스프링으로 프로가 되자!

스프링 프레임워크는 내 손에 [스프1탄]

스프링 기초부터 보안까지 한번에 MASTER

TPC

생각하고(Thinking)-〉 표현하고(Presentation)-〉코딩(Coding)하고

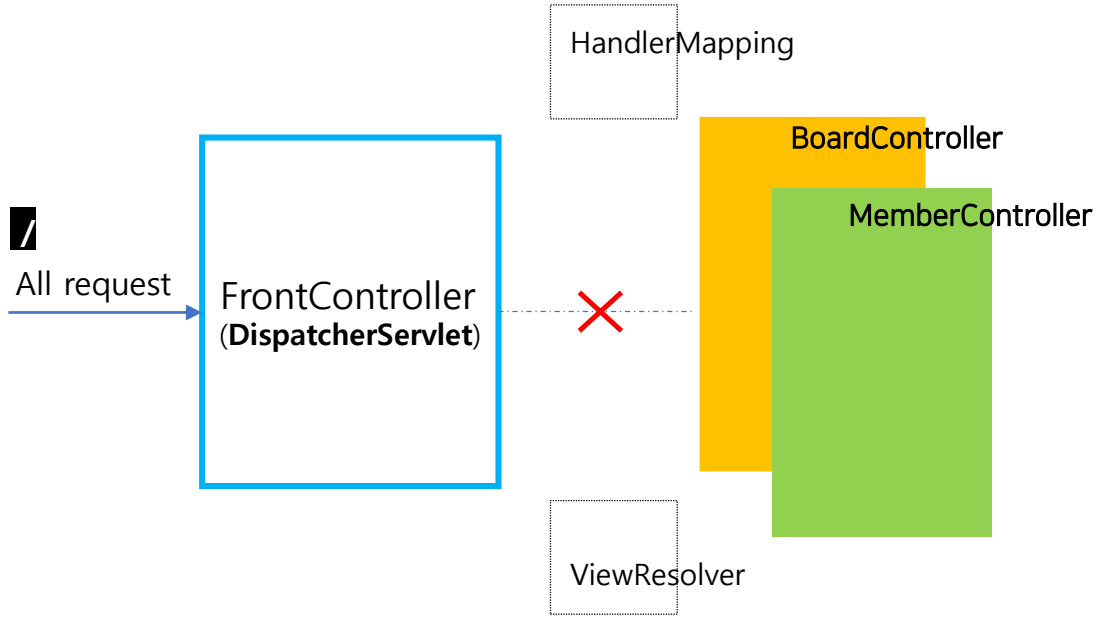
실습 : Spring MVC01

FrontController(DispatcherServlet) + Controller(POJO) loading

WEB Layer

web.xml

```
<servlet>
  <servlet-name>appServlet</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```



servlet-context.xml

```
<context:conponent-scan base-package="kr.board.controller"/>
```

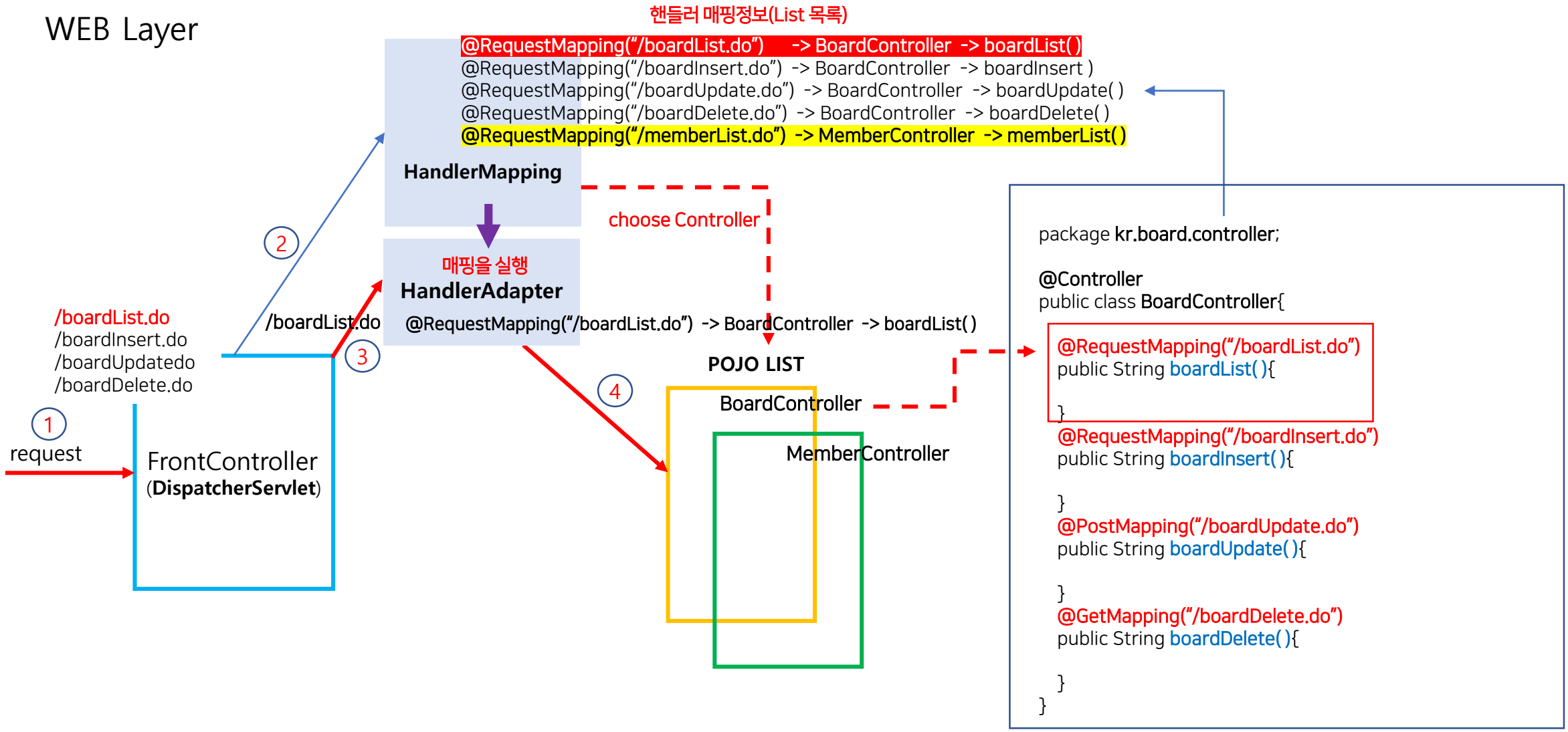
scan

```
package kr.board.controller;

@Controller
public class BoardController{



}

@Controller
public class MemberController{



}
```

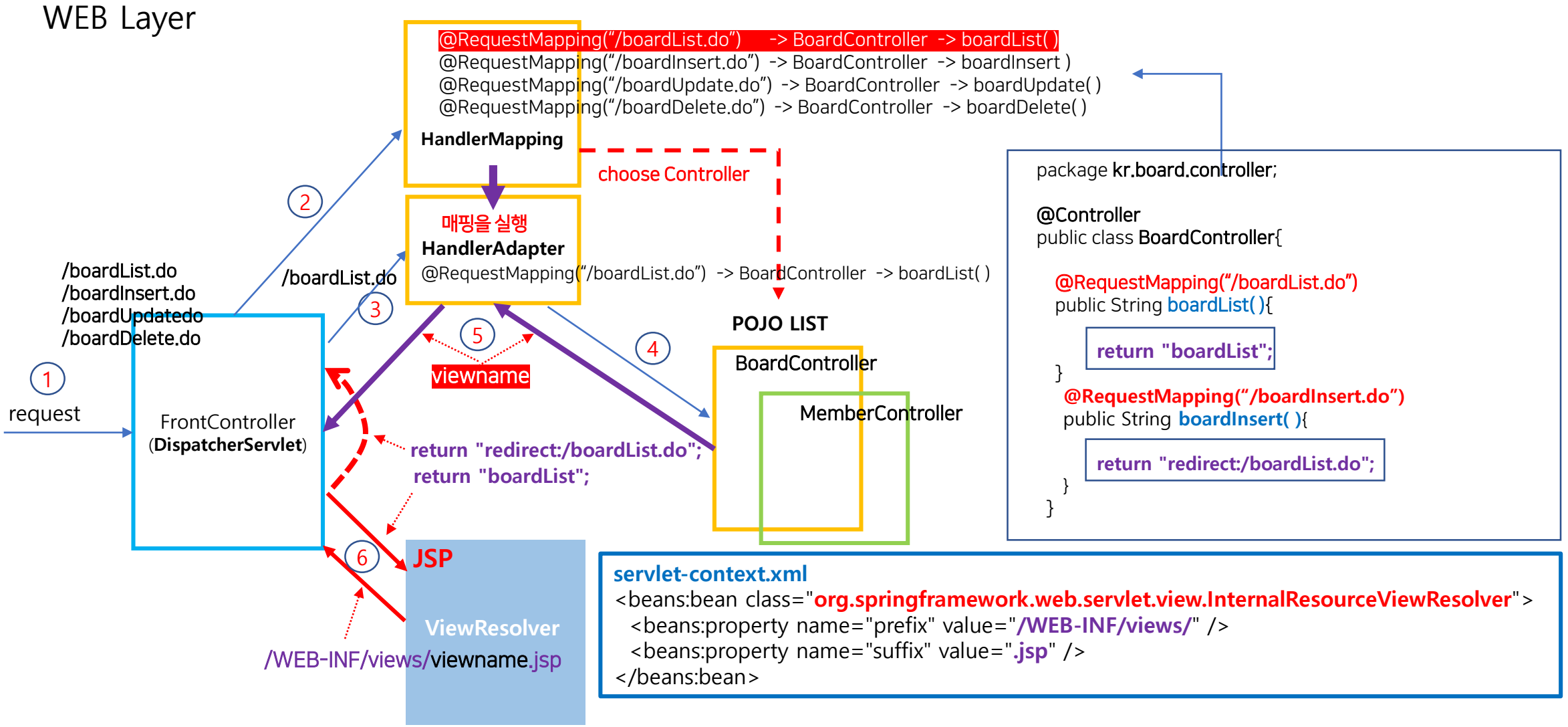
- 객체생성 -

```
BoardController baordController=new BoardController();
MemberController baordController=new MemberController();
```

HandlerMapping, HandlerAdapter Loading (@RequestMapping, @GetMapping, @PostMapping)

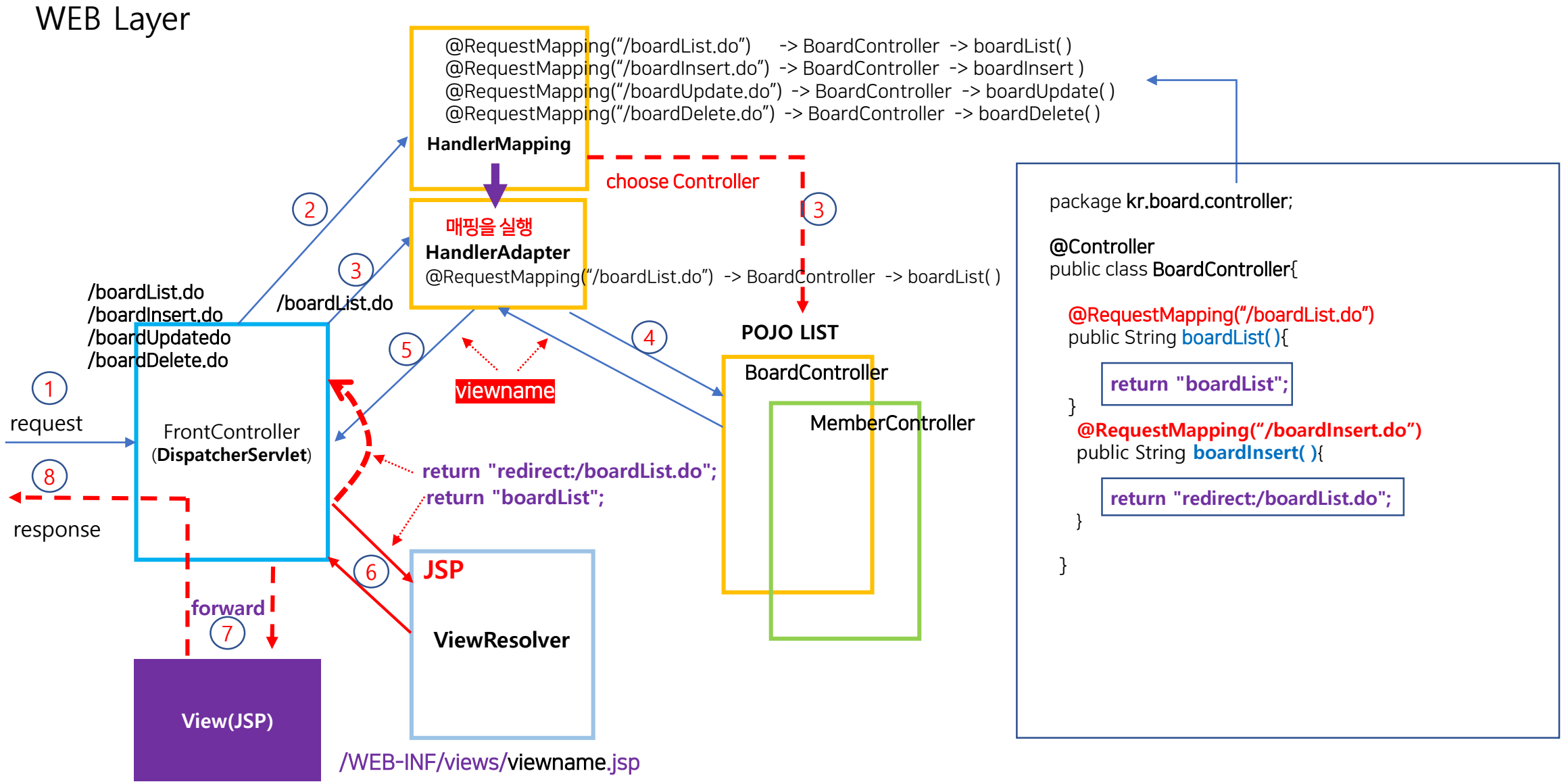


ViewResolver Loading (viewname->/WEB-INF/views/viewname.jsp)



실습 : SpringMVC01

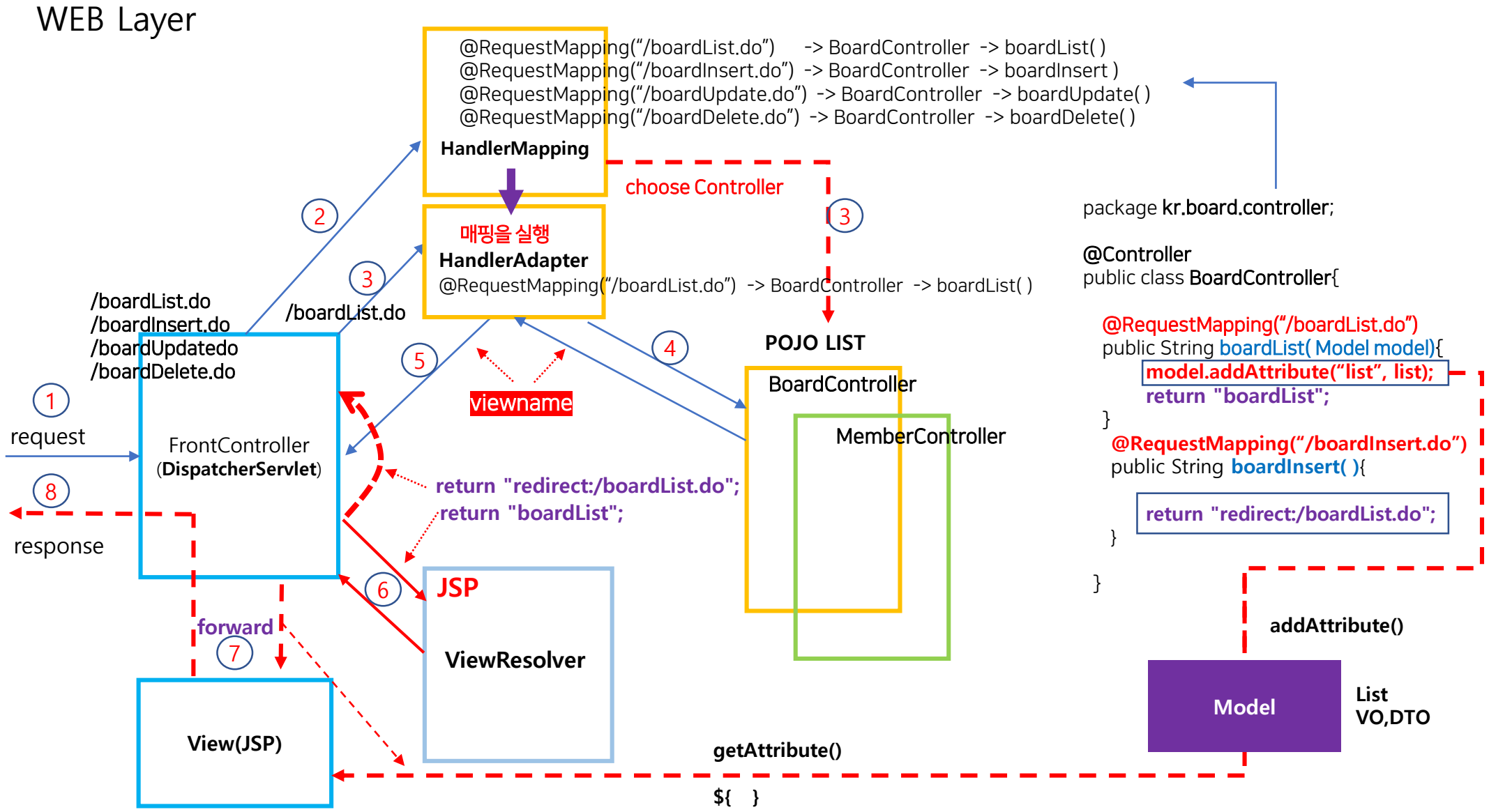
View page forward (viewname->/WEB-INF/views/viewname.jsp)



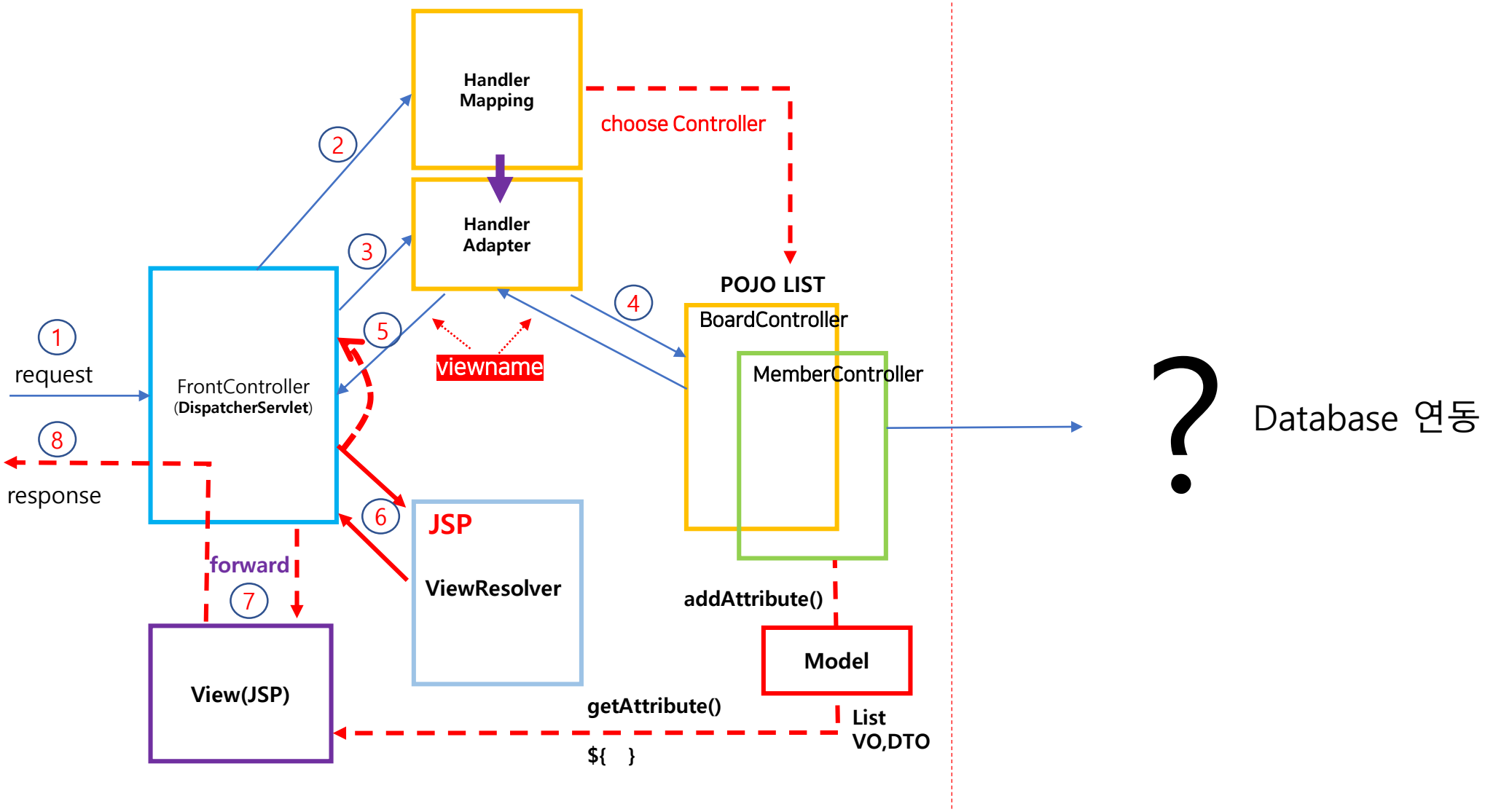

