출");
```

```
    }

    // ✅ POST 방식 요청 처리 (보통 form 전송 시)
    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        // POST 요청도 doGet()과 동일하게 처리
        doGet(request, response);
        System.out.println("doPost 메서드 호출");
    }
}

    // Request: 요청 파라미터 / Response: 응답 파라미터 / Exception: 예외 처리
```

import java.io.IOException; - 입출력 예외 처리용 클래스
import javax.servlet.ServletConfig; - 서블릿의 설정정보(초기화용)
import javax.servlet.ServletException; - 서블릿 실행 중 생기는 일반적인 예외
import javax.servlet.http.HttpServlet; - 서블릿의 부모 클래스 (HTTP 전용 서블릿)
import javax.servlet.http.HttpServletRequest; - 요청(request) 객체
import javax.servlet.http.HttpServletResponse; - 응답(response) 객체

HelloServlet2.java

```
package view;

import java.io.IOException; //입출력 예외 처리용 클래스
import javax.servlet.ServletConfig;  //서블릿의 설정정보(초기화용)
import javax.servlet.ServletException; //서블릿 실행 중 생기는 일반적인 예외
import javax.servlet.http.HttpServlet; //서블릿의 부모 클래스 (HTTP 전용 서블
릿)
import javax.servlet.http.HttpServletRequest; //요청(request) 객체
import javax.servlet.http.HttpServletResponse; //응답(response) 객체

public class HelloServlet2 extends HttpServlet {

    //생명주기 함수 초기화
```

```
    public void init(ServletConfig config) throws ServletException {
     System.out.println("init 초기화 2");
  }

  // doGet
  protected void doGet(HttpServletRequest request, HttpServletResponse r
esponse) throws ServletException, IOException {
     System.out.println("Get 방식으로 데이터 넘어올때의 처리");
  }
  // doPost
  protected void doPost(HttpServletRequest request, HttpServletResponse r
esponse) throws ServletException, IOException {
     System.out.println("Post 방식으로 데이터 넘어올때의 처리");
  }


  public void destroy() {
     System.out.println("destroy 소멸");
  }
}
```

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xml
ns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.
jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xs
d" id="WebApp_ID" version="4.0">
  <display-name>JChap01ServletBasic</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
```

```
  </welcome-file-list>

  <servlet>
    <servlet-name>hello</servlet-name>
    <servlet-class>view.HelloServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>hello</servlet-name>
    <url-pattern>/helloservlet</url-pattern>
  </servlet-mapping>

  <servlet>
    <servlet-name>hello2</servlet-name>
    <servlet-class>view.HelloServlet2</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>hello2</servlet-name>
    <url-pattern>/helloservlet2</url-pattern>
  </servlet-mapping>
</web-app>
```

💡결론

👉 web.xml은 “주소와 자바 클래스(서블릿)를 연결해주는 지도(map)”이다.

> 즉, 클라이언트(브라우저)가 “/helloservlet”으로 들어왔을 때
>
> 톰캣이 “그럼 이건 view.HelloServlet 클래스가 처리해야 하는구나!”
>
> 하고 연결시켜주는 역할이며, 이걸 “**서블릿 매핑(mapping)**”이라고 부르며
>
> **서블릿 매핑 호출 @WebServlet("/HelloServlet")**

✅GET 방식 예제

```
<form action="/search" method="get">
  <input type="text" name="keyword" value="java">
  <button>검색</button>
</form>
```

```
http://localhost:8080/search?keyword=java
```

→ **서버로 가는 요청 URL 주소창에 검색어가 그대로 보임**

✅ POST 방식 예제

```
<form action="/login" method="post">
  <input type="text" name="id">
  <input type="password" name="pw">
  <button>로그인</button>
</form>
```

→ **주소창엔** `/login` **만 보이고 , ID, PW 데이터는 본문(body) 에 숨겨서 전송**

✅**GET** : 주소(URL)에 데이터를 붙여 보내는 방식 → 주로 **조회용**

✅ **POST** : 데이터가 숨겨진 상태로 전송되는 방식 → 주로 **등록·로그인 등 전송용**

🔷 정리해보면

1️⃣ 사용자가 `index.html` 페이지를 열고

↓

2️⃣ 버튼 클릭으로 `/HelloServletWorld` 요청하게되면

↓

3️⃣ Apache → Tomcat에게 전달된다

↓

4️⃣ Tomcat이 @어노테이션을 보고 해당 `HelloServletWorld.java`(서블릿) 실행하고

↓

5️⃣ 자바가 HTML 코드 만들어서 사용자에게 돌려줌

↓

6️⃣ 화면에 웹페이지가 표시됨

↓

7️⃣ JSP는 HTML + Java 혼합형으로 더 편하게 작성 가능

💡 웹 서버 통신 흐름

👉 `index.html`에서 버튼 누름 →

👉 `HelloServlet.java`가 "안녕!"을 준비 →

👉 `result.jsp`가 그 말을 화면에 보여줌

1️⃣ index.html (시작 화면)

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>시작 화면</title>
</head>
<body>
  <h1>index.html</h1>
  <a href="/hello">서블릿으로 이동</a>
</body>
</html>
```

📌 **설명: a태그** 누르면 `/hello` 주소로 이동 하며 HelloServlet.java로 전달

2️⃣ HelloServlet.java (자바 코드)

```
import java.io.IOException; // 입출력 중 문제 있을 때 예외 던짐
import jakarta.servlet.*;   // 서블릿의 뼈대(기초 도구) 모음
import jakarta.servlet.annotation.WebServlet; // 서블릿 주소 어노테이션을 쓰기 위해 필요
import jakarta.servlet.http.*; // "HTTP 요청·응답 도구 세트"

@WebServlet("/hello") // index.html에서 http 요청 받음
public class HelloServlet extends HttpServlet {

  @Override //doGet 생명주기 함수
  protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException { // 예외처리

      // 데이터 만들기
```

```
        String message = "안녕, 나는 서블릿이야!";

        // JSP에 전달하기
        req.setAttribute("msg", message);

        // JSP로 이동
        RequestDispatcher rd = req.getRequestDispatcher("/result.jsp");
        rd.forward(req, resp);
    }
}
```

📌 **설명:**

1. 누군가 `/hello`로 오면 Tomcat이 이 클래스를 실행.
2. "안녕, 나는 서블릿이야!" 문장을 만들어서
3. `msg`라는 이름으로 JSP에게 전달(`setAttribute`)
4. JSP로 이동(`forward`)

⚙️ 톰캣이 실제로 호출하는 순서

서블릿은 톰캣이 알아서 **생성(init) → 요청 처리(doGet/doPost) → 소멸(destroy)** 단계로 움직임

```
서블릿 최초 요청 시
  ↓
init()  — (톰캣이 자동 호출, 처음 한 번만)
  ↓
service() — (모든 요청 시 자동 호출)
  ↓
doGet() or doPost() — (service() 안에서 호출, 실제 요청 처리)
  ↓
destroy() — (서버 종료 시 자동 호출)
```

3️⃣ result.jsp (결과 화면)

