

// IoT 9일차

// nodeMCU 와 oracle 연동.

// 하드웨어

nodeMCU 빌트인 led 이용, A0핀에 가변저항 연결. nodeMCU 서버 구현.

데스크탑 피씨에 스프링으로 서버를 구현하고, oracle 11g xe 연동.

2개의 서버는 크로스 도메인 허용으로 연결.

// nodeMCU 구현 프로그램.

```
#include <ESP8266WiFi.h>
```

```
#include <WiFiClient.h>
```

```
#include <ESP8266WebServer.h>
```

```
#define LED LED_BUILTIN
```

```
const char* ssid = "you_ssid";
```

```
const char* password = "you_password";
```

```
ESP8266WebServer server(80);
```

// 가변저항 이용.

```
void handleRoot() {
```

```
    Serial.println("You called root page");
```

// 보여줄 웹페이지와 스크립트 생성.

```
String s = "<!DOCTYPE html> <html> <head> <meta charset='UTF-8'> </head> <body>";
```

```

s += "<div><h1>NodeMCU_13</h1>";

s += "<button type='button' onclick='sendData(1)'>LED ON</button>";

s += "<button type='button' onclick='sendData(0)'>LED OFF</button><BR></div>";

s += "<div>ADC Value is : <span id='ADCValue'>0</span><br>";

s += "LED State is : <span id='LEDState'>NA</span></div>";

s += "<script src='https://code.jquery.com/jquery-3.6.0.min.js'></script>";

s += "<script>";

s += " function sendGet(param) {";

s += "$.ajax(";

s += "type : 'get'";

s += "url : 'http://192.168.123.135:8090/node?led=' + param,";

s += "success : function(result, status, xhr) {";

s += "console.log(result);";

s += "},,)}";

s += "function sendData(led) {";

s += "$.ajax({ type : 'get'";

s += "url : '/setLED?LEDstate=' + led,";

s += "success : function(result, status, xhr) { ";

s += "console.log(result);";

s += " $('#LEDState').html(result);";

s += "sendGet(result);";

s += "},,    });    }";

s += "setInterval(function() { ";

s += "getData();";

s += "}, 2000);";

s += "function getData() { ";

```

```

s += "$.ajax({  type : 'get',';

s += "url : '/readADC',';

s += "success : function(result, status, xhr) { ";

s += "        console.log(result);";

s += "        $('#ADCValue').html(result);";

s += "    },    });";

s += "</script><br></body></html>";

```

```

server.send(200, "text/html", s);// 웹 브라우저에 표시.

```

```

}

```

```

void handleADC() {

```

```

    int a = analogRead(A0); // 노드엠의 A0 핀의 값을 읽어와서 정수형 변수에 대입.

```

```

    String adcValue = String(a); // 문자열로 변환.

```

```

    Serial.println("adcValue: " + adcValue); // 시리얼 모니터로 출력.

```

```

    server.send(200, "text/plain", adcValue); // 웹서버에 전송.

```

```

}

```

```

// 잘못된 접근시 안내 메시지 생성.

```

```

void handleNotFound() {

```

```

    String message = "File Not Found\r\n\r\n";

```

```

    message += "URI: ";

```

```

    message += server.uri();

```

```

    message += "\r\nMethod: ";

```

```

    message += (server.method() == HTTP_GET) ? "GET" : "POST";

```

```

message += "\nArguments: ";

message += server.args();

message += "\n";

for (uint8_t i = 0; i < server.args(); i++) {

    message += " " + server.argName(i) + ": " + server.arg(i) + "\n";

}

server.send(404, "text/plain", message);
}

```

```

void handleLED() {

    String ledState = "OFF";

    String t_state = server.arg("LEDstate");

    //Refer  xhttp.open("GET", "setLED?LEDstate="+led, true);

    Serial.print(t_state + " ");

    if (t_state == "1")

    {

        digitalWrite(LED, LOW); //LED ON

        ledState = "ON"; //Feedback parameter

        Serial.println("led_on");

    }

    else

    {

        digitalWrite(LED, HIGH); //LED OFF

        ledState = "OFF"; //Feedback parameter

        Serial.println("led_off");

    }

}

