

사용자 요청 처리

egy@induk.ac.kr
<http://lms.induk.ac.kr>

학습 개요

■ 학습 배경

- 웹 애플리케이션은 사용자의 요청을 받아 동적인 웹 페이지를 생성하거나 서버에 데이터를 저장하는 작업을 수행하고 수행 결과를 응답한다. 따라서 효율적인 개발을 위해서 요청-응답 방법에 대한 정확한 이해가 필요하다.

■ 학습 목표

- HTTP 요청과 응답에 대하여 알아본다.

■ 주요 용어

- GET, POST

학습 내용

- **HTTP에 대하여 알아본다.**
 - 특징, 발전, 단계별 동작
 - HTTP 메시지, HTTP 요청, HTTP 응답
 - HTTP 상태 코드, 콘텐츠 타입
- **HTML 활용에 대하여 알아본다.**
 - 구조, <form>, <div>, <table>
- **클라이언트 요청 방식에 대하여 알아본다.**
 - GET
 - POST
- **한글 처리에 대하여 알아본다.**

HTTP 개요

■ 특징

- 분산되고, 협업가능한 하이퍼미디어 정보 시스템을 위한 통신규약으로 클라이언트의 요청과 서버의 응답을 처리하는 클라이언트-서버 방식을 사용한다.
- HTTP는 무상태(stateless) 프로토콜이다.
- 웹 브라우저가 실행되는 컴퓨터는 각각의 새로운 HTTP GET 또는 POST 요청마다 웹 서버와 TCP 네트워크 연결을 수립(설립, establish)해야 한다. 웹 서버는 하나의 HTTP GET 또는 POST 요청보다 더 길게 TCP 네트워크 연결을 유지할 수 없다.
- HTTP는 데이터의 올바른 전송을 위해 TCP/IP를 사용한다.

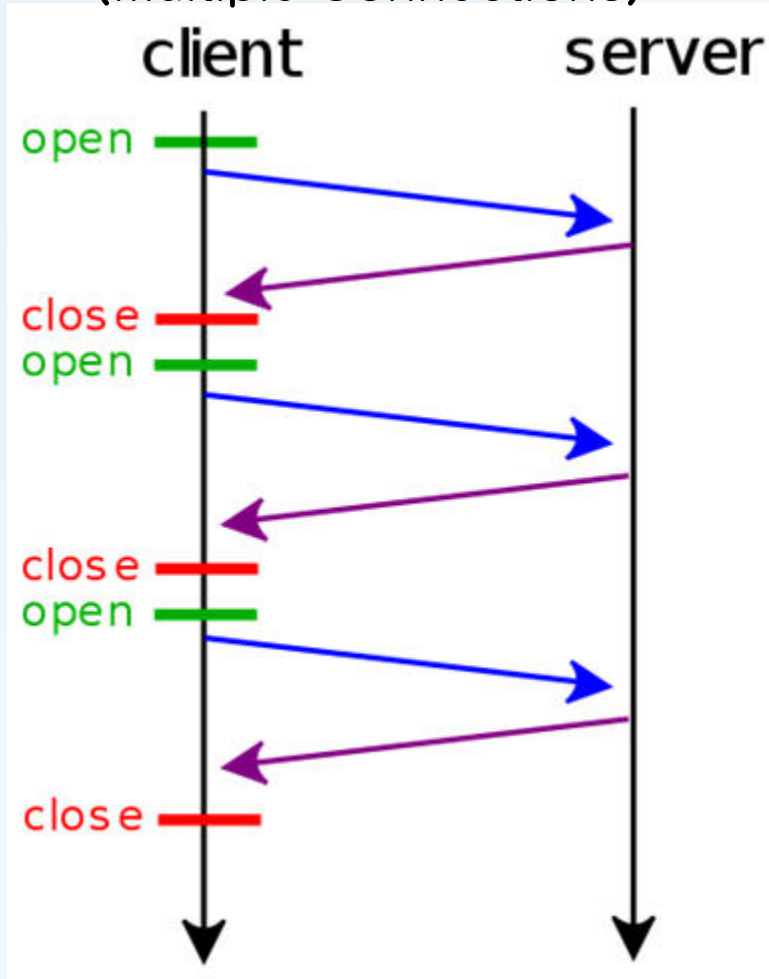
계속

• 발전

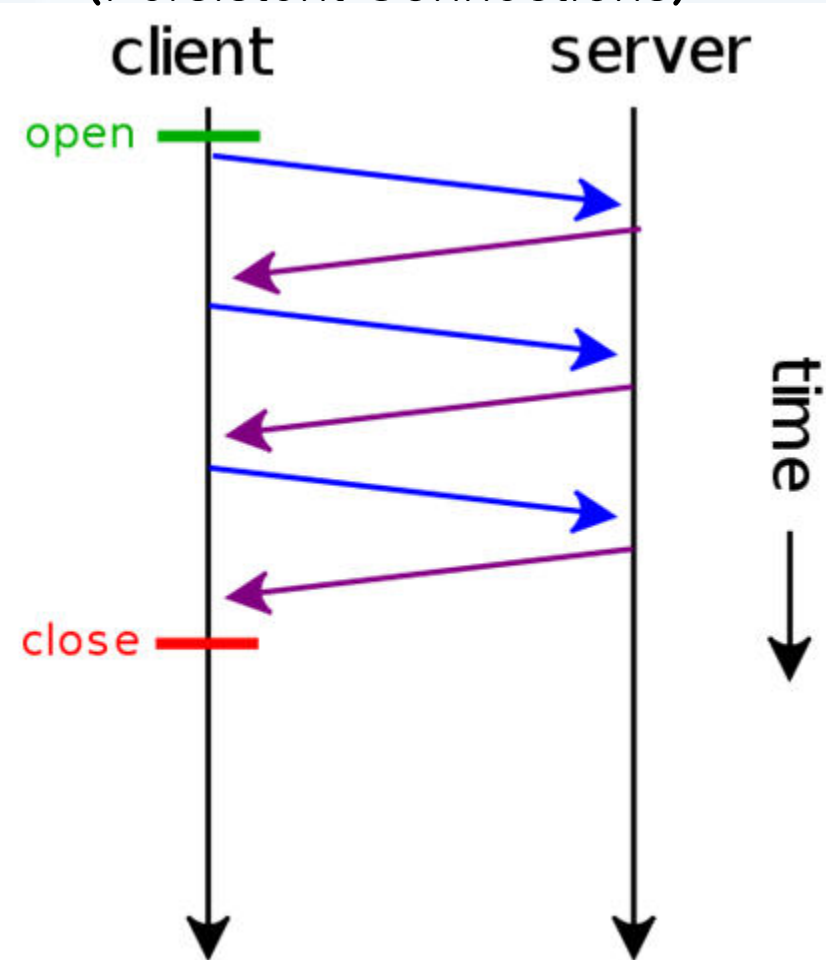
- 1996년 RFC(Requests for Comments) 에서 HTTP/1.0을 소개되었고, 승인되었다.
 - HTTP/1.0은 동일한 서버에 대하여 요청-응답 트랜잭션(request-response transaction)을 처리하는 과정에서 독립적인 또는 다수의 연결(connection)을 사용하는 문제가 있다.
- 현재 사용되고 있는 HTTP 버전은 RFC 2616을 기반으로 1999년 6월에 출시된 HTTP/1.1이 표준으로 사용되고 있다.
 - HTTP 지속적인 연결(persistent connection)을 제공한다.
 - HTTP 연결 재사용(connection reuse) 또는 HTTP keep-alive라고도 하며, 다수의 HTTP 요청들과 응답들을 보내고 받는 경우 매번 연결을 생성하지 않고, 같은 TCP 연결을 사용하는 방법을 의미한다. 이 방법은 TCP 연결 설정(establishment)시 발생하는 지연을 줄일 수 있기 때문에 향상된 처리가 가능하다.