```
package kr.board.controller;

@Controller
public class BoardController{

   @RequestMapping("/boardList.do")
   public String boardList( ){

        return "boardList";
   }
    @RequestMapping("/boardInsert.do")
    public String boardInsert( ){

        return "redirect:/boardList.do";
    }

}
```

실습 : SpringMVC01

객체바인딩(Model, HttpServletRequest, HttpSession)



WEB Layer + Persistence Layer



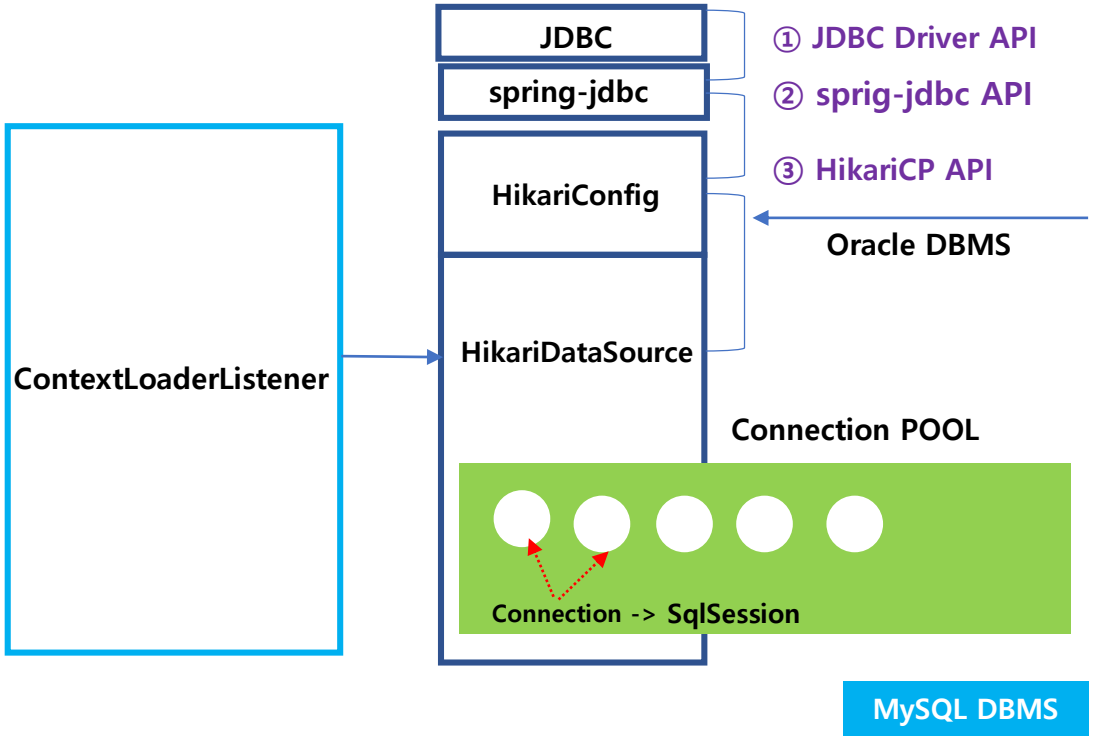
실습 : SpringMVC01

Persistence Layer(Connection POOL 만들기 = DB연동)

Persistence Layer

web.xml

```
<!-- The definition of the Root Spring Container shared by all Servlets and Filters -->
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>/WEB-INF/spring/root-context.xml</param-value>
</context-param>

<!-- Creates the Spring Container shared by all Servlets and Filters -->
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```



root-context.xml

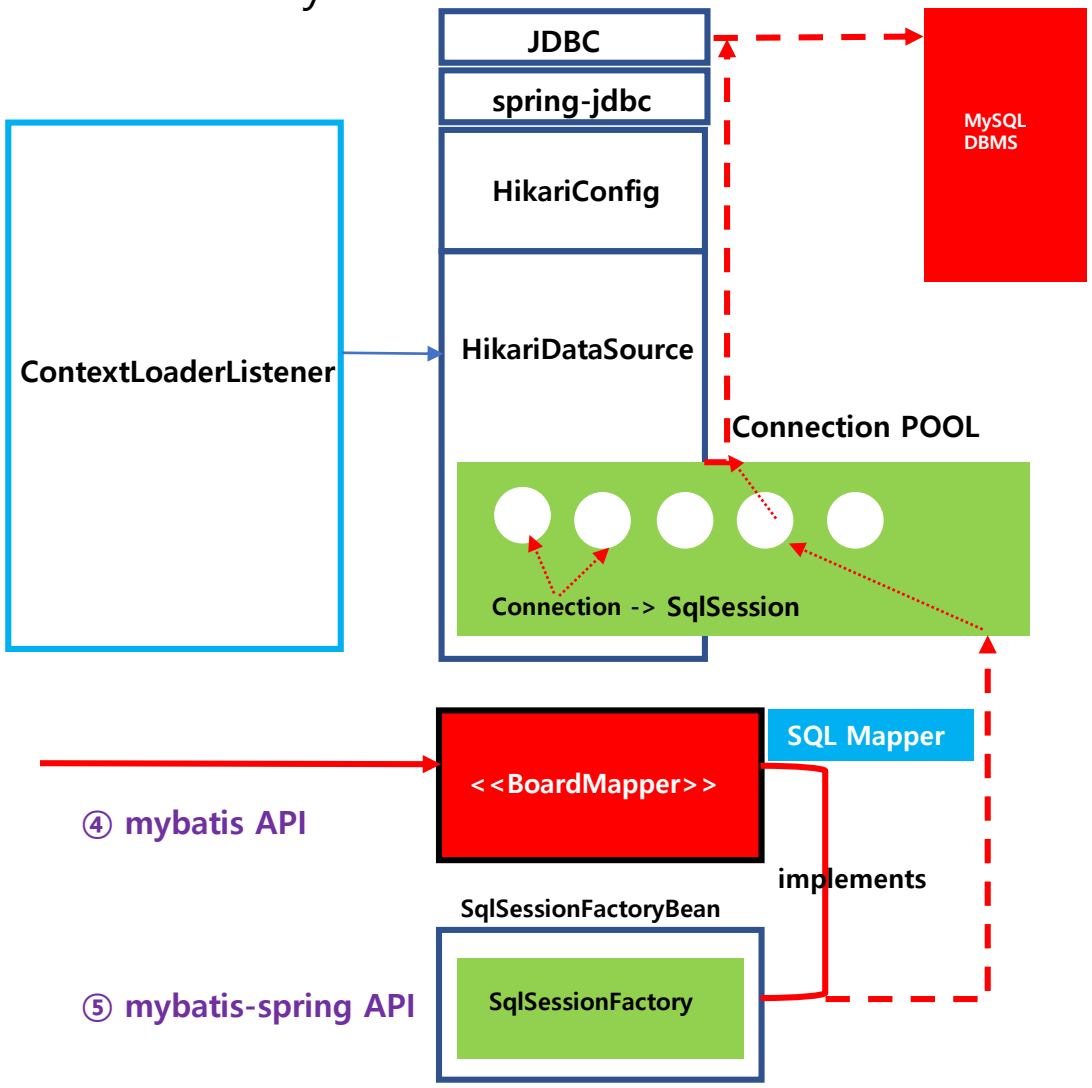
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
    xsi:schemaLocation="http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/mybatis-spring-1.2.xsd http://www.springframework.org/schema/beans https://www.springframework.org/schema/beans/spring-beans.xsd">

<!-- Root Context: defines shared resources visible to all other web components -->
    <!-- API(HikariCP) -->
    <!-- bean : 객체를 생성하는 태그 -->
    <bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
        <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"/>
        <property name="jdbcUrl" value="jdbc:oracle:thin:@127.0.0.1:1521:XE"/>
        <property name="username" value="hr"/>
        <property name="password" value="hr"/>
    </bean>
  <!-- HikariDataSource(Connection POOL을 만드는 역할을 한다) -->
  <bean id="dataSource" class="com.zaxxer.hikari.HikariDataSource" destroy-method="close">
        <constructor-arg ref="hikariConfig" />
  </bean>

</beans>
```

```
    <bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
        <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
        <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/com?serverTimezone=UTC"/>
        <property name="username" value="com"/>
        <property name="password" value="com01"/>
    </bean>
```

실습 : SpringMVC01

Persistence Layer(Mapper loading, MyBatis Framework와 연결)

Persistence Layer



BoardMapper.java

```
package kr.board.mapper;
@Mapper
public interface BoardMapper

}
```

BoardMapper.xml

```
<mapper
namespace="kr.board.mapper.BoardMapper">
```

scan

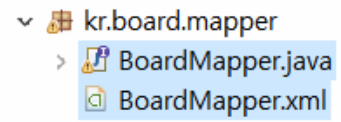
root-context.xml

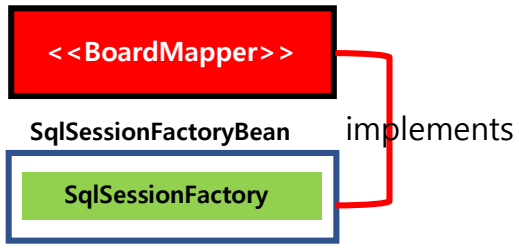
```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
    xsi:schemaLocation="http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/
mybatis-spring-1.2.xsd http://www.springframework.org/schema/beans https://www.springframe
work.org/schema/beans/spring-beans.xsd">


<!-- Mapper interface들을 메모리에 올리는 방법(scan) -->
 <mybatis-spring:scan base-package="kr.board.mapper"/>

<!-- BoardMapper interface의 구현클래스 생성
    SqlSessionFactoryBean(SQL을 실행하는 API) -->
 <bean class="org.mybatis.spring.SqlSessionFactoryBean">
     <property name="dataSource" ref="dataSource" />
 </bean>

</beans>
```

Mapper Interface와 Mapper file(SQL) 연결

BoardMapper.java

```
package kr.board.mapper;
@Mapper
public interface BoardMapper {
    public List<Board> getLists(); //추상메서드
    public void boardInsert(Board vo);
    public Board boardContent(int idx);

    @Delete("delete from board where idx=#{idx}")
    public void boardDelete(int idx); // delete SQL
    public void boardUpdate(Board vo);// update SQL

}
```





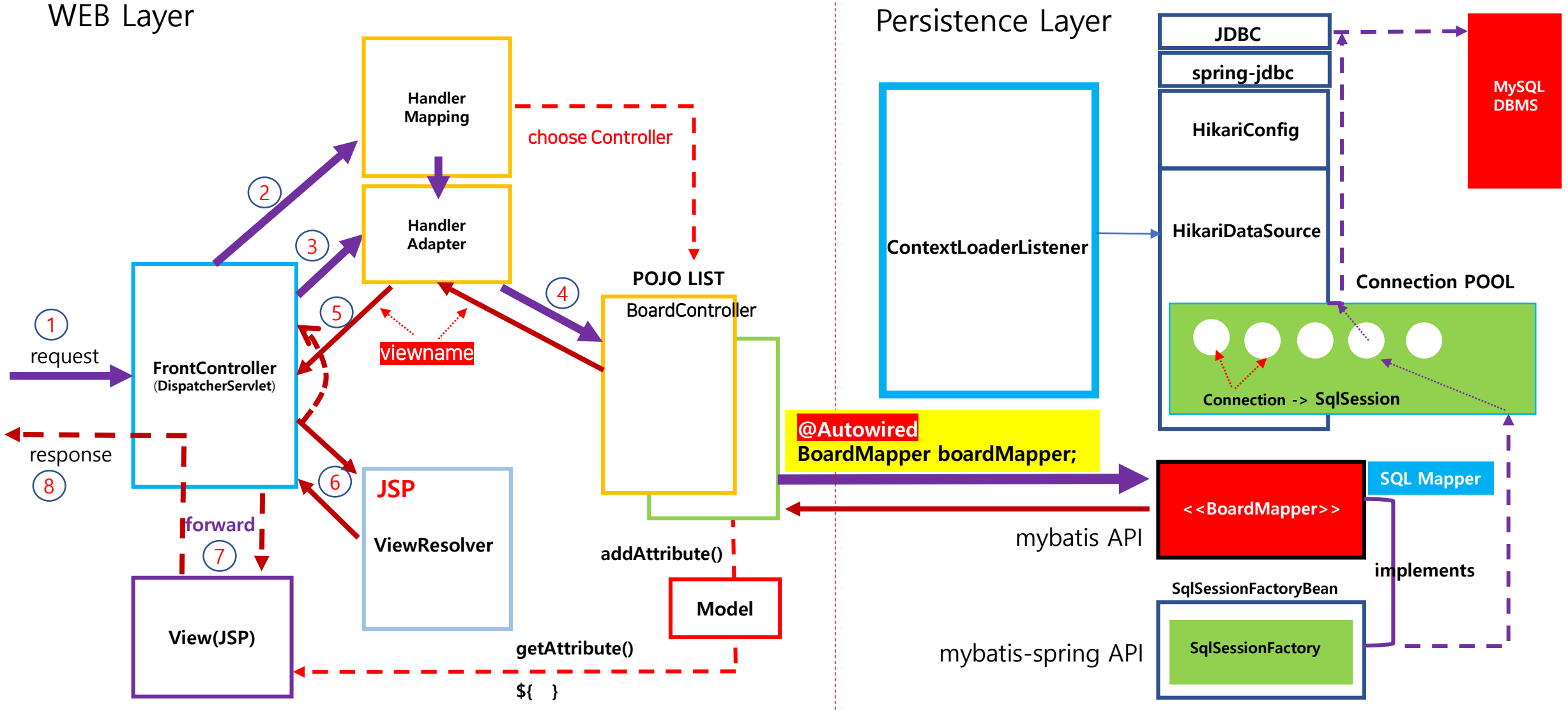

```
public class SqlSessionFactory implements BoardMapper {

    public List<Board> getLists(){

      // 게시판 전체리스트 가져오는 코드가 자동으로 만들어진다.