```

```
server.send(200, "text/plain", ledState); //Send web page  
}
```

```
void setup() {
```

```
    // put your setup code here, to run once:
```

```
    Serial.begin(115200);
```

```
    pinMode(LED, OUTPUT);
```

```
    pinMode(A0, INPUT);
```

```
    WiFi.mode(WIFI_STA);
```

```
    WiFi.begin(ssid, password);
```

```
    Serial.println("");
```

```
    // Wait for connection
```

```
    while (WiFi.status() != WL_CONNECTED) {
```

```
        delay(500);
```

```
        Serial.print(".");
```

```
    }
```

```
    Serial.println("");
```

```
    Serial.print("Connected to ");
```

```
    Serial.println(ssid);
```

```
    Serial.print("IP address: ");
```

```
    Serial.println(WiFi.localIP());
```

```
    server.on("/", handleRoot);
```

```
server.on("/setLED", handleLED);

server.on("/readADC", handleADC);

server.onNotFound(handleNotFound);


server.begin();

Serial.println("HTTP server started");

}
```

```
void loop() {

    server.handleClient();

}
```

// 오라클 디비 구현.

SQL\*Plus: Release 11.2.0.2.0 Production on 토 5월 15 04:55:33 2021

// 오라클 계정 생성

```
create user admin identified by 1234;
```

// 권한 부여

```
grant dba to admin;
```

// 시퀀스 생성

```
Create sequence seq_led;
```

-- 테이블 생성

```
Create table tbl_led(
```

```
Lno number(10,0),  
  
onOff varchar(20) not null,  
  
regDate date default sysdate);  
  
commit;
```

// spring\_MVC 서버 구현.

Spring Tool Suite 3

Version: 3.9.14.RELEASE  
Build Id: 202009151354  
Platform: Eclipse 2020-09 (4.17.0)

프로젝트명 : nodeMcuNoracleV2

시작패키지 : kr.icia.controller

// 스프링 버전을 3.1.1 에서 5.2.9 로 변경.

```
<org.springframework-version>5.2.9.RELEASE</org.springframework-version>
```

// pom.xml에 ojdbc8 의존성 추가.

```
<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
```

```
<dependency>  
    <groupId>org.projectlombok</groupId>  
    <artifactId>lombok</artifactId>  
    <version>1.18.20</version>  
    <scope>provided</scope>  
</dependency>
```

```
<dependency>  
    <groupId>com.zaxxer</groupId>  
    <artifactId>HikariCP</artifactId>
```

```

        <version>3.4.5</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.5.3</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring
-->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>2.0.3</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.springframework/spring-
jdbc -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-jdbc</artifactId>
        <version>5.2.9.RELEASE</version>
    </dependency>

    <!-- https://mvnrepository.com/artifact/org.springframework/spring-
test -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-test</artifactId>
        <version>5.2.8.RELEASE</version>
        <scope>test</scope>
    </dependency>

```



```

        <!--
https://mvnrepository.com/artifact/com.oracle.database.jdbc/ojdbc8 -->
        <dependency>
            <groupId>com.oracle.database.jdbc</groupId>
            <artifactId>ojdbc8</artifactId>
            <version>19.7.0.0</version>
            <scope>test</scope>
        </dependency>

        <!-- https://mvnrepository.com/artifact/org.bgee.log4jdbc-log4j2/log4jdbc-log4j2-jdbc4.1 -->
        <dependency>
            <groupId>org.bgee.log4jdbc-log4j2</groupId>
            <artifactId>log4jdbc-log4j2-jdbc4.1</artifactId>
            <version>1.16</version>
        </dependency>

        <!-- https://mvnrepository.com/artifact/org.springframework/spring-context-support -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context-support</artifactId>
            <version>5.2.8.RELEASE</version>
        </dependency>

