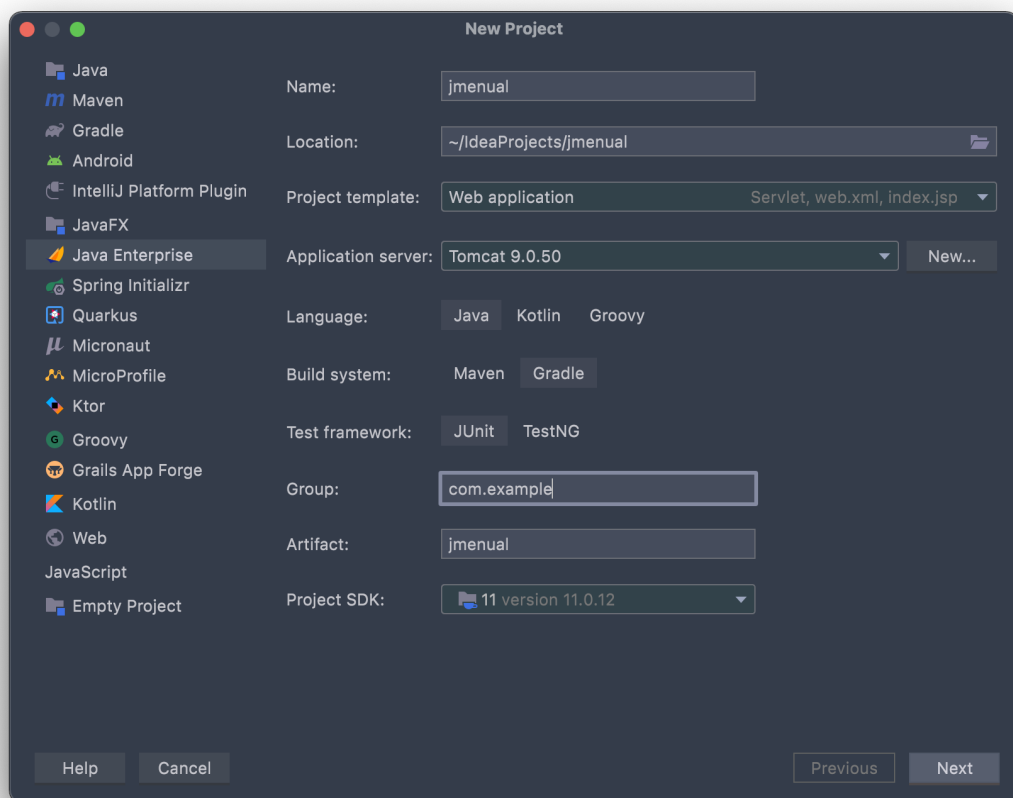
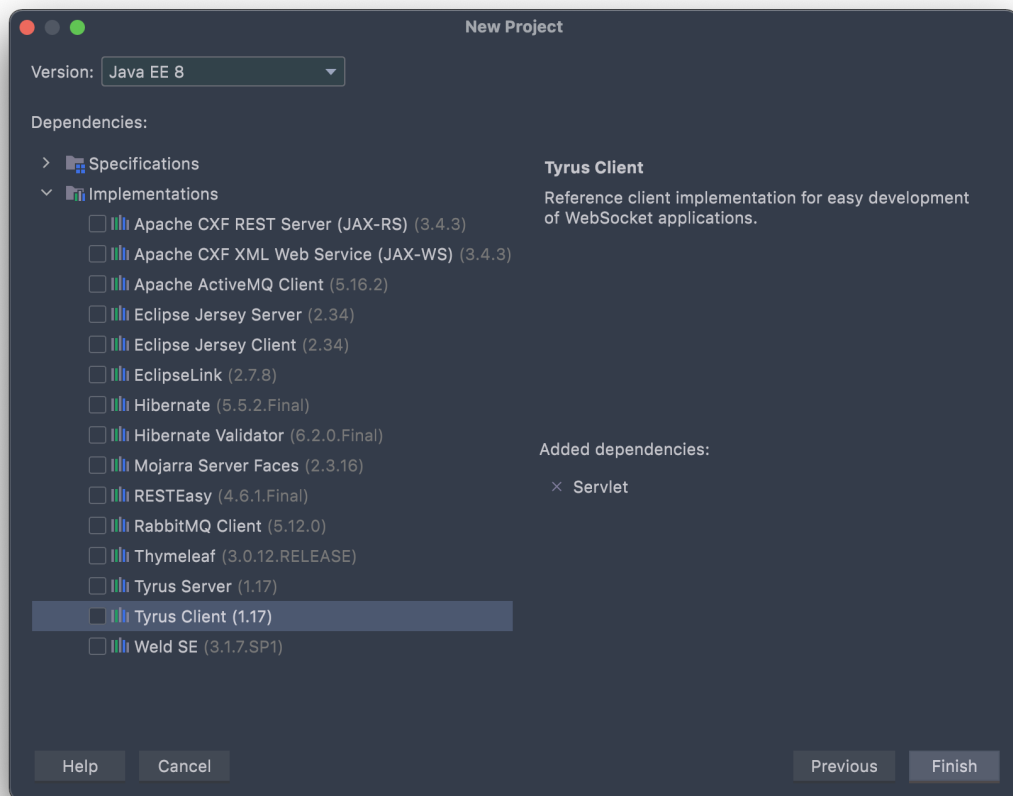