계속

다수 연결
(Multiple Connections)



지속적인 연결
(Persistent Connections)



계속

• 단계별 동작

- 1단계) 연결 설정
- 2단계) 요청 메시지 전송 (HTTP 요청)
- 3단계) 응답 메시지 전송 (HTTP 응답)
- 4단계) 연결 끊기

HTTP 메시지(Message)

- 정의

- HTTP 대화의 기본 단위(basic unit of HTTP communication)이다.

- 구조

- Start-Line

- Request-Line | Status-Line

- Message-Headers

- {field-name ":" [field-value]}^N

- Cache-Control | Connection | Date | Pragma | Trailer | Transfer-Encoding | Upgrade | Via | Warning

- CRLF

- Message Body

- entity-body | <entity-body encoded as per Transfer-Encoding>

요청 메시지(Request Message), 요청 포맷

- **Request-Line**

- Method SP Request-URI SP HTTP-Version CRLF

- Method : HTTP 메소드

- OPTIONS | GET | HEAD | POST | PUT | DELETE | TRACE | CONNECT | extension-method

- Request-URI : 웹 서버 상에 존재하는 자원의 경로

- HTTP-Version : 프로토콜 버전

- **Request-Headers**

- {field-name ":" [field-value]}^N

- Accept | Accept-Charset | Accept-Encoding | Accept-Language | Authorization | Expect | From | Host | If-Match | If-Modified-Since | If-None-Match | If-Range | If-Unmodified-Since | Max-Forwards | Proxy-Authorization | Range | Referer | TE | User-Agent

- **CRLF**

- **[Message-Body(Payload)]**

응답 메시지 (Response Message), 응답 포맷

- **Status-Line**

- HTTP-Version SP Status-Code SP Reason-Phrase CRLF

- HTTP-Version : 프로토콜 버전
 - **Status Code** : HTTP 상태 코드
 - Reason-Phrase : 상태 코드에 대한 텍스트

- **Response-Headers**

- {field-name ":" [field-value]}^N

- Accept_Ranges | Age | ETag | Location | Proxy-Authenticate |
Retry-After | Server | Vary | WWW-Authenticate

- **CRLF**

- **Message-Body**

- Content-Encoding(**Content-Type**(data))

계속

• HTTP 상태 코드

- 1XX : 정보(Informational)
- 2XX : 성공(Success)
 - 200 OK
- 3XX : 재지정(Redirection)
- 4XX : 클라이언트 오류(Client Error)
 - 400 Bad Request - 요청 구문의 형식이 잘못된(malformed) 경우
 - 401 Unauthorized - 접근 권한이 없는 경우
 - 403 Forbidden - 웹 서버가 요청을 이해했지만 처리를 거절하는 경우
 - 404 Not Found - URL 오류로 해당 자원을 찾지 못하는 경우
- 5XX : 서버 오류(Server Error)
 - 500 Internal Server Error - 불특정한 오류가 발생한 경우
 - 501 Not Implemented - 구현되지 않는 기능 요청으로 오류가 발생한 경우
 - 502 Bad Gateway - 업스트림(upstream) 서버로부터 잘못된 응답을 받은 경우
 - 503 Service Unavailable - 과부하 또는 유지보수 등 일시적인 오류가 발생한 경우
 - 504 Gateway Timeout - 업스트림(upstream) 서버나 보조 서버들로부터 정해진 시간 안에 응답을 받지 못한 경우

<http://tools.ietf.org/html/rfc2068#section-6.1.1>

계속

• 콘텐츠 타입

- 웹 브라우저는 응답 헤더의 "Content-Type" 항목에 지정된 값으로 자원의 종류를 구별하는데 사용한다.
- 전송된 콘텐츠의 타입을 구별해주는 식별자를 MIME(Multipurpose Internet Mail Extensions) 타입이라고 한다.
- "Content-Type: **text/plain** " 에서 "text/plain"은 일반적인 텍스트 문서를 의미한다.
 - "text/html"은 HTML 문서를 의미한다
 - "application/vnd.ms-excel"은 마이크로소프트 엑셀 파일을 의미하고, 참고로 "vnd"는 특정한 공급자(vender)를 의미한다.
- http://en.wikipedia.org/wiki/Internet_media_type

요청 헤더와 응답 헤더 정보 확인

■ 방법

- request, response 객체를 활용한 프로그래밍으로 확인
- Postman 등과 같은 프로그램을 확인

계속

■ Postman

- a collaboration platform for API development.
Postman's features simplify each step of building an API and streamline collaboration so you can create better APIs—faster
- API 개발을 위한 협동 플랫폼
 - design and mock, debug, test, document, monitor, and publish Your's API



New

Import

Runner



My Workspace ▾



Invite



Upgrade ▾



Filter

History

Collections

APIs

☒ Save Responses

You haven't sent any requests

Any request you send in this workspace will appear here.



Show me how

Launchpad X

GET jsp

GET http://st...

GET http://st...

GET https://s...



No Environment ▾



Hey there, early bird!

Let's start the day off right. Use Launchpad to start something new, pick up where you left off, or explore some resources to help you master Postman.

Start something new



Create a request



Create a collection ▾



Create an environment



Create an API



View More

Recent workspaces

Customize



Dark mode



Enable Launchpad

Work smarter with Postman

Learn how Postman can help you at every stage of the API development lifecycle with these in-app tutorials.



Designing and mocking APIs

2 lessons



Debugging and manual testing

4 lessons



Automated testing

4 lessons



API documentation

1 lesson



Monitoring

1 lesson



Collaboration

2 lessons

What's new with Postman



Launchpad

GET http://student.induk.a... X GET https://ssl.pstatic.net/tv... (+) ... No Environment

Untitled Request BUILD

GET http://student.induk.ac.kr Send Save

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Code

<input checked="" type="checkbox"/>	Postman-Token ⓘ	<calculated when request is sent>	
<input checked="" type="checkbox"/>	Host ⓘ	<calculated when request is sent>	
<input checked="" type="checkbox"/>	User-Agent ⓘ	PostmanRuntime/7.26.5	
<input checked="" type="checkbox"/>	Accept ⓘ	*/*	
<input checked="" type="checkbox"/>	Accept-Encoding ⓘ	gzip, deflate, br	
<input checked="" type="checkbox"/>	Connection ⓘ	keep-alive	

Get Request 헤더

Status: 200 OK Time: 32 ms Size: 29.8 KB Save Response

Connection ⓘ	keep-alive
Content-Type ⓘ	text/html; charset=utf-8
Cache-Control ⓘ	no-cache
Cache-Control ⓘ	no-store
Pragma ⓘ	no-cache
Transfer-Encoding ⓘ	chunked

Bootcamp Build Browse

Response 콘텐츠

Launchpad POST http://student.induk.a... GET https://ssl.pstatic.net/tv... No Environment

Untitled Request BUILD

POST http://student.induk.ac.kr?USER_ID=egyout&USER_PW=spring0522! Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

<input checked="" type="checkbox"/>	Host ⓘ	<calculated when request is sent>
<input checked="" type="checkbox"/>	User-Agent ⓘ	PostmanRuntime/7.26.5
<input checked="" type="checkbox"/>	Accept ⓘ	*/*
<input checked="" type="checkbox"/>	Accept-Encoding ⓘ	gzip, deflate, br

Status: 200 OK Time: 22 ms Size: 29.8 KB Save Response

Pretty Raw Preview Visualize HTML

```
1 <!DOCTYPE html>
2 <html lang="ko">
3
4 <head>
5
6   <meta charset="utf-8">
7
8   <meta http-equiv="x-ua-compatible" content="ie=edge">
9
10  <meta name="format-detection" content="telephone=no">
11  <meta name="description" content="인덕대학교 종합정보시스템, 인덕대학교, Idu종합정보시스템, 인덕대 종합정보시스템">
12  <meta name="Keywords" content="인덕대학교 종합정보시스템 | 인덕대학교">
13  <meta name="viewport"
```

