



# Application Development

Dept. of Computer Science  
Hanyang University



A blue outline of a cloud shape, with the word "Outline" written inside it in a dark grey font.

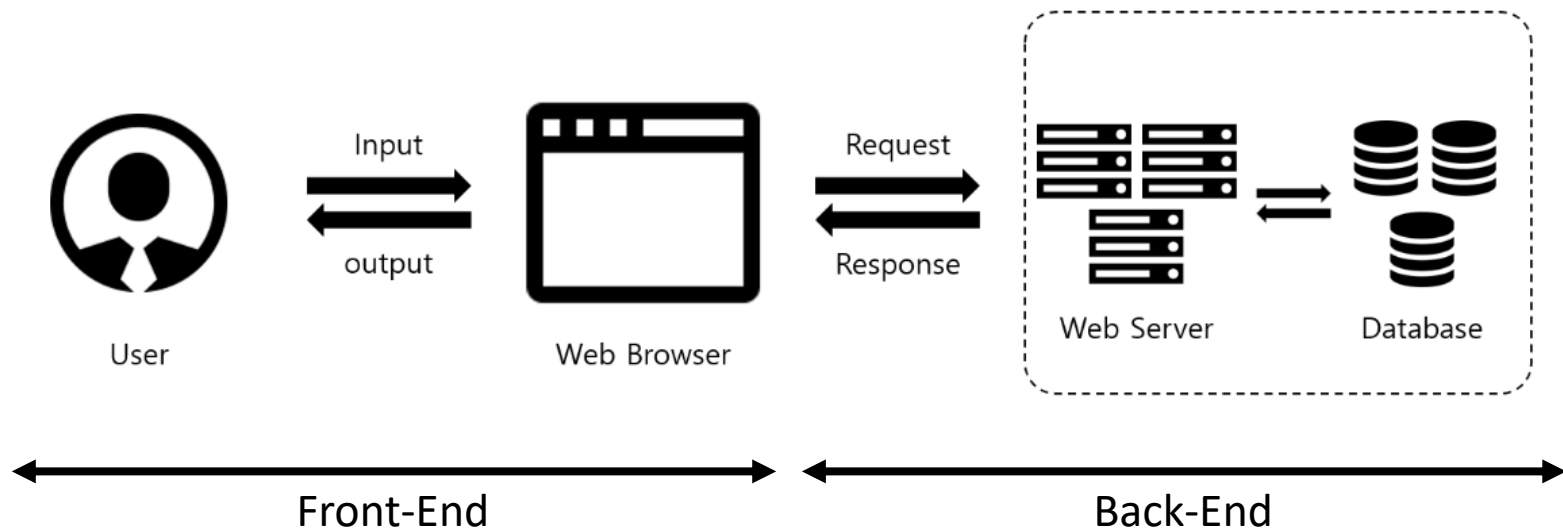
# Outline

---

- Application Programs & User Interfaces
- Servlet
- JSP
- Project#1



- Most database users do *not* use a query language like SQL
- An application program acts as the intermediary between users and the database
  - Applications split into
    - front-end
    - middle layer
    - backend