Spring Java 설정 메뉴얼

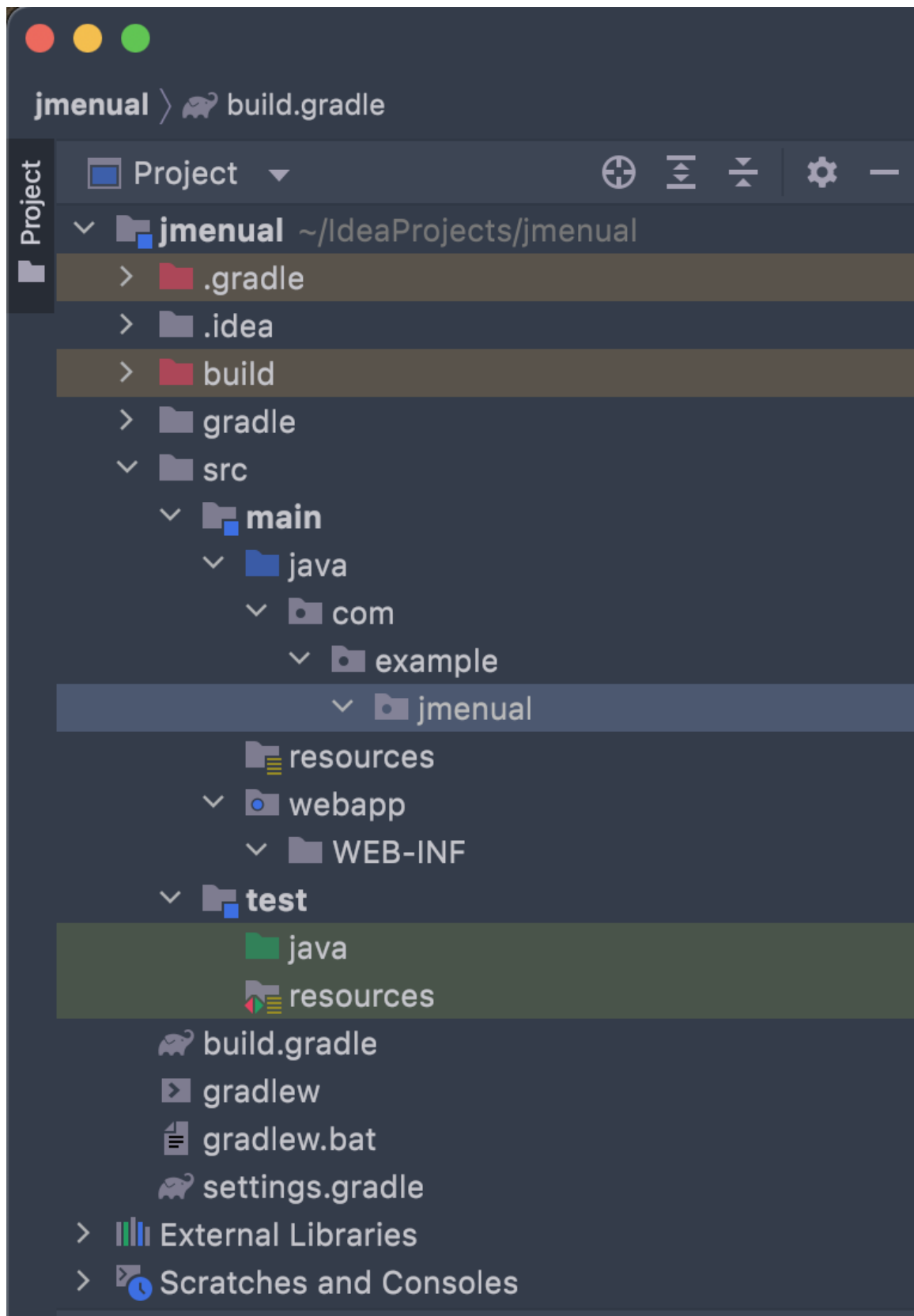
<Spring Java 연결>

1. 프로젝트 생성





2. HelloServlet.java, index.xml, web.xml 파일 삭제



3. 라이브러리 추가

/main/resources/log4j2.xml 추가

```
implementation group: 'org.springframework', name: 'spring-context', version: '5.3.9'
implementation group: 'org.springframework', name: 'spring-webmvc', version: '5.3.9'
implementation group: 'org.springframework', name: 'spring-test', version: '5.3.9'

implementation group: 'javax.servlet', name: 'jstl', version: '1.2'

compileOnly group: 'org.projectlombok', name: 'lombok', version: '1.18.20'
testCompileOnly group: 'org.projectlombok', name: 'lombok', version: '1.18.20'

annotationProcessor group: 'org.projectlombok', name: 'lombok', version: '1.18.20'
testAnnotationProcessor group: 'org.projectlombok', name: 'lombok', version: '1.18.20'

implementation group: 'org.apache.logging.log4j', name: 'log4j-core', version: '2.14.0'
implementation group: 'org.apache.logging.log4j', name: 'log4j-api', version: '2.14.0'

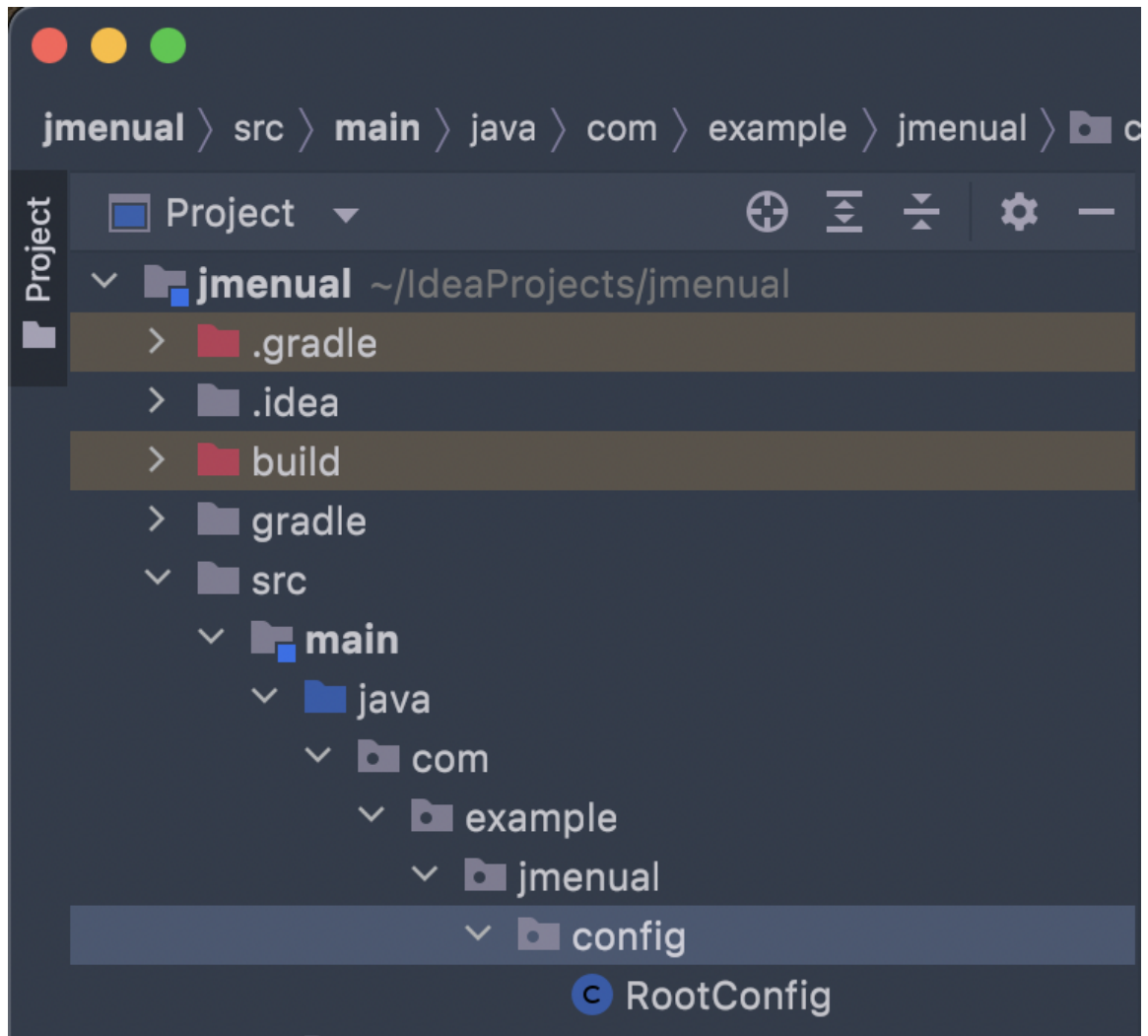
implementation group: 'mysql', name: 'mysql-connector-java', version: '8.0.26'

implementation group: 'com.zaxxer', name: 'HikariCP', version: '5.0.0'

// https://mvnrepository.com/artifact/org.mybatis/mybatis
implementation group: 'org.mybatis', name: 'mybatis', version: '3.5.7'
implementation group: 'org.mybatis', name: 'mybatis-spring', version: '2.0.6'
implementation group: 'org.springframework', name: 'spring-jdbc', version: '5.3.9'
implementation group: 'org.springframework', name: 'spring-tx', version: '5.3.9'

// https://mvnrepository.com/artifact/org.slf4j/slf4j-api
implementation group: 'org.slf4j', name: 'slf4j-api', version: '1.7.32'
implementation group: 'org.slf4j', name: 'slf4j-simple', version: '1.7.32'
```

