38_停下脚步：先来看看原生 JDBC 如何进行数据库操作

1、开篇

不知不觉来到了第五周的课程，在第四周我们介绍了日志和异常处理的功能，它们都属系统级别的功能。数据库访问作为另外一个系统级别的功能会在后面的开发过程中用到，因此这周会给大家介绍这方面的内容。整体脉络会根据 JDBC、连接池、JDBCTemplate 以及 MyBatis 几部分内容展开，本节课主要介绍 JDBC 访问数据库的功能，今天的内容：

- JDBC 简介
- 上传服务
- 启动服务

2、JDBC 简介

JDBC 是 Java Database Connectivity 的简称，可以理解为 Java 数据库连接。为 Java 程序开发者使用和操作数据库所提供的统一的编程接口（API），由一组使用 Java 语言编写的类和接口组成。

JDBC 整个工作包括如下具体流程：

- 加载和注册数据库驱动程序
- 连接数据库，建立连接（获得 Connection 对象）
- 操作数据库（增查改删）：创建数据库操作对象（用于执行 SQL 语句）。这里会涉及到 Statement 对象或者 PrepareStatement 对象的创建，以及基于两者进行的执行 SQL 语句工作，从而获得并处理结果，这个结果通常放在 ResultSet 对象中并且返回。
- 释放资源：释放数据库连接资源。

3、配置 JDBC 访问数据

对 JDBC 进行简介以后，接下来就尝试如何配置 JDBC 的组件和环境。由于 JDBC 的组件包包含在 JDK 中因此不用额外引用，那么先来看看它的配置文件。如图 1

所示，在 jdbc.properties 文件中配置 JDBC 的基本信息，这里包括 MySQL 的驱动、数据库连接串（url）以及访问数据库的用户名和密码。

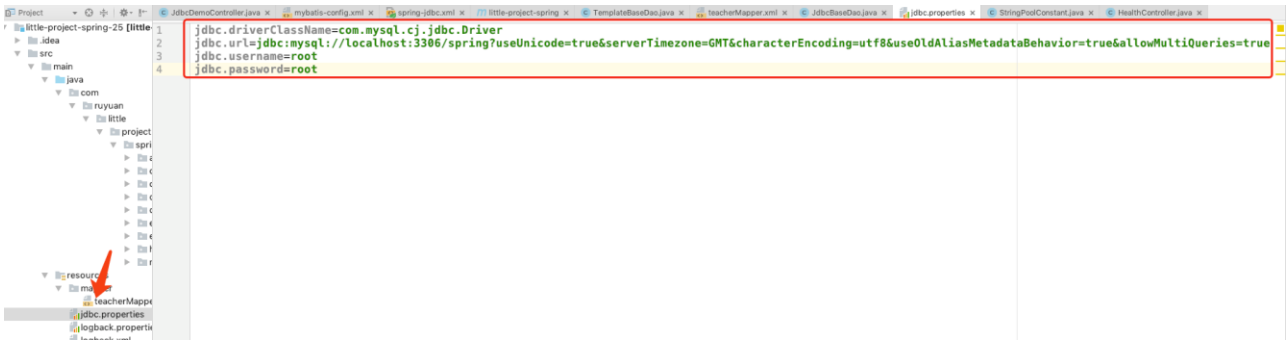

图 1 配置数据库连接信息

然后创建一个 JDBC 的访问类，如图 2 所示，在/spring/dao 目录下面创建 JdbcBaseDao 的类，该类需要对数据库驱动、url 地址、用户名和密码进行初始化，这个初始化的工作后续会通过 Spring IOC 完成。

