

87 | 开源实战五（上）：MyBatis如何权衡易用性、性能和灵活性？

王争 2020-05-22



上几节课我们讲到了 Spring 框架，剖析了背后蕴含的一些通用设计思想，以及用到的十几种设计模式。从今天开始，我们再创析另外一个 Java 项目开发中经常用到的框架：MyBatis。因为内容比较多，同样，我们也分三节课来讲解。

- 第一节课，我们分析 MyBatis 如何权衡代码的易用性、性能和灵活性。
- 第二节课，我们学习如何利用职责链与代理模式实现 MyBatis Plugin。
- 第三节课，我们总结罗列一下 MyBatis 框架中用到的十几种设计模式。

话不多说，让我们正式开始今天的学习吧！

Mybatis 和 ORM 框架介绍

熟悉 Java 的同学应该知道，MyBatis 是一个 ORM (Object Relational Mapping, 对象 - 关系映射) 框架。ORM 框架主要是根据类和数据库表之间的映射关系，帮助程序员自动实现对象与数据库中数据之间的互相转化的。说得更具体点就是，ORM 负责将程序中的对象存储到数据库中、将数据库中的数据转化为程序中的对象。实际上，Java 中的 ORM 框架有很多，除了刚刚提到的 MyBatis 之外，还有 Hibernate、TopLink 等。

在剖析 Spring 框架的时候，我们讲到，如果用一句话来总结框架作用的话，那就是简化开发。MyBatis 框架也不例外。它简化的是数据库方面的开发。那 MyBatis 是如何简化数据库开发的呢？我们结合第 59 讲中的 JdbcTemplate 的例子来说明一下。

在第 59 讲中，我们讲到，Java 提供了 JDBC 类库来封装不同类型数据库操作。不过，直接使用 JDBC 来进行数据库编程，还是有点麻烦的。于是，Spring 提供了 JdbcTemplate，对 JDBC 进一步封装，来进一步简化数据库编程。

使用 JdbcTemplate 进行数据库编程，我们只需要编写跟业务相关的代码（比如，SQL 语句、数据库中数据与对象之间的互相转化的代码），其他流程性质的代码（比如，加载驱动、创建数据库连接、创建 statement、关闭连接、关闭 statement 等）都封装在了 JdbcTemplate 类中，不需要我们重复编写。

当时，为了展示使用 JdbcTemplate 是如何简化数据库编程的，我们还举了一个查询数据库中用户信息的例子。还是同样这个例子，我再来看下，使用 MyBatis 该如何实现，是不是比使用 JdbcTemplate 更加简单。

因为 MyBatis 依赖 JDBC 驱动，所以，在项目中使用 MyBatis，除了需要引入 MyBatis 框架本身 (mybatis.jar) 之外，还需要引入 JDBC 驱动（比如，访问 MySQL 的 JDBC 驱动实现类 mysql-connector-java.jar）。将两个 jar 包引入项目之后，我们就可以开始编程了。使用 MyBatis 来访问数据库中用户信息的代码如下所示：

```
1 // 1. 定义UserDO
2 public class UserDO {
3     private long id;
4     private String name;
5     private String telephone;
6     // 省略setter/getter方法
7 }
8
9 // 2. 定义访问接口
10 public interface UserMapper {
11     public UserDO selectById(long id);
12 }
13
14 // 3. 定义映射关系：UserMapper.xml
15 <?xml version="1.0" encoding="UTF-8"?>
16 <!DOCTYPE mapper PUBLIC "-//mybatis.org/DTD Mapper 3.0//EN"
17     "http://mybatis.org/dtd/mybatis-3-mapper.dtd" >
18 <mapper namespace="cn.xzg.cd.a87.repo.mapper.UserMapper">
19     <select id="selectById" resultType="cn.xzg.cd.a87.repo.UserDO">
20         select * from user where id=#{id}
21     </select>
22 </mapper>
23
24 // 4. 全局配置文件：mybatis.xml
25 <?xml version="1.0" encoding="UTF-8"?>
26 <!DOCTYPE configuration
27     PUBLIC "-//mybatis.org/DTD Config 3.0//EN"
28     "http://mybatis.org/dtd/mybatis-3-config.dtd">
29 <configuration>
30     <environments default="dev">
31         <environment id="dev">
32             <transactionManager type="JDBC"/></transactionManager>
33             <dataSource type="POOLED">
34                 <property name="driver" value="com.mysql.jdbc.Driver" />
35                 <property name="url" value="jdbc:mysql://localhost:3306/test?u
36                 <property name="username" value="root" />
37                 <property name="password" value="..." />
38             </dataSource>
39         </environment>
40     </environments>
41     <mapppers>
42         <mapper resource="mapper/UserMapper.xml"/>
43     </mapppers>
44 </configuration>
```

