



Mybatis 使用技巧培训（未完）

JavaEE 框架培训之一

郝金隆（haojinlong@189.cn）

2014 年 6 月



1

第一个 Mybatis O/R 映射程序

2

使用 Mapper 接口

3

与 Spring、c3p0 集成

4

多表映射与条件查询

第一步：前期准备



- 创建数据库（以 mysql 为例）
 - 安装 mysql 数据库（具体方法和所需软件请自行 baidu ）
 - 创建测试数据库
- 准备系统所需要的 jar 包
 - jdbc 驱动（以 mysql 为例）
 - mysql-connector-java-5.1.24-bin.jar （或其他版本）
 - mybatis 文件
 - 官方地址：<http://blog.mybatis.org/> <http://mybatis.github.io/>
 - 所需 jar 包：mybatis-3.2.2.jar （或其他版本）
 - java 日志相关文件
 - 所需 jar 包：slf4j-api, commons-lang3, jcl-over-slf4j, logback-core, logback-classic
 - 下载地址：参见培训一

第二步：创建数据库表，并写入测试数据



-- 创建库表

```
drop table if exists users;
```

```
create table users(
```

```
    id integer not null primary key auto_increment comment '自增长 ID',
```

```
    name char(20) comment '姓名',
```

```
    passwd char(20) comment '密码',
```

```
    age integer comment '年龄'
```

```
);
```

-- 创建测试数据

```
insert into users(name, passwd, age) values('haojinlong', 'haojinlong', 30);
```

```
insert into users(name, passwd, age) values('jucky', 'jucky', 18);
```

```
insert into users(name, passwd, age) values('william', 'william', 30);
```

```
insert into users(name, passwd, age) values('john', 'john', 30);
```

第三步：创建映射类



```
/**
 * # Users.java -- (2014 年 6 月 29 日 )
 * 作者：郝金隆
 * 联系方式： haojinlong@189.cn
 */
package com.github.haojinlong.trainning.mybatis.entity;
import org.apache.commons.lang3.builder.ReflectionToStringBuilder;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * @author 郝金隆
 */
public class Users {
    static Logger logger = LoggerFactory.getLogger(Users.class);
    private Integer id;
    private String name;
    private String passwd;
    private Integer age;

    // ..... getter 、 setter 方法

    @Override
    public String toString() {
        return ReflectionToStringBuilder.reflectionToString(this);
    }
}
```

第四步：创建 OR 映射配置文件



```
<?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="com.github.haojinlong.trainning.mybatis.mapper.UsersMapper">
    <select id="selectById" parameterType="int"
        resultType="com.github.haojinlong.trainning.mybatis.entity.Users">
        select *
        from users where
        id =#{id}
    </select>

    <select id="listAll" resultType="com.github.haojinlong.trainning.mybatis.entity.Users">
        select * from users
    </select>

    <insert id="insert" useGeneratedKeys="true" keyProperty="id"
        parameterType="com.github.haojinlong.trainning.mybatis.entity.Users">
        insert into users(name, passwd, age)
        values(#{name},
        #{passwd}, #{age})
    </insert>

    <!-- ..... -->

</mapper>
```

Mapper 的命名空间

方法名

第五步：创建 mybatis 配置文件



```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
```

```
  <environments default="development">
    <environment id="development">
      <transactionManager type="JDBC" />
      <dataSource type="POOLED">
```

```
        <property name="driver" value="com.mysql.jdbc.Driver" />
        <property name="url" value="jdbc:mysql://localhost:3306/test" />
        <property name="username" value="root" />
        <property name="password" value="root" />
```

```
      </dataSource>
```

```
    </environment>
```

```
  </environments>
```

```
  <mappers>
```

```
    <mapper
```

```
      resource="com/github/haojinlong/training/mybatis/mapper/UsersMapper.xml" />
```

```
    </mapper>
```

```
  </mappers>
</configuration>
```

数据库配置

映射文件相对地址

第六步：调用 mybatis 接口实现数据库访问



// 读取 mybatis 配置文件进行初始化

```
String resource = "mybatis-config.xml";
```

```
SqlSessionFactory sqlSessionFactory = null;
```

```
try {
```

```
    InputStream inputStream = Resources.getResourceAsStream(resource);
```

```
    sqlSessionFactory = new SqlSessionFactoryBuilder()
```

```
        .build(inputStream);
```

```
} catch (IOException e) {
```

```
    e.printStackTrace();
```

```
}
```

```
SqlSession sqlSession = sqlSessionFactory.openSession();
```

// 调用接口进行查询

```
Users users = sqlSession
```

```
    .selectOne("com.github.haojinlong.training.mybatis.mapper.UsersMapper.selectById", 1);
```

```
logger.debug("the value of users: {}", users);
```

```
sqlSession.close();
```

mybatis 配置文件
地址

UsersMapper 中的命
名空间和方法名称

注：完整版本代码参见附录中代码示例：1-mybatis-basic



1

第一个 Mybatis O/R 映射程序

2

使用 Mapper 接口

3

与 Spring、c3p0 集成

4

多表映射与条件查询



- 传统的 mybatis(ibatis) 中所有的数据读写操作均通过 sqlSession 来完成，通过传入 Mapper 配置文件指定的 namespace 和映射 id 来进行访问，每次方法的执行都需要输入一长串的字符串用来定位具体的映射，使用很不方便，为此 Mapper 接口应运而生
- 我们可以通过创建 Mapper 接口来绑定映射语句，Mapper 接口的实例可以通过 sqlSession 自动生成，这样前端的开发人员只需要调用响应的 Mapper 接口程序即可
 - Mappers are interfaces that you create to bind to your mapped statements. Instances of the mapper interfaces are acquired from the SqlSession
- 使用 Mapper 接口方式进行 mybatis 数据存取需要七步工作，前五步与传统的方式一模一样，唯一不同的是需要创建 Mapper 接口文件，并采用 Mapper 接口方式进行数据的操作