4. /jex00/config/RootConfig.java 생성



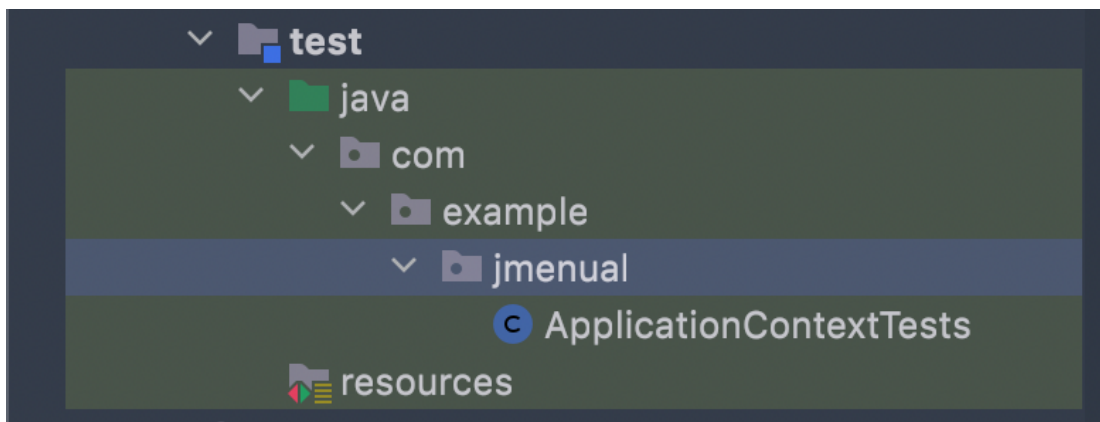
5. RootConfig.java

`@Configuration` : xml로 `<bean>` 태그를 설정해줬던 것을 자바 코드로 설정해 주는 것

```
build.gradle (jmenual) x RootConfig.java x
2
3 import org.springframework.context.annotation.Bean;
4 import org.springframework.context.annotation.Configuration;
5
6 import java.util.ArrayList;
7
8 @Configuration
9 public class RootConfig {
10
11     @Bean(name = "names")
12     public ArrayList<String> names(){
13
14         ArrayList<String> list = new ArrayList<>();
15         list.add("AAA");
16         list.add("BBB");
17         list.add("CCC");
18         return list;
19     }
}
```

6. Test 해주기

- test 폴더에 패키지 생성



- test 코드 작성

`@ExtendWith(SpringExtension.class)`

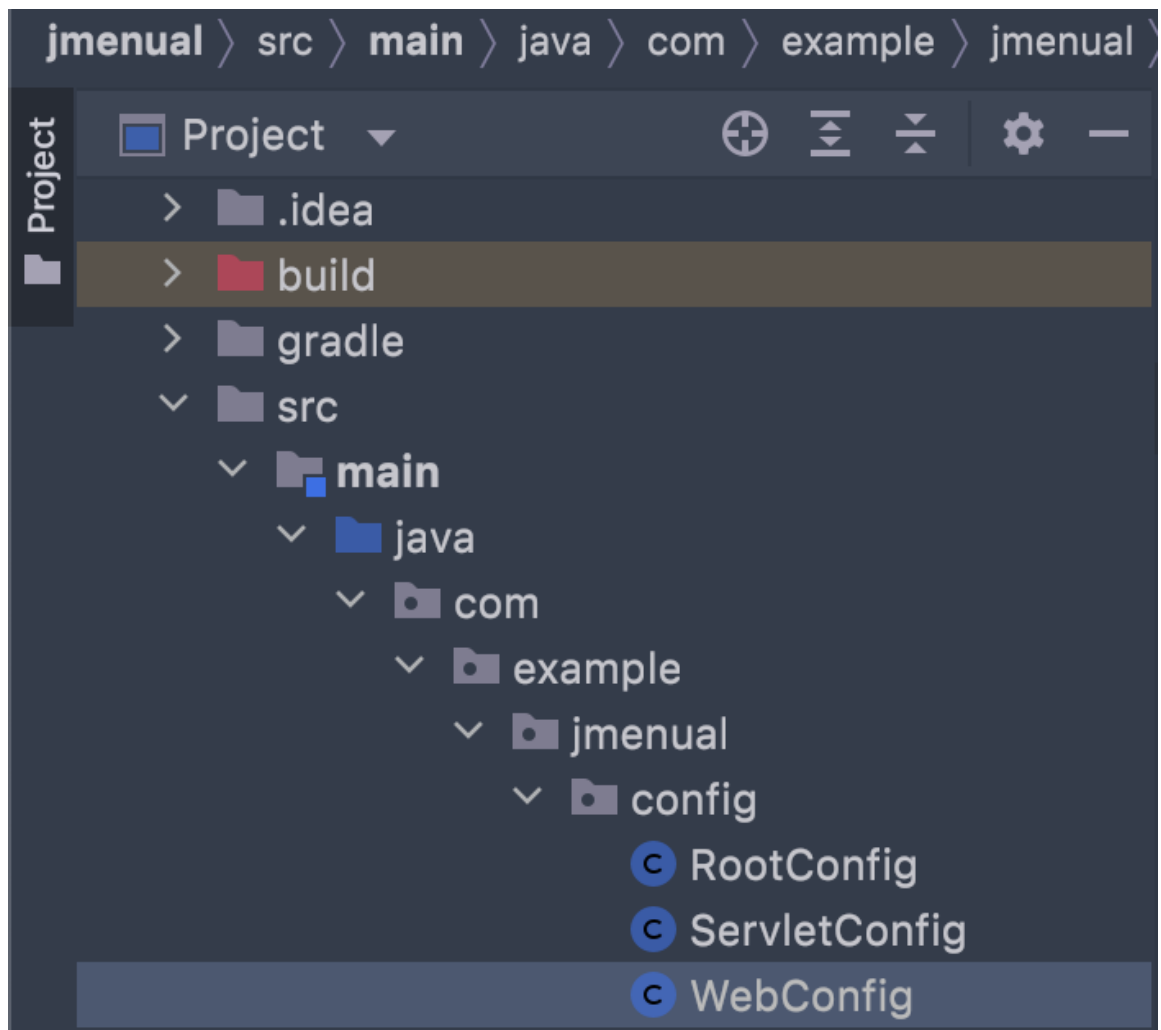
`@ContextConfiguration(classes = {RootConfig.class})` : 설정파일 설정

```
jmenual > src > test > java > com > example > jmenual > ApplicationContextTests > test1
Project build.gradle (jmenual) x RootConfig.java x ApplicationContextTests.java x
13
14 @Log4j2
15 @ExtendWith(SpringExtension.class)
16 @ContextConfiguration(classes = {RootConfig.class})
17 public class ApplicationContextTests {
18
19     @Autowired
20     ApplicationContext applicationContext;
21
22     @Autowired
23     ArrayList<String> names;
24
25     @Test
26     public void test1() {
27         log.info("-----");
28         log.info("-----");
29         log.info(applicationContext); //객체 생성 확인
30         log.info(names);
31         log.info("-----");
32         log.info("-----");
33     }
34
```