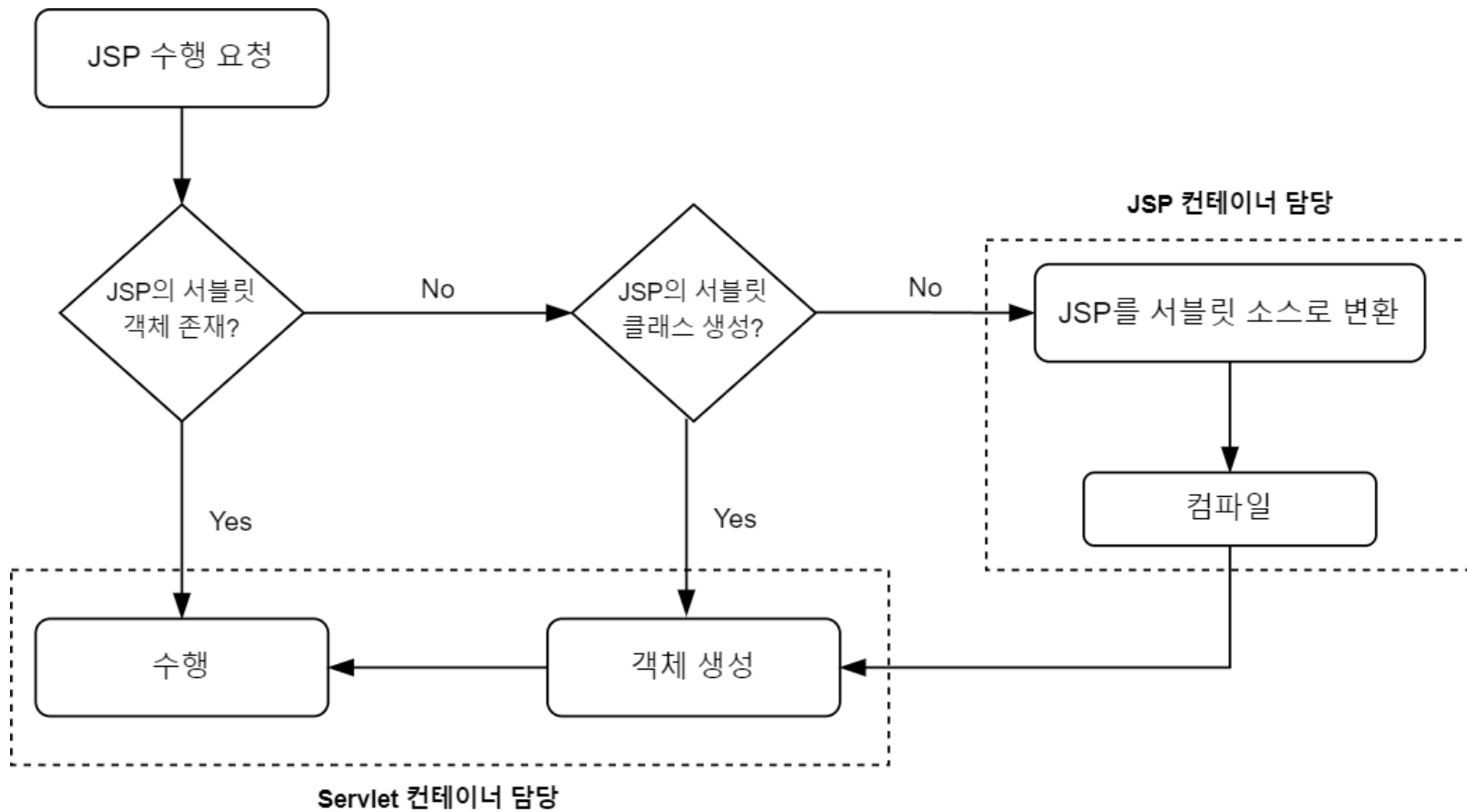
- Java Servlet specification defines an API for communication between the Web/application server and application program running in the server
  - E.g., methods to get parameter values from Web forms, and to send HTML text back to client
- Application program (also called a servlet) is loaded into the server
  - Each request spawns a new thread in the server
    - thread is closed once the request is serviced
  - Programmer creates a class that inherits from HttpServlet
    - And overrides methods doGet, doPost, ...
  - Mapping from servlet name (accessible via HTTP), to the servlet class is done in a file web.xml
    - Done automatically by most IDEs when you create a Servlet using the IDE
- 필요한 객체들은 따로 만들 필요 없이 WAS에서 제공하는 서블릿 API를 가져다 사용 가능
  - <https://tomcat.apache.org/> → 접속 후 왼쪽 [Documentation] 아래 자신의 'Tomcat버전' 을 선택 → Reference 영역에서 자신의 Tomcat버전에서 지원하는 서블릿은 x.0 버전이라는 내용 확인 → 그 옆에 'Javadoc' 링크 클릭

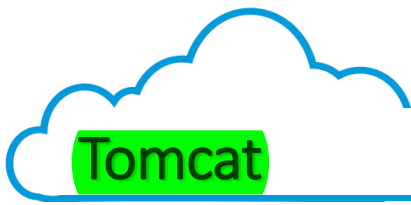


# Java Server Pages (JSP)

- Servlet과 동일한 기술이라 보아도 무방
- 차이가 있다면 표현하는 방법과 web application에서의 역할이 다름
- 차이1 : 표현하는 방법의 차이
  - 서블릿은 자바언어로 구현해야하지만, JSP는 HTML 페이지 안에서 스크립트 형태로 구현
  - 일부 서블릿 객체는 변수 선언과 초기화 작업 없이 바로 사용 가능
  - 내부적으로 자동으로 코드가 생성됨
- 차이2 : Web Application에서의 역할
  - 서블릿은 컨트롤러 페이지를 만들 때 사용되지만 JSP는 뷰 페이지를 만들 때 사용
  - 뷰는 단순히 클라이언트가 보는 화면으로서, 클라이언트로부터 요청받거나 처리된 결과를 보여주는 페이지를 의미
  - 따라서 뷰에서 들어온 요청을 받아 처리하는 페이지가 컨트롤러 페이지이며, 이는 서블릿으로 구현해야 함