第六步：创建 Mapper 接口文件



```
/**
 * # UsersMapper.java -- (2014 年 6 月 29 日)
 * 作者：郝金隆
 * 联系方式：haojinlong@189.cn
 */
package com.github.haojinlong.training.mybatis.mapper;

import java.util.List;
import com.github.haojinlong.training.mybatis.entity.Users;

/**
 * @author 郝金隆
 */
public interface UsersMapper {
    public Users selectById(int id);
    public List<Users> listAll();
    public int insert(Users users);
    public int delete(int id);
    public int update(Users users);
}
```

Mapper 接口包名和接口名称需
要与映射文件中的 namespace
相对应

方法名需要与映射配置文件中单
个映射的 id 值相对应

第七步：通过 Mapper 接口实现数据库访问



// 读取 mybatis 配置文件进行初始化

```
String resource = "mybatis-config.xml";
```

```
SqlSessionFactory sqlSessionFactory = null;
```

```
try {
```

```
    InputStream inputStream = Resources.getResourceAsStream(resource);
```

```
    sqlSessionFactory = new SqlSessionFactoryBuilder()
```

```
        .build(inputStream);
```

```
} catch (IOException e) {
```

```
    e.printStackTrace();
```

```
}
```

```
SqlSession sqlSession = sqlSessionFactory.openSession();
```

// 调用接口进行查询

```
UsersMapper usersMapper = sqlSession.getMapper(UsersMapper.class);
```

```
Users users = usersMapper.selectById(1);
```

```
logger.debug("the value of users: {}", users);
```

```
sqlSession.close();
```

mybatis 配置文件
地址

注：完整版本代码参见附录中代码示例：2-mybatis-mapper



1

第一个 Mybatis O/R 映射程序

2

使用 Mapper 接口

3

与 Spring、c3p0 集成

4

多表映射与条件查询



一、引入的所需要的 jar 包

- mybatis、Java 日志程序、jdbc 驱动
 - 参见 P3：《第一步：前期准备》
- mybatis 的 spring 集成工具
 - mybatis-spring.jar：<http://blog.mybatis.org/> <http://mybatis.github.io/>
- c3p0 数据中连接池程序
 - mchange-commons-java、c3p0.jar
 - 官方地址：<http://www.mchange.com/projects/c3p0/>
- spring 程序
 - spring-aop、spring-beans、spring-context、spring-core、spring-expression、spring-jdbc、spring-tx
 - 官方地址：<http://spring.io>
<http://maven.springframework.org/release/org/springframework/spring/>



二、创建库表并完成 OR 映射文件编写

1. 初始化数据库表

- 参见 P4 : 《第二步：创建数据库表，并写入测试数据》

2. 创建 OR 映射类

- 参见 P5 : 《第三步：创建映射类》

3. 创建 OR 映射配置文件

- 参见 P6 : 《第四步：创建 OR 映射配置文件》

4. 创建 Mapper 接口

- 参见 P11 : 《第六步：创建 Mapper 接口文件》

注：与 spring 集成后，不需要使用 mybatis-config.xml 配置文件

三、创建 applicationContext.xml 配置文件



```
<?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:aop="http://www.springframework.org/schema/aop"
    xmlns:tx="http://www.springframework.org/schema/tx" xmlns:jdbc="http://www.springframework.org/schema/jdbc"
    xmlns:context="http://www.springframework.org/schema/context"
    xsi:schemaLocation="
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd
        http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
        http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-3.0.xsd
        http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd">
    <!-- 数据源 -->
    <bean id="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
        <property name="driverClass" value="com.mysql.jdbc.Driver"></property>
        <property name="jdbcUrl" value="jdbc:mysql://localhost:3306/test"></property>
        <property name="user" value="root"></property>
        <property name="password" value="root"></property>
    </bean>
    <!-- 创建 SqlSessionFactory -->
    <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
        <property name="dataSource" ref="dataSource" />
    </bean>
    <!-- 自动扫描映射器 -->
    <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
        <property name="basePackage" value="com.github.haojinlong" />
    </bean>
</beans>
```


四、通过 Spring 获取 Mapper 接口的实体数据存取



```
ApplicationContext context = new ClassPathXmlApplicationContext(  
    "applicationContext.xml");  
UsersMapper usersMapper = context.getBean(UsersMapper.class);  
Users users = usersMapper.selectById(1);  
logger.debug("the value of users: {}", users);
```

注：完整版本代码参见附录中代码示例：3-mybatis-spring-c3p0



1

第一个 Mybatis O/R 映射程序

2

使用 Mapper 接口

3

与 Spring、c3p0 集成

4

多表映射与条件查询