- 로그 확인

```
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] -----
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] -----
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] org.springframework.context.support.GenericApplicationContext
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] [AAA, BBB, CCC]
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] -----
2021-08-30 05:43:36 INFO [com.example.jmenual.ApplicationContextTests] -----
BUILD SUCCESSFUL in 6s
4 actionable tasks: 4 executed
5:43:37 오후: Task execution finished ':test --tests "com.example.jmenual.ApplicationContextTests.test1"'.
```

7. ServletConfig.java / WebCoinfig.java 생성




```
jmenual > src > main > java > com > example > jmenual > config > ServletConfig > addResourceHandlers
build.gradle (jmenual) x RootConfig.java x ApplicationContextTests.java x ServletConfig.java x WebConfig.java x
9 import org.springframework.web.servlet.view.JstlView;
10
11 @ComponentScan(basePackages = {"com.example.controller"})
12 @EnableWebMvc
13 public class ServletConfig implements WebMvcConfigurer {
14
15     @Override
16     public void configureViewResolvers(ViewResolverRegistry registry) {
17
18         InternalResourceViewResolver bean = new InternalResourceViewResolver();
19         bean.setViewClass(JstlView.class);
20         bean.setPrefix("/WEB-INF/views/");
21         bean.setSuffix(".jsp");
22         registry.viewResolver(bean);
23     }
24
25     @Override
26     public void addResourceHandlers(ResourceHandlerRegistry registry) {
27
28         registry.addResourceHandler(...pathPatterns: "/resources/**").addResourceLocations("/resources/");
29         |
30     }
31 }
```

→ ServletConfig.java 코드

```

@Log4j2
public class WebConfig extends AbstractAnnotationConfigDispatcherServletInitializer {
    @Override
    protected Class<?>[] getRootConfigClasses() {
        log.info("1-----");
        log.info("1-----");

        return new Class[]{RootConfig.class};
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        log.info("2-----");
        log.info("2-----");

        return new Class[]{ServletConfig.class};
    }

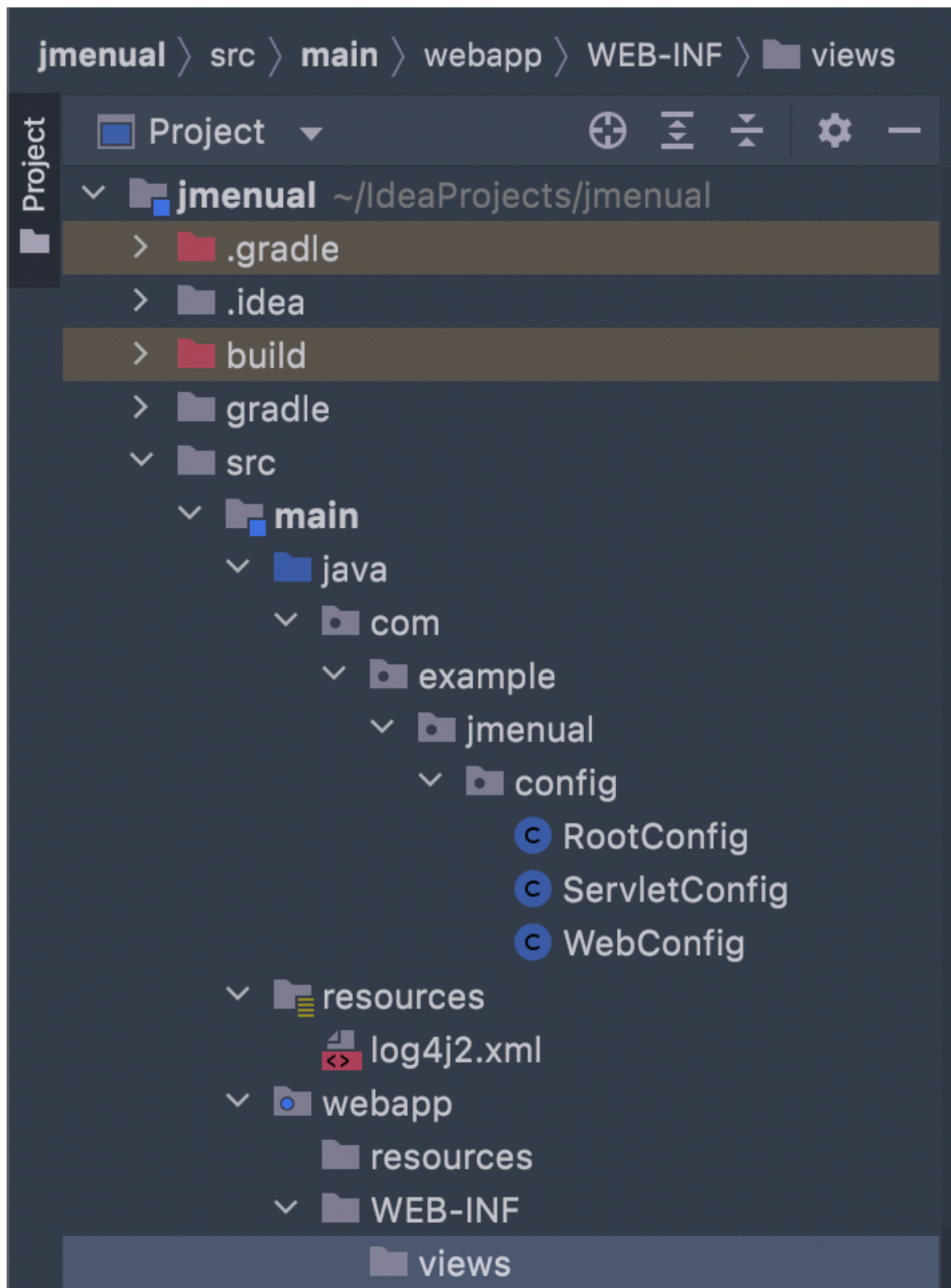
    @Override
    protected String[] getServletMappings() {
        return new String[]{"/"};
    }
}