```
<%@ page contentType="text/html; charset=UTF-8" %>
<html>
```

```
<body>
  <h1>result.jsp</h1>
  <p>${msg}</p>
</body>
</html>
```

📌 **설명:** 서블릿이 보낸 `msg` 변수를 `${msg}`로 꺼내서 보여준다.

`@WebServlet("/hello")` 어노테이션

`@WebServlet("/hello")`는 **"이 클래스는 /hello 요청을 처리한다"** 라고

Tomcat에게 미리 등록해두는 **표시(Annotation)**하고,

- **"이 주소로 요청이 오면 이 클래스를 실행해"** 라는 연결 고리이다.
- 그래서 web.xml에 등록 안 해도 자동 인식되는것.

`doGet()` 메서드 자동 실행

- 링크 클릭은 "GET 방식" 요청이라
- **Tomcat이 HelloServlet의 doGet()을 자동으로 실행**
- 만약 `<form method="post">`였다면 doPost()가 실행된다.

"요청(req)"과 "응답(resp)"의 역할

이 부분이 가장 핵심⚡

이름	역할
`req` (Request)	사용자가 서버로 보낸 모든 정보 담는 "요청 상자"
`resp` (Response)	서버가 사용자의 브라우저로 돌려주는 "응답 상자"

즉,

- JSP로 데이터를 넘길 때 **req 안에 데이터 담기** (`setAttribute`)
- JSP가 응답을 완성하면 **resp로 브라우저에게 HTML 돌려줌**

그래서

👉 req, resp 두 개는 **항상 함께** 있어야 한다. (필수 매개변수!)

JSP로 전달 (forward)

`forward(req, resp)`는

"지금 받은 요청과 응답 상자 그대로 JSP에게 넘겨줘"라는 뜻으로

> ⚠️ 여기서 중요한 점은!
>
> JSP가 **index.html로 데이터를 다시 보내는 건 아니다.**
>
> JSP는 **최종 결과를 브라우저로 직접 보여주는 역할이다.**

즉, 👇 흐름은

```
[사용자 브라우저] → index.html (링크 클릭)
          ↓
        /hello 요청
          ↓
       HelloServlet 실행 (해당 어노테이션실행)
          ↓
  JSP에게 req, resp 전달 (forward)
          ↓
       JSP가 HTML로 응답 작성
          ↓
[브라우저 화면에 JSP 결과 표시]
```

그래서 JSP는 **index.html로 되돌려주는 게 아니라, 브라우저에 바로 결과를 그려주는 화면**.
index.html은 "출발점(입구)"일 뿐.

변수와 전달 순서

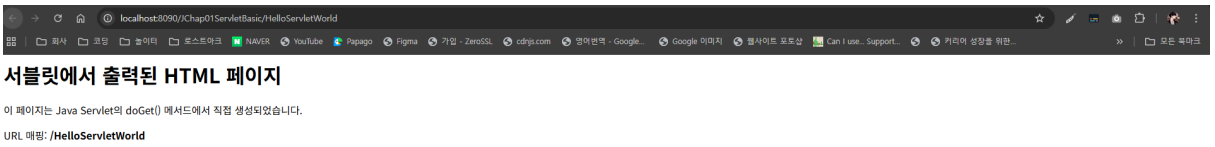
1. 서블릿 안에서 `String message = "안녕";` → 변수에 데이터 저장
2. `req.setAttribute("msg", message);` → req에 데이터 담기
3. `forward(req, resp);` → JSP에 req 전달
4. JSP에서 `${msg}` 로 꺼내서 출력
5. JSP가 결과 HTML을 만들어 브라우저에 보냄

🧩 한 줄로 요약

> <a href="/hello"> 클릭 → Tomcat이 HelloServlet 실행

> → `doGet()` 자동 실행
>
> → 변수(message) 만들어 `req`에 담음
>
> → JSP로 forward
>
> → JSP가 req 안의 데이터를 꺼내 HTML로 만들어
>
> → 브라우저에 결과 표시

🔷 JSP는 **Servlet이 만든 데이터를 "화면으로 보여주는 최종 페이지"**



FormAll.html

style.css

```
    :root{--bg:#f7f8fa;--card:#fff;--border:#e5e7eb;--text:#111;--primary:#0
d6efd;--primary-weak:#eef4ff}
    html,body{height:100%}
    body{background:var(--bg);color:var(--text)}
    .wrap{max-width:920px;margin:40px auto;padding:24px;background:var
(--card);border:1px solid var(--border);border-radius:12px}
    .panel-heading h3{margin:0 0 12px}
    table.table{margin:0}
    .table>tbody>tr>td{vertical-align:middle}
    .table>tbody>tr>td:first-child{width:160px;color:#555}
    input[type="text"],input[type="password"],input[type="email"],input[typ
```

```
e="url"],input[type="number"],input[type="date"],textarea,select{
    max-width:560px;width:100%;padding:7px 14px;border:1px solid #ddd;
border-radius:5px}
  input:focus,select:focus,textarea:focus{outline:none;border-color:var(--p
rimary);box-shadow:0 0 0 3px rgba(13,110,253,.08)}
  .d-flex{display:flex;align-items:center;gap:10px;flex-wrap:wrap}
  .option{display:inline-flex}
  .option input{display:none}
  .option span{display:inline-flex;align-items:center;gap:8px;padding:8px 1
2px;border:1px solid var(--border);border-radius:999px;line-height:1;curso
r:pointer}
  .option input:checked+span{border-color:var(--primary);background:var
(--primary-weak);font-weight:600}
  .option input:disabled+span{opacity:.5;cursor:not-allowed}
  .file-wrap{display:flex;align-items:center;gap:10px}
  .file-hidden{position:absolute;left:-9999px}
  .file-btn{display:inline-flex;align-items:center;justify-content:center;padd
ing:8px 14px;border-radius:8px;border:1px solid var(--border);background:
#fff;cursor:pointer}
  .file-btn.primary{border-color:var(--primary);background:var(--primary);
color:#fff}
  .file-name{color:#555}
  .btns{display:flex;gap:8px}
  .btns input[type="submit"],.btns input[type="reset"],.btns input[type="b
utton"]{padding:10px 16px;border-radius:10px;border:1px solid var(--borde
r);background:#fff;cursor:pointer}
  .btns input[type="submit"]{border-color:var(--primary);background:var
(--primary);color:#fff}
  .btns input[type="reset"]:hover{background:#f2f2f2}
  small.helper{display:block;color:#888;margin-top:6px}
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>다양한 Form 데이터 받기</title>
<link rel="stylesheet"
  href="https://unpkg.com/bootstrap@3.3.7/dist/css/bootstrap.min.css">
```

```
</head>
<body>
  <div class="wrap">
    <div class="panel-heading">
      <h3>Form</h3>
      <p class="text-muted">필요한 값을 입력하고 전송을 눌러주세요.</p>
    </div>
    <div class="panel-body">
      <form id="mainForm" action="formAll" method="post" novalidate>
        <table class="table">
          <tr>
            <td>이름</td>
            <td><input type="text" name="name" value="김현수" maxle
ngth="10"
              autocomplete="name" required></td>
          </tr>
          <tr>
            <td>아이디</td>
            <td><input type="text" name="id" value="acehs91"
              autocomplete="username" required></td>
          </tr>
          <tr>
            <td>패스워드</td>
            <td><input type="password" name="password" value="123
4"
              autocomplete="new-password" required></td>
          </tr>
          <tr>
            <td>주소</td>
            <td><input type="text" name="address" value="서울시 노원
구"
              placeholder="서울시" autocomplete="street-address"></td
>
          </tr>
          <tr>
            <td>주문수량</td>
            <td><input type="number" name="qty" value="1" min="0"
              max="10" step="2" inputmode="numeric"></td>
```

```
            </tr>
            <tr>
              <td>E-Mail</td>
              <td><input type="email" name="email" value="you@exampl
e.com"
                  autocomplete="email" placeholder="you@example.com">
</td>
            </tr>
            <tr>
              <td>URL</td>
              <td><input type="text" name="url" id="urlInput"
                value="example.com" inputmode="url"
                placeholder="example.com 또는 blog.example.com"> <sm
all
                class="helper">https:// 제외하고 작성해주세요.</small></td
>
            </tr>
            <tr>
              <td>날짜</td>
              <td><input type="date" name="date" id="dateInput"
                autocomplete="off"></td>
            </tr>
            <tr>
              <td>Comment</td>
              <td><textarea name="comment" rows="6" cols="50"
                  placeholder="요청사항을 입력하세요.">안녕하세요 풀스택 개
발자 김현수입니다.</textarea></td>
            </tr>