# Servlet-JSP Process





- Web Application Server 의 오픈소스
- Java EE 스펙을 모두 갖추지 있지 않고, JSP와 서블릿을 실행하는 컨테이너와 웹서버만 제공
- 톰캣에서 제공하는 웹서버의 기능은 아주 기본적인 기능만 하므로 일반적으로 독립적인 웹서버를 설치한 후 톰캣과 연동하여 실행되는 구조 구축



## Development Environment

- 서블릿과 JSP를 개발하려면 자바 개발 환경과 실행 환경, 그리고 웹에서의 실행 환경 구축 필요
- 자바 소스를 컴파일하기 위한 도구 **JDK**(Java Development Kit)
- 웹서버 Tomcat
  - 기본적인 기능을 갖춘 웹서버 포함
  - 서블릿을 수행하는 서블릿 컨테이너 포함
  - JSP를 수행하는 JSP 컨테이너 포함
- 자바 및 다양한 언어를 지원하는 프로그래밍 통합 개발 환경을 지원하는 도구
  - ex. eclipse / **IntelliJ** / etc..

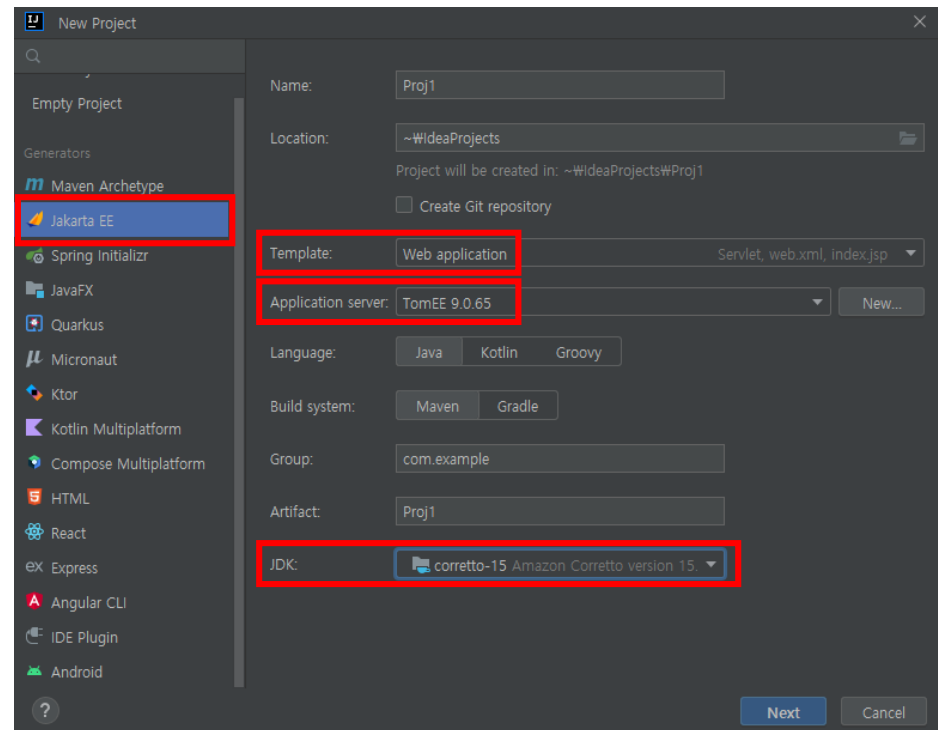
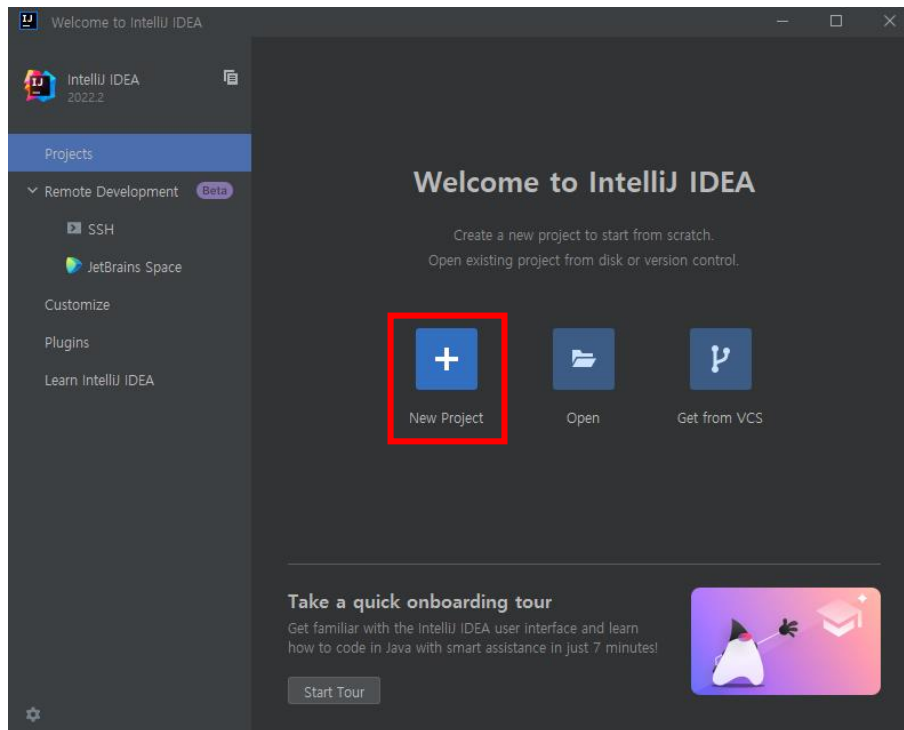