需要注意的是，在 UserMapper.xml 配置文件中，我们只定义了接口和 SQL 语句之间的映射关系，并没有显式地定义类 (UserDO) 字段与数据库表 (user) 字段之间的映射关系。实际上，这就体现了“约定优于配置”的设计原则。类字段与数据库表字段之间使用了默认映射关系：类字段跟数据库表中拼写相同的字段——映射。当然，如果没法做到——映射，我们也可以自定义它们之间的映射关系。

有了上面的代码和配置，我们就可以像下面这样来访问数据库中的用户信息了。

```
public class MybatisDemo {
    public static void main(String[] args) throws IOException {
        Reader reader = Resources.getResourceAsReader("mybatis.xml");
        SqlSessionFactory sessionFactory = new SqlSessionFactoryBuilder().build(re
        SqlSession session = sessionFactory.openSession();
        UserMapper userMapper = session.getMapper(UserMapper.class);
        UserDO userDO = userMapper.selectById(8);
        // ...
    }
}
```

从代码中，我们可以看出，相对于使用 JdbcTemplate 的实现方式，使用 MyBatis 的实现方式更加灵活。在使用 JdbcTemplate 的实现方式中，对象与数据库中数据之间的转化代码、SQL 语句，是硬编码在业务代码中的。而在使用 MyBatis 的实现方式中，类字段与数据库字段之间的映射关系、接口与 SQL 之间的映射关系，是写在 XML 配置文件中的，是跟代码相分离的，这样会更加灵活、清晰，维护起来更加方便。

如何平衡易用性、性能和灵活性？

刚刚我们对 MyBatis 框架做了简单介绍，接下来，我们再对比一下另外两个框架：JdbcTemplate 和 Hibernate，通过对比我们来看，MyBatis 是如何权衡代码的易用性、性能和灵活性的。

我们先来看 JdbcTemplate。相对于 MyBatis 来说，JdbcTemplate 更加轻量级。因为它对 JDBC 只做了很简单的封装，所以性能损耗比较少。相对于其他两个框架来说，它的性能最好。但是，它的缺点也比较明显，那就是 SQL 与代码耦合在一起，而且不具备 ORM 的功能，需要自己编写代码，解析对象跟数据库中的数据之间的映射关系。所以，在易用性上它不及其他两个框架。

我们再来看 Hibernate。相对于 MyBatis 来说，Hibernate 更加重量级。Hibernate 提供了更加高级的映射功能，能够根据业务需求自动生成 SQL 语句。我们不需要像使用 MyBatis 那样自己编写 SQL。因此，有的时候，我们也把 MyBatis 称作半自动化的 ORM 框架，把 Hibernate 称作全自动化的 ORM 框架。不过，虽然自动生成 SQL 简化了开发，但是毕竟是自动生成的，没有针对性的优化。在性能方面，这样得到的 SQL 可能没有程序员编写得好。同时，这样也丧失了程序员自己编写 SQL 的灵活性。

实际上，不管用哪种实现方式，从数据库中取出数据并且转化成对象，这个过程涉及的代码逻辑基本是一致的。不同实现方式的区别，只不过是哪部分代码逻辑放到了哪里。有的框架提供的功能比较强大，大部分代码逻辑都由框架来完成，程序员只需要实现很小的一部分代码就可以了。这样框架的易用性就更好些。但是，框架集成的功能越多，为了处理逻辑的通用性，就会引入更多额外的处理代码。比起针对具体问题具体编程，这样性能损耗就相对大一些。

所以，粗略地讲，有的时候，框架的易用性和性能成对立关系。追求易用性，那性能就差一些。相反，追求性能，易用性就差一些。除此之外，使用起来越简单，那灵活性就越差。这就好比我们用的照相机。傻瓜相机按下快门就能拍照，但没有复杂的单反灵活。

实际上，JdbcTemplate、MyBatis、Hibernate 这几个框架也体现了刚刚说的这个规律。

JdbcTemplate 提供的功能最简单，易用性最差，性能损耗最少，用它编程性能最好。Hibernate 提供的功能最完善，易用性最好，但相对来说性能损耗就最高了。MyBatis 介于两者中间，在易用性、性能、灵活性三个方面做到了均衡。它支撑程序员自己编写 SQL，能够延续程序员对 SQL 知识的积累。相对于完全盒子的 Hibernate，很多程序员反倒是更喜欢 MyBatis 这种半透明的框架。这也提醒我们，过度封装，提供过于简化的开发方式，也会丧失开发的灵活性。