            <tr>
              <td>야구팀</td>
              <td>
                <div class="d-flex">
                  <label class="option"><input type="radio"
                    name="baseball" value="두산" checked><span>두산
</span></label> <label
                    class="option"><input type="radio" name="baseball"
                    value="samsung"><span>삼성</span></label> <labe
```

```
l class="option"><input
                        type="radio" name="baseball" value="LG"><span>L
G</span></label>
                    <label class="option"><input type="radio"
                        name="baseball" value="OB"><span>OB</span></la
bel> <label
                        class="option"><input type="radio" name="baseball"
                        value="KIA"><span>KIA</span></label> <label class
="option"><input
                        type="radio" name="baseball" value="KT"><span>K
T</span></label>
                </div>
            </td>
        </tr>

        <tr>
            <td>4강팀</td>
            <td>
                <div class="d-flex">
                    <label class="option"><input type="checkbox" name
="four"
                        value="두산" checked><span>두산</span></label> <l
abel
                        class="option"><input type="checkbox" name="fou
r"
                        value="samsung"><span>삼성</span></label> <labe
l class="option"><input
                        type="checkbox" name="four" value="LG"><span>L
G</span></label>
                    <label class="option"><input type="checkbox" name
="four"
                        value="OB"><span>OB</span></label> <label class
="option"><input
                        type="checkbox" name="four" value="KIA"><span>
KIA</span></label>
                    <label class="option"><input type="checkbox" name
="four"
                        value="KT"><span>KT</span></label>
```

```
                </div>
              </td>
            </tr>

            <tr>
              <td>인기구단</td>
              <td><select name="base" required>
                  <option value="" disabled selected hidden>구단을 선택해
주세요</option>
                  <option value="두산">두산</option>
                  <option value="samsung">삼성</option>
                  <option value="LG">LG</option>
                  <option value="OB">OB</option>
                  <option value="KIA">KIA</option>
                  <option value="KT">KT</option>
              </select></td>
            </tr>


            <!-- ✅ 새롭게 개선된 "인기구단2" 영역 -->
            <tr>
              <td>인기구단 (최대 2)</td>
              <td>
                <div class="d-flex" id="fav2Group" data-limit="2">
                  <label class="option"><input type="checkbox" name
="five"
                      value="두산"><span>두산</span></label> <label clas
s="option"><input
                      type="checkbox" name="five" value="samsung"><s
pan>삼성</span></label>
                  <label class="option"><input type="checkbox" name
="five"
                      value="LG"><span>LG</span></label> <label class
="option"><input
                      type="checkbox" name="five" value="OB"><span>O
B</span></label>
                  <label class="option"><input type="checkbox" name
="five"
```

```
                        value="KIA"><span>KIA</span></label> <label class
="option"><input
                        type="checkbox" name="five" value="KT"><span>K
T</span></label>
                    </div> <small class="helper" aria-live="polite"> <span
                      id="fav2Count">0</span>/2 선택 • 최대 2개까지 선택 가능
합니다.
                  </small>
                    <button type="button" class="file-btn" id="fav2Clear"
                      style="margin-top: 6px">선택 해제</button>
                  </td>
                </tr>

                <tr>
                  <td>File Upload</td>
                  <td>
                    <div class="file-wrap">
                      <input id="upload" class="file-hidden" type="file"
                        aria-label="파일 선택"> <label for="upload"
                        class="file-btn primary">파일 선택</label> <span id
="fileName"
                        class="file-name">선택된 파일 없음</span> <input typ
e="hidden"
                        name="upload" id="uploadName">
                    </div>
                  </td>
                </tr>

                <tr>
                  <td></td>
                  <td class="btns"><input id="submitBtn" type="submit"
                    value="전송"> <input type="reset" value="리셋"> <input
                    type="button" value="버튼"></td>
                </tr>
              </table>
            </form>
          </div>
        </div>
```

```
    <script>
        (function() {
            // 폼/버튼 참조
            var form = document.getElementById('mainForm');
            var submitBtn = document.getElementById('submitBtn');

            // 날짜: 클릭/포커스 시 달력 오픈
            (function() {
                var d = document.getElementById('dateInput');
                if (!d)
                    return;
                var openPicker = function() {
                    if (typeof d.showPicker === 'function')
                        d.showPicker();
                };
                d.addEventListener('focus', openPicker);
                d.addEventListener('click', openPicker);
            })();

            // 파일명 표시 + hidden 전송
            (function() {
                var fileInput = document.getElementById('upload');
                var fileNameSpan = document.getElementById('fileName');
                var uploadNameHidden = document.getElementById('uploadNam
e');
                if (!(fileInput && fileNameSpan && uploadNameHidden))
                    return;

                fileInput
                        .addEventListener(
                                'change',
                                function() {
                                    var name = (fileInput.files && fileInput.files.length) ? fil
eInput.files[0].name
                                            : '';
                                    fileNameSpan.textContent = name
                                            || '선택된 파일 없음';
```

```
                        uploadNameHidden.value = name; // FormAll.java: get
Parameter("upload")
                    });
        })();

        // 인기구단(최대 2개) 제한 로직
        (function() {
            var favGroup = document.getElementById('fav2Group');
            if (!favGroup)
                return;

            var limit = parseInt(favGroup.getAttribute('data-limit'), 10) || 2;
            var boxes = favGroup
                    .querySelectorAll('input[type="checkbox"][name="five"]');
            var counter = document.getElementById('fav2Count');
            var clearBtn = document.getElementById('fav2Clear');

            function update() {
                var checked = favGroup
                        .querySelectorAll('input[name="five"]:checked');
                var full = checked.length >= limit;
                boxes.forEach(function(b) {
                    if (!b.checked)
                        b.disabled = full;
                });
                if (counter)
                    counter.textContent = checked.length;
            }

            boxes.forEach(function(b) {
                b.addEventListener('change', update);
            });
            if (clearBtn) {
                clearBtn.addEventListener('click', function() {
                    boxes.forEach(function(b) {
                        b.checked = false;
                        b.disabled = false;
                    });
```

```
                update();
            });
        }
        update();
    })();

    // 제출 전 정리/검증/중복방지
    if (form) {
        form
            .addEventListener(
                'submit',
                function(e) {
                    // 1) 트림
                    var fields = form
                        .querySelectorAll('input[type="text"], input[type
="password"], input[type="email"], textarea');
                    fields.forEach(function(el) {
                        if (typeof el.value === 'string')
                            el.value = el.value.trim();
                    });

                    // 2) URL 자동 보정
                    var urlEl = document
                        .getElementById('urlInput');
                    if (urlEl && urlEl.value) {
                        var v = urlEl.value.trim().replace(
                            /[;,\.\u3002]+$/g, '');
                        if (!/^[a-zA-Z][a-zA-Z0-9+\-.]*:\/\//
                            .test(v))
                            v = 'https://' + v;
                        urlEl.value = v;
                    }

                    // 3) 인기구단(select name="base") 선택 확인
                    var baseSel = form
                        .querySelector('select[name="base"]');
                    if (baseSel && !baseSel.value) {
                        e.preventDefault();
```

```
                        alert('인기구단을 선택해주세요.');
                        baseSel.focus();
                        return;
                    }

                    // 4) 중복 제출 방지(간단)
                    if (submitBtn) {
                        submitBtn.disabled = true;
                        submitBtn.value = '전송 중...';
                        setTimeout(function() {
                            submitBtn.disabled = false;
                            submitBtn.value = '전송';
                        }, 4000);
                    }
                });
        }
    })();
  </script>
</body>
</html>
```

FormAll.java