    }
}
```

BoardMapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="kr.board.mapper.BoardMapper">

    <select id="getLists" resultType="kr.board.entity.Board">
        select * from board order by idx desc
    </select>
    <insert id="boardInsert" parameterType="kr.board.entity.Board">
        insert into board(idx, title, content, writer)
        values(board_idx.nextval,#{title},#{content},#{writer})
    </insert>

    <select id="boardContent"
         resultType="kr.board.entity.Board"
         parameterType="int">
       select * from board where idx=#{idx}
    </select>

    <update id="boardUpdate" parameterType="kr.board.entity.Board">
        update board set title=#{title}, content=#{content}
        where idx=#{idx}
    </update>
</mapper>
```

실습 : SpringMVC01

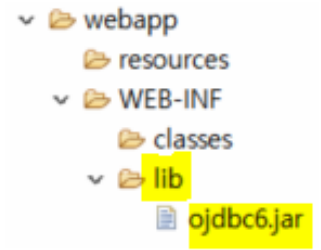
WEB Layer + Persistence Layer 연결



API 추가(pom.xml)

```
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>javax.servlet-api</artifactId>
  <version>3.1.0</version>
  <scope>provided</scope>
</dependency>

<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>javax.servlet.jsp-api</artifactId>
  <version>2.3.1</version>
  <scope>provided</scope>
</dependency>
```

Oracle jdbc driver



MySQL jdbc driver

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.42</version>
</dependency>
```

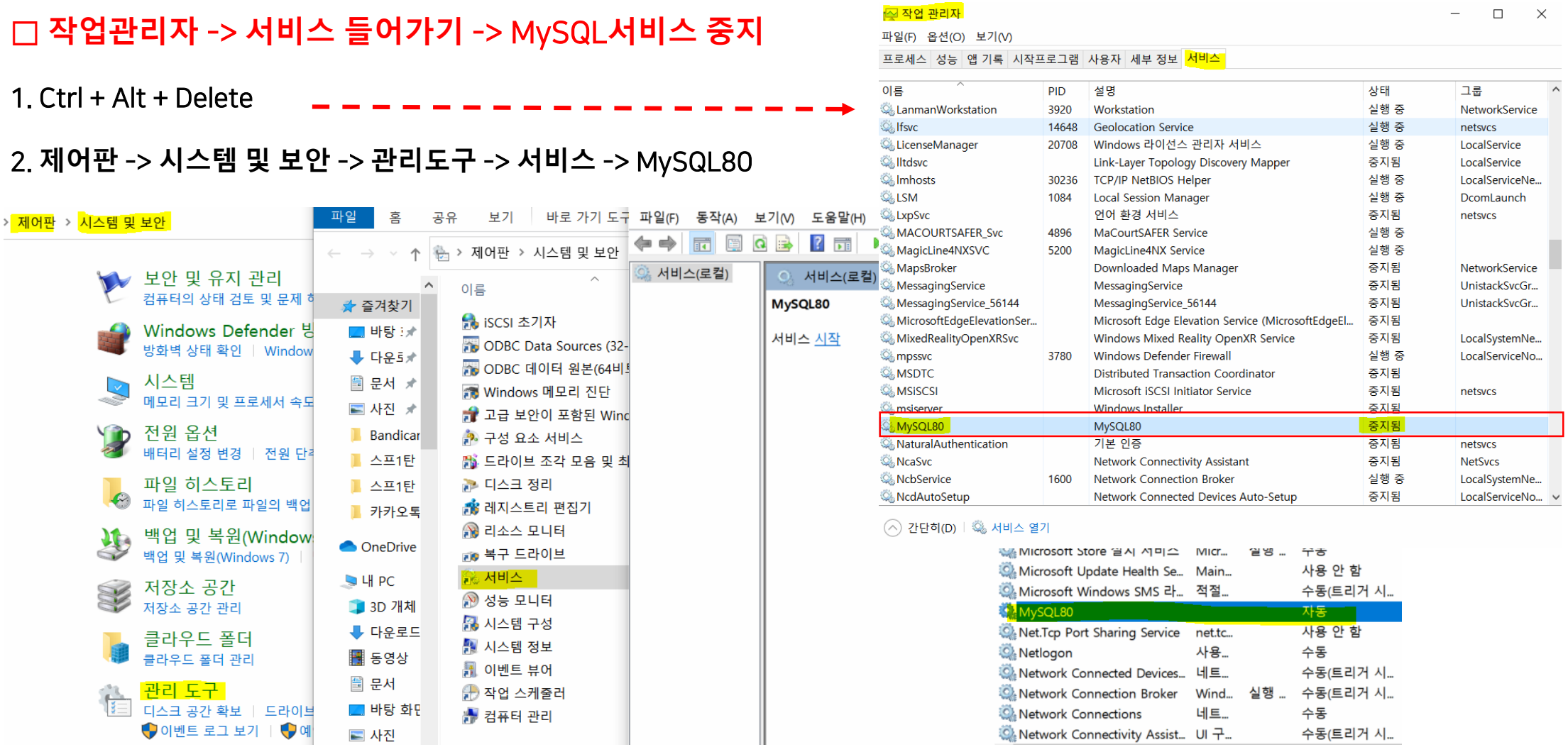
```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.16</version>
</dependency>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>5.0.2.RELEASE</version>
</dependency>
<dependency>
    <groupId>com.zaxxer</groupId>
    <artifactId>HikariCP</artifactId>
    <version>3.4.1</version>
</dependency>
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis</artifactId>
  <version>3.4.6</version>
  </dependency>
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis-spring</artifactId>
  <version>1.3.2</version>
</dependency>
```

MySQL DB 접속방법 - 1

□ 작업관리자 -> 서비스 들어가기 -> MySQL서비스 중지

1. Ctrl + Alt + Delete

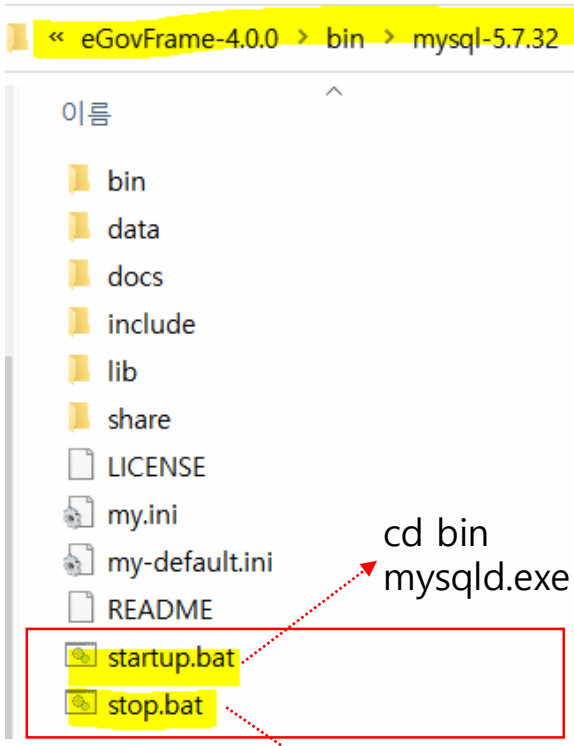
2. 제어판 -> 시스템 및 보안 -> 관리도구 -> 서비스 -> MySQL80



MySQL DB 접속방법 - 2

☐ MySQL server start / stop

C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32
디렉토리에서 startup.bat(실행), stop.bat(중지) 할 수 있다.



cd bin
mysqld.exe --console

cd bin
mysqladmin -uroot shutdown

```
C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32>cd bin

C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32₩bin>mysqld.exe --console
mysqld: Could not create or access the registry key needed for the MySQL applicat
to log to the Windows EventLog. Run the application with sufficient
privileges once to create the key, add the key manually, or turn off
logging for that application.
2022-05-22T09:37:48.641357Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is
tion (see documentation for more details).
2022-05-22T09:37:48.641582Z 0 [Note] --secure-file-priv is set to NULL. Operations
2022-05-22T09:37:48.645474Z 0 [ERROR] Cannot open Windows EventLog; check privile
2022-05-22T09:37:48.645520Z 0 [Note] mysqld.exe (mysqld 5.7.32-log) starting as pi
2022-05-22T09:37:48.682128Z 0 [Note] InnoDB: Mutexes and rw_locks use Windows inte
2022-05-22T09:37:48.682394Z 0 [Note] InnoDB: Uses event mutexes
2022-05-22T09:37:48.683154Z 0 [Note] InnoDB: _mm_lfence() and _mm_sfence() are use
2022-05-22T09:37:48.683428Z 0 [Note] InnoDB: Compressed tables use zlib 1.2.11
2022-05-22T09:37:48.684439Z 0 [Note] InnoDB: Number of pools: 1
2022-05-22T09:37:48.685060Z 0 [Note] InnoDB: Not using CPU crc32 instructions
2022-05-22T09:37:48.693444Z 0 [Note] InnoDB: Initializing buffer pool, total size
2022-05-22T09:37:48.705364Z 0 [Note] InnoDB: Completed initialization of buffer po
2022-05-22T09:37:48.778162Z 0 [Note] InnoDB: Highest supported file format is Barr
2022-05-22T09:37:50.230812Z 0 [Note] InnoDB: Creating shared tablespace for tempor
2022-05-22T09:37:50.232185Z 0 [Note] InnoDB: Setting file '.₩ibtmp1' size to 12 MB
2022-05-22T09:37:50.316854Z 0 [Note] InnoDB: File '.₩ibtmp1' size is now 12 MB.
2022-05-22T09:37:50.341067Z 0 [Note] InnoDB: 96 redo rollback segment(s) found. 96
2022-05-22T09:37:50.341586Z 0 [Note] InnoDB: 32 non-redo rollback segment(s) are a
2022-05-22T09:37:50.345136Z 0 [Note] InnoDB: Waiting for purge to start
2022-05-22T09:37:50.411556Z 0 [Note] InnoDB: 5.7.32 started; log sequence number
2022-05-22T09:37:50.412917Z 0 [Note] Plugin 'FEDERATED' is disabled.
2022-05-22T09:37:50.414980Z 0 [Note] InnoDB: Loading buffer pool(s) from C:₩eGovFr
2022-05-22T09:37:50.468823Z 0 [Note] Found ca.pem, server-cert.pem and server-key
2022-05-22T09:37:50.469585Z 0 [Note] Skipping generation of SSL certificates as ce
2022-05-22T09:37:50.478484Z 0 [Warning] CA certificate ca.pem is self signed.
2022-05-22T09:37:50.479384Z 0 [Note] Skipping generation of RSA key pair as key f
2022-05-22T09:37:50.484382Z 0 [Note] Server hostname (bind-address): '*'; port: 33
2022-05-22T09:37:50.487287Z 0 [Note] IPv6 is available.
2022-05-22T09:37:50.487675Z 0 [Note]   - '::' resolves to '::';
2022-05-22T09:37:50.490030Z 0 [Note] Server socket created on IP: '::'.
2022-05-22T09:37:50.507262Z 0 [Note] InnoDB: Buffer pool(s) load completed at 2205
2022-05-22T09:37:50.591758Z 0 [Note] Failed to start slave threads for channel ''
2022-05-22T09:37:50.698025Z 0 [Note] Event Scheduler: Loaded 0 events
2022-05-22T09:37:50.698805Z 0 [Note] mysqld.exe: ready for connections.
Version: '5.7.32-log'  socket: ''  port: 3306  MySQL Community Server (GPL)
```

MySQL DB 접속방법(cmd) - 3

☐ MySQL 서버구동 후 MySQL 접속하기(mysql참고.txt)

mysql-5.7.32
* mysql ROOT 계정
root /

심플홈페이지 템플릿
db : sht
id : sht
pw : sht01

엔터프라이즈 비지니스 템플릿
db : ebt
id : ebt
pw : ebt01

포털사이트 템플릿
db : pst
id : pst
pw : pst01

실습용 공통컴포넌트
db : com
id : com
pw : com01

공통컴포넌트 all-in-one
db : comall
id : com
pw : com01

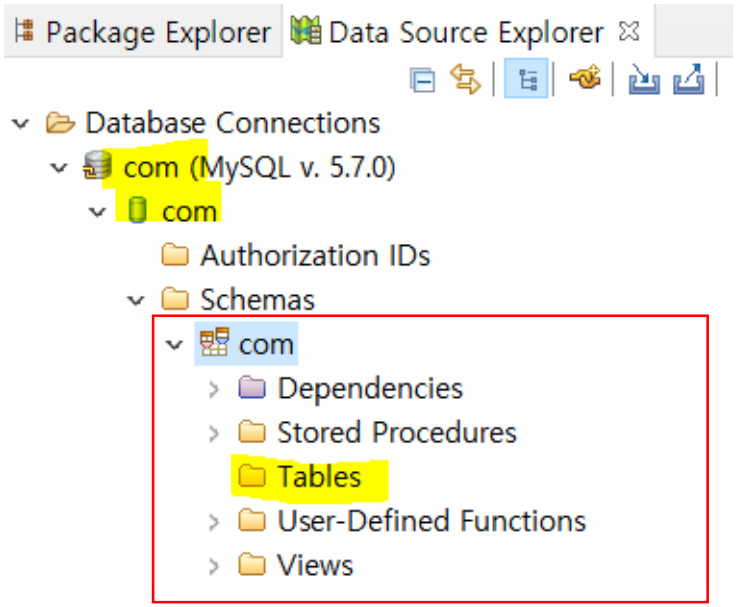
모바일
db : mobile
id : mobile
pw : mobile01

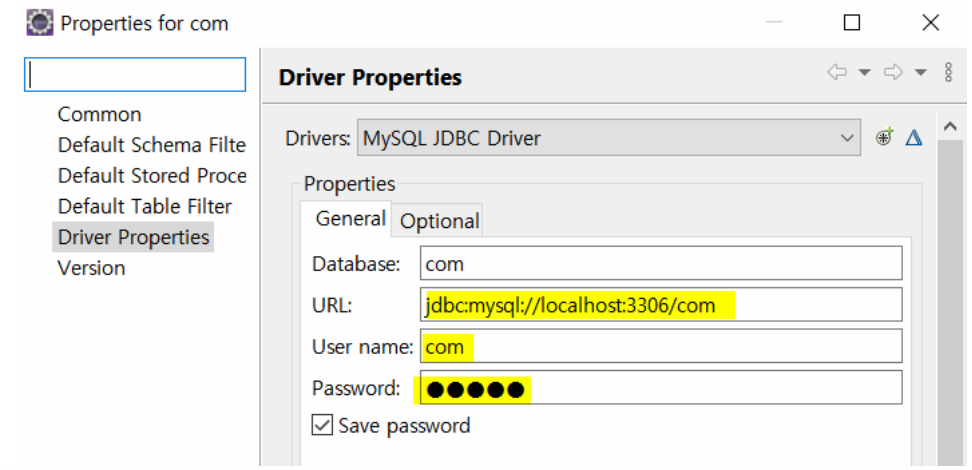
디바이스API
db : hyb
id : hyb
pw : hyb01

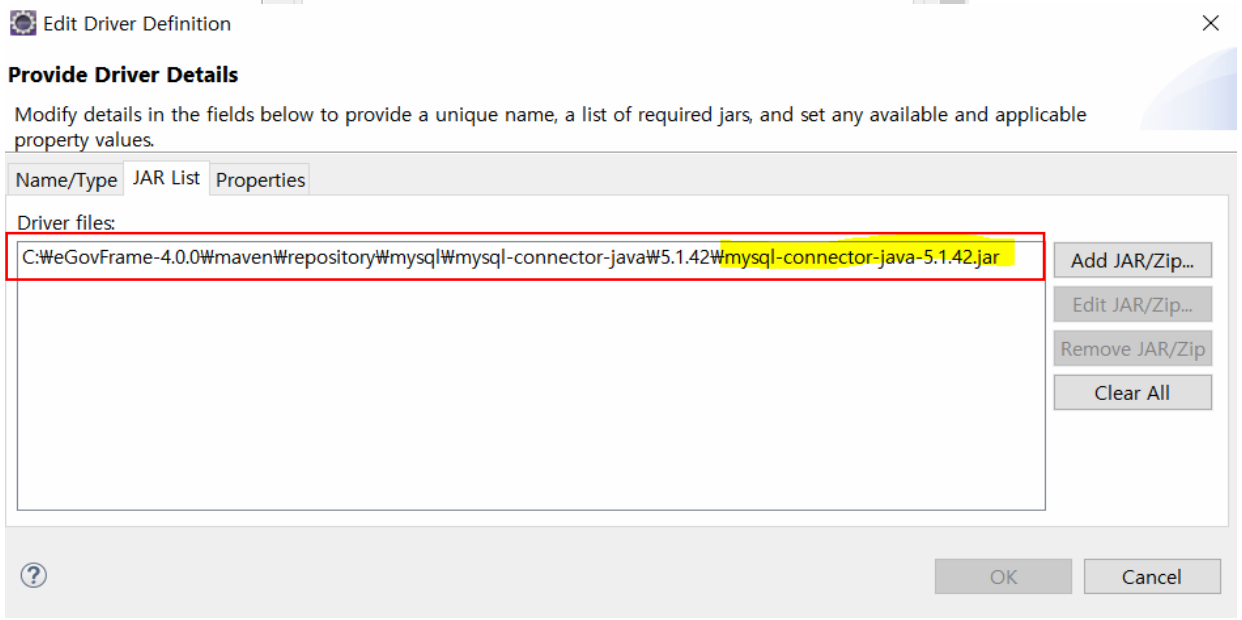
```
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:₩Users₩GSM>cd C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32₩bin

C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32₩bin>
C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32₩bin>mysql -u com -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or ₩g.
Your MySQL connection id is 9
Server version: 5.7.32-log MySQL Community Server (GPL)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '₩h' for help. Type '₩c' to clear the current input statement.

mysql>
```

MySQL DB 접속방법(eclipse) - 4

☐ MySQL 서버구동 후 MySQL 접속하기







MySQL DB table 만들기 - 5

MySQL

```
create table myboard(
  idx int not null auto_increment,
  title varchar(100) not null,
  content varchar(2000) not null,
  writer varchar(30) not null,
  indate datetime default now(),
  count int default 0,
  primary key(idx)
);

insert into myboard(title,content,writer)
values('게시판 연습','게시판 연습','관리자');
insert into myboard(title,content,writer)
values('게시판 연습','게시판 연습','박매일');
insert into myboard(title,content,writer)
values('게시판 연습','게시판 연습','선생님');

select * from myboard order by idx desc;
```

Oracle

```
create table myboard(
  idx number not null,
  title varchar2(100) not null,
  content varchar2(2000) not null,
  writer varchar2(30) not null,
  indate date default sysdate,
  count number default 0,
  primary key(idx)
);

create sequence myboard_idx;

insert into myboard(idx, title, content, writer)
values(myboard_idx.nextval, '스프링게시판','스프링게시판','관리자');
insert into myboard(idx, title, content, writer)
values(myboard_idx.nextval, '스프링게시판','스프링게시판','박매일');

select * from myboard order by idx desc;
```

MySQL 접속 TEST(JUnit)

DBTest.java

☐ MySQL 서버가 구동된 상태에서 실습

```
package kr.board.test;
import java.sql.Connection;
import java.sql.DriverManager;

import org.junit.Test;

public class DBTest {

    @Test
    public void testConnection() {
        try(Connection conn=DriverManager.getConnection
                ("jdbc:mysql://localhost:3306/com", "com", "com01")){
            System.out.println(conn);
        }catch(Exception e){
            e.printStackTrace();
        }
    }