重点回顾

好了，今天的内容到此就讲完了。我们一块来总结回顾一下，你需要重点掌握的内容。

如果你熟悉 Java 和 MyBatis，那你应该掌握今天讲到 JDBC、JdbcTemplate、MyBatis、Hibernate 之间的区别。JDBC 是 Java 访问数据库的开发规范，提供了一套抽象的统一的开发接口，隐藏不同数据库的访问细节。

JdbcTemplate、MyBatis、Hibernate 都是对 JDBC 的二次封装，为的是进一步简化数据库开发。其中，JdbcTemplate 不能算得上是 ORM 框架，因为还需要程序员自己编程来实现对象和数据库数据之间的互相转化。相对于 Hibernate 这种连 SQL 都不用程序员自己写的全自动 ORM 框架，MyBatis 算是一种半自动化的 ORM 框架。

如果你不熟悉 Java 和 MyBatis，作为背景介绍，那你简单了解一下 MyBatis 和 ORM 就可以了。不过，在你熟悉的语言中，应该也有相应的 ORM 框架，你也可以对比着去分析一下。

今天的内容除了起到对 MyBatis 做背景介绍之外，我们还学习了代码的易用性、性能、灵活性之间的关系。一般来讲，提供的高级功能越多，那性能损耗就会越大些；用起来越简单，提供越简化的开发方式，那灵活性也就相对越低。

课堂讨论

在你的项目开发中，有没有用过哪些框架，能够切实提高开发效率，减少不必要的体力劳动？

欢迎留言和我分享你的想法。如果有收获，也欢迎你把这篇文章分享给你的朋友。

课程预告

6月-7月课表抢先看

充 ¥500 得 ¥580

赠「¥118 月球主题 AR 笔记本」

【点击】图片，立即查看 >>>

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

志恒Z

由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。

Ctrl + Enter 发表

Q/2000字

提交留言

精选留言 (13)

工作中做过一些c++的东西,做起来相当复杂,每引入一个第三方类库,都要自己去github上找,找到再clone下来,打包,才能引入,搜编程面向对象面向过程基于对象函数式,眼花缭乱,指针引用const像傻分不清是cmake打包异常,只有求助大佬才能维持生活

做java就像回家一样,做开发有spring全家桶,maven在csdn比家里感觉好多了,里面各个是人,说话又好听,只需要CTRL C V就能完成工作,我喜欢里面的!

2020-05-22

5

Demon.Lee

易用性: Hibernate > MyBatis > JdbcTemplate
性能: JdbcTemplate > MyBatis > Hibernate
灵活性: MyBatis > JdbcTemplate > Hibernate

2020-05-22

5

test

SpringCloud全家桶

2020-05-22

4

jinjunzhu

比如做限流的时候可以直接使用guava中的限流器

2020-05-22

3

子豪sirius

mybatis可以让开发人员自己写SQL,相比hibernate给了更多控制权。不过在实际开发中有个问题,有些开发人员会写很复杂的SQL,美其名曰是性能更好,但实际性能提升多少,不清楚;反而因为SQL写得巨冗复杂,带来了阅读困难、调试和查错不便等等问题。明明这部分代码用Java写,业务逻辑是更清晰的~

2020-05-22

3

Monday

mybatis系列
1、mybatis-plus 作用如其名, mybatis增强功能封装好了一些crud的方法
2、mybatis-generator自动生成器, 自动生成实体、mapper、Mapper.xml等
3、mybatis分页插件PageHelper, 无需关心分页的问题

2020-05-22

2

白马度和

netty是个反例, 兼顾易用性和性能。

2020-05-22

2

Javaadu

目前用过的最好的用的框架还是Spring Boot

2020-05-24

1

whistleman

打卡~
MyBatis 介于JDBC Template 和 MyBatis中间, 在易用性、性能、灵活性三个方面做到T权衡。

2020-05-23

1

xianhai

jetty, 容易集成

2020-05-23

1

守拙

对于"减少不必要的体力劳动"这一限定条件来说, Gson + Idea插件GsonFormatter当仁不让, 无法想象没有它们该怎么写业务代码。

2020-05-22

1

Jxin

ruby on rails

2020-05-22

1

小晏子

rails框架功能也异常强大, 全自动ORM, 不用写sql语句, 所有东西框架帮你搞定, 快速搞出产品原型利器!

2020-05-22

1