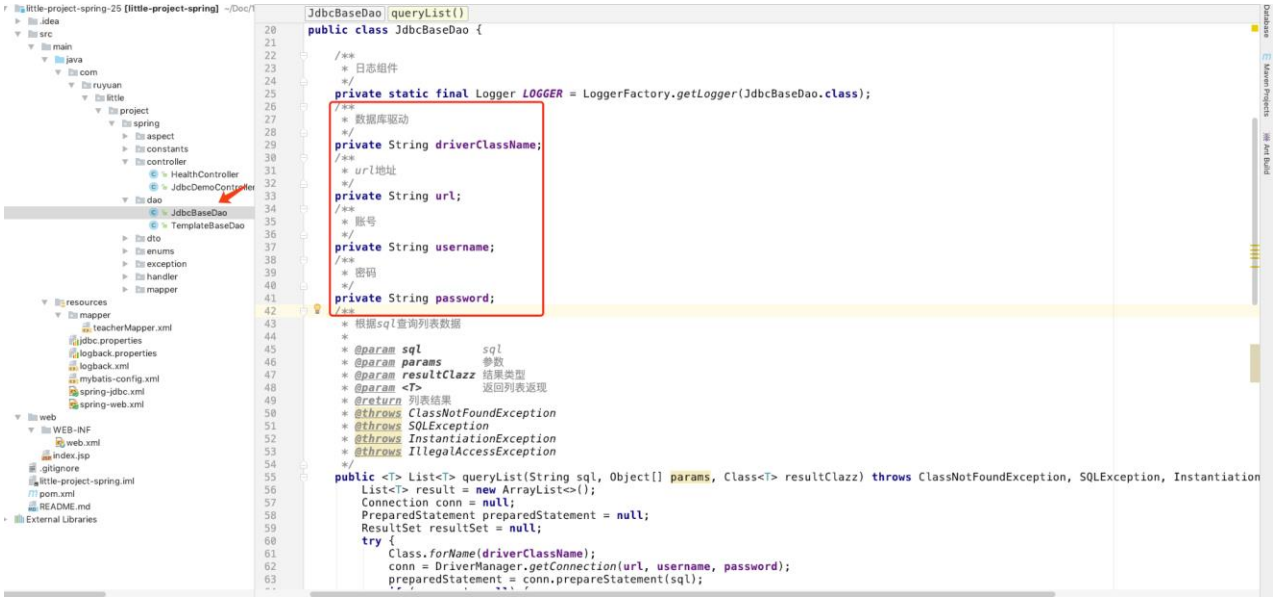

图 2 创建 JdbcBaseDao 类

在这个 JdbcBaseDao 类文件中定义了 queryList、queryOne 以及 update 等访问数据库的方法。这里我们以 queryList 方法为例给大家介绍。

如图 3 所示，首先通过 DriverManager 中的 getConnection 方法传入 url、用户名和密码等信息创建数据库的连接。然后调用数据库连接的 prepareStatement 方法

传入 SQL 语句用来处理查询操作。queryList 方法通过 params 接受了一个参数的数组，在代码中会通过 prepareStatement 的 setObject 方法放入其中。prepareStatement 将 SQL 语句以及对应的参数做了分离，从而实现动态 SQL 提高了运行效率。最后通过 prepareStatement 的 executeQuery 方法从数据库中获取查询结果。这个结果返回给了 resultSet 变量，后面通过 while 循环读出 resultSet 变量中的信息，放入到 List<T> result 中并且返回给调用者。

```
public <T> List<T> queryList(String sql, Object[] params, Class<T> resultClazz) throws ClassNotFoundException, SQLException, InstantiationException, IllegalAccessException {
    List<T> result = new ArrayList<>();
    Connection conn = null;
    PreparedStatement preparedStatement = null;
    ResultSet resultSet = null;
    try {
        Class.forName(driverClassName);
        conn = DriverManager.getConnection(url, username, password);
        preparedStatement = conn.prepareStatement(sql);
        if (params != null) {
            for (int i = 0; i < params.length; i++) {
                preparedStatement.setObject( parameterIndex: i + 1, params[i]);
            }
        }
        resultSet = preparedStatement.executeQuery();
        while (resultSet.next()) {
            T o = resultClazz.newInstance();
            Field[] fields = resultClazz.getDeclaredFields();
            for (Field field : fields) {
                field.setAccessible(true);
                String fieldName = field.getName();
                try {
                    Object value = resultSet.getObject(fieldName);
                    field.set(o, value);
                }catch (SQLException e){
                    LOGGER.debug("结果集中无 {} 参数", fieldName);
                }
            }
            result.add(o);
        }
        conn.close();
```

图 3 JdbcBaseDao 中的 queryList 方法

上面描述了 JDBC 访问数据库的整个过程，这里需要注意的是 PreparedStatement，它会经常在 JDBC 的代码中遇到，这里总结一下它的作用：

1）PreparedStatement 实例包含已编译的 SQL 语句。这就是使语句“准备好”。包含于 PreparedStatement 对象中的 SQL 语句可具有一个或多个 IN 参数。IN 参数的值在 SQL 语句创建时未被指定。相反的，该语句为每个 IN 参数保留一个问号（“？”）作为占位符。每个问号的值必须在该语句执行之前，通过适当的 setXXX 方法来提供。

2）由于 PreparedStatement 对象已预编译过，所以其执行速度要快于 Statement 对象。因此，多次执行的 SQL 语句经常创建为 PreparedStatement 对象，以提高效率。

有了 JdbcBaseDao 以后，就有了通过 JDBC 访问数据库的基本类，为了测试访问数据库的功能还需要创建一个 Controller 类。如图 4 所示，在/spring/controller 目录下面创建 JdbcDemoController 进行测试，加入 GetAll 方法编写对应的 SQL 语