계속

• Get 요청 헤더

HTTP Request Header

Connect to 220.67.174.104 on port 8080 ... ok

GET /prj01/index.html HTTP/1.1[CRLF]

Host: 220.67.174.104[CRLF]

Connection: close[CRLF]

User-Agent: Web-sniffer/1.0.44 (+http://web-sniffer.net/) [CRLF]

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 [CRLF]

Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4 [CRLF]

Accept-Charset: ISO-8859-1,UTF-8;q=0.7,*;q=0.7 [CRLF]

Cache-Control: no-cache [CRLF]

Referer: http://web-sniffer.net/ [CRLF]

[CRLF]

계속

• Post 요청 헤더

HTTP Request Header

Connect to 220.67.174.104 on port 8080 ... ok

```
POST /prj01/index.html HTTP/1.1[CRLF]
Host: 220.67.174.104[CRLF]
Connection: close[CRLF]
User-Agent: Web-sniffer/1.0.44 (+http://web-sniffer.net/) [CRLF]
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 [CRLF]
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4 [CRLF]
Accept-Charset: ISO-8859-1,UTF-8;q=0.7,*;q=0.7 [CRLF]
Cache-Control: no-cache[CRLF]
Referer: http://web-sniffer.net/ [CRLF]
Content-type: application/x-www-form-urlencoded[CRLF]
Content-length: 0 [CRLF]
[CRLF]
```

Simplest HTML

- 구조

```
<html>
<head>
  <title> ~ </title>
  <meta> ~ </meta>
</head>
<body>
  <div>
    <table> ~ </table>
    <form> ~ </form>
  </div>
</body>
</html>
```

- Layout

- using <div> elements
- using <table> elements
- 참고
 - http://www.w3schools.com/html/html_layout.asp

- Listing

- ordered list
- unordered list
- 참고
 - http://www.w3schools.com/html/html_lists.asp

변화

▪ HTML4 의미론적 요소

- `<div id="nav">`
`<div class="header">`
`<div id="footer">`
- 개발자별로 id나 class 속성으로 지정하기 때문에 검색엔진이 html 파일을 분석할 때 정확하게 콘텐츠를 식별하기 어렵다.

▪ HTML5 offers new semantic elements to define different parts of a web page:

- `<article>`
- `<aside>`
- `<details>`
- `<figcaption>`
- `<figure>`
- `<footer>`
- `<header>`
- `<main>`
- `<mark>`
- `<nav>`
- `<section>`
- `<summary>`
- `<time>`

계속

■ <table>

- width
- border
- <thead>, <tbody>, <tfoot>
 - <tbody>만 여러번 나타날 수 있다.
- <tr>, <th>
- <td>
 - colspan, rowspan
- 참조
 - http://www.w3schools.com/html/html_tables.asp

■ <form> 태그

- action
 - 사용자 요청 파라미터를 전송할 웹 애플리케이션을 지정한다.
- method
 - 사용자 요청을 전송하는 방식을 지정한다.
- name
 - 폼의 이름을 지정한다.
- 참조
 - http://www.w3schools.com/html/html_for_ms.asp

계속

▪ `<input>` 태그

- type

- text
 - readonly
- password
- submit
- reset
- button

- checkbox

- checked

- radio

- hidden

- file

- name

- value

계속

■ <select> 태그

- name
- multiple
- size
- <option>
 - value
 - selected

■ <textarea> 태그

- cols
- rows
- name
- readonly
- wrap
 - off, soft, hard

계속

■ Internal Style Sheet

- `<style type="text/css">`
`body {background-color:yellow;}`
`p {color:blue;}`
`</style>`

■ External Style Sheet

- `<link rel="stylesheet" type="text/css" href="mystyle.css">`

■ Internal JavaScript

- `<script> ~ </script>`

■ External JavaScript

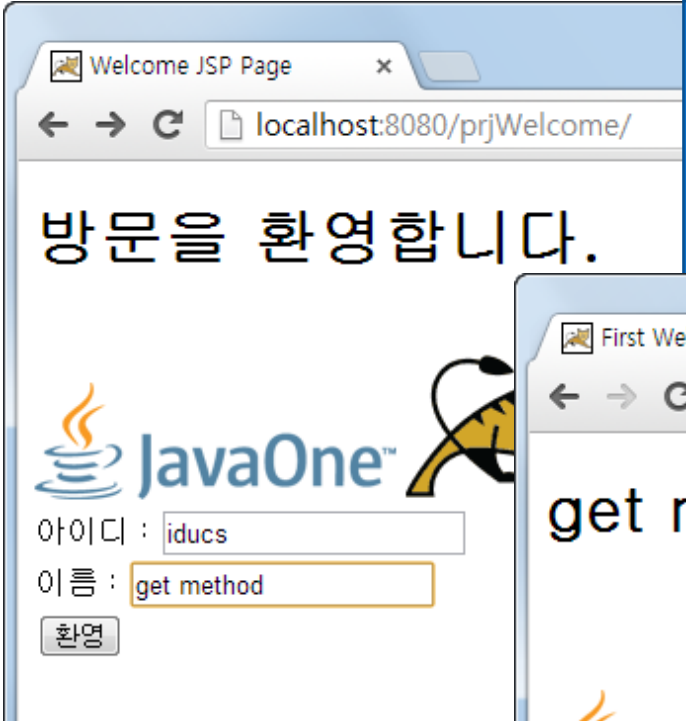
- `<script language="javascript" src="myscript.js" type="text/javascript"> ~ </script>`

사용자 요청 전송을 위한 HTML 폼 작성


- HTTP를 이용한 사용자 요청을 전송하기 위한 방법
 - 쿼리스트링을 이용하는 방법
 - `<http>://<domain_name>[:<port_number>]/<file_path>?<search_part>`
 - 폼을 이용한 방법
 - `<form method="get or post"> ~ </form>`

계속

- 폼을 이용한 Get 방식으로 요청

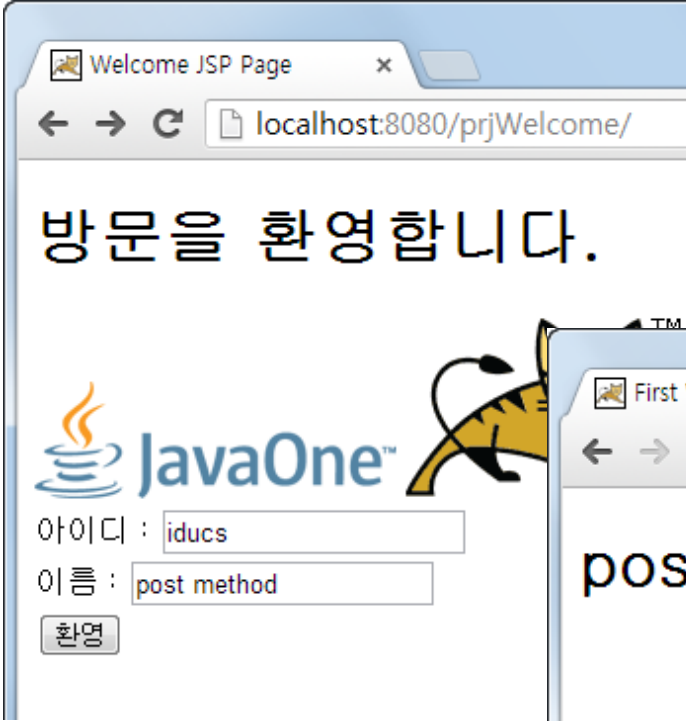


```
<form action="todayHello.jsp" method="get">  
아이디 : <input type="text" name="id" ><br>  
이름 : <input type="text" name="name"><br>  
<input type="submit" value="환영">
```




계속

- 폼을 이용한 Post 방식으로 요청



```
<form action="todayHello.jsp" method="post">  
아이디 : <input type="text" name="id" ><br>  
이름 : <input type="text" name="name"><br>  
<input type="submit" value="환영">  
</form>
```



