狂神说SpringBoot07：整合JDBC

秦疆 狂神说 2020-03-14

SpringData简介

对于数据访问层，无论是 SQL(关系型数据库) 还是 NOSQL(非关系型数据库)，Spring Boot 底层都是采用 Spring Data 的方式进行统一处理。

Spring Boot 底层都是采用 Spring Data 的方式进行统一处理各种数据库，Spring Data 也是 Spring 中与 Spring Boot、Spring Cloud 等齐名的知名项目。

Sping Data 官网：https://spring.io/projects/spring-data

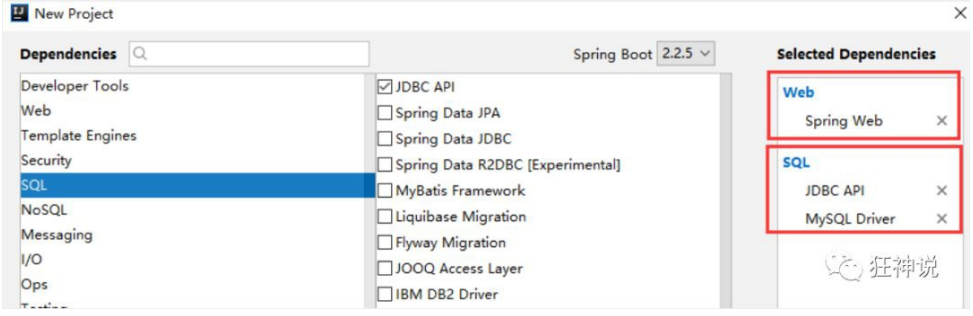
数据库相关的启动器 ：可以参考官方文档：

https://docs.spring.io/spring-boot/docs/2.2.5.RELEASE/reference/htmlsingle/#using-boot-starter

整合JDBC

创建测试项目测试数据源

1、我去新建一个项目测试：springboot-data-jdbc ; 引入相应的模块！基础模块



2、项目建好之后，发现自动帮我们导入了如下的启动器：

```
<dependency>    <groupId>org.springframework.boot</groupId>    <artifactId>spring-boot-starter-jdbc</artifactId></dependency><dependency>    <groupId>mysql</groupId>    <artifactId>mysql-connector-java</artifactId>    <scope>runtime</scope></dependency>
```

3、编写yaml配置文件连接数据库；

```
spring:  datasource:    username: root    password: 123456    #?serverTimezone=UTC解决时区的报错    url: jdbc:mysql://localhost:3306/springboot?serverTimezone=UTC&useUnicode=true&characterEncoding=utf-8    driver-class-name: com.mysql.cj.jdbc.Driver
```

4、配置完这一些东西后，我们就可以直接去使用了，因为SpringBoot已经默认帮我们进行了自动配置；去测试类测试一下

```
@SpringBootTestclass SpringbootDataJdbcApplicationTests {
    //DI注入数据源    @Autowired    DataSource dataSource;
    @Test    public void contextLoads() throws SQLException {         //看一下默认数据源         System.out.println(dataSource.getClass());         //获得连接
```

```
        Connection connection =    dataSource.getConnection();         System
.out.println(connection);         //关闭连接         connection.close();     }}
```

结果：我们可以看到他默认给我们配置的数据源为 : class com.zaxxer.hikari.HikariDataSource ， 我们并没有手动配置

我们来全局搜索一下，找到数据源的所有自动配置都在 ：DataSourceAutoConfiguration文件：

```
@Import(    {Hikari.class, Tomcat.class, Dbcp2.class, Generic.class, DataSo
urceJmxConfiguration.class})protected static class PooledDataSourceConfigur
ation {    protected PooledDataSourceConfiguration() {    }}
```

这里导入的类都在 DataSourceConfiguration 配置类下，可以看出 Spring Boot 2.2.5 默认使用 HikariDataSource 数据源，而以前版本，如 Spring Boot 1.5 默认使用 org.apache.tomcat.jdbc.pool.DataSource 作为数据源；