```

→ [WebConfig.java](#) 코드

8. /webapp/resouces 폴더 생성

/WEB-INF/view 폴더 생성



- 로그확인
객체 생성되고 실행되는지 확인함

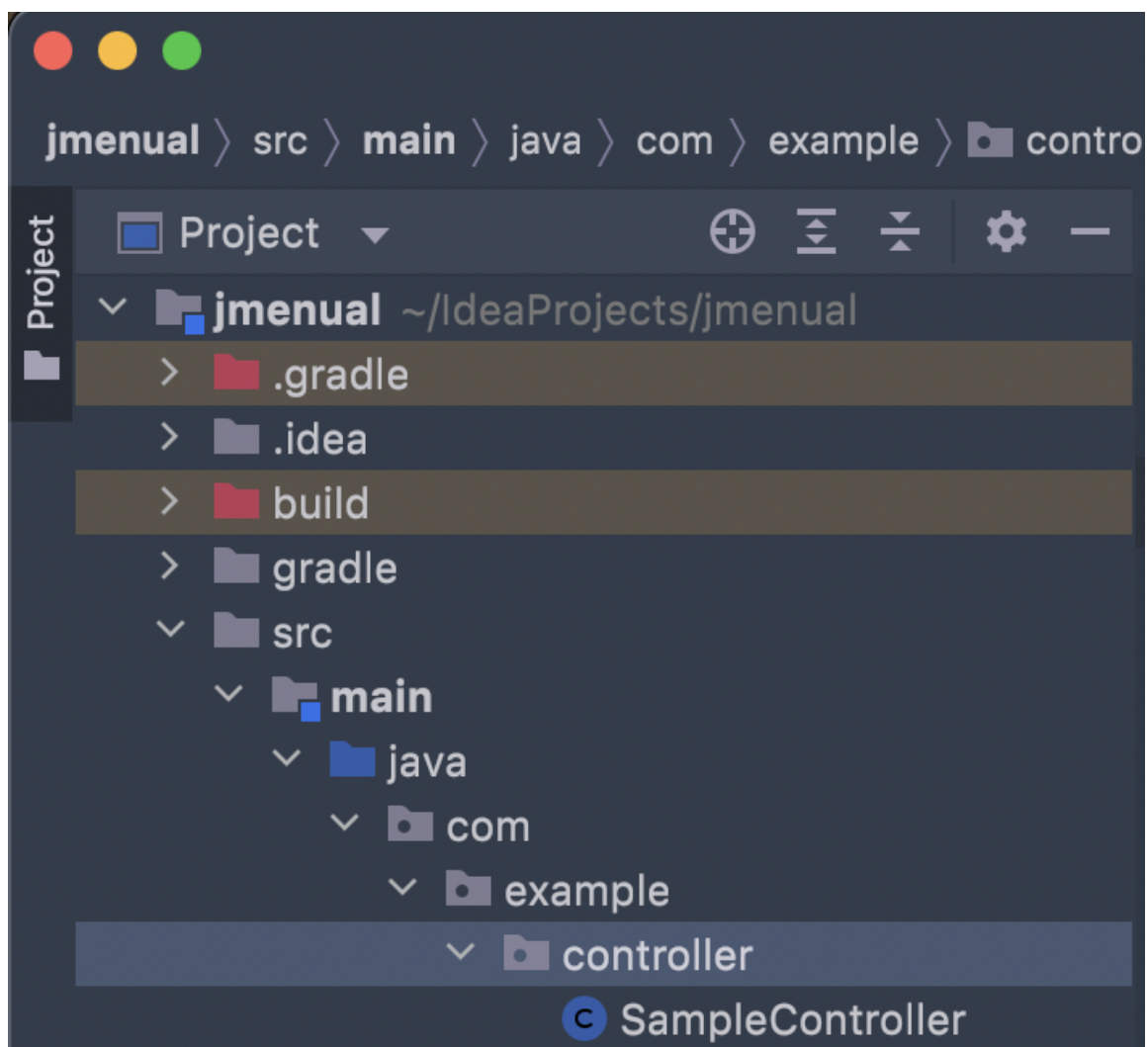
```

2021-08-30 05:55:12 INFO [com.example.jmenvul.config.WebConfig] 1-----
2021-08-30 17:55:12,083 RMI TCP Connection(2)-127.0.0.1 DEBUG AsyncLogger.ThreadNameStrateg
2021-08-30 17:55:12,083 RMI TCP Connection(2)-127.0.0.1 DEBUG org.apache.logging.log4j.core
2021-08-30 05:55:12 INFO [com.example.jmenvul.config.WebConfig] 1-----
2021-08-30 05:55:12 INFO [com.example.jmenvul.config.WebConfig] 2-----
2021-08-30 05:55:12 INFO [com.example.jmenvul.config.WebConfig] 2-----

```

9. 컨트롤러 생성

/controller/SampleController.java 생성



- 코드 작성

```

@Controller
@Log4j2
public class SampleController {

    @GetMapping("/hello")
    public void hello(){

        log.info("hello.....");
        log.info("hello.....");
        log.info("hello.....");
        log.info("hello.....");
        log.info("hello.....");

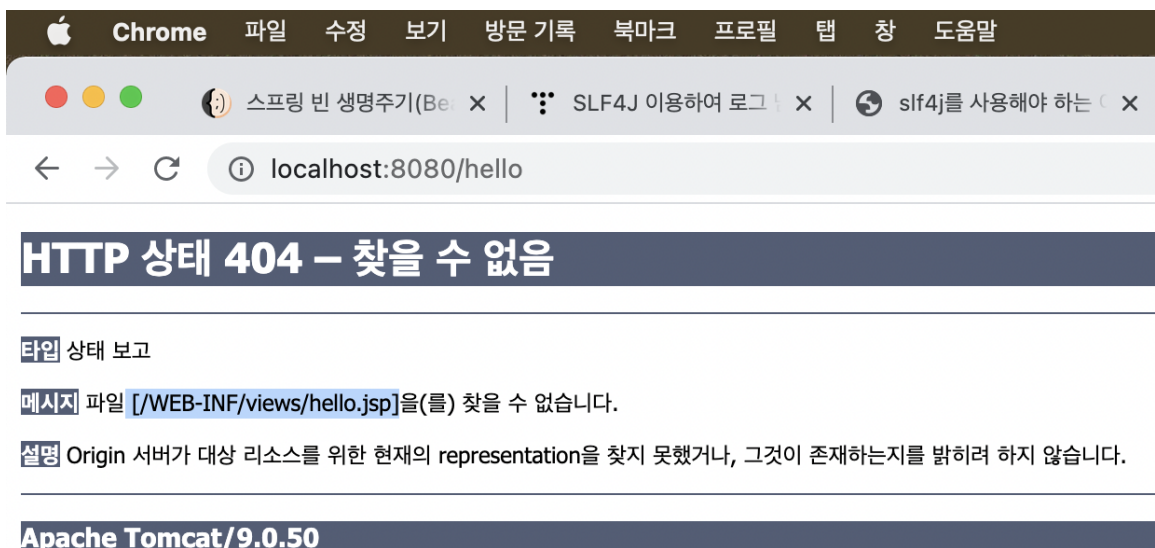
    }

}

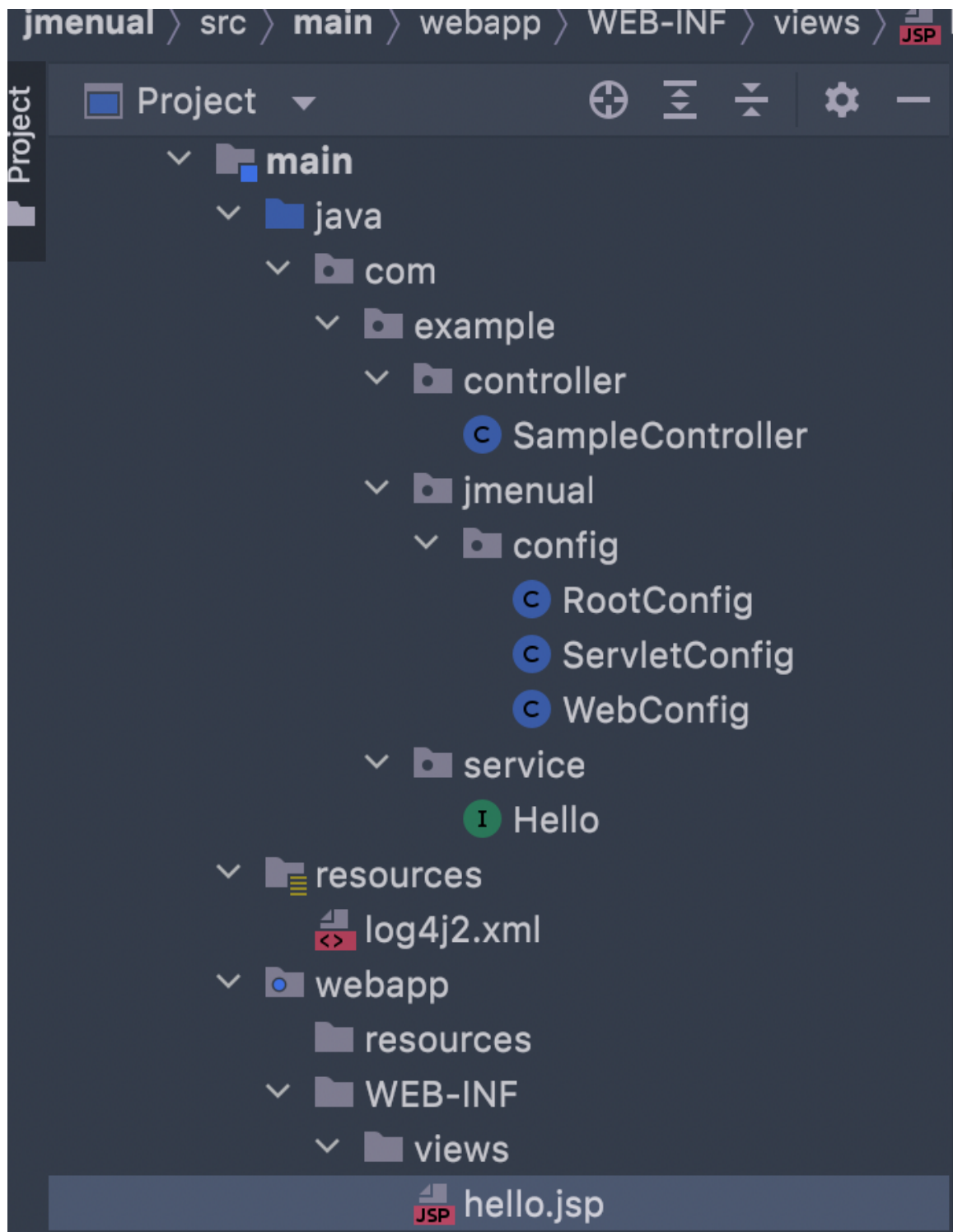
```