클라이언트 요청 방식 비교

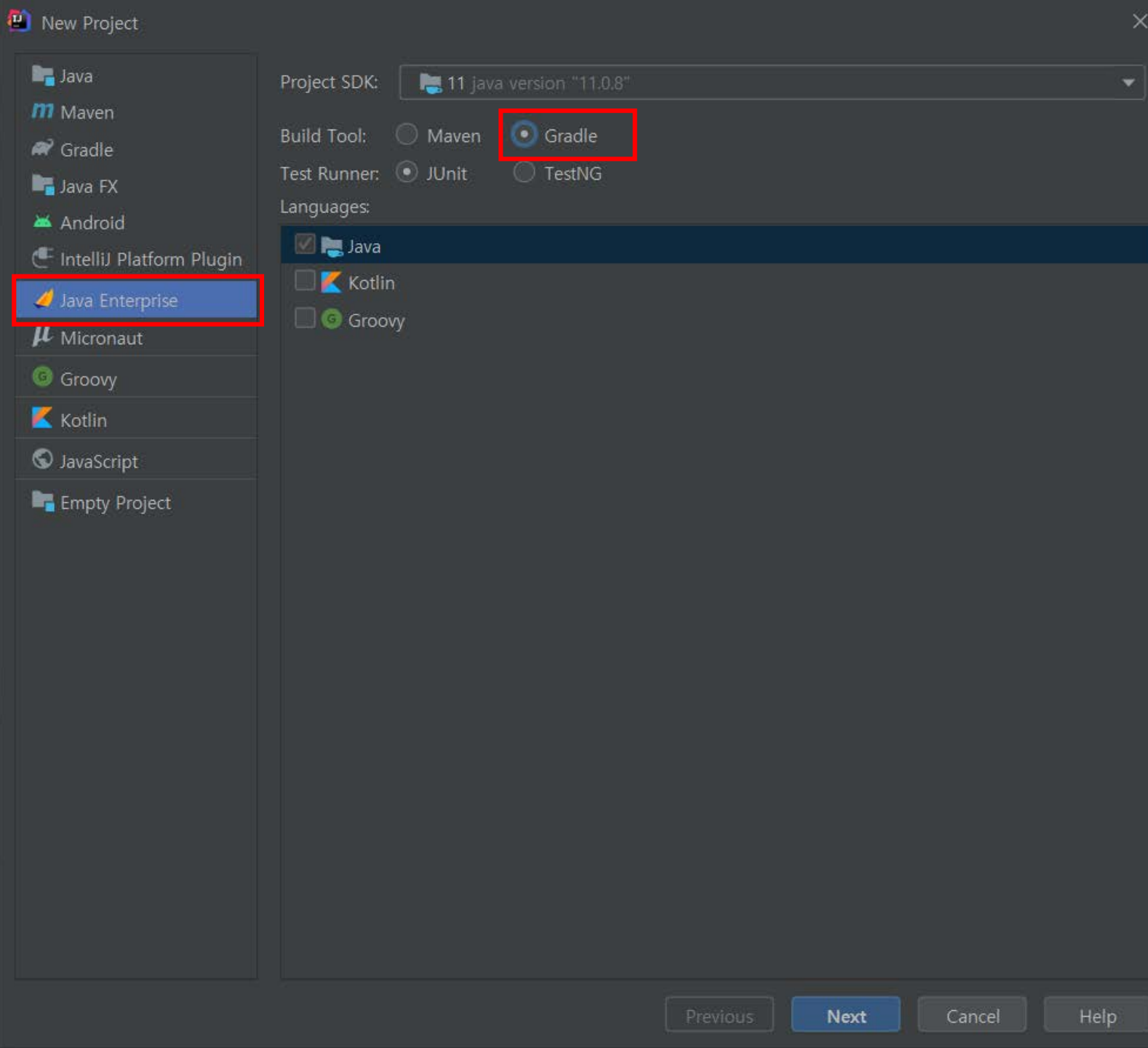
• GET 방식 (기본값)

- 서버에 있는 **정보를 가져오기 위해 설계된 방법**이다. HTML, 이미지 등을 웹 클라이언트에게 가져올 때 사용된다.
- 테스트 과정에서 손쉽게 상호작용할 수 있다.
- 서버로 전달할 수 있는 데이터 크기는 최대 2048 Byte까지이다.
- 서버로 데이터를 전송하는 경우 QUERY_STRING이라는 환경 변수를 통해서 가능하고, URL을 이용한다. 형식은 URI?"속성=값 &속성=값 ... "
- 요청 URL에 값들이 노출되기 때문에 **보안 문제**가 발생할 수 있다.
- 결과페이지를 북마크 또는 즐겨찾기에 저장할 수 있다.
- 웹 클라이언트들이 결과를 캐쉬에 저장할 수 있다.

계속

• POST 방식

- 서버로 **정보를 전달하기 위해 설계된 방법**이다. HTML 폼에 입력한 내용을 서버에 전달할 때 사용된다. 요청이 서버상에 존재하는 데이터들의 수정하는 경우 항상 사용되어야 한다.
- 서버에 전달할 수 있는 데이터 크기에 대한 제한이 없다.
- URL이 간결해진다.
- URL에 전달하려는 정보 값이 표시되지 않기 때문에 상대적으로 높은 보안성을 제공한다.
- 특별한 문자들 또는 업로드 파일 등을 전송할 수 있다.
- 웹 클라이언트들이 결과를 캐쉬에 저장할 수 없다.