}
```

com.mysql.jdbc.JDBC4Connection@2d6e8792

```
C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32>cd bin

C:₩eGovFrame-4.0.0₩bin₩mysql-5.7.32₩bin>mysqld.exe --console
mysqld: Could not create or access the registry key needed for the MySQL applicat
to log to the Windows EventLog. Run the application with sufficient
privileges once to create the key, add the key manually, or turn off
logging for that application.
2022-05-22T09:37:48.641357Z 0 [Warning] TIMESTAMP with implicit DEFAULT value is
tion (see documentation for more details).
2022-05-22T09:37:48.641582Z 0 [Note] --secure-file-priv is set to NULL. Operations
2022-05-22T09:37:48.645474Z 0 [ERROR] Cannot open Windows EventLog; check privile
2022-05-22T09:37:48.645520Z 0 [Note] mysqld.exe (mysqld 5.7.32-log) starting as pr
2022-05-22T09:37:48.682128Z 0 [Note] InnoDB: Mutexes and rw_locks use Windows inte
2022-05-22T09:37:48.682394Z 0 [Note] InnoDB: Uses event mutexes
2022-05-22T09:37:48.683154Z 0 [Note] InnoDB: _mm_lfence() and _mm_sfence() are use
2022-05-22T09:37:48.683428Z 0 [Note] InnoDB: Compressed tables use zlib 1.2.11
2022-05-22T09:37:48.684439Z 0 [Note] InnoDB: Number of pools: 1
2022-05-22T09:37:48.685060Z 0 [Note] InnoDB: Not using CPU crc32 instructions
2022-05-22T09:37:48.693444Z 0 [Note] InnoDB: Initializing buffer pool, total size
2022-05-22T09:37:48.705364Z 0 [Note] InnoDB: Completed initialization of buffer po
2022-05-22T09:37:48.778162Z 0 [Note] InnoDB: Highest supported file format is Barr
2022-05-22T09:37:50.230812Z 0 [Note] InnoDB: Creating shared tablespace for tempor
2022-05-22T09:37:50.232185Z 0 [Note] InnoDB: Setting file '.₩ibtmp1' size to 12 MB
2022-05-22T09:37:50.316854Z 0 [Note] InnoDB: File '.₩ibtmp1' size is now 12 MB.
2022-05-22T09:37:50.341067Z 0 [Note] InnoDB: 96 redo rollback segment(s) found. 96
2022-05-22T09:37:50.341586Z 0 [Note] InnoDB: 32 non-redo rollback segment(s) are a
2022-05-22T09:37:50.345136Z 0 [Note] InnoDB: Waiting for purge to start
2022-05-22T09:37:50.411556Z 0 [Note] InnoDB: 5.7.32 started; log sequence number
2022-05-22T09:37:50.412917Z 0 [Note] Plugin 'FEDERATED' is disabled.
2022-05-22T09:37:50.414980Z 0 [Note] InnoDB: Loading buffer pool(s) from C:₩eGovFr
2022-05-22T09:37:50.468823Z 0 [Note] Found ca.pem, server-cert.pem and server-key
2022-05-22T09:37:50.469585Z 0 [Note] Skipping generation of SSL certificates as ce
2022-05-22T09:37:50.478484Z 0 [Warning] CA certificate ca.pem is self signed.
2022-05-22T09:37:50.479384Z 0 [Note] Skipping generation of RSA key pair as key f
2022-05-22T09:37:50.484382Z 0 [Note] Server hostname (bind-address): '*'; port: 33
2022-05-22T09:37:50.487287Z 0 [Note] IPv6 is available.
2022-05-22T09:37:50.487675Z 0 [Note]   - '::' resolves to '::';
2022-05-22T09:37:50.490030Z 0 [Note] Server socket created on IP: '::'.
2022-05-22T09:37:50.507262Z 0 [Note] InnoDB: Buffer pool(s) load completed at 2205
2022-05-22T09:37:50.591758Z 0 [Note] Failed to start slave threads for channel ''
2022-05-22T09:37:50.698025Z 0 [Note] Event Scheduler: Loaded 0 events
2022-05-22T09:37:50.698805Z 0 [Note] mysqld.exe: ready for connections.
Version: '5.7.32-log'  socket: ''  port: 3306  MySQL Community Server (GPL)
```

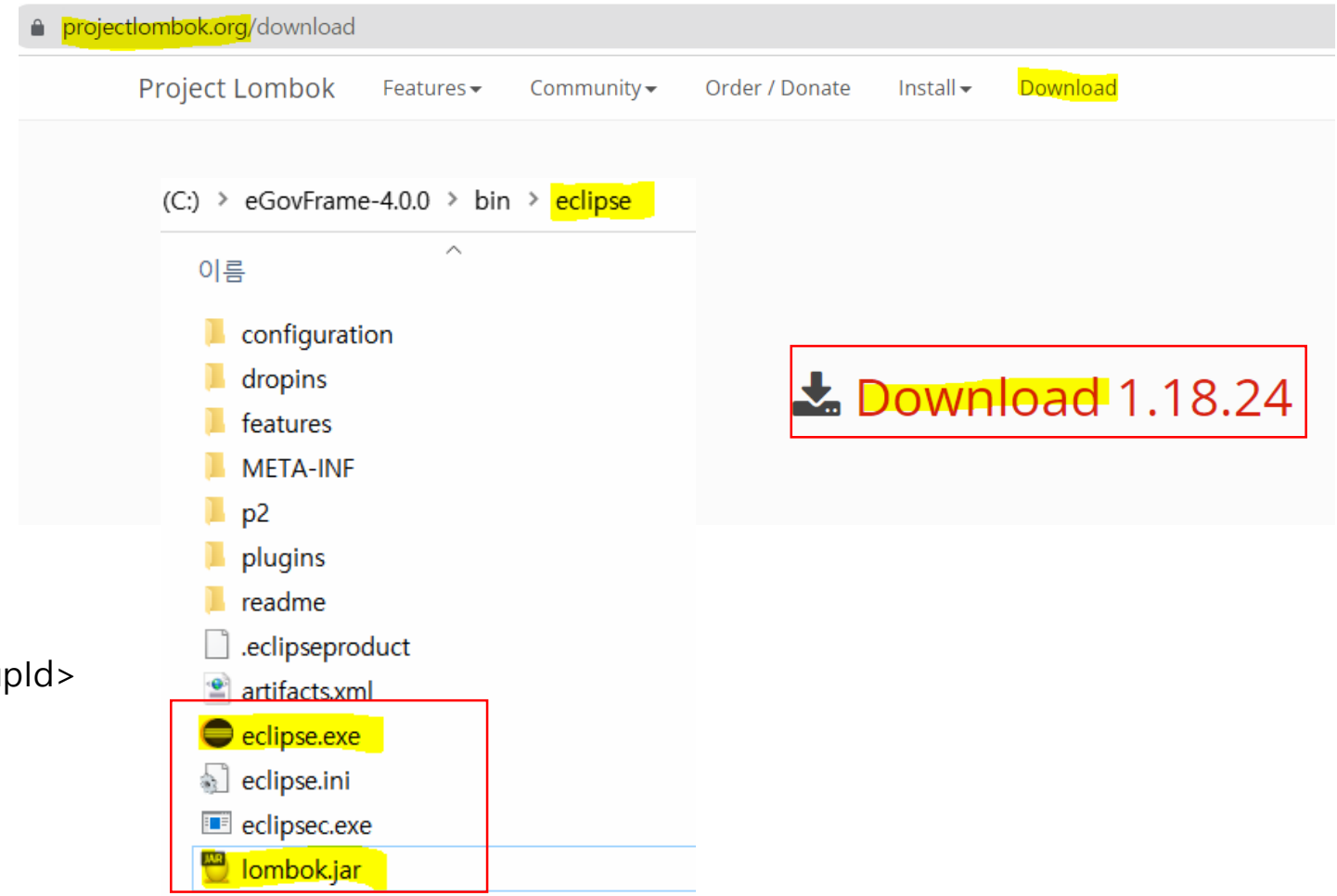
Lombok API 설치 - 1

```
package kr.board.entity;

@Data  //- Lombok API
public class Board {
  private int idx; // 번호
  private String title; // 제목
  private String content; // 내용
  private String writer; // 작성자
  private String indate; // 작성일
  private int count; // 조회수
  // setter , getter
}
```

pom.xml

```
<dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <version>1.18.12</version>
        <scope>provided</scope>
</dependency>
```

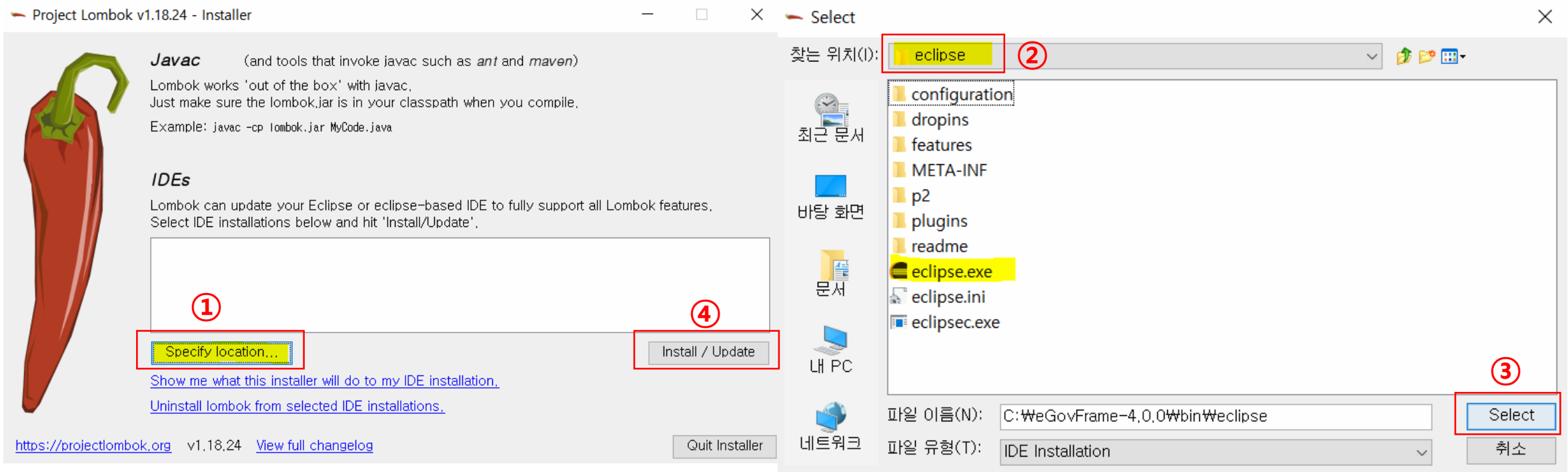
☐ Lombok API 다운로드 및 설치(https://projectlombok.org)



Lombok API 설치(cmd) - 2

1. C:₩Users₩GSM>**cd** C:₩eGovFrame-4.0.0₩bin₩jdk8u242-b08₩bin

2. C:₩eGovFrame-4.0.0₩bin₩jdk8u242-b08₩bin>java -jar C:₩eGovFrame-4.0.0₩bin₩eclipse₩lombok.jar



3. 이클립스를(eGovFram) 재시작 한다.

TPC

생각하고(Thinking)-> 표현하고(Presentation)->코딩(Coding)하고

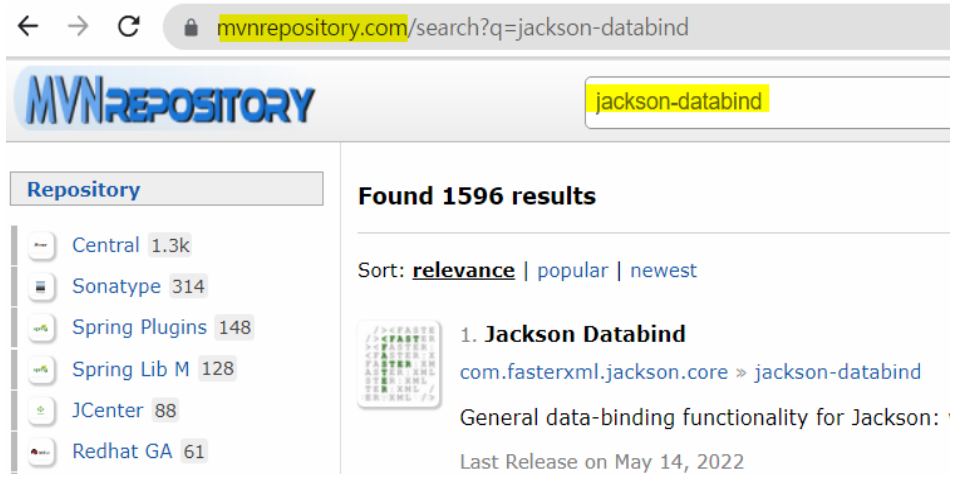
실습 : Spring MVC02

Controller의 리턴타입

- String : jsp를 이용하는 경우에는 jsp파일의 경로와 파일이름을 나타내기 위해서 사용
- void : 호출하는 URL과 동일한 이름의 jsp를 의미
- **VO, DTO 타입** : 주로 **JSON타입**의 데이터를 만들어서 반환하는 용도
- ResponseEntity 타입 : response할 때 Http헤더정보와 내용을 가공하는 용도

객체 타입

Controller의 메서드 리턴 타입을 VO(Value Object)나 DTO(Data Transfer Object)타입 등 복합적인 데이터가 들어간 객체 타입으로 지정할 수 있는데, 이 경우는 주로 JSON 데이터를 만들어 내는 용도로 사용합니다.



- jackson-databind API를 pom.xml에 추가하기 –

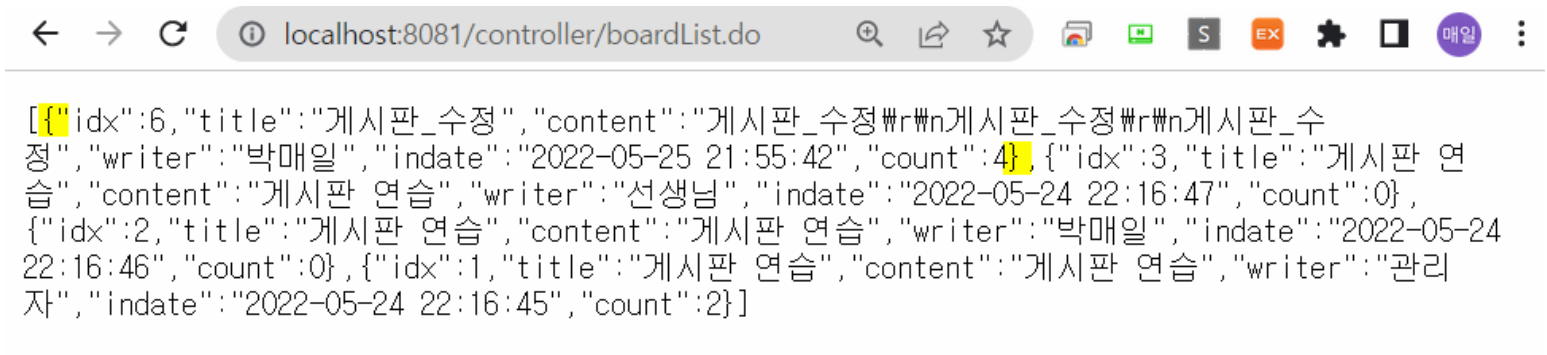
```
<!--
https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-databind -->
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.9.8</version>
</dependency>
```

Jackson-databind를 통해 JSON으로 변환

Board

```
@Data
public class Board {
  private int idx; // 번호
  private String title; // 제목
  private String content; // 내용
  private String writer; // 작성자
  private String indate; // 작성일
  private int count; // 조회수
}
```

String(JSON)



Board	idx	title	content	writer	indate	count
	6	게시판_수정	게시판_수정	박매일	2022-05-25	4

{ key : value, key : value , , , }

JSON(jackson-databind)

String

{"**idx**":6,"**title**":"게시판_수정","**content**":"게시판_수정₩r₩n게시판_수정₩r₩n게시판_수정","**writer**":"박매일","**indate**":"2022-05-25 21:55:42","**count**":4}

[{"**idx**":6,"**title**":"게시판_수정","**content**":"게시판_수정₩r₩n게시판_수정₩r₩n게시판_수정","**writer**":"박매일","**indate**":"2022-05-25 21:55:42","**count**":4},
{"**idx**":6,"**title**":"게시판_수정","**content**":"게시판_수정₩r₩n게시판_수정₩r₩n게시판_수정","**writer**":"박매일","**indate**":"2022-05-25 21:55:42","**count**":4}]

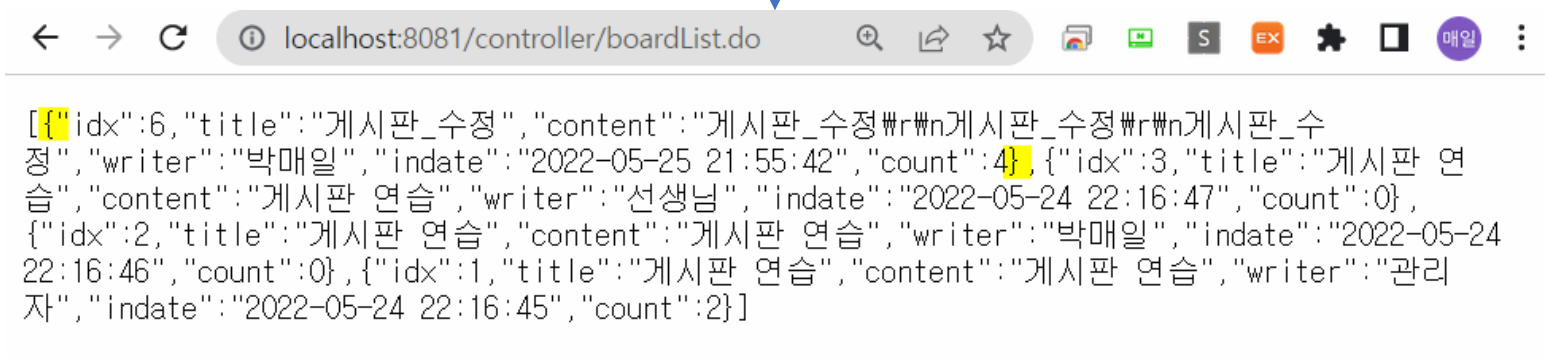
@ResponseBody를 통해 클라이언트로 전달

```
@RequestMapping("/boardList.do")
 public @ResponseBody List<Board> boardList(){
    List<Board> list=boardMapper.getLists();
    return list;
}
```

List<Board>

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.9.8</version>
</dependency>
```

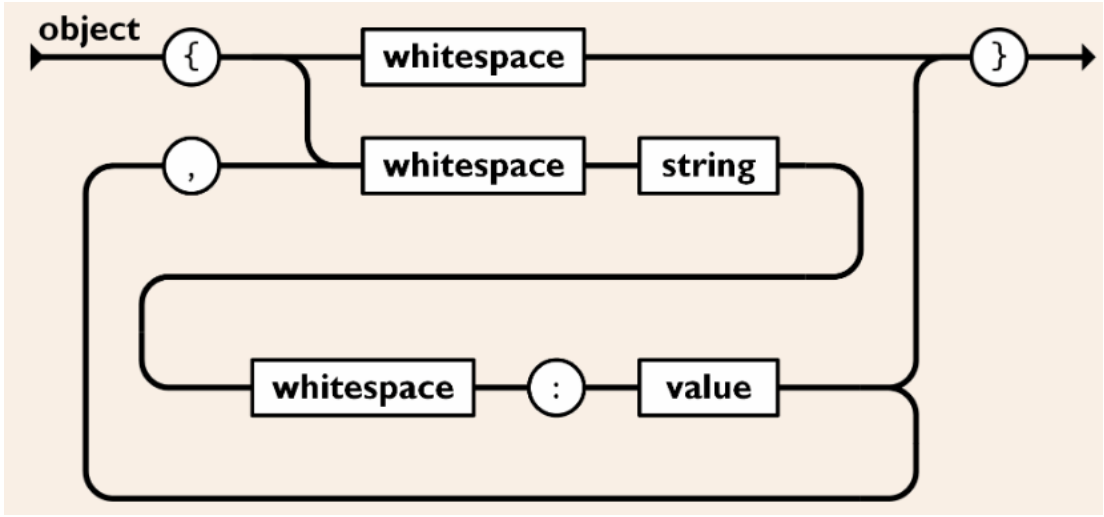
String(JSON)