HikariDataSource 号称 Java WEB 当前速度最快的数据源，相比于传统的 C3P0 、DBCP、Tomcat jdbc 等连接池更加优秀；

可以使用 spring.datasource.type 指定自定义的数据源类型，值为 要使用的连接池实现的完全限定名。

关于数据源我们并不做介绍，有了数据库连接，显然就可以 CRUD 操作数据库了。但是我们需要先了解一个对象 JdbcTemplate

JDBCTemplate

1、有了数据源(com.zaxxer.hikari.HikariDataSource)，然后可以拿到数据库连接(java.sql.Connection)，有了连接，就可以使用原生的 JDBC 语句来操作数据库；

2、即使不使用第三方第数据库操作框架，如 MyBatis等，Spring 本身也对原生的JDBC 做了轻量级的封装，即JdbcTemplate。

3、数据库操作的所有 CRUD 方法都在 JdbcTemplate 中。

4、Spring Boot 不仅提供了默认的数据源，同时默认已经配置好了 JdbcTemplate 放在了容器中，程序员只需自己注入即可使用

5、JdbcTemplate 的自动配置是依赖 org.springframework.boot.autoconfigure.jdbc 包下的 JdbcTemplateConfiguration 类

JdbcTemplate主要提供以下几类方法：

- execute方法：可以用于执行任何SQL语句，一般用于执行DDL语句；

- update方法及batchUpdate方法：update方法用于执行新增、修改、删除等语句；batchUpdate方法用于执行批处理相关语句；
- query方法及queryForXXX方法：用于执行查询相关语句；
- call方法：用于执行存储过程、函数相关语句。

测试

编写一个Controller，注入 jdbcTemplate，编写测试方法进行访问测试；

```
package com.kuang.controller;
import org.springframework.beans.factory.annotation.Autowired;import org.sp
ringframework.jdbc.core.JdbcTemplate;import org.springframework.web.bind.an
notation.GetMapping;import org.springframework.web.bind.annotation.PathVari
able;import org.springframework.web.bind.annotation.RequestMapping;import o
rg.springframework.web.bind.annotation.RestController;
import java.util.Date;import java.util.List;import java.util.Map;
@RestController@RequestMapping("/jdbc")public class JdbcController {
    /**     * Spring Boot 默认提供了数据源，默认提供了 org.springframework.jdbc.c
```

```
ore.JdbcTemplate     * JdbcTemplate 中会自己注入数据源，用于简化 JDBC操作     * 还能避免一些常见的错误,使用起来也不用再自己来关闭数据库连接     */    @Autowired    JdbcTemplate jdbcTemplate;
    //查询employee表中所有数据    //List 中的1个 Map 对应数据库的 1行数据    //Map 中的 key 对应数据库的字段名，value 对应数据库的字段值    @GetMapping("/list")    public List<Map<String, Object>> userList(){        String sql = "select * from employee";        List<Map<String, Object>> maps = jdbcTemplate.queryForList(sql);        return maps;    }        //新增一个用户    @GetMapping("/add")    public String addUser(){        //插入语句，注意时间问题        String sql = "insert into employee(last_name, email,gender,department,birth)" +
            " values ('狂神说','24736743@qq.com',1,101,'"+ new Date().toLocaleString() +"')";        jdbcTemplate.update(sql);        //查询        return "addOk";    }
    //修改用户信息    @GetMapping("/update/{id}")    public String updateUser(@PathVariable("id") int id){        //插入语句        String sql = "update employee set last_name=?,email=? where id="+id;        //数据        Object[] objects = new Object[2];        objects[0] = "秦疆";        objects[1] = "24736743@sina.com";        jdbcTemplate.update(sql,objects);        //查询
    return "updateOk";    }
    //删除用户    @GetMapping("/delete/{id}")    public String delUser(@PathVariable("id") int id){        //插入语句        String sql = "delete from employee where id=?";        jdbcTemplate.update(sql,id);        //查询
    return "deleteOk";    }    }
```

测试请求，结果正常；

到此，CURD的基本操作，使用 JDBC 就搞定了。