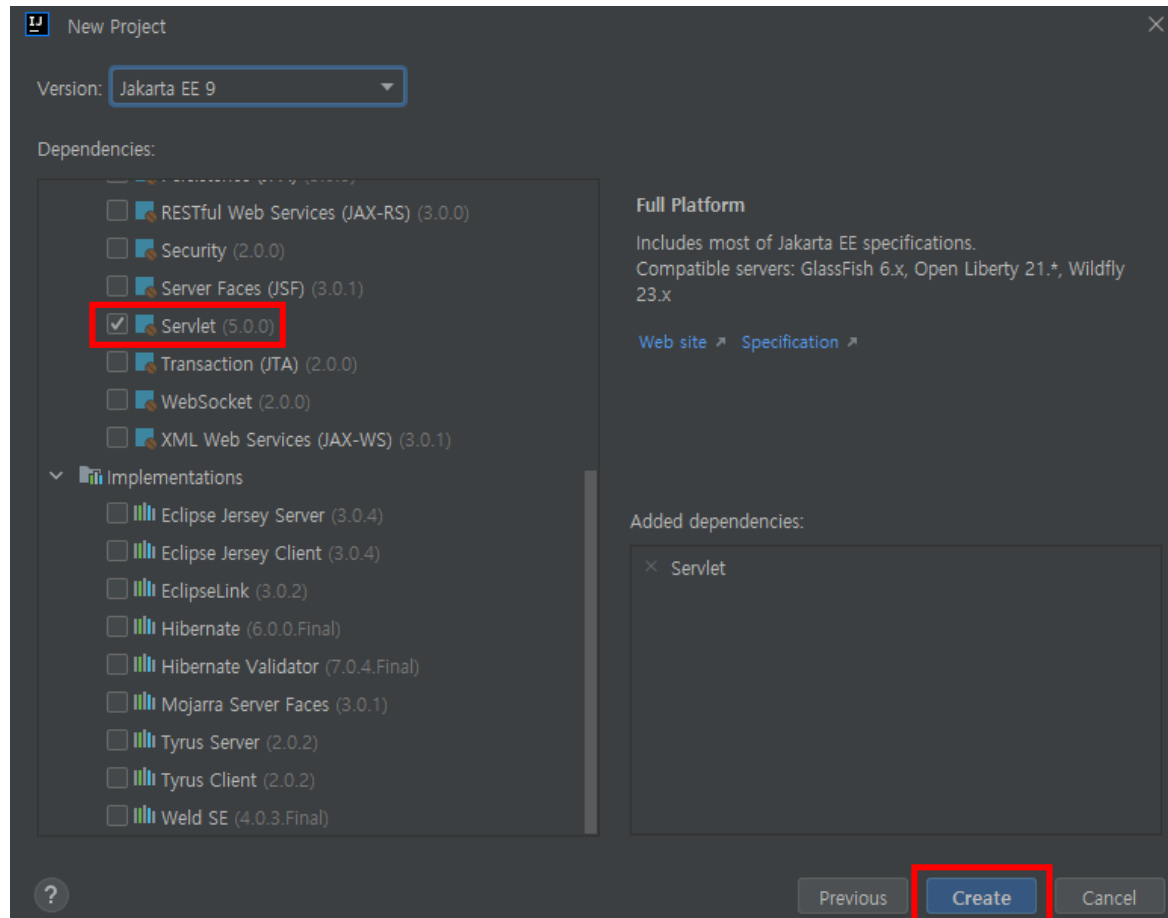
## Setting & Option : Web Application Dvelop Project #1