```

```
// junit 의존성 버전을 4.7 >> 4.12
```

```
// 아래 코드 구성하고, src/test/java
```

```
package kr.icia.controller;
```

```
import static org.junit.Assert.fail;
```

```

import java.sql.Connection;
import java.sql.DriverManager;

import org.junit.Test;

import lombok.extern.log4j.Log4j;

@Log4j
public class JDBCTests {

    static {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void testConnection() {
        try {
            Connection con =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "admin",
"1234");

            Log.info(con);
        } catch (Exception e) {
            fail(e.getMessage());
        }
    }
}

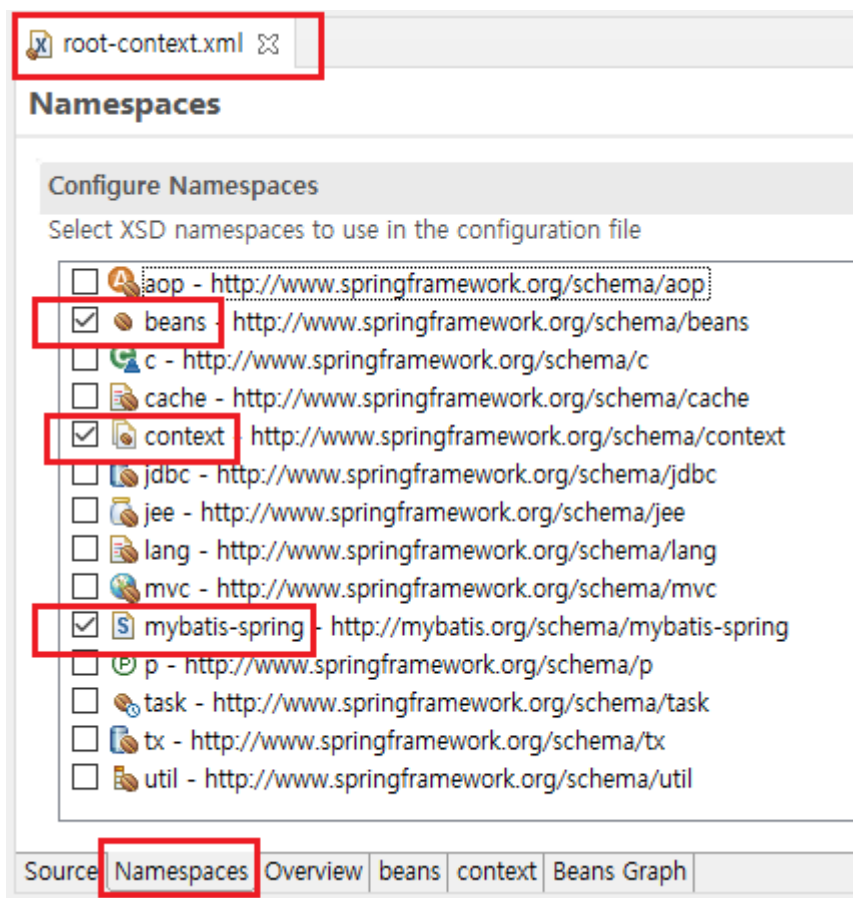
```

// 롬복 적용후, 프로젝트 클린, maven update project, ctrl + shift + O (auto import)

// run as >> junit test 숫자는 다르지만 아래와 같은 객체명이 콘솔에 출력.

INFO : kr.icia.controller.JDBCTests - oracle.jdbc.driver.T4CConnection@5ce81285

// root-context.xml 추가 작성



```
<?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:context="http://www.springframework.org/schema/context"
```

```
    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>

<http://www.springframework.org/schema/context>

<http://www.springframework.org/schema/context/spring-context-3.1.xsd>>

```
<!-- Root Context: defines shared resources visible to all other web
components -->

<bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
    <property name="driverClassName"
        value="net.sf.log4jdbc.sql.jdbcapi.DriverSpy"></property>
    <property name="jdbcUrl"
        value="jdbc:log4jdbc:oracle:thin:@localhost:1521:XE">
    </property>
    <property name="username" value="admin"></property>
    <property name="password" value="1234"></property>
</bean>

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

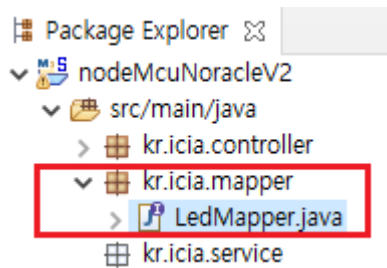
<bean id="sqlSessionFactory"
    class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource" />
</bean>

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

<context:component-scan
    base-package="kr.icia.service">
</context:component-scan>

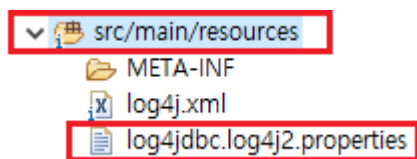
</beans>
```

// 아래 처럼 패키지를 생성하고, 인터페이스 생성.



```
package kr.icia.mapper;
```

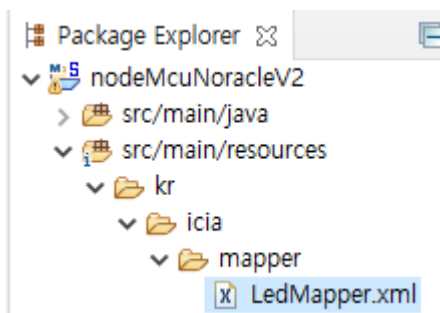
```
public interface LedMapper {  
    public void insert(String onOff);  
}
```



// 아래와 같이 작성

```
log4jdbc.spylogdelegator.name=net.sf.log4jdbc.log.slf4j.Slf4jSpyLogDelegator
```

// 아래와 같이 폴더를 3개 생성하고, xml 파일 생성.



// LedMapper.xml 에 아래와 같이 작성.