句从 t_teacher 表中获取教师的信息，并且通过 JdbcBaseDao 中的 queryList 方法返回结果。

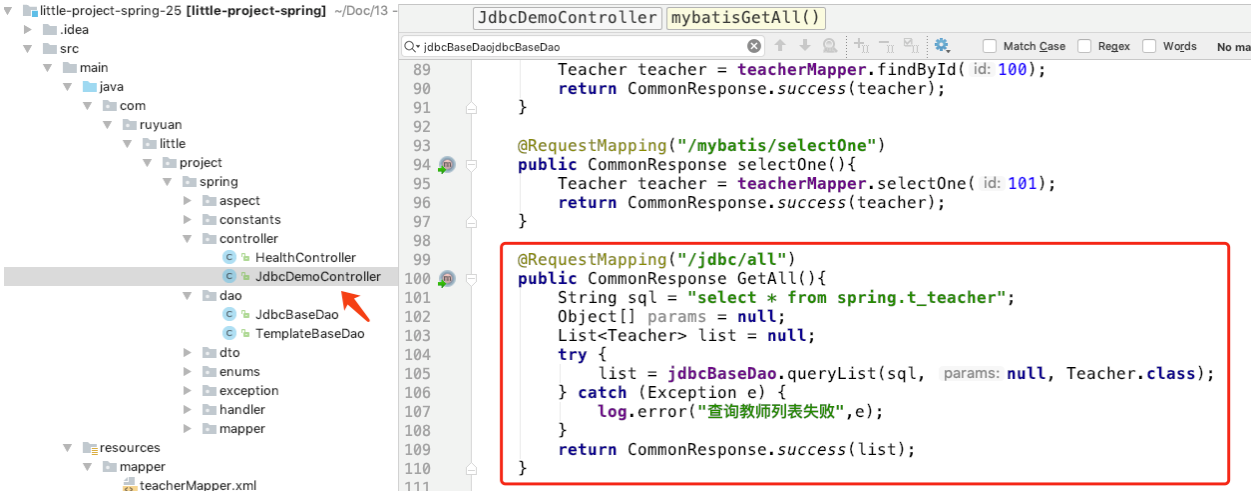

图 4 JdbcDemoController

在开始测试之前还需要对 JdbcBaseDao 进行初始化，如图 5 所示，在 spring-jdbc.xml 中定义 bean，其 id 为 jdbcBaseDao，并且填写对应的 class namespace。这里需要对 driverClassName、url、username 以及 password 进行初始化。这些初始化的信息都来自于 jdbc.properties 文件，这里通过参数的方式引入到 bean 定义中。

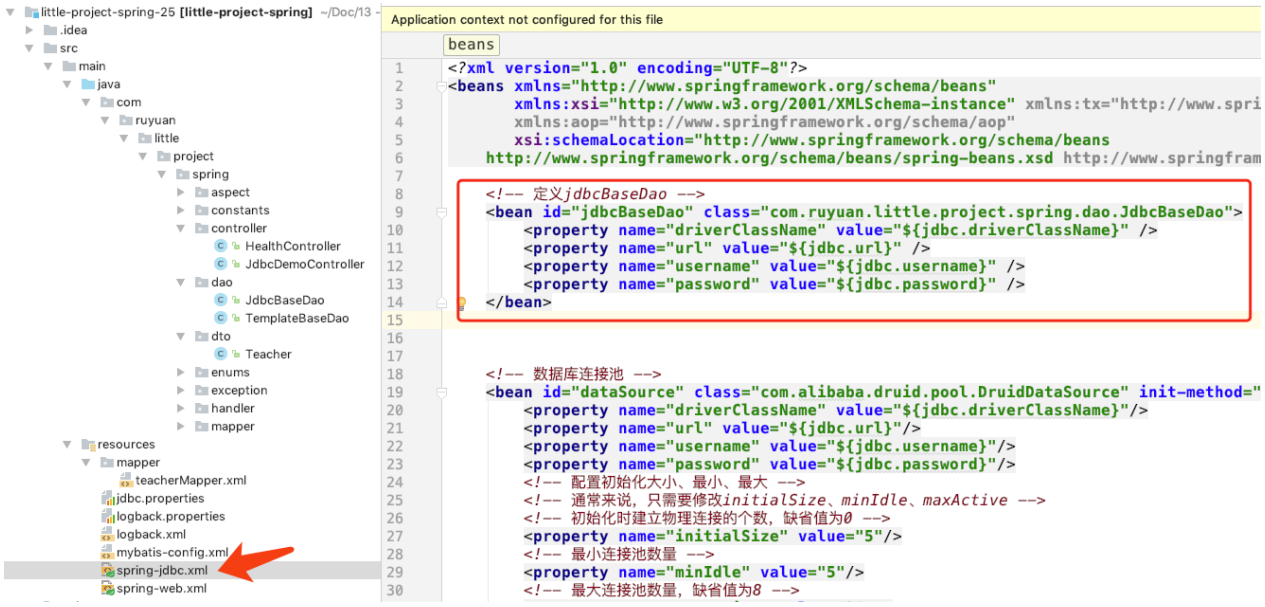

图 5 spring-jdbc.xml 文件

4、总结

这节课对 JDBC 做了简单介绍，知道 JDBC 访问数据库的基本流程。然后，通过在原有代码中加入 JDBC 配置和访问类，达到访问数据库的效果。下期我们会测试一下这次编写的代码，同时思考一下原生 JDBC 操作数据库存在的问题。我们下期见，拜拜。