Name :
webapp2020

Group :
iducs.jsp<학번>

경로 :
영문 디렉터리
활용

New Project

Name: webapp2020

Location: D:\2019\workspace\webapp2020

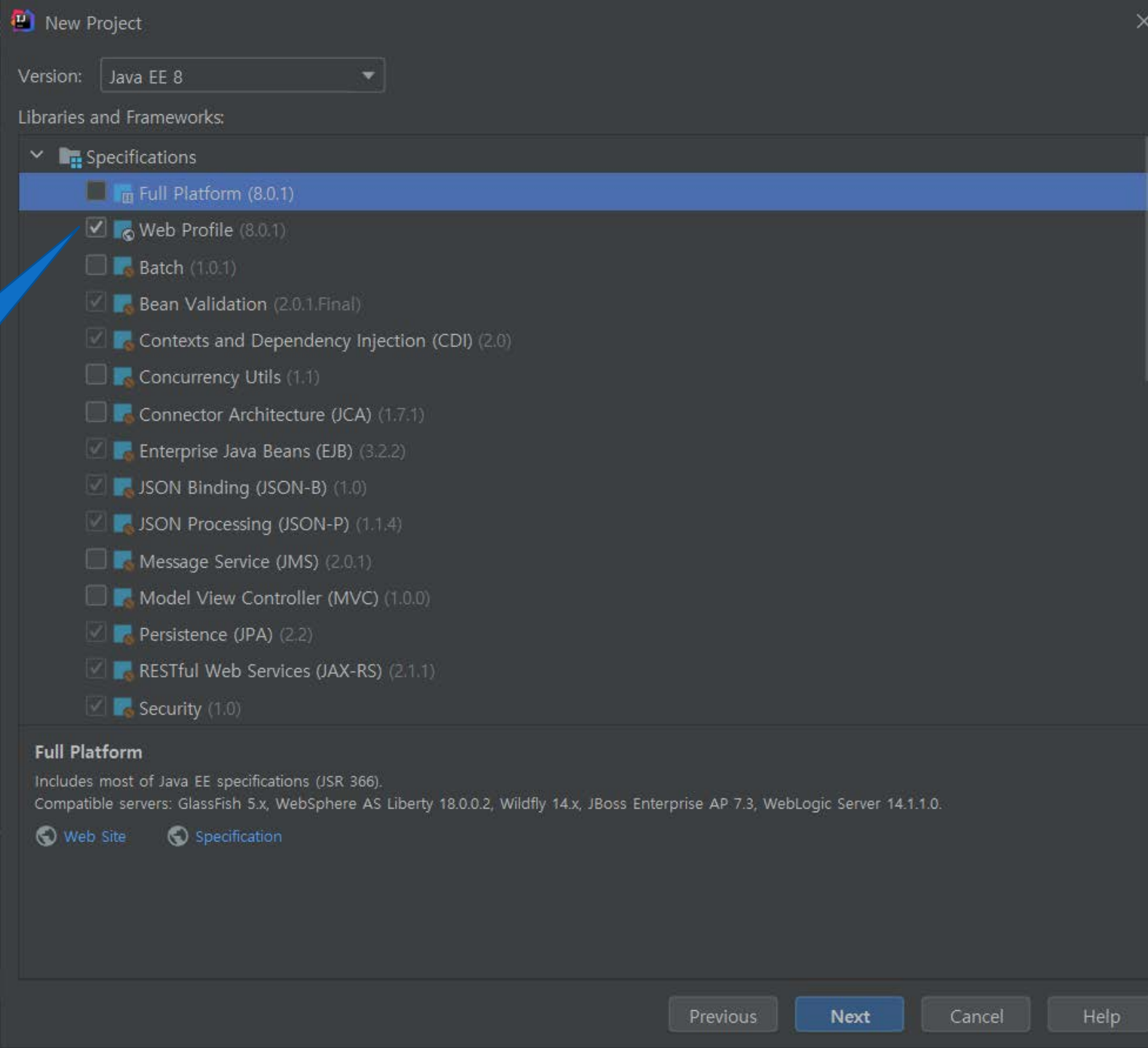
Artifact Coordinates

Group: iducs.jsp202012000
The name of the artifact group, usually a company domain

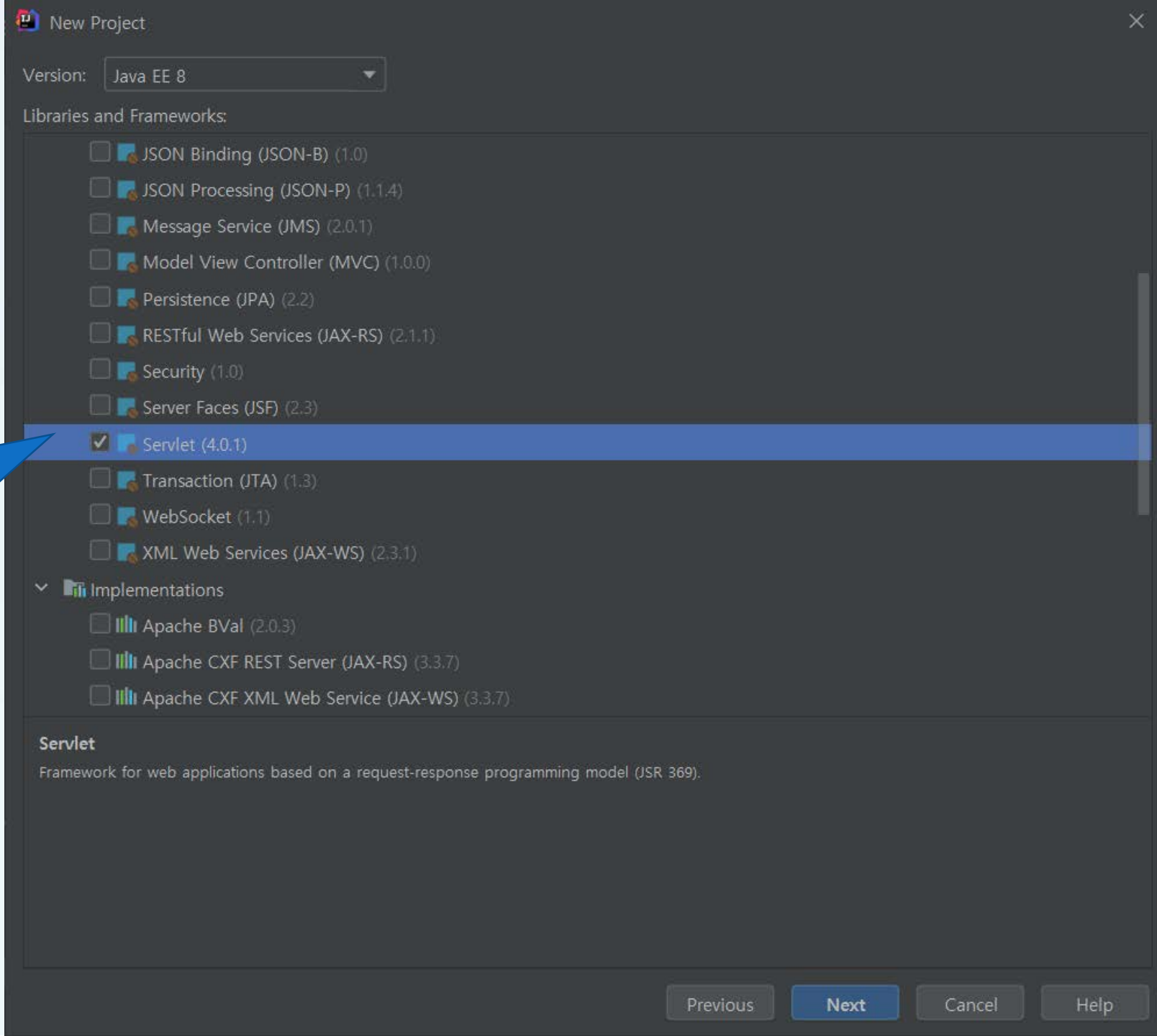
Artifact: webapp2020
The name of the artifact within the group, usually a project name

Version: 1.0-SNAPSHOT

Previous Finish Cancel Help



Library,
Framework 선택 :
Web Profile



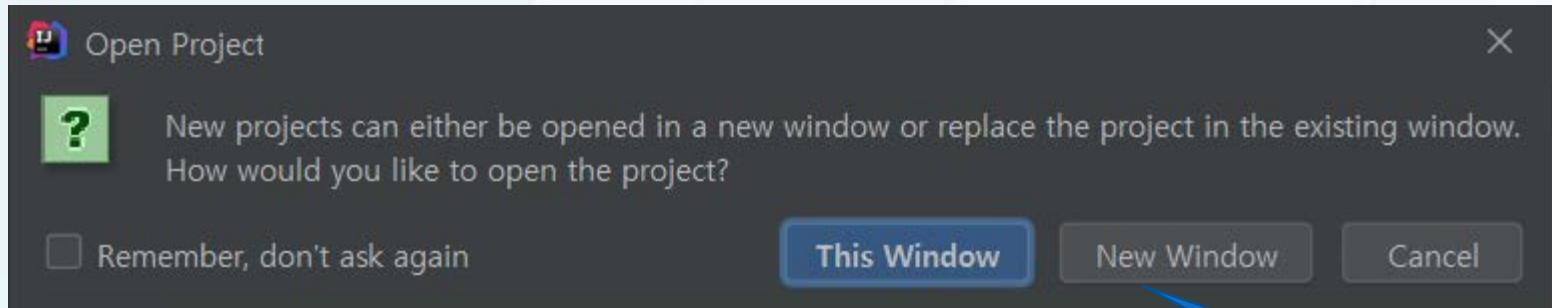
Library,
Framework 선택 :
Servlet 만

```
1 plugins {  
2     id 'java'  
3     id 'war'  
4 }  
5  
6 group 'iducs.jsp202012000'  
7 version '1.0'  
8  
9 repositories {  
10     mavenCentral()  
11 }  
12  
13 ext {  
14     junitVersion = '5.6.2'  
15 }  
16  
17 sourceCompatibility = 1.8  
18 targetCompatibility = 1.8  
19  
20 dependencies {  
21     compileOnly('javax:javaee-web-api:8.0.1')  
22  
23     testImplementation("org.junit.jupiter:junit-jupiter-api:${junitVersion}")  
24     testRuntimeOnly("org.junit.jupiter:junit-jupiter-engine:${junitVersion}")  
25 }  
26  
27 test {  
28     useJUnitPlatform()  
29 }
```