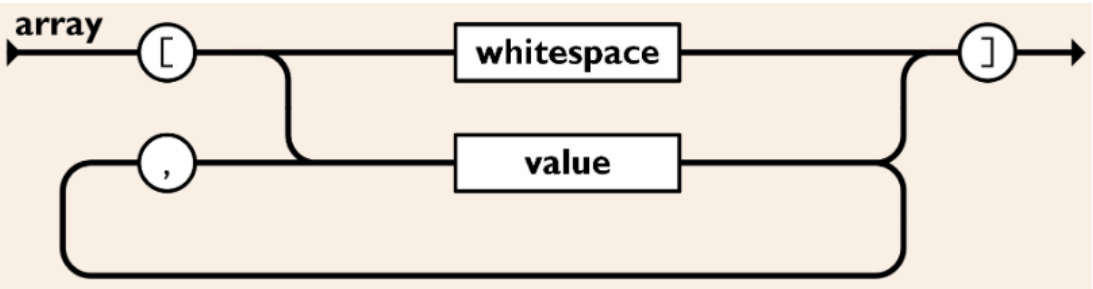


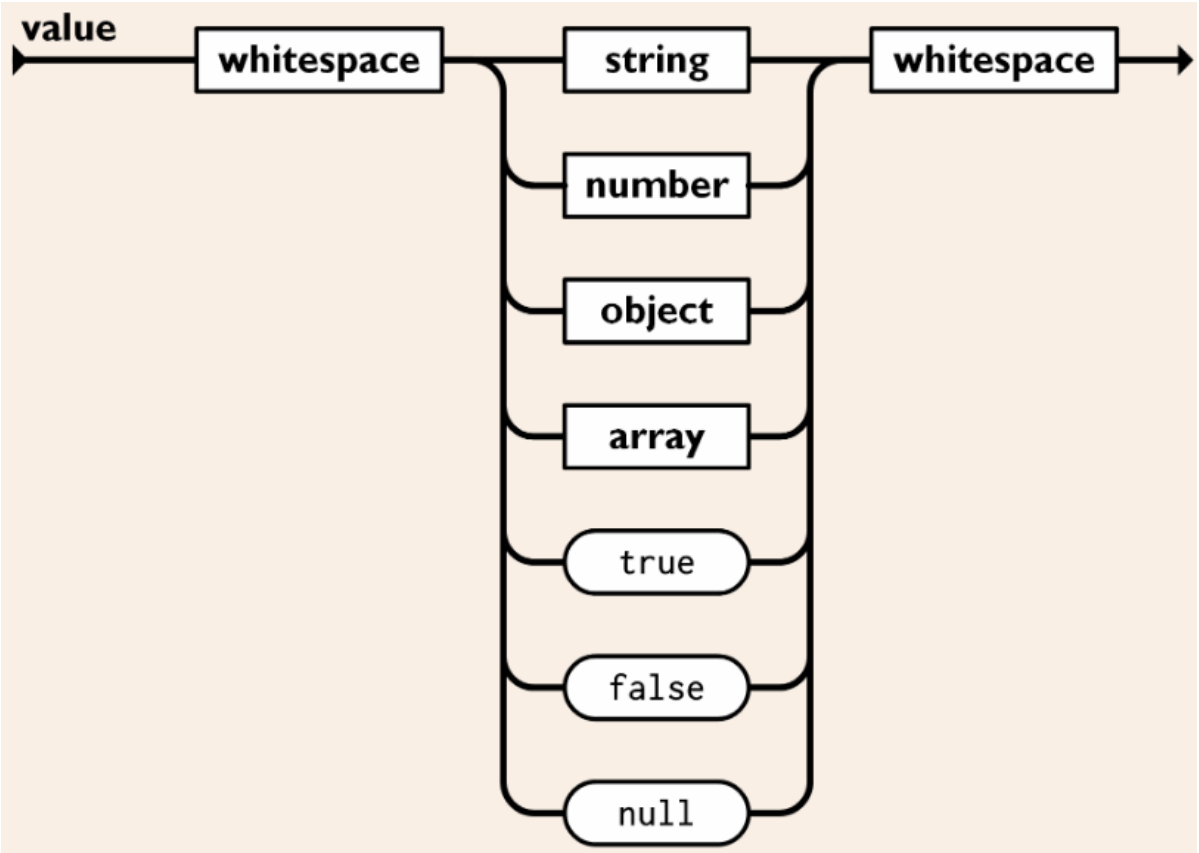
JSON (JavaScript Object Notation)

https://www.json.org/json-en.html

JSON (JavaScript Object Notation) is a lightweight **data-interchange format**.



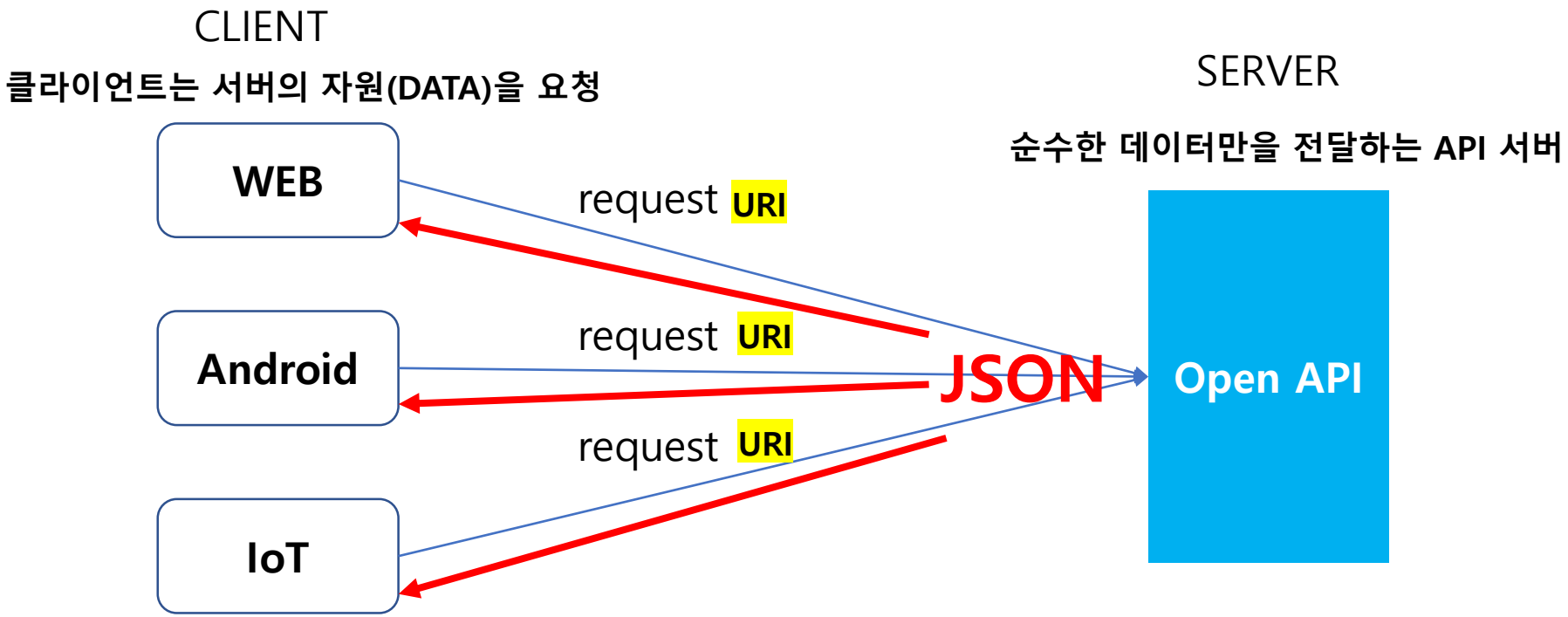




REST 방식과 Ajax

모바일 환경의 대두로 서버 역할의 변화

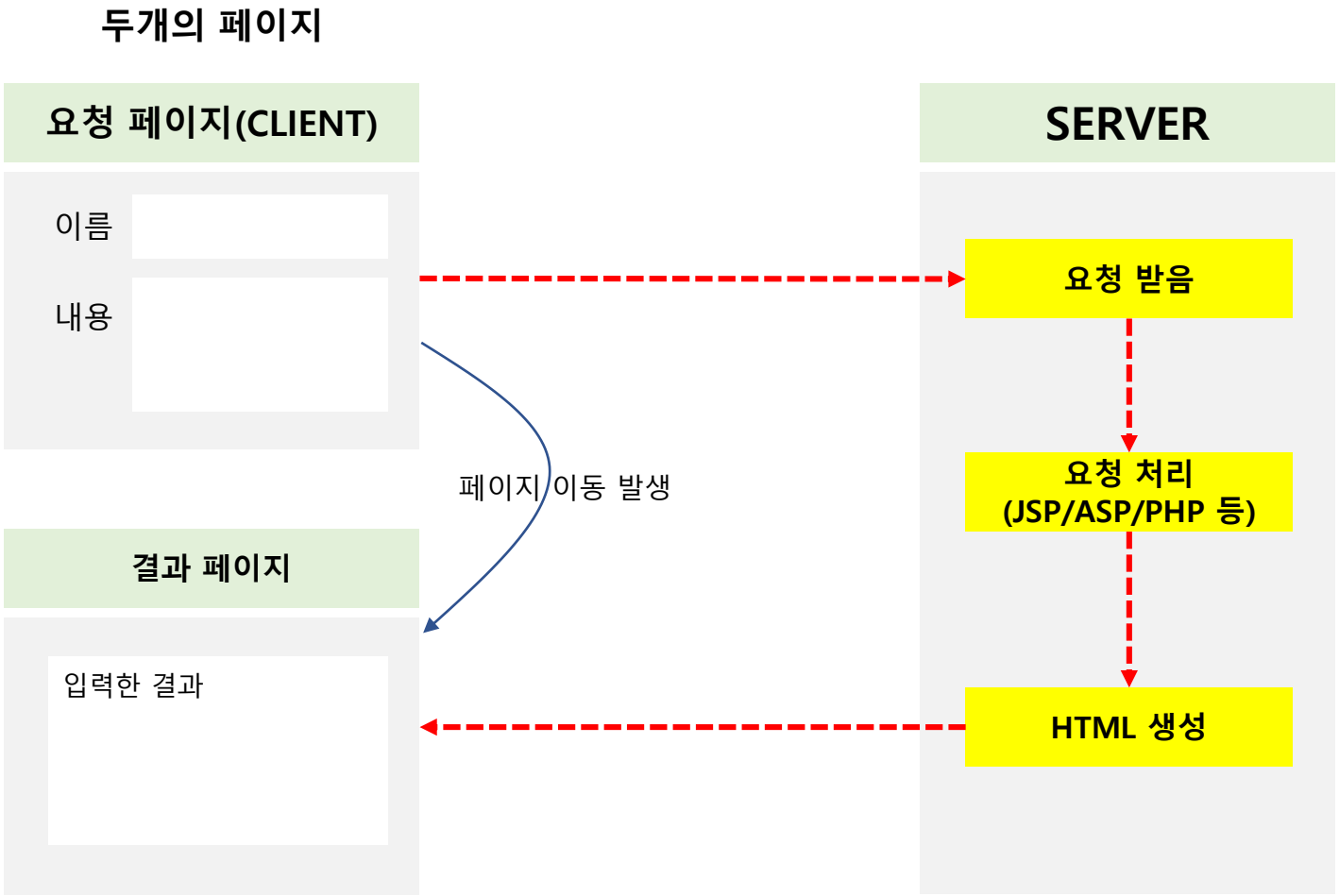
서버는 브라우저나 모바일에서 필요한 순수한 데이터만을 전달하는 API 서버의 형태로 변화
서버는 클라이언트의 요청 결과를 **XML**이나 **JSON**의 형태로 전달하고, 브라우저나 모바일에서는 이를 가공해서 사용자에게 보여주는 방식



URI(Uniform Resource Identifier) : 자원의 식별자(주소)

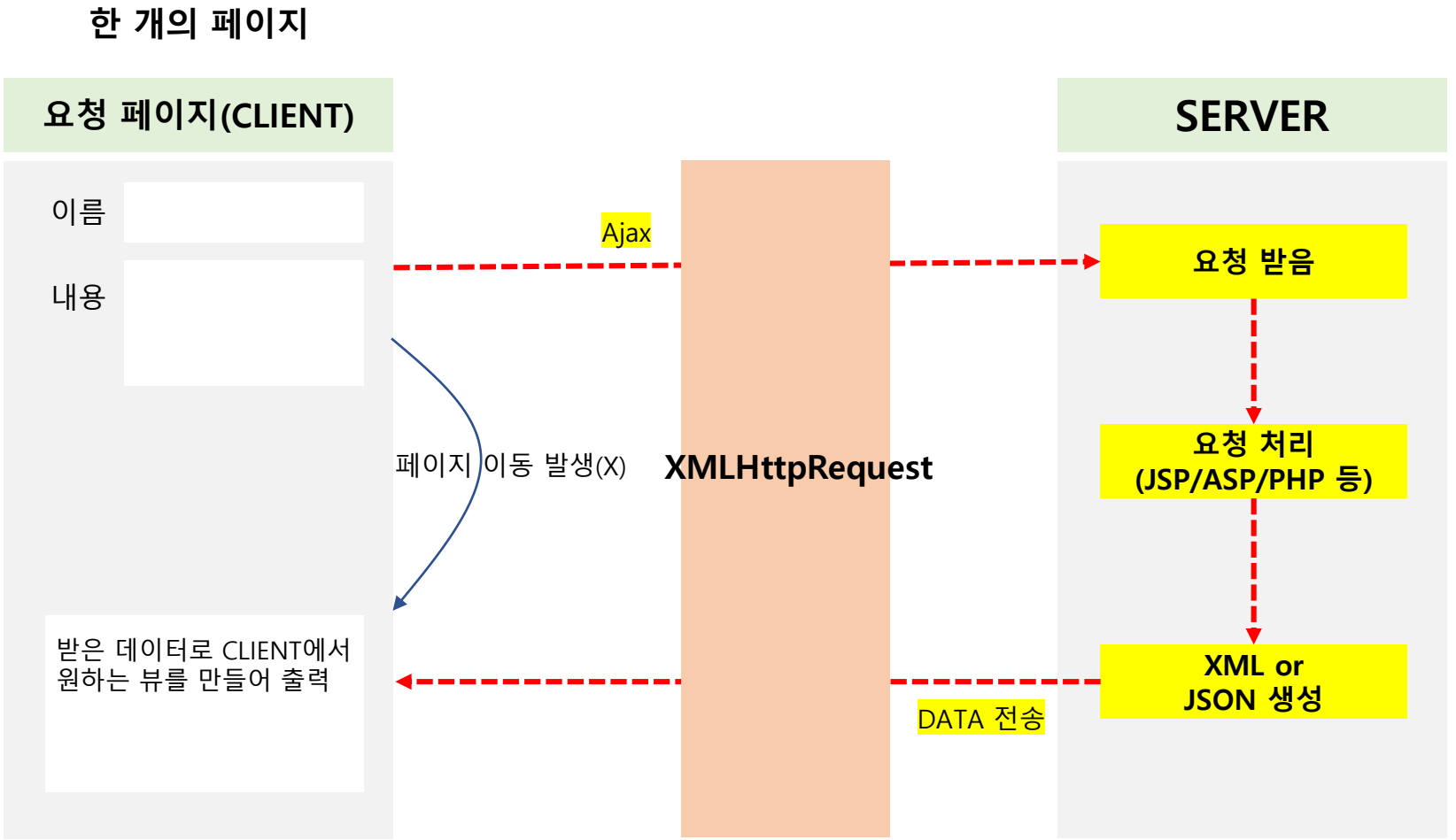
jQuery Ajax 기능

기존 웹 페이지 동작 방식



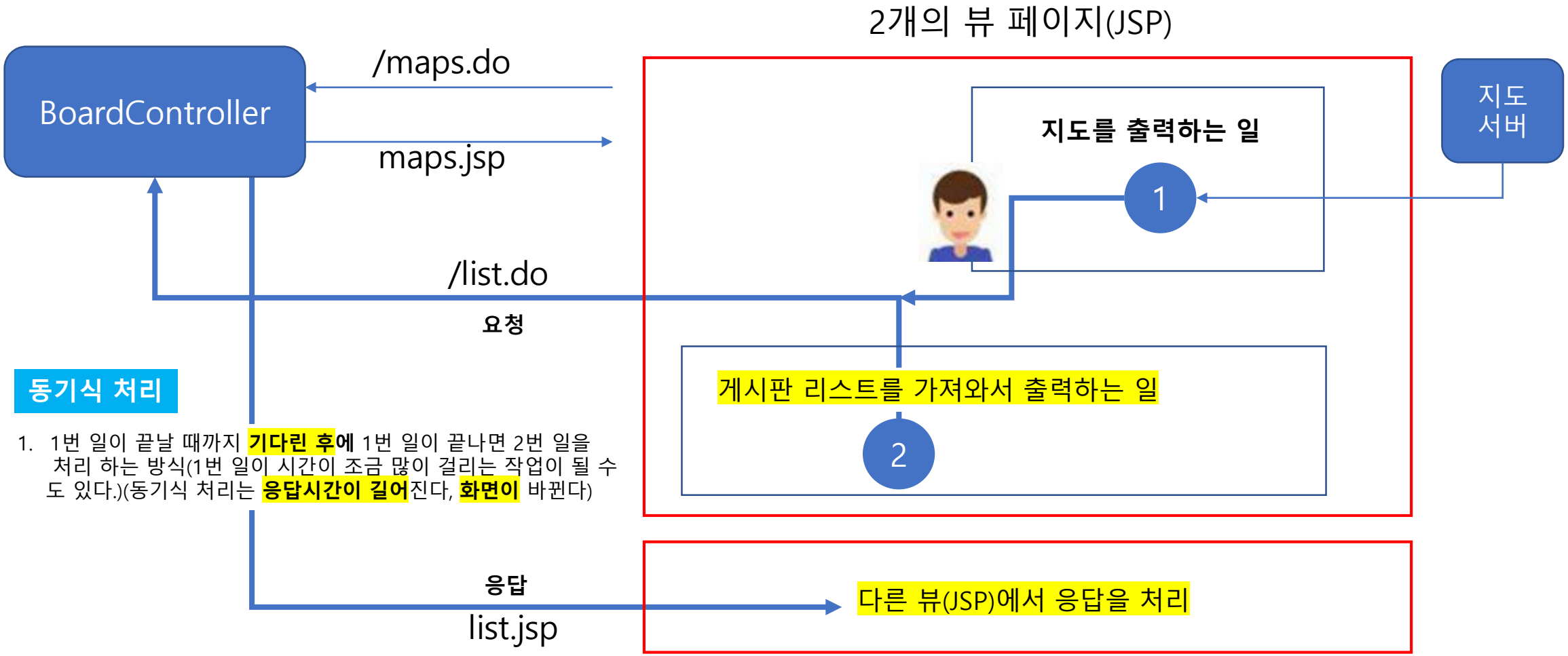
jQuery Ajax 기능

Ajax 웹 페이지 동작 방식



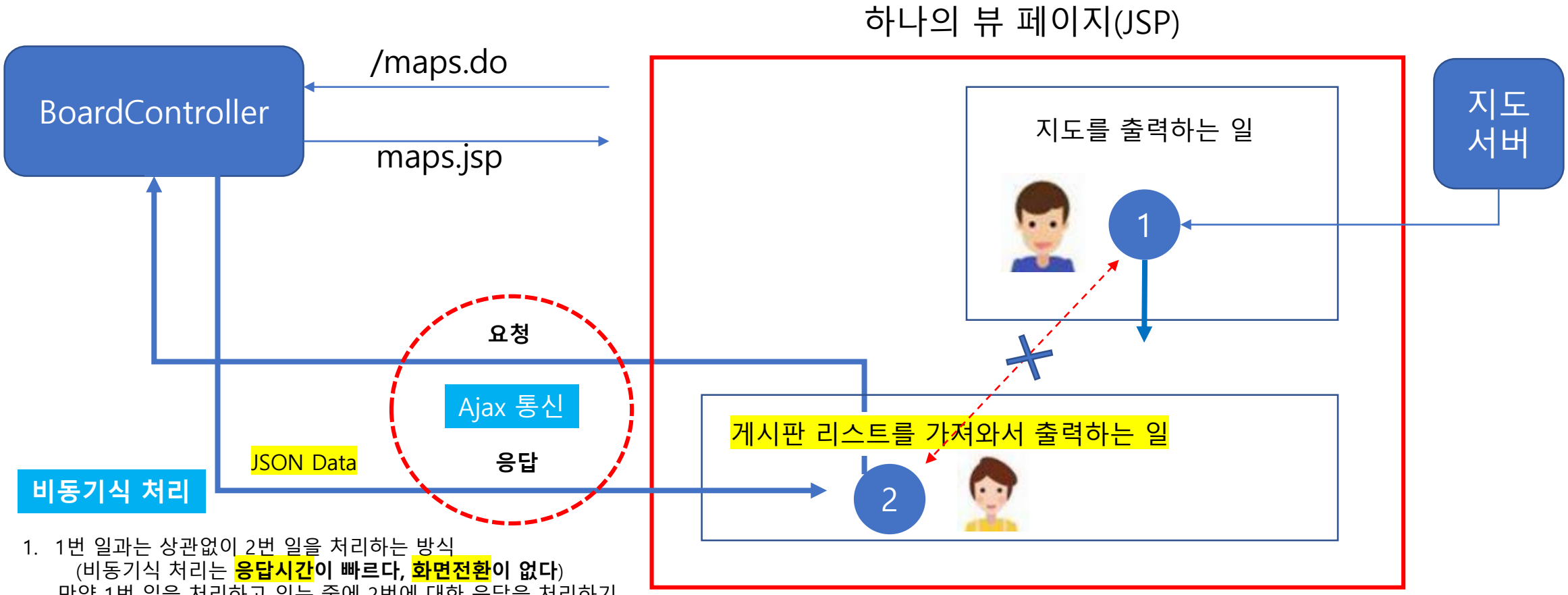
동기식 처리방법

하나의 뷰 페이지에서(JSP) 여러가지 처리를 해야 되는 경우에 어떻게 2가지 일을 처리 할까?
예를 들어서 지도를 출력하는 일, 게시판 리스트를 출력하는 일을 동시에 하는 방법을 생각해보자.