→ [SampleController.java](#) 코드

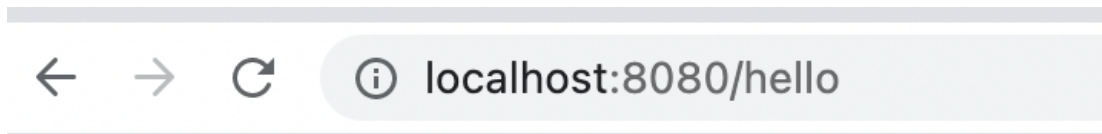
- 서버 실행 후 경로 확인



10. /views/hello.jsp 생성



- 화면출력

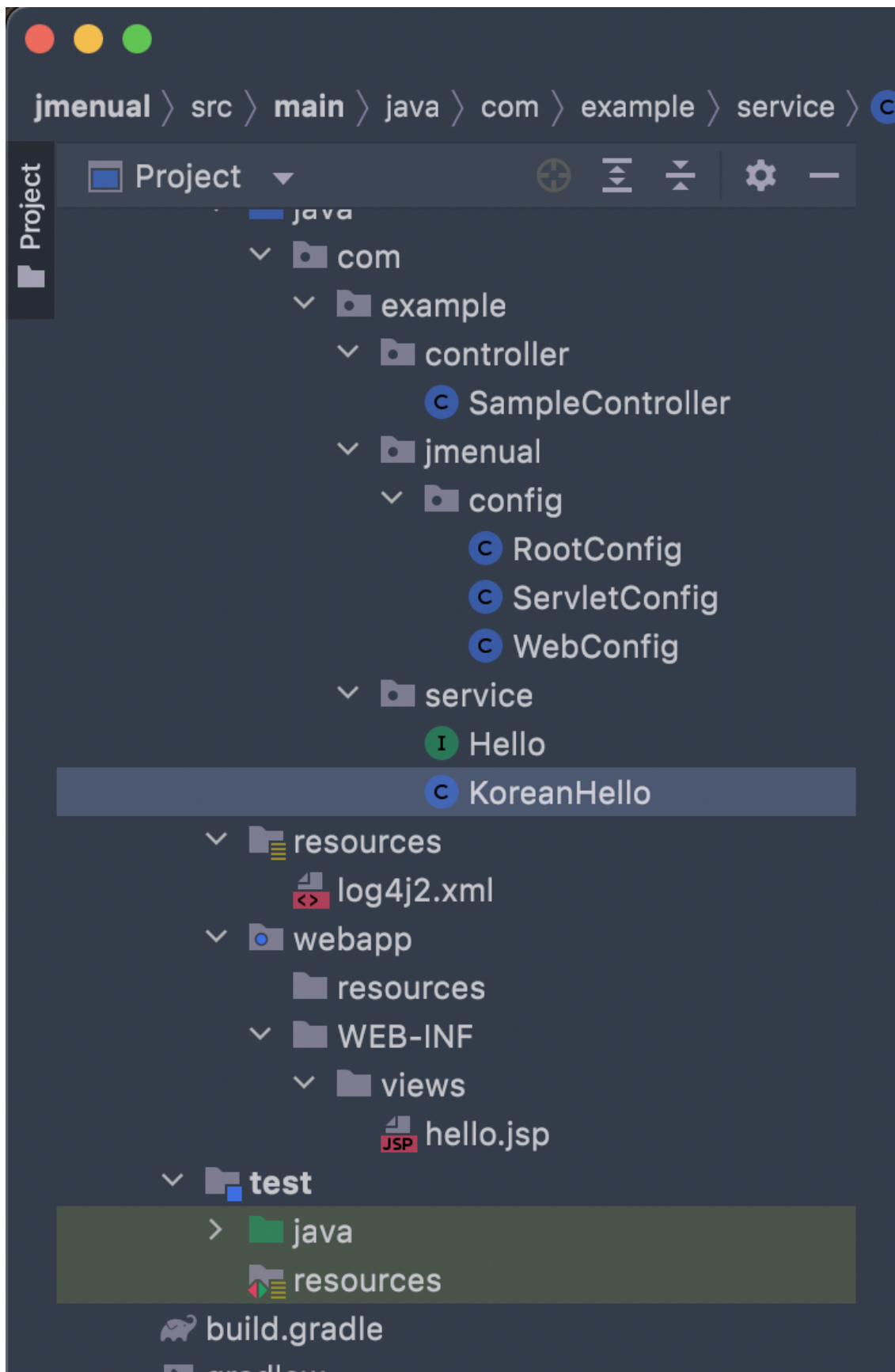


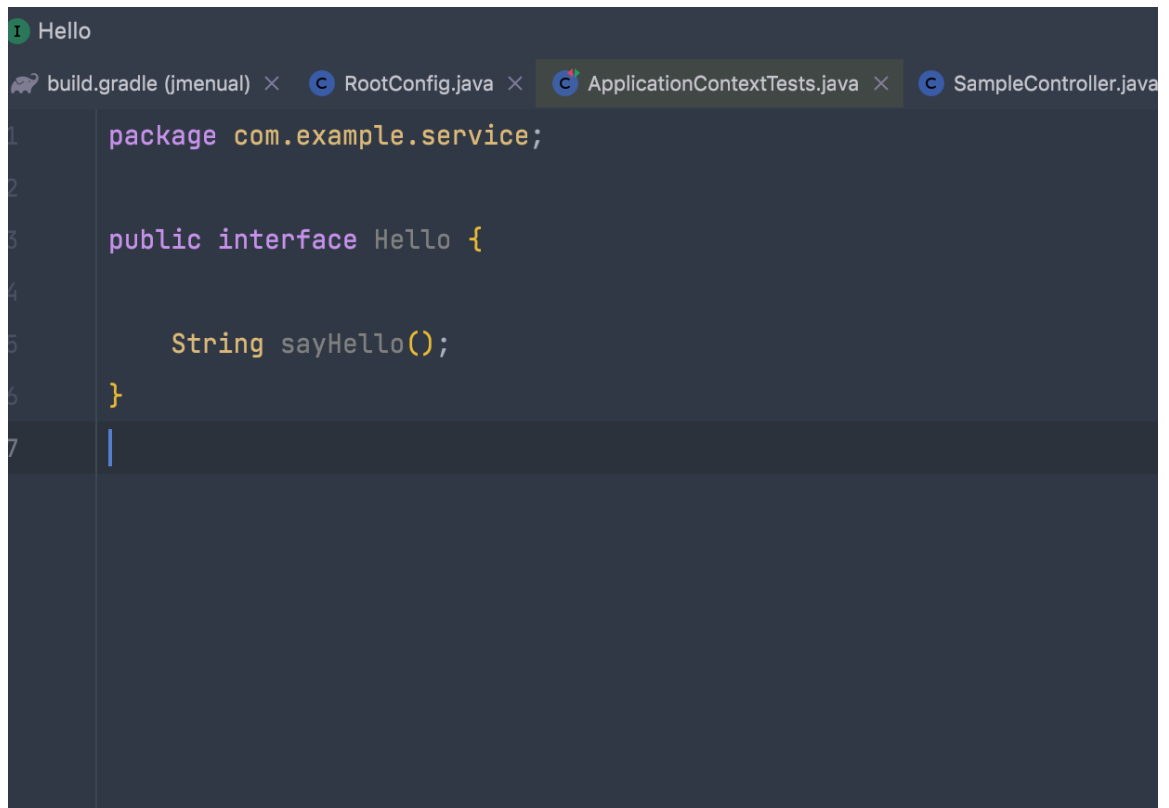
Hello JSP File

10. 서비스 생성

/service/Hello.java 인터페이스 생성

/service/KoreanHello.java 생성





The screenshot shows an IDE window titled 'Hello'. The tab bar at the top contains four tabs: 'build.gradle (jmenual)', 'RootConfig.java', 'ApplicationContextTests.java' (which is the active tab), and 'SampleController.java'. The main editor area displays the following Java code:

```
1 package com.example.service;
2
3
4
5 public interface Hello {
6
7     String sayHello();
8 }
9
```

→ [Hello.java](#) 인터페이스 코드

```
jmenual - KoreanHello.java [jmenual.main]

C KoreanHello
C KoreanHello.java x

1 package com.example.service;
2
3 import org.springframework.stereotype.Service;
4
5 @Service
6 public class KoreanHello implements Hello{
7
8     @Override
9     public String sayHello(){
10         return "An Nenung ha se yo";
11     }
12 }
13
```

→ [KoreanHello.java 코드](#)

12. 서비스와 RootConfig 연결 확인

- `@ComponentScan(basePackages = {"com.example.service"})` : 패키지 스캔해서 객체연결

```

1 @Configuration
2 @Log4j2
3 @ComponentScan(basePackages = {"com.example.service"})
4 public class RootConfig {
5
6     @Bean(name = "names")
7     public ArrayList<String> names(){
8
9         ArrayList<String> list = new ArrayList<>();
10        list.add("AAA");
11        list.add("BBB");
12        list.add("CCC");
13        return list;
14    }
15 }

```

→ [RootConfig.java](#)

- `@Autowired` 사용해서 객체 연결 후 로그 찍어주기

```
jmenual - SampleController.java [jmenual.main]
> C SampleController > m hello
C KoreanHello.java x C SampleController.java x I Hello.java x
8
9 @Controller
10 @Log4j2
11 public class SampleController {
12
13     @Autowired
14     private Hello hello;
15
16     @GetMapping("/hello")
17     public void hello(){
18