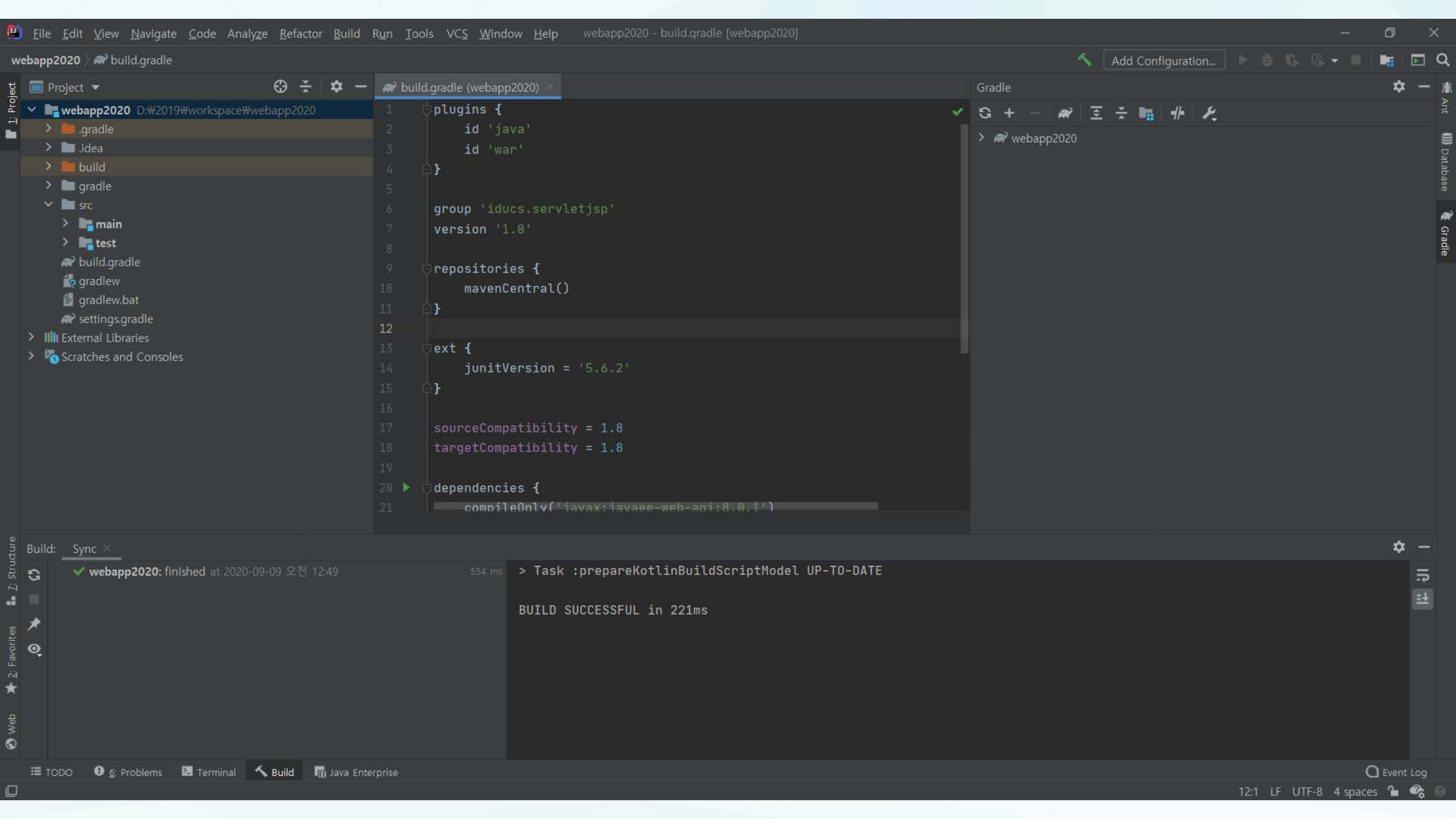
Library,
Framework 선택 :
Web profile

```
1 plugins {  
2     id 'java'  
3     id 'war'  
4 }  
5  
6 group 'com.example'  
7 version '1.0-SNAPSHOT'  
8  
9 repositories {  
10     mavenCentral()  
11 }  
12  
13 ext {  
14     junitVersion = '5.6.2'  
15 }  
16  
17 sourceCompatibility = 1.8  
18 targetCompatibility = 1.8  
19  
20 dependencies {  
21     compileOnly('javax.servlet:javax.servlet-api:4.0.1')  
22  
23     testImplementation("org.junit.jupiter:junit-jupiter-api:${junitVersion}")  
24     testRuntimeOnly("org.junit.jupiter:junit-jupiter-engine:${junitVersion}")  
25 }  
26  
27 test {  
28     useJUnitPlatform()  
29 }
```

Library,
Framework 선택 :
Servlet



작성한 프로젝트를 새로운 윈도우에서 열기



File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help webapp2020 - pring-request-header.jsp [webapp2020.main]

webapp2020 > src > main > webapp

Project ▾

- webapp2020 D:\2019\workspace\webapp2020
 - .gradle
 - .idea
 - build
 - gradle
 - out
 - src
 - main
 - java
 - resources
 - META-INF
 - persistence.xml
 - webapp
 - WEB-INF
 - web.xml

build.gradle (webapp2020) × web.xml × persistence.xml × pring-request-header

31 padding-bottom: 12px;
32 text-align: left;
33 background-color: #4CAF50;
34 color: white;

New

- Cut Ctrl+X
- Copy
- Paste Ctrl+V
- Find Usages Alt+F7
- Find in Path... Ctrl+Shift+F
- Replace in Path... Ctrl+Shift+R
- Analyze
- Refactor
- Add to Favorites
- Delete... Delete
- Build Module 'webapp2020.main'
- Run 'Tests in 'webapp2020...'' Ctrl+Shift+F10
- Debug 'Tests in 'webapp2020...''
- Run 'Tests in 'webapp2020...'' with Coverage
- Run 'Tests in 'webapp2020...'' with 'Java Flight Recorder'

JSP/JSPX

- File
- Scratch File Ctrl+Alt+Shift+Insert
- Directory
- HTML File
- Stylesheet
- JavaScript File
- TypeScript File
- package.json File
- Kotlin Script
- Kotlin Worksheet
- OpenAPI Specification
- EditorConfig File
- Swing UI Designer
- Resource Bundle
- XML Configuration File
- Diagram
- Data Source

spring-request-header.jsp

```

1  <!--
2      Created by IntelliJ IDEA.
3      User: egyou@induk.ac.kr
4      Date: 2020-09-09
5      Time: 오전 12:57
6      To change this template use File | Settings | File Templates.
7  -->
8  <%@ page contentType="text/html;charset=UTF-8" pageEncoding="UTF-8" language="java" %>
9  <%@ page import="java.util.Enumeration" %>
10 <html>
11 <head>
12     <title>요청 헤더 출력</title>
13     <style>
14         #customers {
15             font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
16             border-collapse: collapse;
17             width: 100%;
18         }
19
20         #customers td, #customers th {
21             border: 1px solid #ddd;
22             padding: 8px;
23         }
24
25         #customers tr:nth-child(even){background-color: #f2f2f2;}
26
27         #customers tr:hover {background-color: #ddd;}
28
29         #customers th {
30             padding-top: 12px;
31             padding-bottom: 12px;
32             text-align: left;