비동기식 처리방법

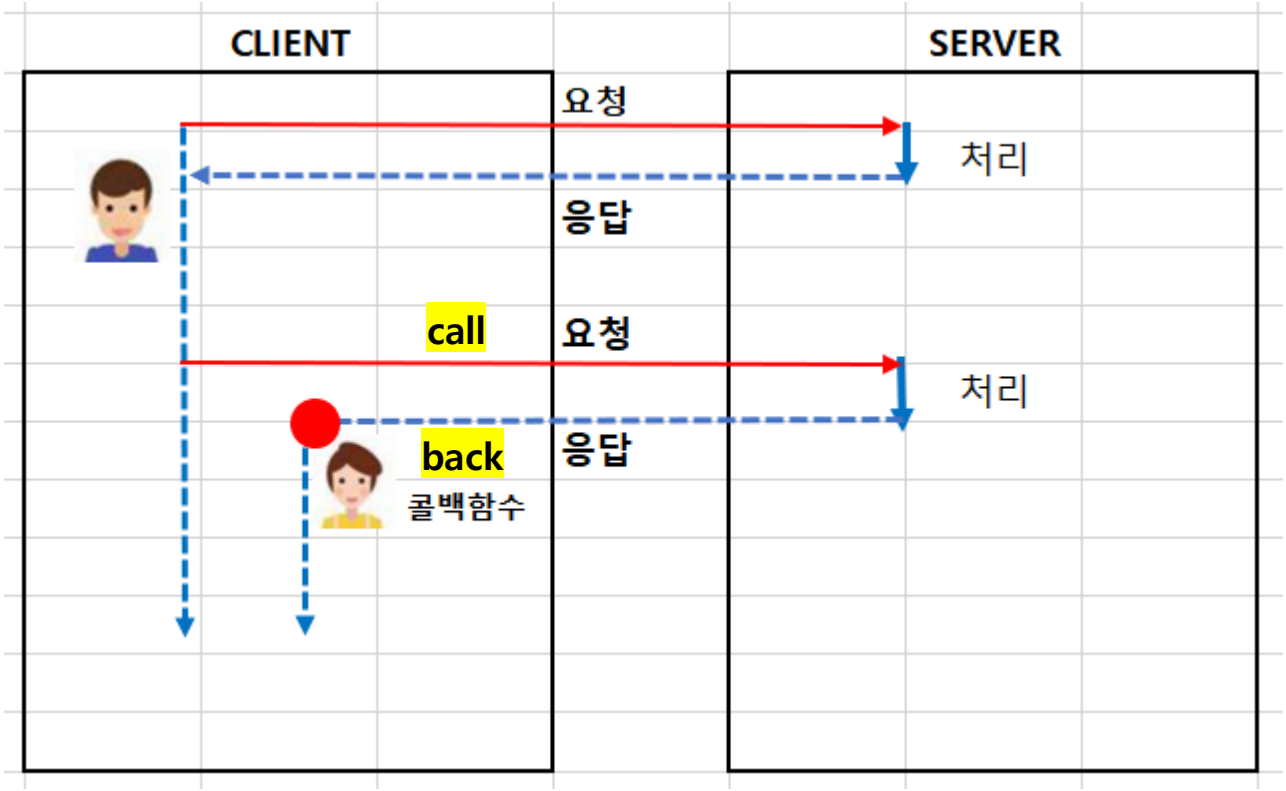
하나의 뷰 페이지에서(JSP) 여러가지 처리를 해야 되는 경우에 어떻게 2가지 일을 처리 할까?
예를 들어서 지도를 출력하는 일, 게시판 리스트를 출력하는 일을 동시에 하는 방법을 생각해보자.



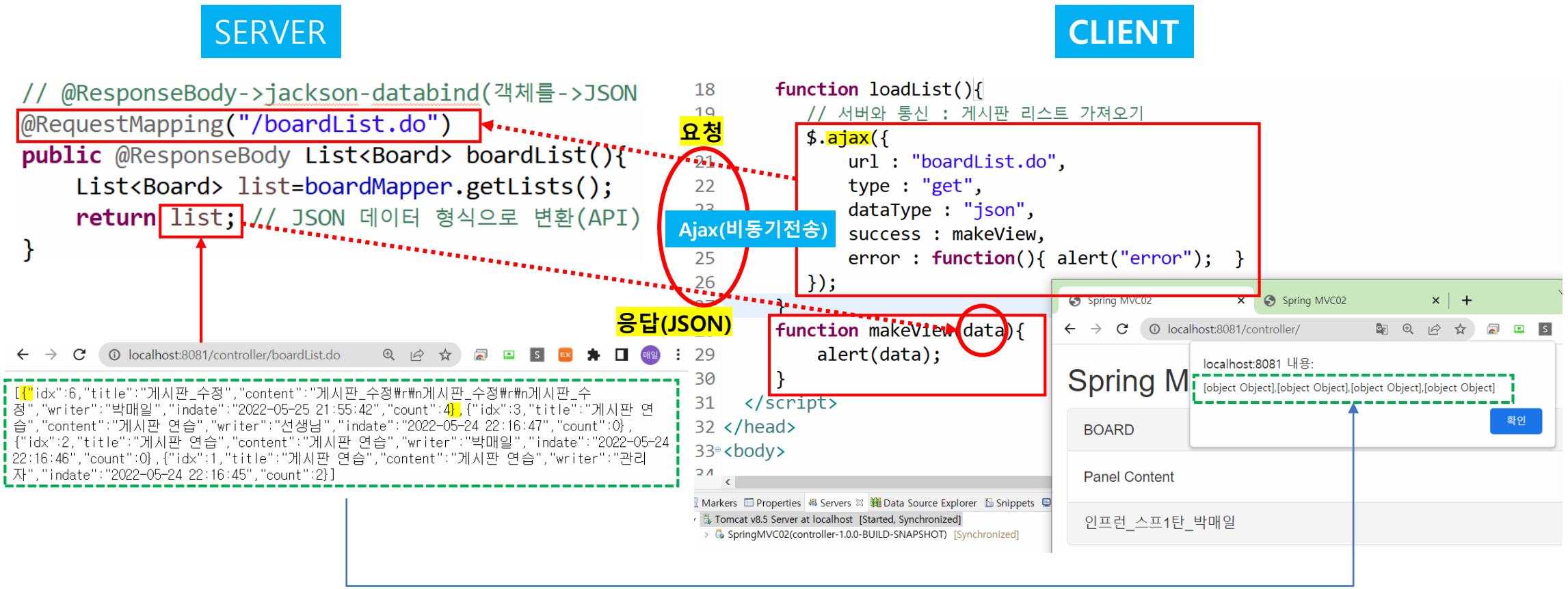
1. 1번 일과는 상관없이 2번 일을 처리하는 방식
   (비동기식 처리는 **응답시간이 빠르다, 화면전환이 없다**)
   만약 1번 일을 처리하고 있는 중에 2번에 대한 응답을 처리하기 위해서는 콜백함수를 만들어서 처리하게 해야 한다.

콜백함수(callback) 란?

: 클라이언트가 서버로 요청 후 서버에서 응답하는 데이터를 클라이언트에서 받아서 처리하는 함수
-> 클라이언트에서는 JavaScript(자바스크립트)에서 처리한다.



jQuery Ajax 기능



jQuery Ajax 기능

Ajax 웹 페이지 동작 방식

```
$.ajax(
    type : “post” or “get”,
    url : “요청할 URL”,
    data : { 서버로 전송할 데이터},
    dataType : “서버에서 전송받을 데이터형식”,
    success : function(서버로부터 데이터 받기){
        // 정상 요청, 응답시 처리
     },
     error : function(오류정보){
        // 오류 발생시 처리
     }
);
```

REST(Representational State Transfer : **대표상태전송**) = URI + GET/POST/PUT/DELETE/...

REST : 서버의 고유한 리소스를 접근하는 **대표 상태 전송**

REST 구성

URI + GET/POST/PUT/DELETE/...

서버의 고유한 리소스를 접근하는 **주소**

요청방식(GET, POST 방식)으로 결정

어노테이션	기능
@RestController	Controller가 REST방식을 처리하기 위한 것임을 명시합니다.
@ResponseBody	일반적인 JSP와 같은 뷰로 전달되는 게 아니라 데이터 자체를 전달하기 위한 용도
@PathVariable	URL 경로에 있는 값을 파라메터로 추출하려고 할때 사용
@CrossOrigin	Ajax의 크로스 도메인 문제를 해결해주는 어노테이션
@RequestBody	JSON 데이터를 원하는 타입으로 바인딩 처리

REST 전송방식(GET/POST/PUT/DELETE/...)

작업	전송방식	URI	의미
등록(Create)	POST	/boardInsert.do	등록 해주세요
조회(Read)	GET	/boardContent.do	해당(idx=5) 게시판 내용을 보여주세요
전체조회	GET	/boardList.do	전체리스트를 보여주세요
수정(Update)	POST	/boardUpdate.do	해당(idx=5) 게시판을 수정해주세요.
수정(조회수)	GET	/boardCount.do	해당(idx=5) 게시판 조회수를 누적해주세요.
삭제(Delete)	GET	/boardDelete.do	해당(idx=5) 게시판을 삭제해주세요.

작업	전송방식	URI
등록(Create)	POST	/board/new
조회(Read)	GET	/board/{idx}
전체조회	GET	/board/all
수정(Update)	PUT	/board/update + body(json 데이터 등)
수정(조회수)	PUT	/board/count/{idx}
삭제(Delete)	DELETE	/board/{idx}

Ajax(Asynchronous JavaScript and XML)

Ajax(Asynchronous JavaScript and XML)
: 웹에서 화면을 갱신하지 않고 Server로 부터 Data를 가져와서 동적으로 뷰를 만들 수 있는 방법

key	설명
url	요청이 전송되는 URL이 포함된 문자열 입니다.
type	Http요청방식 입니다. (Get/Post)
timeout	Http요청에 대한 제한 시간을 지정합니다.(단위 : ms)
success	Http요청 성공시 이벤트 핸들러 입니다.
error	Http요청 실패시 이벤트 핸들러 입니다.
complete	Http요청 완료시 이벤트 핸들러 입니다.
data	Http요청 후 return하는 값 입니다.
dataType	Http요청 후 return하는 데이터의 Type을 지정합니다. (xml,html,sript,json,jsonp,text)
async	요청시 동기 유무를 선택할 수 있습니다.(True/False)
dataType	return된 데이터의 Type 입니다. (xml,html,json,jsonp,script,text)
cache	브라우저에 의해 요청되는 페이지를 캐시할 수 있습니다 (True/False)
beforeSend	Http요청 전에 발생하는 이벤트 핸들러 입니다.
global	전역함수 활성 여부를 설정합니다. (True/False)

$.ajax({

Ajax통신시 필요한
대표적인 Property

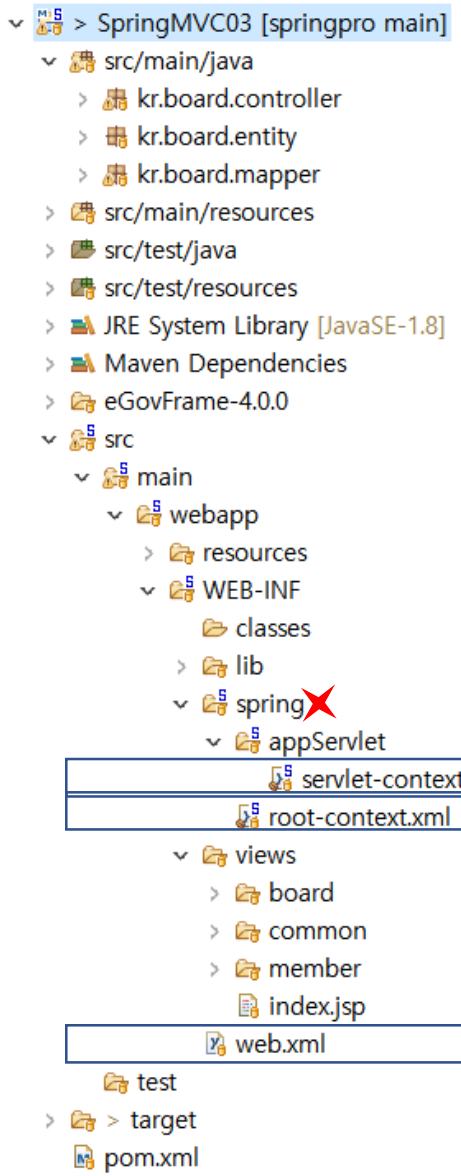
});

TPC

생각하고(Thinking)->표현하고(Presentation)->코딩(Coding)하고

실습 : Spring MVC04

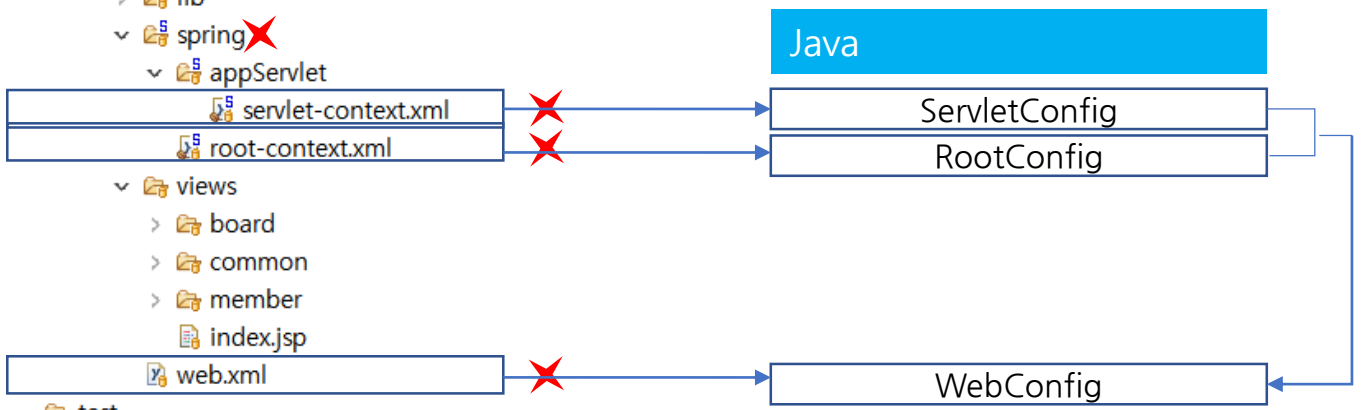
Java Configuration(WebConfig.java)



pom.xml 수정

web.xml 파일을 삭제하고
pom.xml 파일에 web.xml파일을 사용하지 않겠다고 지정한다.

```
<plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-war-plugin</artifactId>
        <version>3.2.0</version>
        <configuration>
          <failOnMissingWebXml>false</failOnMissingWebXml>
        </configuration>
</plugin>
```



@EnableWebMvc
@Configuration

Java Configuration(WebConfig.java)

web.xml을 대신할 자바 파일 생성(WebConfig.java)

```
package kr.board.config;
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer{

    @Override
    protected Class<?>[] getRootConfigClasses() {
            // TODO Auto-generated method stub
            return null;
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
            return null;
    }

    @Override
    protected String[] getServletMappings() {
            return null;
    }
}
```

3개의 추상 메서드를 Override(재정의)한다.

web.xml → WebConfig.java

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app
 <filter>
    <filter-name>encodingFilter</filter-name>
    <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
    <init-param>
       <param-name>encoding</param-name>
       <param-value>UTF-8</param-value>
    </init-param>
    <init-param>
       <param-name>forceEncoding</param-name>
       <param-value>true</param-value>
    </init-param>
 </filter>
 <filter-mapping>
    <filter-name>encodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
 </filter-mapping>
```

```java
@Override
protected Filter[] getServletFilters() {
CharacterEncodingFilter encodingFilter = new CharacterEncodingFilter();
        encodingFilter.setEncoding("UTF-8");
        encodingFilter.setForceEncoding(true);
        return new Filter[]{encodingFilter};
}
```

```xml
<context-param>
<param-name>contextConfigLocation</param-name>
<param-value>/WEB-INF/spring/root-context.xml</param-value>
</context-param>