19         log.info("hello.....");
20         log.info("hello.....");
21         log.info(hello);
22         log.info("hello.....");
23         log.info("hello.....");
24
25     }
```

→ SampleController.java

- 서버 실행 후 /hello 경로 들어가서 로그 확인

```
INFO [com.example.controller.SampleController] hello.....
INFO [com.example.controller.SampleController] hello.....
INFO [com.example.controller.SampleController] com.example.service.KoreanHello@279e516c
INFO [com.example.controller.SampleController] hello.....
INFO [com.example.controller.SampleController] hello.....
```

<MyBatis 연결>

1. 히카리CP 데이터 소스 연결

```
@Configuration
@ComponentScan(basePackages = {"com.example.service"})
public class RootConfig {

    @Bean
    public DataSource dataSource(){
        HikariConfig config = new HikariConfig();
        config.setDriverClassName("com.mysql.cj.jdbc.Driver");
        config.setJdbcUrl("jdbc:mysql://localhost:3306/springdb");
        config.setUsername("springuser");
        config.setPassword("springuser");

        HikariDataSource dataSource = new HikariDataSource(config);

        return dataSource;
    }
}
```

→ RootConfig.java

2. Test 해보기

```

@Autowired
DataSource dataSource;

@Test
public void testConnection() throws Exception{

    Connection con = dataSource.getConnection();
    log.info(con);
    con.close();
}

```

→ ApplicationContextTests.java

- HikariCP 연결 proxy 확인

```

com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Starting...
com.zaxxer.hikari.pool.HikariPool - HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@206ae9a4
com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Start completed.
INFO [com.example.jmenvul.ApplicationContextTests] HikariProxyConnection@1472505461 wrapping com.mysql.cj.jdbc.ConnectionImpl@206ae9a4
WARNHook] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Shutdown initiated...
WARNHook] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Shutdown completed.