```



```

33 ■      background-color: #4CAF50;
34 ■      color: white;
35      }
36  </style>
37  </head>
38  <body>
39  <table id="customers">
40  <%
41      String methodName = request.getMethod();
42      String requestUri = request.getRequestURI();
43      String protocol = request.getProtocol();
44
45      out.print("<tr><th>Request Line </th>");
46      out.print("<th>" + methodName + " " + requestUri + " " + protocol + "</th></tr>");
47      out.print("<tr><td>" + "Request Headers " + "</td>");
48      Enumeration headerNames = request.getHeaderNames();
49      String headerValues = "";
50      while(headerNames.hasMoreElements()) {
51          String paramName = (String) headerNames.nextElement();
52          String paramValue = request.getHeader(paramName);
53          headerValues = headerValues + paramName + " >> " + paramValue + "<br/>";
54      }
55      out.print("<td>" + headerValues + "</td></tr>");
56  <%>
57  </table>
58  </body>
59  </html>

```

요청 헤더 출력	
localhost:8080/Gradle__iducs_ jsp2012000 _webapp2020_1_0_war/pring-request-header.jsp	
Request Line	GET /Gradle__iducs_servletjsp__webapp2020_1_0_war/pring-request-header.jsp HTTP/1.1
Request Headers	<pre> host >> localhost:8080 connection >> keep-alive cache-control >> max-age=0 upgrade-insecure-requests >> 1 user-agent >> Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.83 Safari/537.36 accept >> text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9 sec-fetch-site >> none sec-fetch-mode >> navigate sec-fetch-user >> ?1 sec-fetch-dest >> document accept-encoding >> gzip, deflate, br accept-language >> ko,en-US;q=0.9,en;q=0.8 cookie >> JSESSIONID=658A2AE0E256AB14B55A7A2F022934D3; JSESSIONID=768D81A72A9805B9203CC637260CC246 </pre>

index.jsp

<%--
JSP 주석 시작

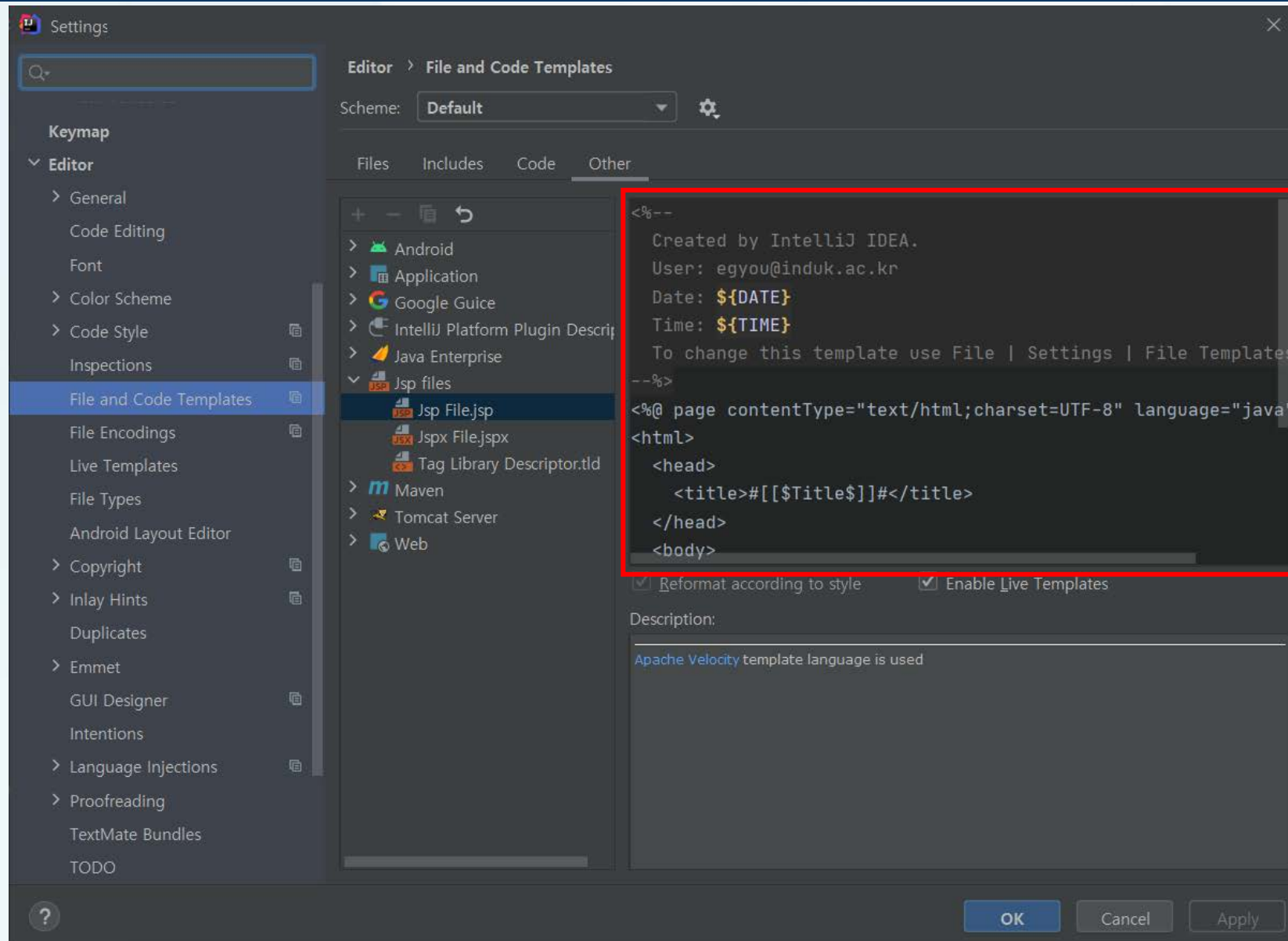
<%=
JSP 구성요소
Expression,
표현식

--%>
JSP 주석 종료

IntelliJ의
File and Code
Templates

```
1  <%--
2      Created by IntelliJ IDEA.
      User: blessyeg
      Date: 2020-09-08
      Time: 오후 8:04
      To change this template use File | Settings | File Templates.
3  --%>
4  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
5
6  9  <html>
7     <head>
8         <title>Title</title>
9     </head>
10    <body>
11        <%= "Hello servlet jsp" %>
12    </body>
13 </html>
```

Editor - File and Code Templates - Other - Jsp files



<%
JSP 태그의 시작
<%@
지시자 태그의
시작

<%@ page
page 지시자의
시작

contentType
page 지시자의
속성, 생성할 파
일에 대한 설정

생성할 파일의
유형은 text/html
문자셋은 UTF-8

**<%@ page contentType="text/html;charset=UTF-8"
pageEncoding="UTF-8" language="java" %> 의미**

pageEncoding
JSP를 읽을 때 문
자셋을 지정,
UTF-8

language
JSP에서 사용할
스크립트 언어
지정

%>
JSP 태그의 끝

JSP 구성 요소

▪ Directive, 지시자

- 정의

- JSP 페이지의 속성을 지정하는데 사용하는 태그
- 웹 컨테이너가 처리 방법을 결정하는데 필요한 정보를 제공

- 종류

- `<%@ page ~ %>` : 페이지 관련 다양한 프로퍼티를 지정한다.
- `<%@ include ~ %>` : 변환 시점에 현재 페이지에 포함할 코드나 문서를 지정한다.
- `<%@ taglib ~ %>` : JSP에서 이용 가능한 **태그 라이브러리**를 지정한다.