<listener>
<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

```java
@Override
protected Class<?>[] getRootConfigClasses() {
        // TODO Auto-generated method stub
        return new Class[] { RootConfig.class };
}
```

```xml
<servlet>
       <servlet-name>appServlet</servlet-name>
       <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
       <init-param>
       <param-name>contextConfigLocation</param-name>
       <param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
       </init-param>
       <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
       <servlet-name>appServlet</servlet-name>
       <url-pattern>/</url-pattern>
</servlet-mapping>

</web-app>
```

```java
@Override
protected Class<?>[] getServletConfigClasses() {
        return new Class[] { ServletConfig.class };
}
```

```java
@Override
protected String[] getServletMappings() {
        return new String[] { "/" };
}
```

Java Configuration(ServletConfig.java)

servlet-context.xml ----------> ServletConfig.java

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:beans="http://www.springframework.org/schema/beans"
xmlns:context="http://www.springframework.org/schema/context"
xsi:schemaLocation="http://www.springframework.org/schema/mvc
https://www.springframework.org/schema/mvc/spring-mvc.xsd
http://www.springframework.org/schema/beans
https://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/context
https://www.springframework.org/schema/context/spring-
context.xsd">

<annotation-driven />

<resources mapping="/resources/**" location="/resources/" />

<beans:bean
class="org.springframework.web.servlet.view.InternalResourceView
Resolver">
  <beans:property name="prefix" value="/WEB-INF/views/" />
  <beans:property name="suffix" value=".jsp" />
</beans:bean>

<context:component-scan base-package="kr.board.controller" />

</beans:beans>
```

```
@Configuration
@EnableWebMvc

@ComponentScan(basePackages = {"kr.board.controller"})

public class ServletConfig implements WebMvcConfigurer{

    @Override
    public void configureViewResolvers(ViewResolverRegistry registry) {
                InternalResourceViewResolver bean=new InternalResourceViewResolver();
                bean.setPrefix("/WEB-INF/views/");
                bean.setSuffix(".jsp");
                registry.viewResolver(bean);
    }

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        registry.addResourceHandler("/resources/**").addResourceLocations("/resources/");
    }
}
```

Java Configuration(RootConfig.java)

root-context.xml ----------> RootConfig.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:mybatis-spring="http://mybatis.org/schema/mybatis-spring"
xsi:schemaLocation="http://mybatis.org/schema/mybatis-spring http://mybatis.org/schema/mybatis-spring-1.2.xsd
http://www.springframework.org/schema/beans
https://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
        <property name="driverClassName" value="com.mysql.cj.jdbc.Driver"/>
        <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/com?serverTimezone=UTC"/>
        <property name="username" value="com"/>
        <property name="password" value="com01"/>
</bean>

<bean id="dataSource" class="com.zaxxer.hikari.HikariDataSource" destroy-method="close">
    <constructor-arg ref="hikariConfig" />
</bean>

<bean class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource" />
</bean>

<mybatis-spring:scan base-package="kr.board.mapper"/>

</beans>
```

```java
@Configuration
@MapperScan(basePackages = {"kr.board.mapper"})
@PropertySource({ "classpath:persistence-mysql.properties"})
public class RootConfig {

@Autowired
private Environment env;

@Bean
 public DataSource myDataSource() {
     HikariConfig hikariConfig=new HikariConfig();
     hikariConfig.setDriverClassName(env.getProperty("jdbc.driver"));
     hikariConfig.setJdbcUrl(env.getProperty("jdbc.url"));
     hikariConfig.setUsername(env.getProperty("jdbc.user"));
     hikariConfig.setPassword(env.getProperty("jdbc.password"));

     HikariDataSource myDataSource=new HikariDataSource(hikariConfig);

     return myDataSource;
}

@Bean
public SqlSessionFactory sessionFactory() throws Exception{

     SqlSessionFactoryBean sessionFactory=new SqlSessionFactoryBean();
     sessionFactory.setDataSource(myDataSource());
     return (SqlSessionFactory)sessionFactory.getObject();
 }
}
```

```
# persistence-mysql.properties
jdbc.driver=com.mysql.cj.jdbc.Driver
jdbc.url=jdbc:mysql://localhost:3306/com?serverTimezone=UTC
jdbc.user=com
jdbc.password=com01
```

Spring Web Security - Java Configuration(SecurityConfig.java)

pom.xml에 API를 추가한다.

```
<org.springframework-version>5.0.2.RELEASE</org.springframework-version>
<org.springsecurity-version>5.0.2.RELEASE</org.springsecurity-version>

<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>${org.springsecurity-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>${org.springsecurity-version}</version>
</dependency>

<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-taglibs</artifactId>
    <version>${org.springsecurity-version}</version>
</dependency>
```

Spring Web Security - Java Configuration(SecurityInitializer.java)

Spring Security 동작 클래스 만들기

AbstractSecurityWebApplicationInitializer클래스를 상속하여 SecurityInitializer를 생성한다.
- 내부적으로 DelegatingFilterProxy를 스프링에 등록하여 스프링 시큐리티를 내부적으로 동작시킨다.

```
public class SecurityInitializer
                        extends AbstractSecurityWebApplicationInitializer{



}
```

Spring Web Security - Java Configuration(SecurityConfig.java)

Spring Security 환경설정파일 만들기

WebSecurityConfigurerAdapter클래스를 상속하여 SecurityConfig객체를 생성한다.
- @EnableWebSecurity는 스프링MVC와 스프링 시큐리티를 결합하는 클래스이다.
- configure() 메서드를 Override하고 관련 설정을 한다.

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter{
   //요청에대한 설정
 @Override
 protected void configure(HttpSecurity http) throws Exception {
        CharacterEncodingFilter filter = new CharacterEncodingFilter();
        filter.setEncoding("UTF-8");
        filter.setForceEncoding(true);
        http.addFilterBefore(filter,CsrfFilter.class);
 }
}
```

Spring Web Security - Java Configuration(SecurityConfig.java)

WebConfig 클래스에 등록하기

```
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer{

    @Override
    protected Class<?>[] getRootConfigClasses() {
    // TODO Auto-generated method stub
        return new Class[] { RootConfig.class , SecurityConfig.class};
    }

}
```

Spring Web Security – form(POST), ajax 통신시 반드시 CSRF 토큰을 추가 해주어야 한다.

```
<script type="text/javascript">
 var csrfHeaderName = "${_csrf.headerName}";
 var csrfTokenValue = "${_csrf.token}";

function boardList(){
    $.ajax({
    url : "${cpath}/board",
    type : "get",
    dataType : "json",
    beforeSend: function(xhr){
                xhr.setRequestHeader(csrfHeaderName, csrfTokenValue)
            },
    success : callBack,
    error : function(){ alert("error");  }
    });
}
```

CSRF(Cross-site request forgery)
: 사이트간 위조 방지를 목적으로 특정한 값의 토큰을 사용하는 방식
예) URI를 통해서 접속자의 권한을 관리자 권한으로 바꾸는 것

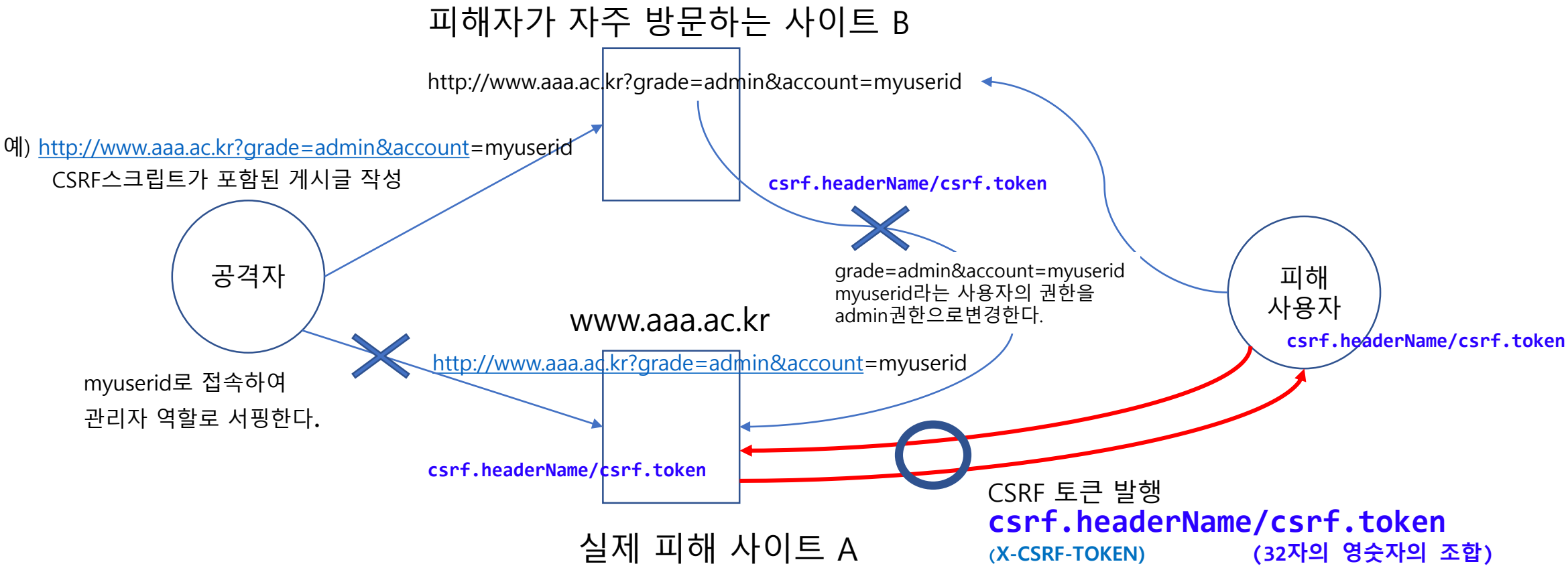
- 서버가 접속한 클라이언트에게 특정 CSRF토큰을 전달한다.
- 클라이언트는 서버에 접속 할 때마다 CSRF값을 가지고 온다.
- 서버는 클라이언트의 CSRF값과 서버에 보관된 CSRF값을 비교하여
- 동일한 사용자 접속인지 확인하고 서비스를 제공해준다.

```
<form>
 <input type="hidden" name="${_csrf.parameterName}" value="${_csrf.token}"/>
</form>
```

```
<form action="${contextPath}/memImageUpdate.do?${_csrf.parameterName}=${_csrf.token}"
                                                    method="post" enctype="multipart/form-data">
</form>
```

Spring Web Security –CSRF 토큰(CSRF(Cross-site request forgery)

공격자가 특정 게시물이나, 이미지에 A사이트에
접속할 수 있는 특정 URL을 등록하여(URL뒤에 특정 파라메터를 전달)
B사이트를 방문한 사용자들에 A사이트를 공격하는 기법



TPC

생각하고(Thinking)->표현하고(Presentation)->코딩(Coding)하고

실습 : Spring MVC05

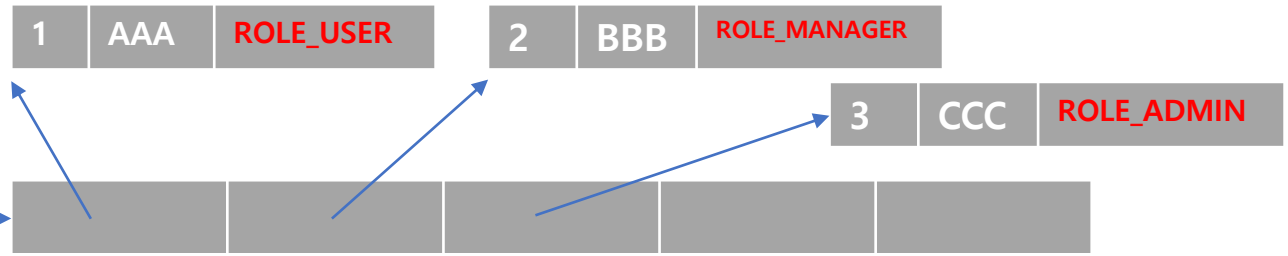
Spring Web Security – Table 구성(회원 Table + 권한 Table)

```sql
create table mem_stbl(
  memIdx int not null, -- 자동증가X
  memID varchar(20) not null,
  memPassword varchar(20) not null,
  memName varchar(20) not null,
  memAge int,
  memGender varchar(20),
  memEmail varchar(50),
  memProfile varchar(50),
  primary key(memID)
);
```

```sql
create table mem_auth(
  no int not null auto_increment,
  memID varchar(50) not null,
  auth varchar(50) not null,
  primary key(no),
  constraint fk_member_auth foreign key(memID)
                            references mem_stbl(memID)
);
```

```java
@Data
public class Member {
  private int memIdx;
  private String memID;
  private String memPassword;
  private String memName;
  private int memAge; // <-null, 0
  private String memGender;
  private String memEmail;
  private String memProfile; //사진정보
  private List<AuthVO> authList; // 추가
}
```

```java
@Data
public class AuthVO {
   private int no; // 일련번호
   private String memID; //회원 아이디
   private String auth; // 회원권한
}
```



Spring Web Security
– 회원가입 폼에 권한 체크박스 추가하기
– 회원가입 버튼을 클릭시 권한체크박스 체크 유무 확인

회원가입 폼에 사용자 권한 설정 체크 박스 추가하기

```
<tr>
  <td style="width: 110px; vertical-align: middle;">사용자 권한</td>
  <td colspan="2">
     <input type="checkbox" name="authList[0].auth" value="ROLE_USER"> ROLE_USER
     <input type="checkbox" name="authList[1].auth" value="ROLE_MANAGER"> ROLE_MANAGER
     <input type="checkbox" name="authList[2].auth" value="ROLE_ADMIN"> ROLE_ADMIN
  </td>
</tr>
```

```
@Data
public class Member {
  private int memIdx;
  private String memID;
  private String memPassword;
  private String memName;
  private int memAge; // <-null, 0
  private String memGender;
  private String memEmail;
  private String memProfile; //사진정보
  private List<AuthVO> authList; // 추가
}
```

```
@RequestMapping("/memRegister.do")
public String memRegister(Member m, String memPassword1, String memPassword2,
                          RedirectAttributes rttr, HttpSession session) {
if(m.getMemID()==null || m.getMemID().equals("") ||
   memPassword1==null || memPassword1.equals("") ||
   memPassword2==null || memPassword2.equals("") ||
   m.getMemName()==null || m.getMemName().equals("") ||
   m.getMemAge()==0 || m.getAuthList().size()==0 ||
   m.getMemGender()==null || m.getMemGender().equals("") ||
   m.getMemEmail()==null || m.getMemEmail().equals("")) {
   // 누락메세지를 가지고 가기? =>객체바인딩(Model, HttpServletRequest, HttpSession)
   rttr.addFlashAttribute("msgType", "실패 메세지");
   rttr.addFlashAttribute("msg", "모든 내용을 입력하세요.");
   return "redirect:/memJoin.do";  // ${msgType} , ${msg}
}
```

Spring Web Security
– 회원가입시 패스워드 암호화 하여 저장하기
– 회원가입시 회원의 권한도 권한 테이블에 저장하기 – 한 명의 회원이 여러 권한을 가질 수 있다)

MemberController에 PasswordEncoder 객체 연결하기

```
// 회원가입시 패스워드 암호화
@Autowired
PasswordEncoder pwEncoder;

// 패스워드 암호화 하기
String encyptPw=pwEncoder.encode(m.getMemPassword());
m.setMemPassword(encyptPw);
// 회원을 테이블에 저장하기
int result=memberMapper.register(m);
```

```
if(result==1) { // 회원가입 성공 메세지
   //권한테이블에 회원권한 저장하기
  List<AuthVO> list=m.getAuthList();
  for(AuthVO authVO : list) {
  if(authVO.getAuth()!=null) {
          AuthVO saveVO=new AuthVO();
          saveVO.setMemID(m.getMemID());
          saveVO.setAuth(authVO.getAuth());
          memberMapper.authInsert(saveVO);
     }
   }
   rttr.addFlashAttribute("msgType", "성공 메세지");
   rttr.addFlashAttribute("msg", "회원가입에 성공했습니다.");
   // 회원가입이 성공하면=>가입정보 다시 가져와서(로그인처리하기)
   Member mvo=memberMapper.getMember(m.getMemID());
   session.setAttribute("mvo", mvo); // ${!empty mvo}
   System.out.println("회원가입성공" + mvo);
   return "redirect:/";
```

```
<insert id="register" parameterType="kr.board.entity.Member">
    insert into
    mem_stbl(memIdx,
    memID,memPassword,memName,memAge,memGender,memEmail,memProfile)
    values((select IFNULL(MAX(memIdx)+1,1) from mem_stbl mem),
    #{memID},#{memPassword},#{memName},#{memAge},#{memGender},#{memEmail},#{memProfile})
 </insert>
```

MemberMapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
<mapper namespace="kr.board.mapper.MemberMapper">

 <resultMap type="kr.board.entity.Member" id="memberMap">
      <id property="memIdx" column="memIdx"/>
      <result property="memID" column="memID"/>
      <result property="memPassword" column="memPassword"/>
      <result property="memName" column="memName"/>
      <result property="memAge" column="memAge"/>
      <result property="memGender" column="memGender"/>
      <result property="memEmail" column="memEmail"/>
      <result property="memProfile" column="memProfile"/>
      <collection property="authList" resultMap="authMap">
      </collection>
  </resultMap>

  <resultMap type="kr.board.entity.AuthVO" id="authMap">
      <id property="no" column="no"/>
      <result property="memID" column="memID"/>
      <result property="auth" column="auth"/>
  </resultMap>

 <select id="registerCheck" resultMap="memberMap">
   select * from mem_stbl where memID=#{memID}
 </select>

<insert id="register" parameterType="kr.board.entity.Member">
   insert into
   mem_stbl(memIdx,
   memID,memPassword,memName,memAge,memGender,memEmail,memProfile)
   values((select IFNULL(MAX(memIdx)+1,1) from mem_stbl mem),
   #{memID},#{memPassword},#{memName},#{memAge},
   #{memGender},#{memEmail},#{memProfile})
</insert>

 <insert id="authInsert" parameterType="kr.board.entity.AuthVO">
    INSERT INTO mem_auth(memID, auth) values(#{memID}, #{auth})
 </insert>

 <delete id="authDelete">
    delete from mem_auth where memID=#{memID}
 </delete>

 <select id="memLogin" parameterType="kr.board.entity.Member"
                       resultMap="memberMap">
       select * from mem_stbl mem LEFT OUTER JOIN mem_auth auth on
       mem.memID=auth.memID where mem.memID=#{memID}

 </select>

 <update id="memUpdate" parameterType="kr.board.entity.Member">
     update mem_stbl set memPassword=#{memPassword}, memName=#{memName},
            memAge=#{memAge},memGender=#{memGender},memEmail=#{memEmail}
            where memID=#{memID}
 </update>

 <select id="getMember" resultMap="memberMap">
       select * from mem_stbl mem LEFT OUTER JOIN mem_auth auth on
       mem.memID=auth.memID where mem.memID=#{memID}
 </select>

 <update id="memProfileUpdate" parameterType="kr.board.entity.Member">
     update mem_stbl set memProfile=#{memProfile} where memID=#{memID}
 </update>

</mapper>
```