```
package dao;

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

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

    // GET 요청 처리
```

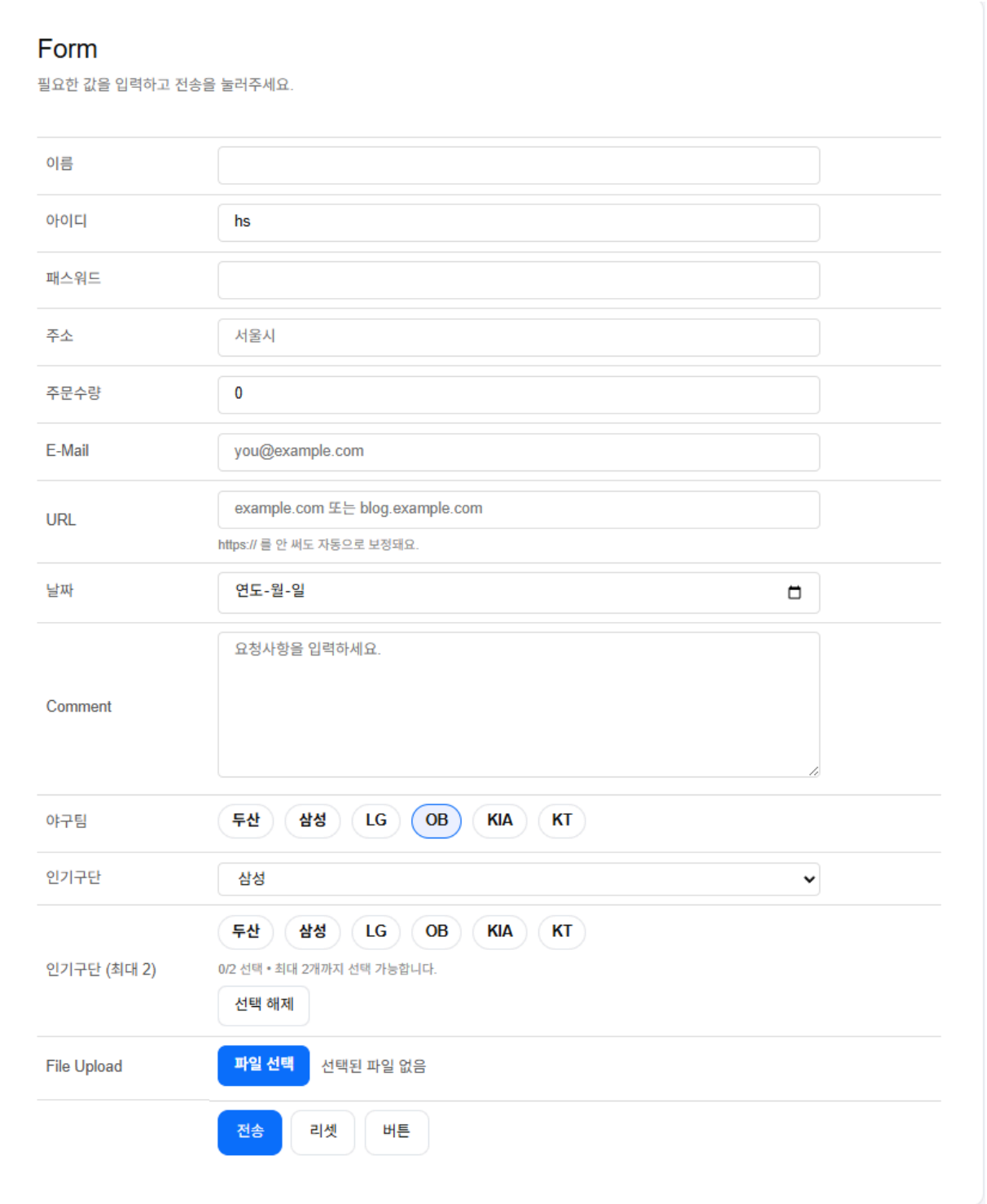
```
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        doGetPost(request, response);
    }

    // POST 요청 처리
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        doGetPost(request, response);
    }

    // 공통 처리 메서드
    protected void doGetPost(HttpServletRequest request, HttpServletRespo
nse response)
            throws ServletException, IOException {

        // 한글 인코딩 설정
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");

        // 출력 객체 생성
        PrintWriter out = response.getWriter();

        // 단일 입력 데이터 수신
        String name = request.getParameter("name");
        String id = request.getParameter("id");
        String password = request.getParameter("password");
        String address = request.getParameter("address");

        // 숫자형 데이터 변환
        int qty = Integer.parseInt(request.getParameter("qty"));

        String email = request.getParameter("email");
        String url = request.getParameter("url");
        String date = request.getParameter("date");
        String comment = request.getParameter("comment");
```

```
    // 라디오 버튼 (단일 선택)
    String baseball = request.getParameter("baseball");

    // 체크박스 (다중 선택)
    String[] four = request.getParameterValues("four");

    // 셀렉트 (단일 선택)
    String base = request.getParameter("base");

    // 셀렉트 (다중 선택)
    String[] five = request.getParameterValues("five");

    // 파일 업로드 (단일 데이터)
    String upload = request.getParameter("upload");

    // 콘솔 출력
    System.out.println(
        "이름: " + name +
        ", 아이디: " + id +
        ", 비밀번호: " + password +
        ", 주소: " + address +
        ", 수량: " + qty +
        ", 이메일: " + email +
        ", URL: " + url +
        ", 날짜: " + date +
        ", 코멘트: " + comment +
        ", 야구팀: " + baseball +
        ", 4강팀: " + arrayToString(four) +
        ", 인기구단: " + base +
        ", 인기구단(다중): " + arrayToString(five) +
        ", 업로드: " + upload
    );

  // 브라우저로 결과 HTML 출력
    out.println("<!doctype html>");
    out.println("<html><head><meta charset='utf-8'>");
    out.println("<title>제출 결과</title>");
```

```
        out.println("<link rel='stylesheet' href='https://unpkg.com/bootstrap@3.3.7/dist/css/bootstrap.min.css'>");
        out.println("<style>body{padding:24px;} .table td:first-child{width:180px;color:#555}</style>");
        out.println("</head><body>");
        out.println("<h3>제출 결과</h3>");
        out.println("<p class='text-muted'>폼으로 전송된 값을 그대로 표시합니다.</p>");

        out.println("<table class='table table-bordered'>");
        out.println("<tr><td>이름</td><td>" + name + "</td></tr>");
        out.println("<tr><td>아이디</td><td>" + id + "</td></tr>");
        out.println("<tr><td>비밀번호</td><td>" + password + "</td></tr>");
        out.println("<tr><td>주소</td><td>" + address + "</td></tr>");
        out.println("<tr><td>주문수량</td><td>" + qty + "</td></tr>");
        out.println("<tr><td>E-Mail</td><td>" + email + "</td></tr>");
        out.println("<tr><td>URL</td><td>" + url + "</td></tr>");
        out.println("<tr><td>날짜</td><td>" + date + "</td></tr>");
        out.println("<tr><td>Comment</td><td><pre style='margin:0;white-space:pre-wrap'>" + comment + "</pre></td></tr>");
        out.println("<tr><td>야구팀(라디오)</td><td>" + baseball + "</td></tr>");
        out.println("<tr><td>4강팀(체크박스)</td><td>" + arrayToString(four) + "</td></tr>");
        out.println("<tr><td>인기구단(셀렉트)</td><td>" + base + "</td></tr>");
        out.println("<tr><td>인기구단(최대2 체크)</td><td>" + arrayToString(five) + "</td></tr>");
        out.println("<tr><td>업로드(파일명)</td><td>" + upload + "</td></tr>");
        out.println("</table>");

        out.println("<div style='margin-top:16px'>");
        out.println("<a href='javascript:history.back()' class='btn btn-default'>뒤로</a> ");
        out.println("<a href='" + request.getContextPath() + "/' class='btn btn-primary'>홈으로</a>");
        out.println("</div>");
```