```

2. MyBatis 라이브러리 추가 확인 후 연결

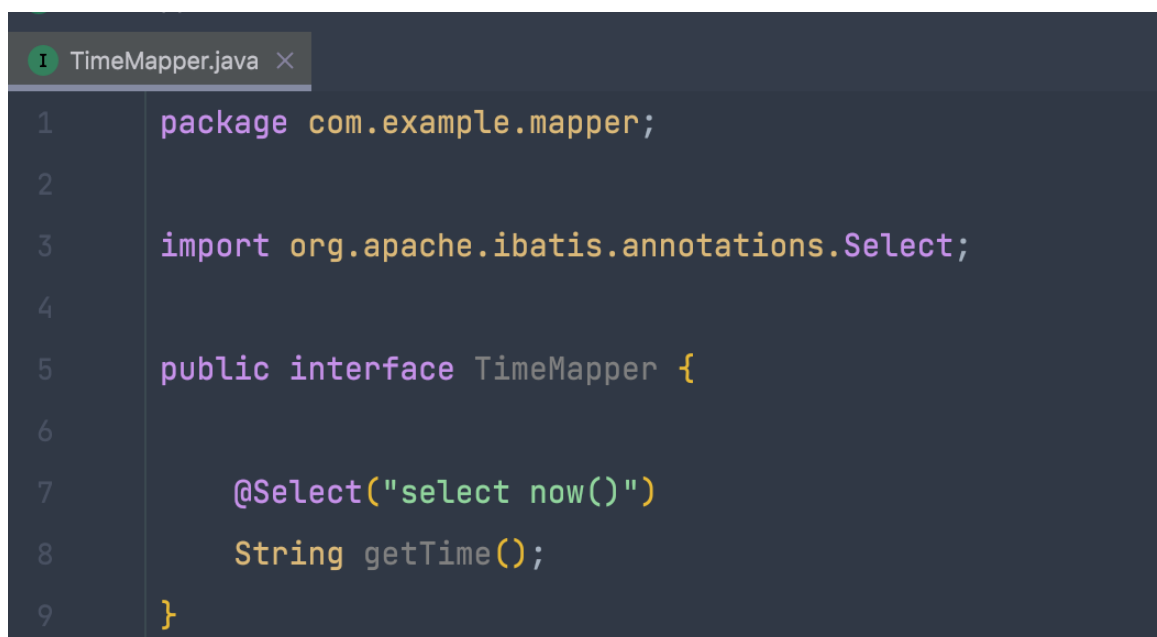
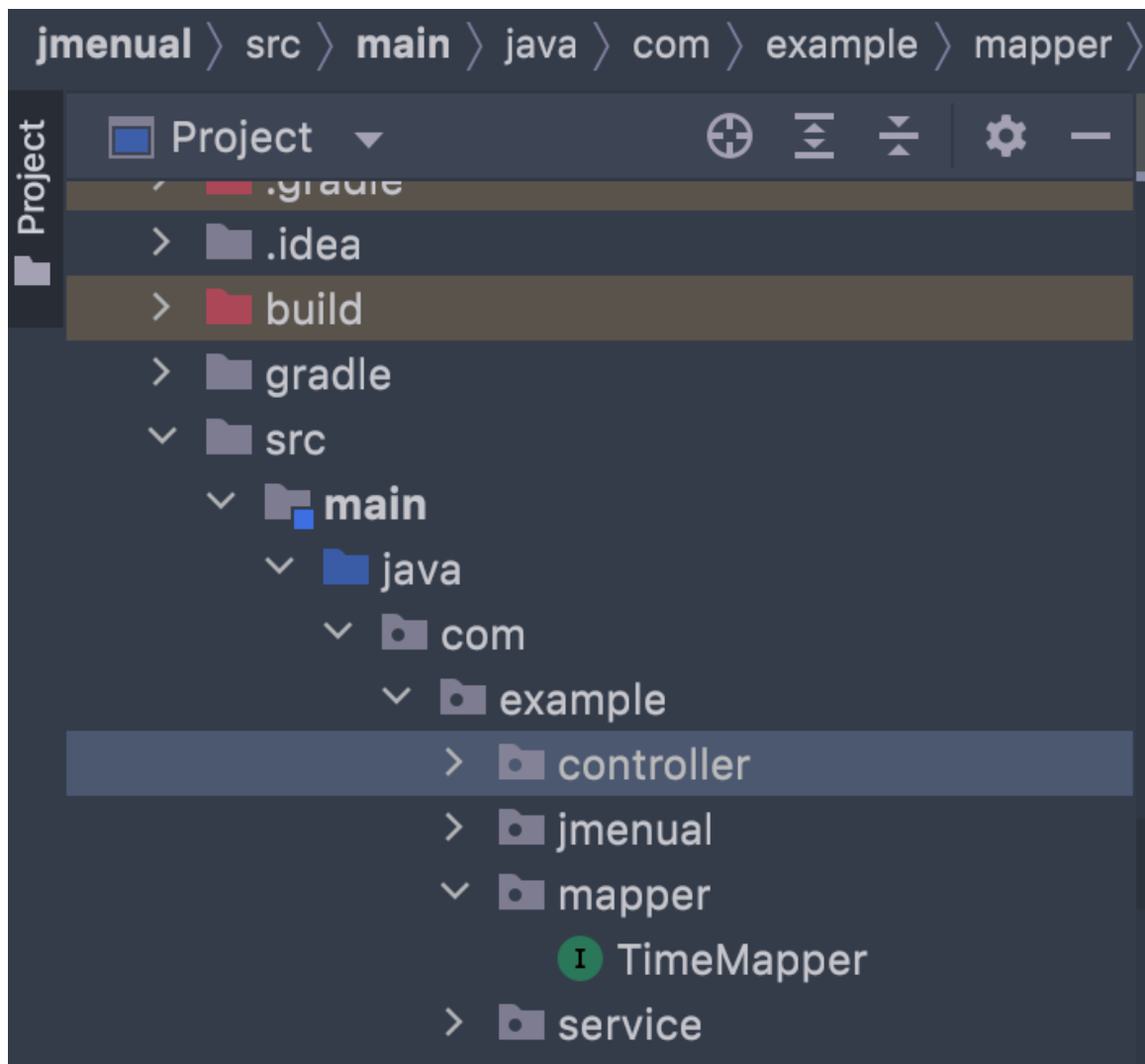
```

0 @Bean
1 public SqlSessionFactory sqlSessionFactory() throws Exception{
2     SqlSessionFactoryBean sqlSessionFactoryBean = new SqlSessionFactoryBean();
3     sqlSessionFactoryBean.setDataSource(dataSource());
4     return sqlSessionFactoryBean.getObject();
5 }

```

→ RootConfig.java

2. /mapper/TimeMapper.java 생성



→ TimeMapper.java

- `@MapperScan` 사용해서 경로 지정 후 mapper 연결

```
@Configuration
@ComponentScan(basePackages = {"com.example.service"})
@MapperScan(basePackages = {"com.example.mapper"})
public class RootConfig {
```

→ RootConfig.java

5. Test 해보기

```
@Log4j2
@ExtendWith(SpringExtension.class)
@ContextConfiguration(classes = {RootConfig.class})
public class TimeMapperTests {

    @Autowired
    TimeMapper timeMapper;

    @Test
    public void testGetTime1(){

        log.info(timeMapper);
        log.info(timeMapper.getTime());
    }
}
```

- 연결된 로그 확인

```
[Test worker] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Starting...
[Test worker] INFO com.zaxxer.hikari.pool.HikariPool - HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionIn
[Test worker] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Start completed.
2021-08-30 07:18:52 INFO [com.example.mapper.TimeMapperTests] org.apache.ibatis.binding.MapperProxy@327d8d04
2021-08-30 07:18:52 INFO [com.example.mapper.TimeMapperTests] 2021-08-30 19:18:52
[SpringContextShutdownHook] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Shutdown initiated...
[SpringContextShutdownHook] INFO com.zaxxer.hikari.HikariDataSource - HikariPool-1 - Shutdown completed.
```