Spring Web Security
– 회원정보수정하기

```
<!-- 권한정보출력  -->
          <tr>
            <td style="width: 110px; vertical-align: middle;">사용자 권한</td>
            <td colspan="2">
                 <input type="checkbox" name="authList[0].auth" value="ROLE_USER"
                 <c:forEach var="authVO" items="${mvo.authList}">
                      <c:if test="${authVO.auth eq 'ROLE_USER'}">
                        checked
                      </c:if>
                 </c:forEach>
                 /> ROLE_USER
                <input type="checkbox" name="authList[1].auth" value="ROLE_MANAGER"
                   <c:forEach var="authVO" items="${mvo.authList}">
                     <c:if test="${authVO.auth eq 'ROLE_MANAGER'}">
                        checked
                     </c:if>
                   </c:forEach>
                /> ROLE_MANAGER
                <input type="checkbox" name="authList[2].auth" value="ROLE_ADMIN"
                   <c:forEach var="authVO" items="${mvo.authList}">
                     <c:if test="${authVO.auth eq 'ROLE_ADMIN'}">
                        checked
                     </c:if>
                   </c:forEach>
                /> ROLE_ADMIN
            </td>
          </tr>
```

Spring Web Security
– 회원정보수정하기

```
// 비밀번호 암호화 수정 추가~~
String encyptPw=pwEncoder.encode(m.getMemPassword());
m.setMemPassword(encyptPw);
int result=memberMapper.memUpdate(m);
if(result==1) { // 수정성공 메세지
//기존 권한을 모두 삭제하고
memberMapper.authDelete(m.getMemID());

//새로운 권한을 추가한다.
List<AuthVO> list=m.getAuthList();
for(AuthVO authVO : list) {
 if(authVO.getAuth()!=null) {
         AuthVO saveVO=new AuthVO();
         saveVO.setMemID(m.getMemID());
         saveVO.setAuth(authVO.getAuth());
         memberMapper.authInsert(saveVO);
     }
 }
```

TPC

생각하고(Thinking)->표현하고(Presentation)->코딩(Coding)하고

실습 : Spring MVC06

Spring Web Security(스프링시큐리티 설정)

```
@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {
	@Bean
	public UserDetailsService memberUserDetailsService() {
		return new MemberUserDetailsService();
	@Override
	protected void configure(AuthenticationManagerBuilder auth) throws Exception {
		auth.userDetailsService(memberUserDetailsService()).passwordEncoder(passwordEncoder());
		System.out.println("인증매니저 시작");
	}
	@Override
	protected void configure(HttpSecurity http) throws Exception {
	    CharacterEncodingFilter filter = new CharacterEncodingFilter();
	    filter.setEncoding("UTF-8");
	    filter.setForceEncoding(true);
	    http.addFilterBefore(filter,CsrfFilter.class);

	    http
		.authorizeRequests()
			.antMatchers("/").permitAll()
			.and()
		.formLogin()
			.loginPage("/memLoginForm.do")
			.loginProcessingUrl("/memLogin.do")
			.defaultSuccessUrl("/")
			.and()
		.logout()
			.invalidateHttpSession(true)
			.logoutSuccessUrl("/")
			.and()
		.exceptionHandling().accessDeniedPage("/access-denied");
	}

}
```

리소스의 접근
antMatchers 설정한 리소스의 접근을 인증절차 없이 허용한다는 의미

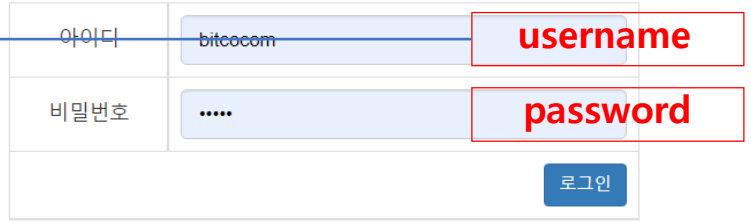
로그인 페이지와 기타 로그인 처리 및 성공 실패 처리를 사용하겠다는 의미
사용자가 따로 만든 로그인 페이지를 사용하려고 할때 설정
로그인 즉 인증 처리를 하는 URL을 설정(인증처리필터 호출)
정상적으로 인증성공 했을 경우 이동하는 페이지를 설정

로그아웃 처리
세션 제거
로그아웃 처리
성공 처리 후 이동(/)

오류페이지 이동

Spring Web Security(UserDetailsService 클래스 만들기, 데이터베이스에 회원인증처리 클래스)

```
public class MemberUserDetailsService implements UserDetailsService{

    @Autowired
    private MemberMapper memberMapper;

    @Override
    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
        System.out.println("로그인 요청이 들어오면 실행");
        Member vo=memberMapper.memberLogin(username);
        // Member Type이 <--X--> UserDetails 타입과 맞지 않다
        // Member타입의 인스턴스를 UserDetails로 처리하려면
        // 1. Member클래스에 UserDetails인터페이스를 구현해 주는 방법
        // 2. Member클래스가 이미 UserDetails인터페이스를 구현한 User클래스를 상속하는 방법
        // 3. 조합을 이용해서 Member를 포함하는 별도의 클래스를 만드는 방법
        //    =>별도의 클래스를 만들고 Member의 인스턴스를 감싸는 형태로 만들면
        //      Member의 모든 정보를 추가적으로 사용가능 하다는 장점
        if(vo!=null) {
          return new MemberUser(vo);
        }else{
          System.out.println("user with username " + username + " does not exist.");
          throw new UsernameNotFoundException("user with username" + username + "does not exist.");
        }
    }
}
```



Spring Web Security(User 클래스 만들기, 회원정보(Member)+권한정보(User) 저장 클래스)

```
public class MemberUser extends User{

	private Member member;

	public MemberUser(String username, String password,
		Collection<? extends GrantedAuthority> authorities) {
		super(username, password, authorities);
	}

	public MemberUser(Member vo) {
		// User클래스에 생성자를 호출하는 코드 작성
		super(vo.getMemId(), vo.getMemPwd(), vo.getAuthList().stream()
		.map(auth->new SimpleGrantedAuthority(auth.getAuth())).
		collect(Collectors.toList()));
		System.out.println("User 생성자 호출");
		this.member=vo;
	}

	public Member getMember() {
		return member;
	}

	public void setMember(Member member) {
		this.member = member;
	}
}
```

UserDetails interface

MemberUser extends User	
Member(회원정보)	Authorities(권한정보)

```
private int memIdx;
private String memID;
private String memPassword;
private String memName;
private int memAge;
private String memGender;
private String memEmail;
private String memProfile;
private List<AuthVO> authList;
```

username(회원아이디)
password(회원패스워드)
authorities(권한정보)

Spring Web Security(User 클래스 만들기, 회원정보(Member)+권한정보(User) 저장 클래스)

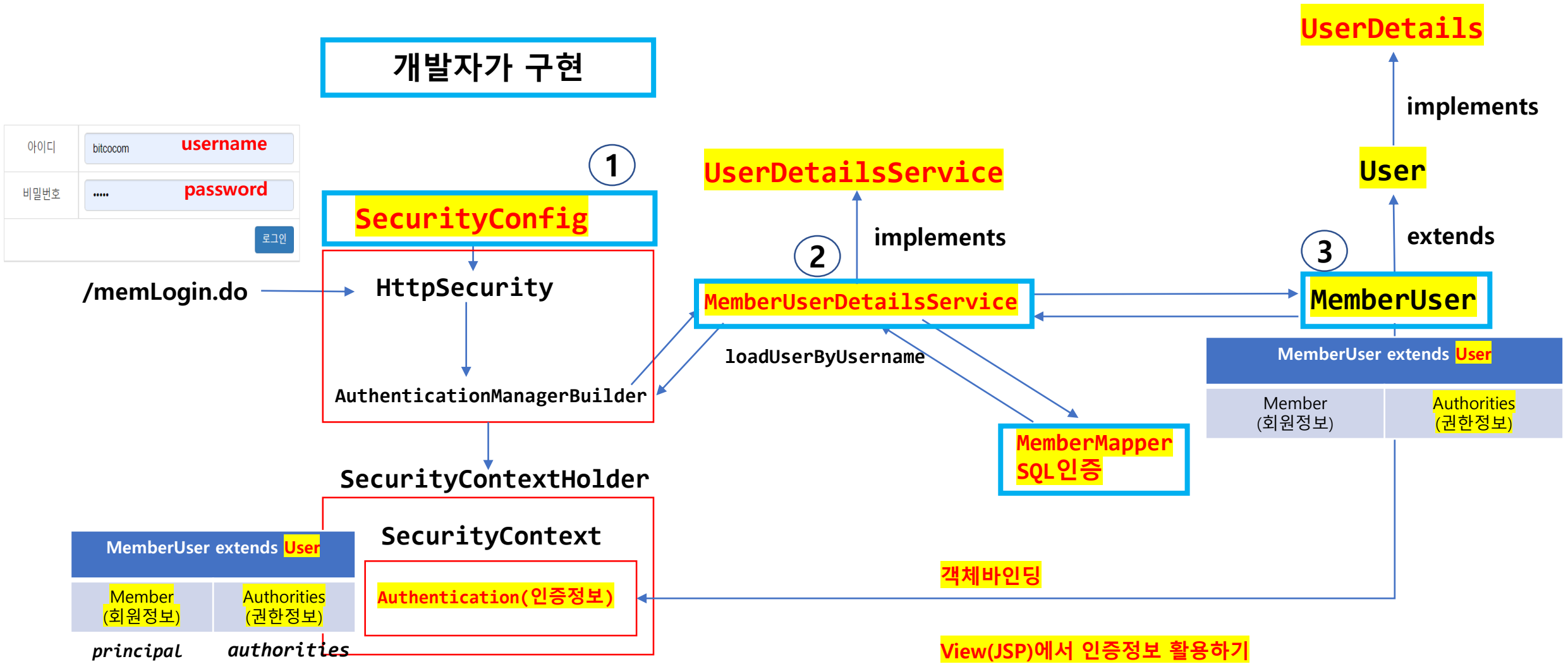
```
public class User implements UserDetails, CredentialsContainer{
        private String password;
        private final String username;
        private final Set<GrantedAuthority> authorities;
        private final boolean accountNonExpired;
        private final boolean accountNonLocked;
        private final boolean credentialsNonExpired;
        private final boolean enabled;
        public User(String username, String password, Collection<? extends GrantedAuthority> authorities) {
                this(username, password, true, true, true, true, authorities);
         }


}
```

implements

```
public MemberUser(Member vo) {
      // User클래스에 생성자를 호출하는 코드 작성
      super(vo.getMemId(), vo.getMemPwd(), vo.getAuthList().stream().map(auth->new SimpleGrantedAuthority(auth.getAuth())).collect(Collectors.toList()));
      System.out.println("User 생성자 호출");
      this.member=vo;
}
```

권한정보를 문자열로 저장
[ROLE_USER, ROLE_MANAGER, ROLE_ADMIN]

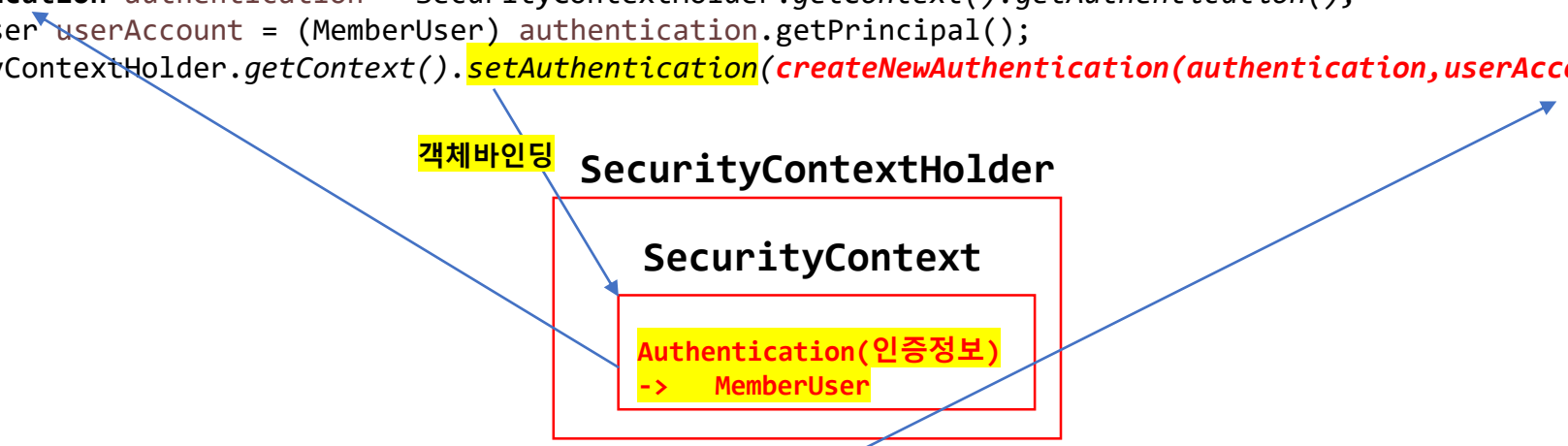
Spring Web Security(인증처리과정)




```
<c:set var="mvo" value="${SPRING_SECURITY_CONTEXT.authentication.principal}"/>
<c:set var="auth" value="${SPRING_SECURITY_CONTEXT.authentication.authorities}"/>
```

Spring Web Security(회원정보 변경시 세션 재설정)
- 회원정보수정 후
- 회원 이미지 등록 후

```
// 스프링보안(새로운 세션 생성)
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
MemberUser userAccount = (MemberUser) authentication.getPrincipal();
SecurityContextHolder.getContext().setAuthentication(createNewAuthentication(authentication,userAccount.getMember().getMemID()));
```




```
// 스프링 보안(새로운 세션 생성 메서드)
// UsernamePasswordAuthenticationToken -> 회원정보+권한정보
protected Authentication createNewAuthentication(Authentication currentAuth, String username) {
    UserDetails newPrincipal = memberUserDetailsService.loadUserByUsername(username);
    UsernamePasswordAuthenticationToken newAuth =
            new UsernamePasswordAuthenticationToken(newPrincipal, currentAuth.getCredentials(), newPrincipal.getAuthorities());
    newAuth.setDetails(currentAuth.getDetails());
    return newAuth;
}
```

- newPrincipal : 현재 로그인 된 사용자의 username을 이용해 이미 업데이트 된 사용자 조회 및 바인딩
- newAuth : 사용자의 ① 새로운 정보(newPrincipal)와 ② 다시 조회된 사용자 권한(newPrincipal.getAuthorities())과 ③ 아직 업데이트 되지 않은 현재 사용자의 자격증명(currentAuth.getCredentials())을 통해 인증된 Authentication 객체를 생성
- newAuth.setDetails() : 새 인증 객체에 기존 details 바인딩

Spring Web Security(View에서 스프링 시큐리티 사용하기)

```
<%@taglib prefix="security" uri="http://www.springframework.org/security/tags" %>
```

```
<c:set var="mvo" value="${SPRING_SECURITY_CONTEXT.authentication.principal}"/>
<c:set var="auth" value="${SPRING_SECURITY_CONTEXT.authentication.authorities}"/>
```

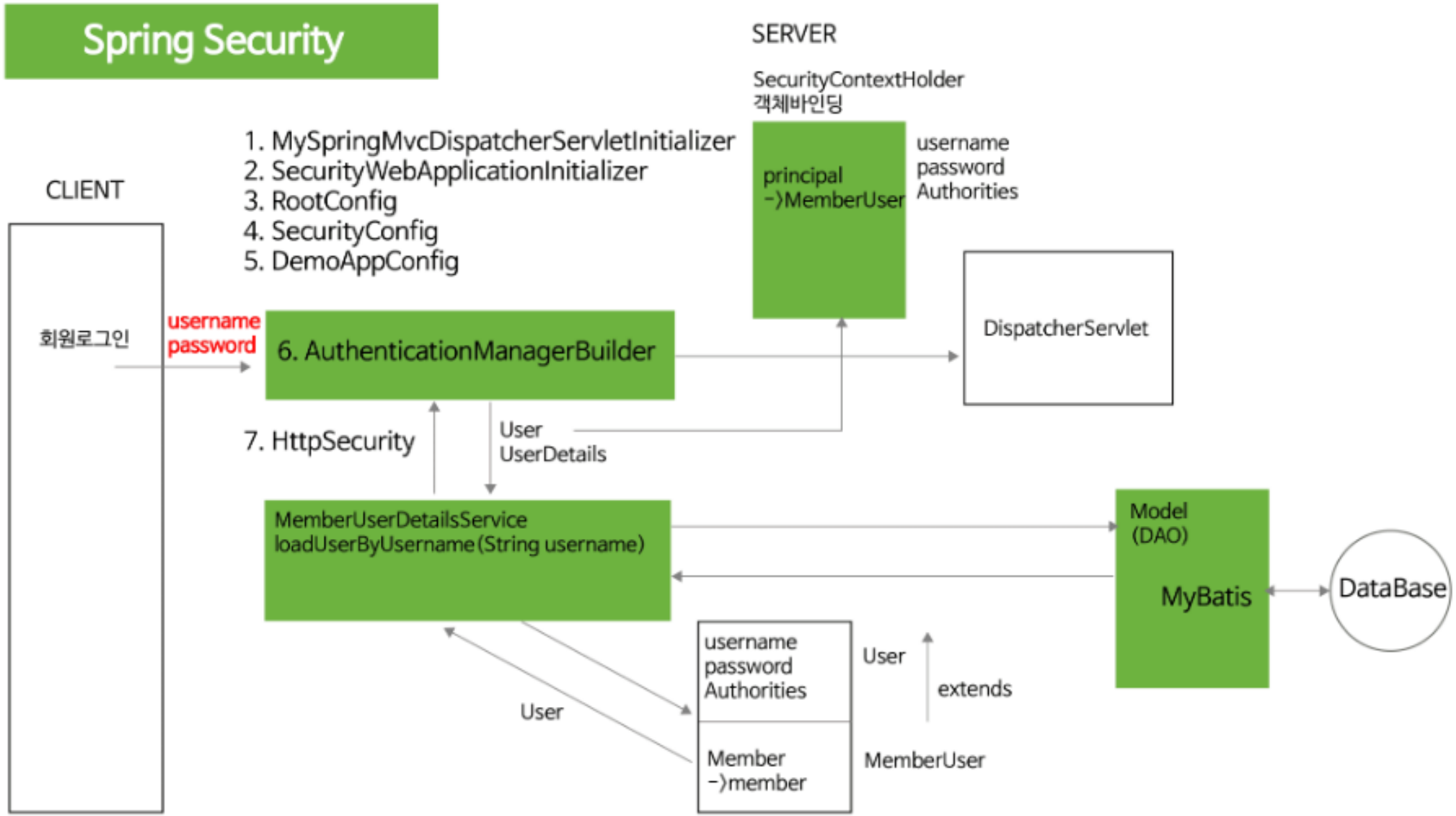
```
<security:authorize access="isAnonymous()">

</security:authorize>
```

```
<security:authorize access="isAuthenticated()">

</security:authorize>
```

```
<security:authentication property="principal.member.memName"/>
```

```
<security:authorize access="hasRole('ROLE_USER')">

</security:authorize>
```

Spring Web Security(동작원리)



Spring Web Security(동작원리)