```
        out.println("</body></html>");
        out.flush();
    }

    // 배열 null 체크 후 문자열로 변환
    private String arrayToString(String[] arr) {
        if (arr == null) return "선택 없음";
        return String.join(", ", arr);
    }


}
```

완성된화면



index.html

```
<!DOCTYPE html>
<html>
<head>
```

```
  <meta charset="UTF-8">
  <title>회원가입</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://unpkg.com/bootstrap@3.3.7/dist/css/
bootstrap.min.css">
  <style>
    .wrap{max-width:768px;margin:56px auto;padding:0 14px}
    .card{border:1px solid #e5e7eb;border-radius:14px;background:#fff;box-
shadow:0 10px 30px rgba(0,0,0,.06)}
    .card-head{padding:22px 26px;border-bottom:1px solid #f1f5f9;border-t
op-left-radius:14px;border-top-right-radius:14px;background:#fff}
    .card-title{margin:0;font-size:22px;font-weight:700;color:#111827}
    .card-sub{margin:6px 0 0;color:#6b7280}
    .card-body{padding:26px}

    .row-g{margin-left:-10px;margin-right:-10px}
    .row-g>[class*="col-"]{padding-left:10px;padding-right:10px}

    label{font-size:14px;color:#111827;margin-bottom:8px;font-weight:600}
    .form-control{height:46px;padding:10px 14px;border:1px solid #e5e7eb;b
order-radius:12px;box-shadow:none;transition:border .15s ease, box-shado
w .15s ease}
    .form-control:focus{border-color:#2563eb;box-shadow:0 0 0 4px rgba(3
7,99,235,.15)}
    .form-control::placeholder{color:#9ca3af}
    textarea.form-control{height:auto;min-height:100px}

    select.form-control{appearance:none;-webkit-appearance:none;-moz-a
ppearance:none;background-image:url("data:image/svg+xml;utf8,<svg xml
ns='http://www.w3.org/2000/svg' width='20' height='20' viewBox='0 0 20
20'><path fill='%23737B87' d='M5.5 7.5l4.5 4.5 4.5-4.5H5.5z'/></svg>");b
ackground-repeat:no-repeat;background-position:right 12px center;backgr
ound-size:18px;padding-right:40px}

    .btn-primary{background:#2563eb;border-color:#2563eb;font-weight:6
00;border-radius:10px;padding:12px 18px}
    .btn-primary:hover{background:#1d4ed8;border-color:#1d4ed8}
    .btn-primary:active,.btn-primary:focus{background:#1e40af;border-colo
```

```
r:#1e40af}

    .hint{font-size:12px;color:#9ca3af;margin-top:6px}

    @media (max-width:768px){
      .btn-block-xs{display:block;width:100%}
    }

    @media (prefers-color-scheme:dark){
      body{background:#0b0f14}
      .card{background:#0f172a;border-color:#1f2937;box-shadow:0 10px 30
px rgba(0,0,0,.4)}
      .card-head{background:#0f172a;border-bottom-color:#1f2937}
      .card-title{color:#e5e7eb}
      .card-sub{color:#94a3b8}
      label{color:#e5e7eb}
      .form-control{background:#0b1220;border-color:#243044;color:#e5e7
eb}
      .form-control::placeholder{color:#64748b}
      .form-control:focus{border-color:#60a5fa;box-shadow:0 0 0 4px rgba
(96,165,250,.16)}
      .btn-primary{background:#3b82f6;border-color:#3b82f6}
      .btn-primary:hover{background:#2563eb;border-color:#2563eb}
    }
  </style>
</head>
<body>
  <div class="wrap">
    <div class="card">
      <div class="card-head">
        <h3 class="card-title">회원가입</h3>
      </div>
      <div class="card-body">
        <form id="signupForm" action="JoinServlet" method="post" novalidat
e>
          <div class="row row-g">
            <div class="col-sm-6">
              <div class="form-group">
```

```
            <label for="name">이름</label>
            <input id="name" name="name" value="홍길동" type="text" class
="form-control" required>
          </div>
        </div>
        <div class="col-sm-6">
          <div class="form-group">
            <label for="nick">별명</label>
            <input id="nick" name="nick" value="KD" type="text" class="for
m-control">
          </div>
        </div>

        <div class="col-sm-12">
          <div class="form-group">
            <label for="email">이메일</label>
            <input id="email" name="email" value="kd@naver.com" type="e
mail" class="form-control" placeholder="you@example.com" required>
          </div>
        </div>

        <div class="col-sm-12">
          <div class="form-group">
            <label for="addr">주소</label>
            <input id="addr" name="addr" type="text" value="서울특별시 강남
구" class="form-control" placeholder="서울특별시 강남구">
            <p class="hint">도로명 주소를 입력하세요</p>
          </div>
        </div>

        <div class="col-sm-12">
          <div class="form-group">
            <label for="addrd">상세주소 (필수 아님)</label>
            <input id="addrd" name="addrd" type="text" value="테헤란로 42
4" class="form-control" placeholder="상세주소를 입력해주세요.">
          </div>
        </div>
```

```
          <div class="col-sm-6">
            <div class="form-group">
              <label for="path">가입 경로</label>
              <select id="path" name="path" class="form-control">
                <option value="" disabled selected>선택하세요</option>
                <option value="search">검색</option>
                <option value="ad">광고</option>
                <option value="friend">지인추천</option>
                <option value="etc">기타</option>
              </select>
            </div>
          </div>

          <div class="col-sm-6">
            <div class="form-group">
              <label for="code">추천인 코드</label>
              <input id="code" name="code" value="KD123"type="text" class
="form-control">
            </div>
          </div>
        </div>

        <div class="text-right">
          <button type="submit" id="submitBtn" class="btn btn-primary btn-l
g btn-block">전송</button>
        </div>
      </form>
    </div>
  </div>
</div>

<script>
(function(){
  var form = document.getElementById('signupForm');
  var btn  = document.getElementById('submitBtn');
  var path = document.getElementById('path');

  form.addEventListener('submit', function(e){
```

```
    // 가입경로
    if (path && (!path.value || path.value === '')) {
      e.preventDefault();
      alert('가입 경로를 선택해주세요.');
      path.focus();
      return false;
    }

    // 공백 제거
    [].forEach.call(form.querySelectorAll('input,textarea'), function(el){
      if (el.type !== 'file') el.value = el.value.trim();
    });