```
<?xml version="1.0" encoding="UTF-8"?>
```

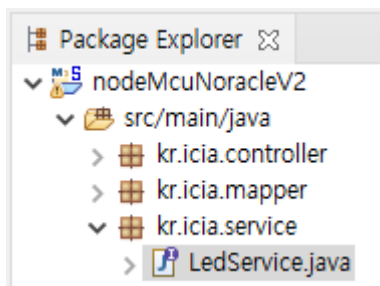
```

<!DOCTYPE mapper
PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="kr.icia.mapper.LedMapper">
    <insert id="insert">
        insert into tbl_led(lno,onOff) values
        (seq_led.nextval, #{onOff})
    </insert>
</mapper>

```

// 아래와 같이 인터페이스 생성



// 아래와 같이 LedService 인터페이스에 작성

```

package kr.icia.service;

public interface LedService {
    public void register(String onOff);
}

```

// 동일 패키지에 아래와 같이 서비스를 구현하는 클래스 생성.

New Java Class

**Java Class**  
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

---

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static

Superclass:

Interfaces:

// 아래와 같이 코드 작성.

```
package kr.icia.service;
```

```
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Service;
```

```
import kr.icia.mapper.LedMapper;
```

```
import lombok.Setter;
```

```
@Service
```

```
public class LedServiceImp implements LedService {
```

```
    @Setter(onMethod_ = @Autowired)
```

```
    private LedMapper mapper;
```

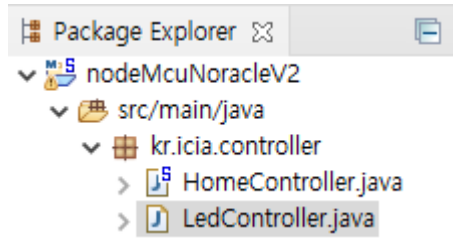
```
    @Override
```

```
    public void register(String onOff) {
```

```
        mapper.insert(onOff);
```

```
    }  
}
```

// 아래와 같이 컨트롤러 작성



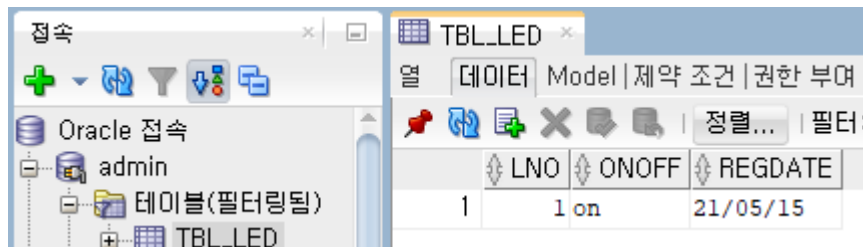
// 아래와 같이 코드 작성.

```
package kr.icia.controller;  
  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
  
import kr.icia.service.LedService;  
import lombok.AllArgsConstructor;  
  
@Controller  
@AllArgsConstructor  
public class LedController {  
    private LedService service;  
  
    @GetMapping("/node")  
    public void node(@RequestParam("led") String s) {  
        service.register(s);  
    }  
  