IntelliJ 단축키

검색 - Actions	Ctrl + Shift + a
Settings 창 띄우기	Ctrl + Alt + s
다시 컴파일	Ctrl + Shift + F9
프로젝트 빌드 (컴파일, 링킹)	Ctrl + F9
(설정 후) 실행	Alt + Shift + F10
(대상) 실행	Shift + F10
실행 중지	Ctrl + F2
Gradle 변화 로딩	Ctrl + Shift + o

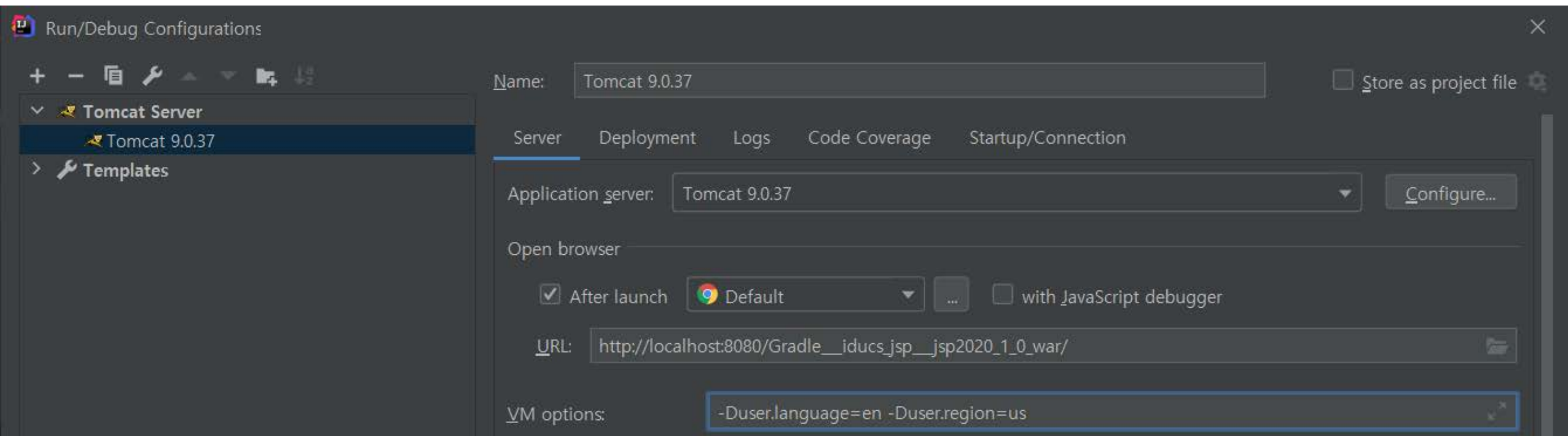
tomcat log window 한글 문제

■ 증상

- 로그 메시지의 한글이 정상적으로 표현되지 않음

■ 해결 방법 1

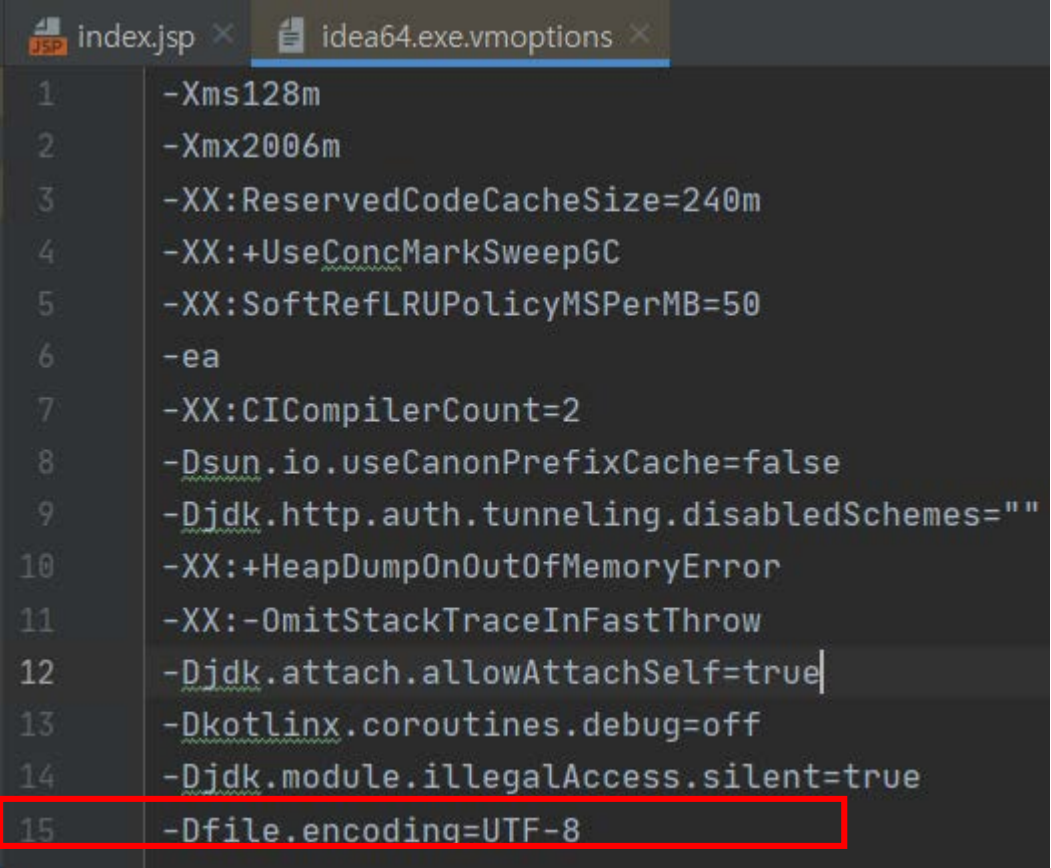
- 로그를 영어로 표기함
 - Run/Debug Configurations의 'VM options' 항목의 값을 '-Duser.language=en -Duser.region=us'으로 지정



계속

■ 해결 방법2(추천)

- Actions 검색 (Ctrl + Shift + a)하여 'Edit VM Options' 항목 선택
 - '-Dfile.encoding=UTF-8' 추가 (실제 idea64.exe.vmoptions 파일에 저장됨)



```
1 -Xms128m
2 -Xmx2006m
3 -XX:ReservedCodeCacheSize=240m
4 -XX:+UseConcMarkSweepGC
5 -XX:SoftRefLRUPolicyMSPerMB=50
6 -ea
7 -XX:CICompilerCount=2
8 -Dsun.io.useCanonPrefixCache=false
9 -Djdk.http.auth.tunneling.disabledSchemes=""
10 -XX:+HeapDumpOnOutOfMemoryError
11 -XX:-OmitStackTraceInFastThrow
12 -Djdk.attach.allowAttachSelf=true
13 -Dkotlinx.coroutines.debug=off
14 -Djdk.module.illegalAccess.silent=true
15 -Dfile.encoding=UTF-8
```

JSP 페이지에서 한글 처리

- 응답(클라이언트에게 보낼 때)
 - <%@ page language="java"
contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
- 요청(클라이언트의 요청을 읽을 때)
 - <% request.setCharacterEncoding("utf-8") %>

계속

▪ get 방식의 경우

- Tomcat 9.x 이전, Servlet 4.x 이전 추가 설정이 필요함
 - <톰캣설치경로>/conf/server.xml 편집
 - <Connector port="8080"protocol="HTTP/1.1"
connectionTimeout="20000"
redirectPort="8443"
URIEncoding="utf-8" />

▪ post 방식의 경우

- 별다른 설정 없이 사용 가능

계속

■서블릿에서 한글 처리

- 응답(클라이언트에게 보낼 때)
 - `response.setContentType("text/html; charset=UTF-8");`
- 요청(클라이언트의 요청을 읽을 때)
 - `request.setCharacterEncoding("utf-8");`

학습 후 기대효과

- HTTP의 특징, 발전, 단계별 동작에 대하여 설명할 수 있다.
- HTTP 요청 및 응답과 HTTP 상태 코드, 콘텐츠 타입 등에 대하여 설명할 수 있다.
- HTML의 구조와 <form>, <div>, <table> 등의 주요 태그에 대하여 설명할 수 있다.
- 클라이언트 요청 방식인 GET과 POST에 대하여 설명할 수 있다.
- 한글 처리에 대하여 설명할 수 있다.