  });
})();
</script>
</body>
</html>
```

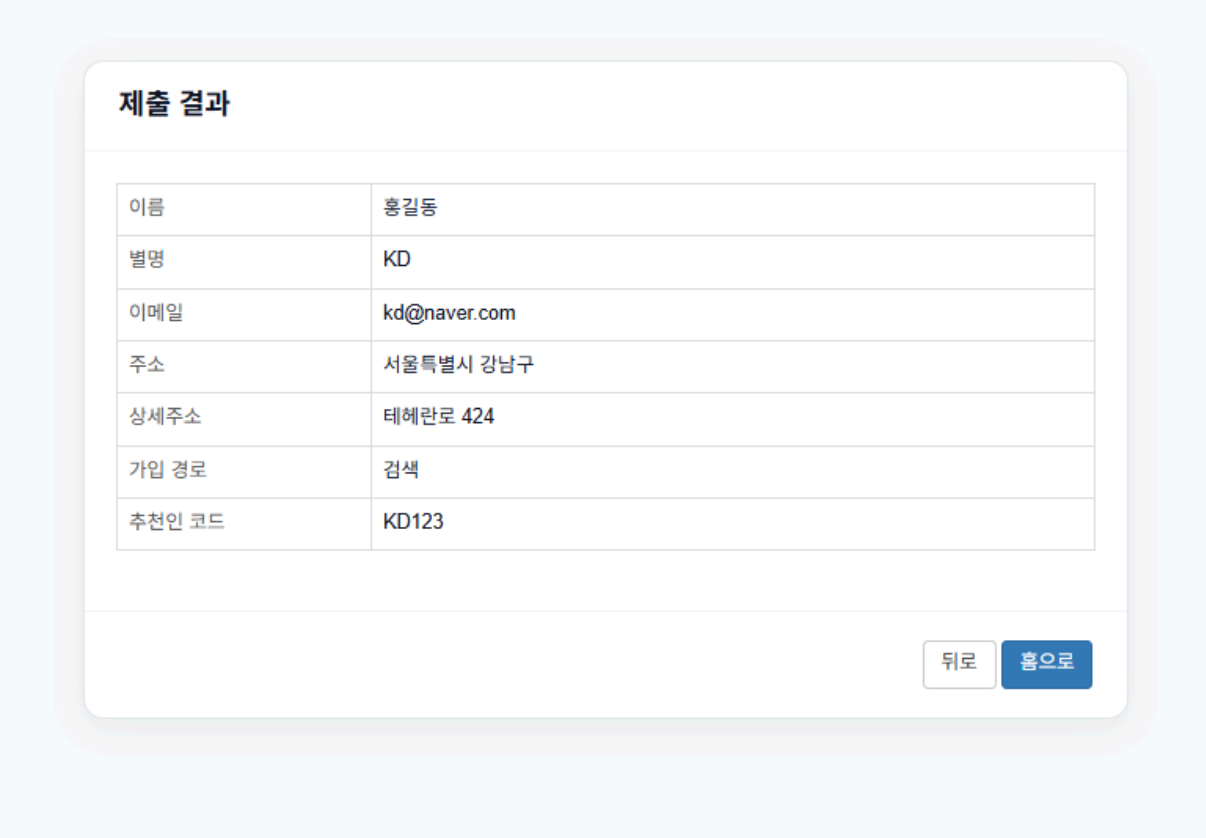
Join.java

```
package JoinServlet;

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

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

```
    protected void doGet(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException { doGetPost(request, respons
e); }

    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException { doGetPost(request, respons
e); }

    protected void doGetPost(HttpServletRequest request, HttpServletRespo
nse response)
            throws ServletException, IOException {
        request.setCharacterEncoding("utf-8");
        response.setContentType("text/html; charset=utf-8");

        PrintWriter out = response.getWriter();

        String name  = request.getParameter("name");
        String nick  = request.getParameter("nick");
        String email = request.getParameter("email");
        String addr  = request.getParameter("addr");
        String addrd = request.getParameter("addrd");
        String path  = request.getParameter("path");
        String code  = request.getParameter("code");

        System.out.println("이름: " + name + ", 별명: " + nick + ", 이메일: " + em
ail +
                ", 주소: " + addr + " " + addrd + ", 가입경로: " + path + ", 추천코드: "
+ code);

        out.println("<!doctype html>");
        out.println("<html><head><meta charset='utf-8'>");
        out.println("<title>제출 결과</title>");
        out.println("<meta name='viewport' content='width=device-width, initi
al-scale=1'>");
        out.println("<link rel='stylesheet' href='https://unpkg.com/bootstrap@
```

```
3.3.7/dist/css/bootstrap.min.css'>");
        out.println("<style>");
        out.println("body{background:#f9fafb;color:#111827}");
        out.println(".wrap{max-width:768px;margin:48px auto;padding:0 14p
x}");
        out.println(".card{background:#fff;border:1px solid #e5e7eb;border-ra
dius:14px;box-shadow:0 6px 24px rgba(0,0,0,.06)}");
        out.println(".card-head{padding:20px 24px;border-bottom:1px solid #f
1f5f9;border-radius:14px 14px 0 0}");
        out.println(".card-title{margin:0;font-size:20px;font-weight:700}");
        out.println(".card-body{padding:22px}");
        out.println(".table>thead>tr>th{background:#f8fafc;font-weight:60
0}");
        out.println(".table td:first-child{width:180px;color:#555}");
        out.println(".actions{padding:20px 24px;border-top:1px solid #f1f5f9;b
order-radius:0 0 14px 14px;text-align:right}");
        out.println(".no-print{display:inline-block}");
        out.println("@media print{body{background:#fff}.wrap{max-width:10
0%;margin:0;padding:0}.card{border:0;box-shadow:none;border-radius:
0}.card-head,.actions{display:none}.table{font-size:12pt}}");
        out.println("</style>");
        out.println("</head><body>");
        out.println("<div class='wrap'>");
        out.println("  <div class='card'>");
        out.println("    <div class='card-head'><h3 class='card-title'>제출 결과
</h3></div>");

        out.println("    <div class='card-body'>");
        out.println("      <table class='table table-bordered table-hover'>");
        out.println("        <tbody>");
        out.println("          <tr><td>이름</td><td>" + nv(name) + "</td></tr>");
        out.println("          <tr><td>별명</td><td>" + nv(nick) + "</td></tr>");
        out.println("          <tr><td>이메일</td><td>" + nv(email) + "</td></tr
>");
        out.println("          <tr><td>주소</td><td>" + nv(addr) + "</td></tr>");
        out.println("          <tr><td>상세주소</td><td>" + nv(addrd) + "</td></t
r>");
        out.println("          <tr><td>가입 경로</td><td>" + pathLabel(path) + "
```

```
</td></tr>");
        out.println("         <tr><td>추천인 코드</td><td>" + nv(code) + "</td>
</tr>");
        out.println("       </tbody>");
        out.println("     </table>");

        out.println("    </div>");
        out.println("    <div class='actions'>");
        out.println("      <a href='javascript:history.back()' class='btn btn-defa
ult'>뒤로</a> ");
        out.println("      <a href='" + request.getContextPath() + "/' class='btn
btn-primary'>홈으로</a> ");
        out.println("    </div>");
        out.println("  </div>");
        out.println("</div>");

        out.println("</body></html>");
        out.flush();
    }

    private String pathLabel(String v) {
        if (v == null || v.isEmpty()) return "";
        switch (v) {
            case "search": return "검색";
            case "ad": return "광고";
            case "friend": return "지인추천";
            case "etc": return "기타";
            default: return v;
        }
    }

    private String nv(String s){
        return s == null ? "" : s;
    }
}
```



🧩 서버통신 정리

1️⃣ forward (포워드)

사용자가 브라우저에서 서버로 요청을 보낸다.

서버는 그 요청을 받아서 하나의 `request` 객체를 만든다.

`request` 객체 안에는 파라미터나 `setAttribute()` 데이터가 들어 있다.

이후 서버 내부에서 `RequestDispatcher.forward()` 를 사용하면

같은 `request` 객체를 다음 서블릿이나 JSP로 그대로 넘긴다.

브라우저는 새 요청을 하지 않고, 주소창도 그대로 유지된다.

결과적으로 한 번의 요청 안에서 데이터가 이어진다.

주소창은 그대로고, 데이터도 그대로 남아있다.

👉 데이터 유지됨 / request 동일 / 주소 안 바뀜

쓰임새:

로그인 결과를 JSP로 넘길 때,

DB 조회 후 결과 페이지로 이동할 때,

같은 요청 안에서 데이터를 전달할 때.