}
```

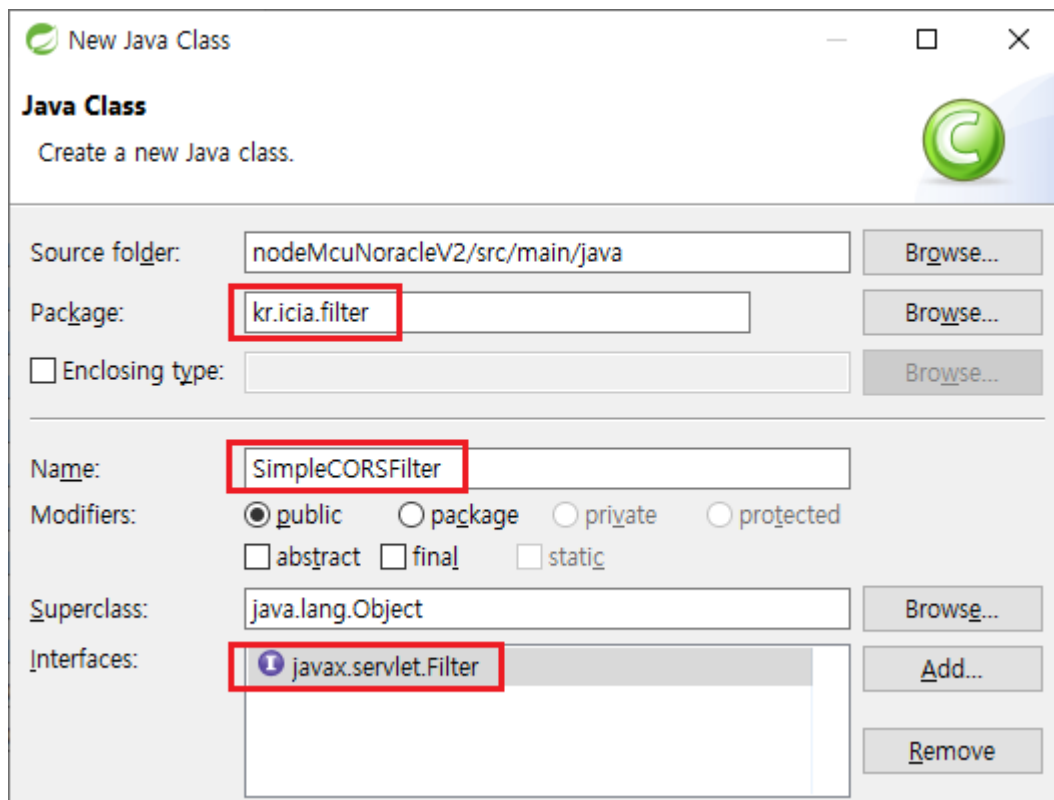


// 아래 주소 접근시, 디비에 데이터 등록됨.

<http://localhost:9090/controller/node?led=on>



// nodeMCU 웹서버에서 스프링 서버 접근시 cors 오류가 발생하므로, 필터를 추가해서 개선.



// 다음과 같이 클래스 작성.

```
package kr.icia.filter;
```

```
import java.io.IOException;
```

```
import javax.servlet.Filter;

import javax.servlet.FilterChain;

import javax.servlet.FilterConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import javax.servlet.http.HttpServletRequestResponse;
```

```
public class SimpleCORSFilter implements Filter {
```

```
    @Override
```

```
    public void init(FilterConfig filterConfig) throws ServletException {
```

```
        // TODO Auto-generated method stub
```

```
    }
```

```
    @Override
```

```
    public void doFilter(ServletRequest request, ServletResponse response, FilterChain chain)
```

```
        throws IOException, ServletException {
```

```
        // TODO Auto-generated method stub
```

```
        HttpServletResponse res = (HttpServletResponse) response;
```

```
        res.setHeader("Access-Control-Allow-Origin", "*");
```

```
        res.setHeader("Access-Control-Allow-Methods", "POST, GET, DELETE, PUT");
```

```
        res.setHeader("Access-Control-Max-Age", "3600");
```

```
        res.setHeader("Access-Control-Allow-Headers", "x-requested-with, origin,  
content-type, accept");
```

```
        chain.doFilter(request, response);
```

```
    }
```

```
    @Override
```

```
    public void destroy() {
```

```
        // TODO Auto-generated method stub
```

```
    }
```

```
}
```

```
// web.xml 에 한글필터와 cors 필터 적용.
```

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
```

```
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee  
https://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
```

```
    <!-- 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>

<!-- Processes application requests -->
<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>

<!-- 한글 필터 시작 -->
<filter>
    <filter-name>encoding</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>encoding</filter-name>
```

```
<servlet-name>appServlet</servlet-name>
```

```
<url-pattern>/*</url-pattern>
```

```
</filter-mapping>
```

```
<!-- 한글 필터 끝 -->
```

```
<filter>
```

```
<filter-name>SimpleCORSFilter</filter-name>
```

```
<filter-class>kr.icia.filter.SimpleCORSFilter</filter-class>
```

```
</filter>
```

```
<filter-mapping>
```

```
<filter-name>SimpleCORSFilter</filter-name>
```

```
<url-pattern>/*</url-pattern>
```

```
</filter-mapping>
```

```
</web-app>
```

// 정보를 공유해 주신 여러분 감사합니다.

// mysql 연동은 아래 링크를 참조 바랍니다.

<https://cafe.naver.com/arduinostory/112154>