# Make Project

EE?

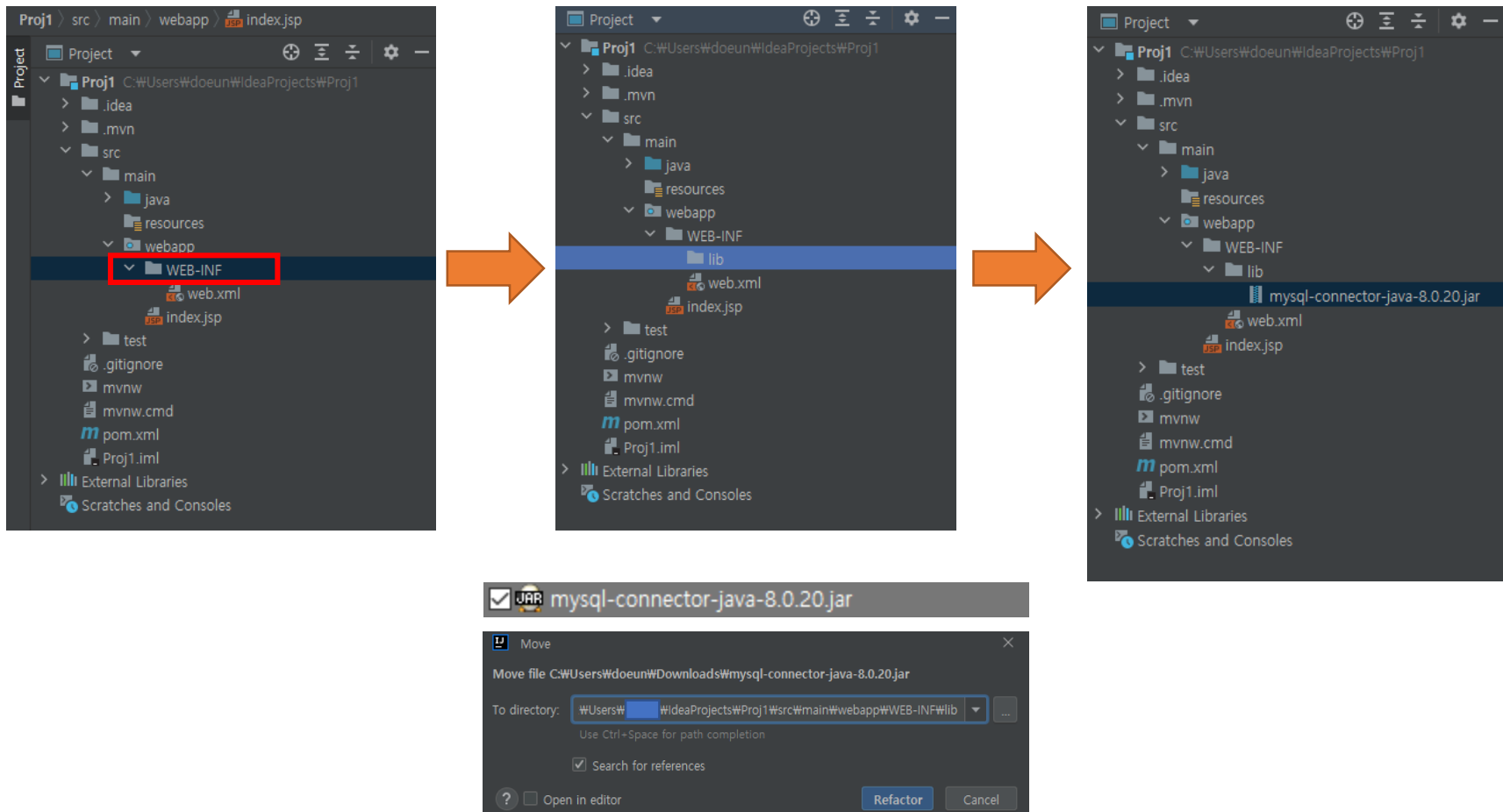


# Make Project



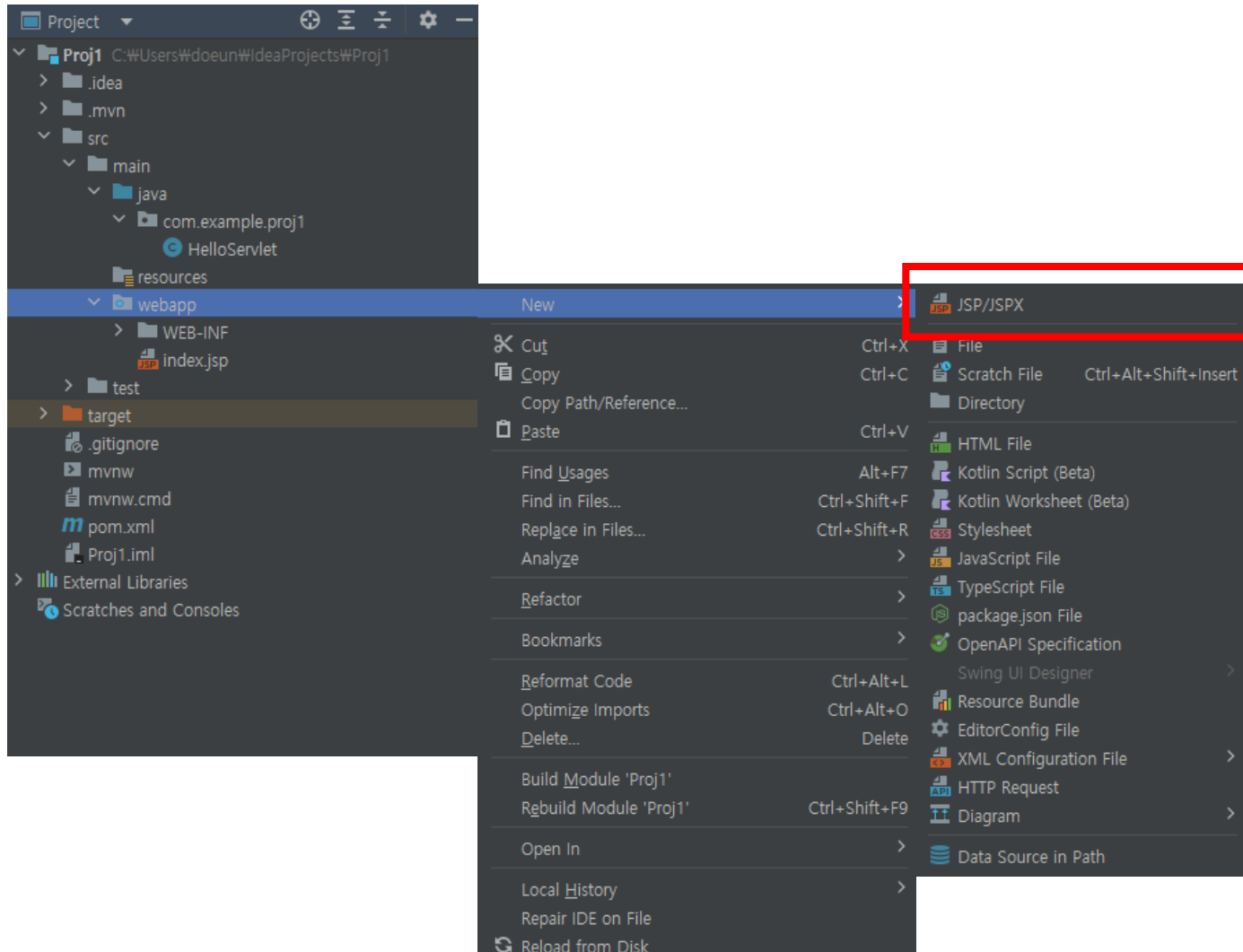
# Make Project

- 다운받은 .jar 파일을 WEB-INF/lib 디렉토리 하위로 이동



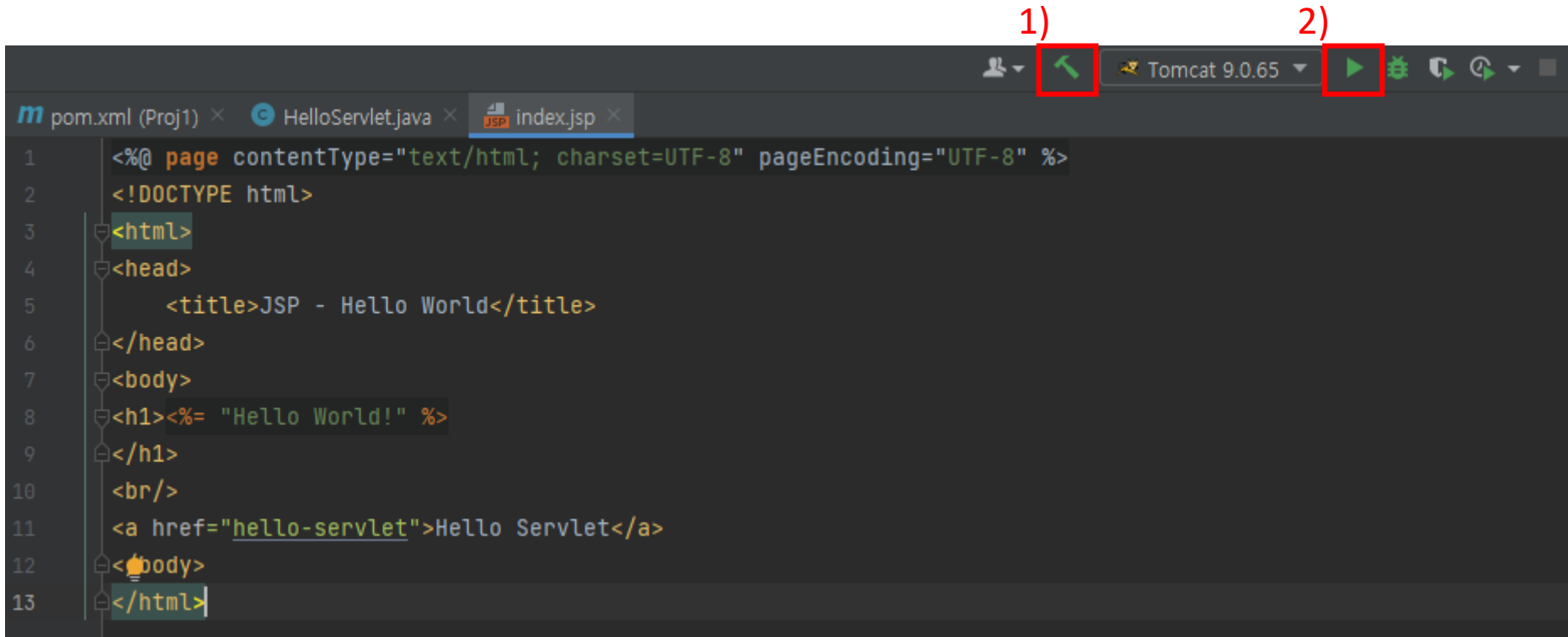
# Make Project

- Web에서 서비스할 문서들은 모두 webapp 폴더 아래에 생성



# Make Project

- Build → Run Tomcat



The screenshot shows an IDE with a dark theme. At the top, there are tabs for 'pom.xml (Proj1)', 'HelloServlet.java', and 'index.jsp'. The 'index.jsp' tab is active, showing the following code:

```
1 <%@ page contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
2 <!DOCTYPE html>
3 <html>
4 <head>
5     <title>JSP - Hello World</title>
6 </head>
7 <body>
8 <h1><%= "Hello World!" %>
9 </h1>
10 <br/>
11 <a href="hello-servlet">Hello Servlet</a>
12 </body>
13 </html>
```

In the top right corner, there is a toolbar. Two buttons are highlighted with red boxes and labeled with red numbers:

- 1) A green left-pointing arrow button (Run button).
- 2) A green right-pointing arrow button (Run button).

Next to these buttons is a dropdown menu showing 'Tomcat 9.0.65'.

# Make Project

- Server connection



**Hello World!**

[Hello Servlet](#)