2️⃣ redirect (리디렉트)

브라우저가 서버에 요청을 보낸다.

서버는 응답으로 "다른 주소로 다시 요청해라"는 명령을 보낸다.

브라우저는 서버의 지시에 따라 다시 요청을 보낸다.

이때는 완전히 새로운 요청이기 때문에

기존의 `request` 객체는 사라지고,

새로운 `request`가 만들어진다.

그래서 `setAttribute()` 로 저장했던 데이터는 사라진다.

데이터를 이어서 보내고 싶다면 쿼리스트링이나 세션을 사용해야 한다.

👉 새 요청 / request 새로 생성 / 주소 바뀜 / 데이터 사라짐

쓰임새:

회원가입 완료 후 로그인 페이지로 이동할 때,

게시글 작성 후 목록으로 돌아갈 때,

폼 재전송을 방지해야 할 때.

3️⃣ binding (바인딩)

서블릿에서 데이터를 잠시 저장해 두는 방법이다.

`setAttribute("키", 값)` 으로 데이터를 저장하고,

`getAttribute("키")` 로 꺼낸다.

이 데이터는 저장된 위치(스코프)에 따라 유효 범위가 다르다.

request에 저장하면 요청 한 번 동안만,

session에 저장하면 브라우저 닫기 전까지,

application에 저장하면 서버 전체에서 공유된다.

👉 request(데이터)는 한 번만 임시 저장 / session은 브라우저 닫을 때까지 / 스코프에 따라 유지 범위 다름

쓰임새:

서블릿 간 데이터 전달,

로그인 정보나 사용자 데이터 임시 저장,

서버 전체에서 공유되는 설정값 관리.

4️⃣🔒 세션 (Session)

사용자가 사이트에 접속하면 서버는 세션을 하나 만든다.

세션에는 사용자의 로그인 정보나 장바구니 같은 데이터를 저장할 수 있다.

브라우저에는 그 세션을 구분할 수 있는 "세션 ID"만 쿠키 형태로 남는다.

다음 요청이 들어올 때 서버는 그 세션 ID를 보고

같은 사용자임을 알아차리고, 저장된 데이터를 불러온다.

브라우저를 닫거나, 일정 시간이 지나면 세션은 사라진다.

👉 **서버가 기억함 / 브라우저엔 세션 ID만 남음 / 로그인 유지용**

쓰임새:

로그인 상태 유지,

장바구니 관리,

사용자별 데이터 저장,

페이지 이동 간 사용자 정보 유지.

5️⃣🍪 쿠키 (Cookie)

- **브라우저(클라이언트)** 에 저장되는 데이터.
- 서버가 쿠키를 내려주면 브라우저가 그걸 보관하고
  다음 요청마다 자동으로 서버에 다시 보낸다.
- 누구나 볼 수 있어서 **보안이 약하다.**

👉 **브라우저가 기억함 / 자동 전송 / 보안 약함**

쓰임새:

자동 로그인,

최근 본 상품 저장,

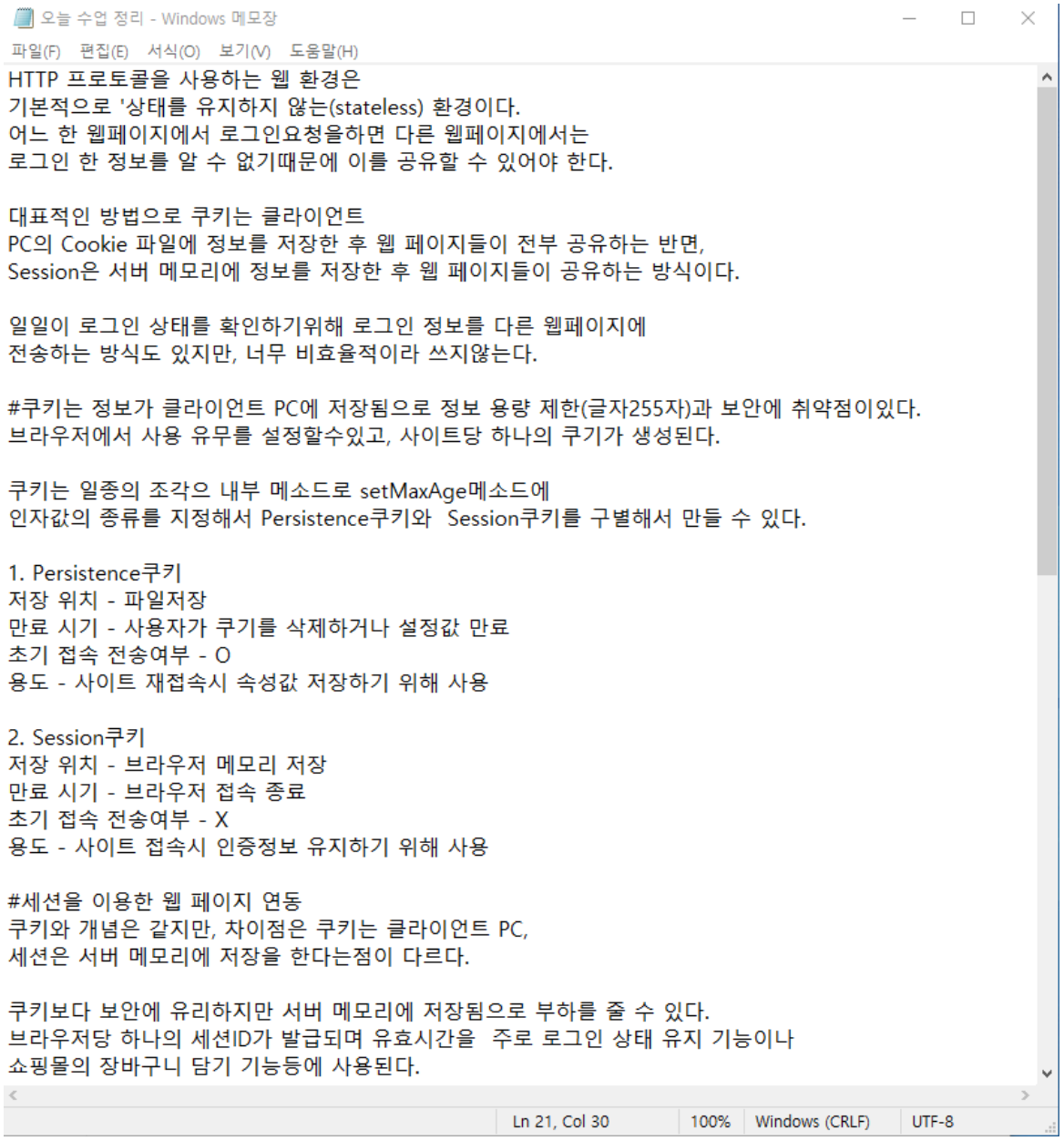
팝업창 닫기 여부 기억,

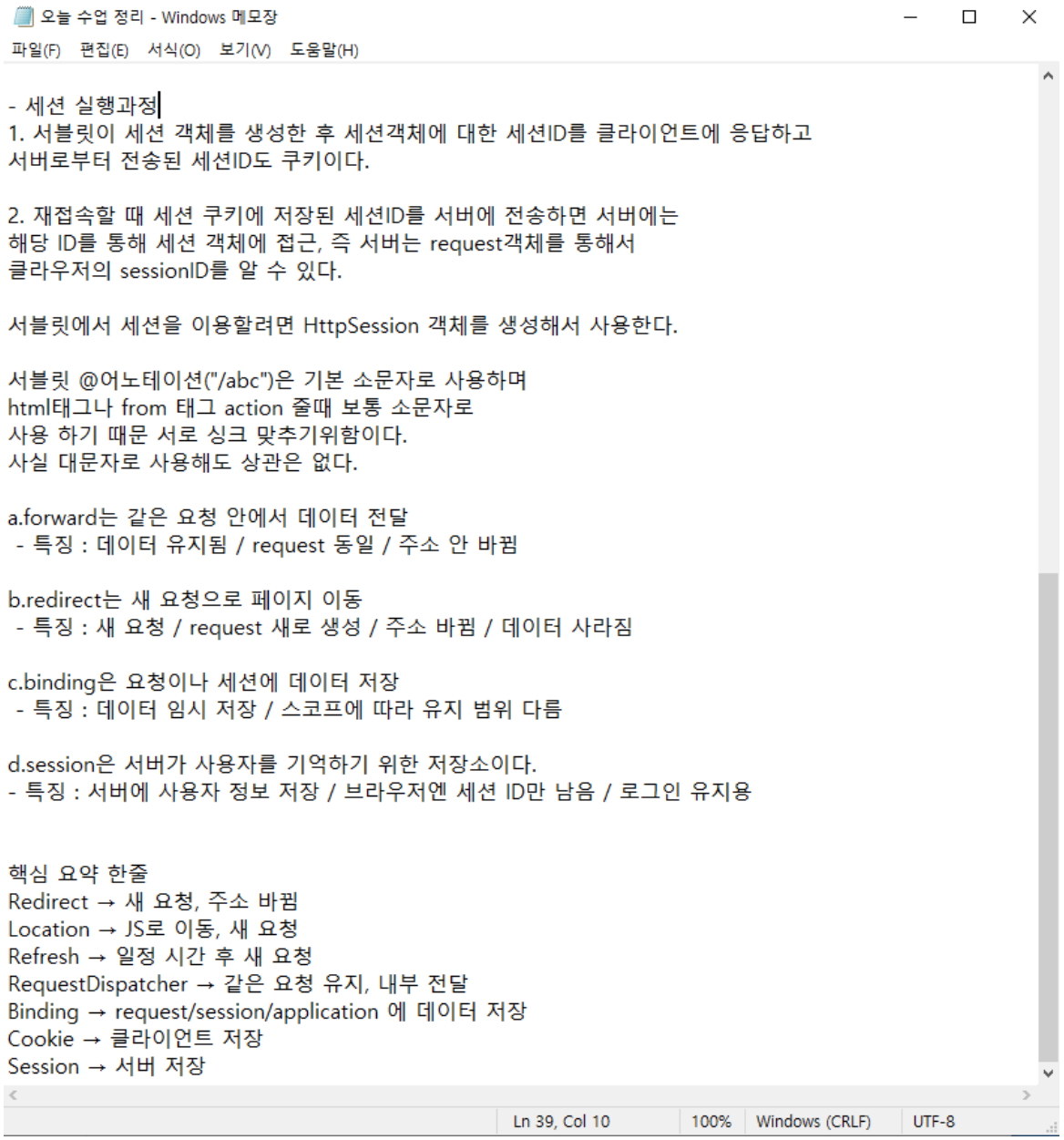
사용자 맞춤 설정 저장.

✅ 핵심 요약 한 줄로

- Redirect → 새 요청, 주소 바뀜
- Location → JS로 이동, 새 요청
- Refresh → 일정 시간 후 새 요청
- RequestDispatcher → 같은 요청 유지, 내부 전달
- Binding → request/session/application 에 데이터 저장
- Cookie → 클라이언트 저장
- Session → 서버 저장

> 쿠키,세션,포워드 개념.txt


오늘 수업 정리 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
HTTP 프로토콜을 사용하는 웹 환경은
기본적으로 '상태를 유지하지 않는(stateless) 환경이다.
어느 한 웹페이지에서 로그인요청을하면 다른 웹페이지에서는
로그인 한 정보를 알 수 없기때문에 이를 공유할 수 있어야 한다.
대표적인 방법으로 쿠키는 클라이언트
PC의 Cookie 파일에 정보를 저장한 후 웹 페이지들이 전부 공유하는 반면,
Session은 서버 메모리에 정보를 저장한 후 웹 페이지들이 공유하는 방식이다.
일일이 로그인 상태를 확인하기위해 로그인 정보를 다른 웹페이지에
전송하는 방식도 있지만, 너무 비효율적이라 쓰지않는다.
#쿠키는 정보가 클라이언트 PC에 저장됨으로 정보 용량 제한(글자255자)과 보안에 취약점이있다.
브라우저에서 사용 유무를 설정할수있고, 사이트당 하나의 쿠기가 생성된다.
쿠키는 일종의 조각으 내부 메소드로 setMaxAge메소드에
인자값의 종류를 지정해서 Persistence쿠키와 Session쿠키를 구별해서 만들 수 있다.
1. Persistence쿠키
저장 위치 - 파일저장
만료 시기 - 사용자가 쿠기를 삭제하거나 설정값 만료
초기 접속 전송여부 - O
용도 - 사이트 재접속시 속성값 저장하기 위해 사용
2. Session쿠키
저장 위치 - 브라우저 메모리 저장
만료 시기 - 브라우저 접속 종료
초기 접속 전송여부 - X
용도 - 사이트 접속시 인증정보 유지하기 위해 사용
#세션을 이용한 웹 페이지 연동
쿠키와 개념은 같지만, 차이점은 쿠키는 클라이언트 PC,
세션은 서버 메모리에 저장을 한다는점이 다르다.
쿠키보다 보안에 유리하지만 서버 메모리에 저장됨으로 부하를 줄 수 있다.
브라우저당 하나의 세션ID가 발급되며 유효시간을 주로 로그인 상태 유지 기능이나
쇼핑몰의 장바구니 담기 기능등에 사용된다.
Ln 21, Col 30
100%
Windows (CRLF)
UTF-8


오늘 수업 정리 - Windows 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
- 세션 실행과정
1. 서블릿이 세션 객체를 생성한 후 세션객체에 대한 세션ID를 클라이언트에 응답하고
서버로부터 전송된 세션ID도 쿠키이다.
2. 재접속할 때 세션 쿠키에 저장된 세션ID를 서버에 전송하면 서버에는
해당 ID를 통해 세션 객체에 접근, 즉 서버는 request객체를 통해서
클라우저의 sessionID를 알 수 있다.
서블릿에서 세션을 이용할려면 HttpSession 객체를 생성해서 사용한다.
서블릿 @어노테이션("/abc")은 기본 소문자로 사용하며
html태그나 from 태그 action 줄때 보통 소문자로
사용 하기 때문 서로 싱크 맞추기위함이다.
사실 대문자로 사용해도 상관은 없다.
a.forward는 같은 요청 안에서 데이터 전달
- 특징 : 데이터 유지됨 / request 동일 / 주소 안 바뀜
b.redirect는 새 요청으로 페이지 이동
- 특징 : 새 요청 / request 새로 생성 / 주소 바뀜 / 데이터 사라짐
c.binding은 요청이나 세션에 데이터 저장
- 특징 : 데이터 임시 저장 / 스코프에 따라 유지 범위 다름
d.session은 서버가 사용자를 기억하기 위한 저장소이다.
- 특징 : 서버에 사용자 정보 저장 / 브라우저엔 세션 ID만 남음 / 로그인 유지용
핵심 요약 한줄
Redirect → 새 요청, 주소 바뀜
Location → JS로 이동, 새 요청
Refresh → 일정 시간 후 새 요청
RequestDispatcher → 같은 요청 유지, 내부 전달
Binding → request/session/application 에 데이터 저장
Cookie → 클라이언트 저장
Session → 서버 저장
Ln 39, Col 10
100%
Windows (CRLF)
UTF-8