计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(4):188−193 [doi: 10.15888/j.cnki.csa.006824]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

Java EE 数据初始化应用中的设计模式解决方案①

欧阳宏基, 葛　萌

(咸阳师范学院 计算机学院, 咸阳 712000)
通讯作者: 欧阳宏基, E-mail: oyhj_nicholas@163.com

摘　要: 针对 Java EE 应用数据初始化时的先后顺序及依赖关系, 提出一种基于设计模式的数据初始化方法. 该方法首先将要初始化的数据看做一个复杂对象, 通过建造者模式的产品角色来表示. 利用抽象建造者角色定义数据初始化的先后顺序, 使用 Hibernate 框架将各部分数据保存到数据库中; 然后通过策略模式定义应用开发阶段和正式运行阶段的初始化算法; 最后利用单例模式封装 JDBC 和 Hibernate 操作的通用步骤. 以高等院校目标考核管理系统的数据初始化操作为背景, 通过 JXL 读取存储在 Excel 文件中的初始化数据作为具体建造者角色的实现逻辑, 详细阐述了该方法的具体实现过程. 通过测试与实践表明该数据初始化方法能够满足系统应用要求, 具有较高的复用度, 能够提高数据初始化操作的扩展性和维护性.

关键词: 数据初始化; 建造者设计模式; 策略设计模式; 单例设计模式; Hibernate 框架

引用格式: 欧阳宏基,葛萌.Java EE 数据初始化应用中的设计模式解决方案.计算机系统应用,2019,28(4):188–193. http://www.c-s-a.org.cn/1003-3254/6824.html

Design Pattern Solution for Data Initialization of Java EE Application

OUYANG Hong-Ji, GE Meng

(College of Computer, Xianyang Normal University, Xianyang 712000, China)

Abstract: According to the order and dependencies of data initialization for Java EE application, a data initialization method based on design pattern is proposed. First, the data to be initialized is regarded as a complex object, and it is represented by the product role of the builder mode, using the abstract builder role to define the order of data initialization, and using Hibernate framework to save every part of the data to the database. Then, it defines the initialization algorithm in the application development stage and the formal operation stage through the strategy pattern. Finally, it uses singleton pattern to encapsulate the general steps of JDBC and Hibernate operation. Taking the data initialization operation of target assessment management system in colleges and universities for example, through reading the initial data stored in the Excel file by JXL as the implementation logic of the concrete builder's role, the concrete realization process of the method is described in detail. The test and practice show that the data initialization method can meet the application requirements of the system, it has high reusability and can improve the expansibility and maintainability of data initialization operation.

Key words: data initialization; builder pattern; strategy pattern; singleton pattern; Hibernate framework

① 基金项目: 陕西省教育厅专项科研计划项目 (17JK0828); 咸阳师范学院“青年骨干教师”培养项目 (XSYGG201615); 咸阳师范学院专项科研计划项目 (XSYK17029)

Foundation item: Special Scientific and Technological Research Program of Education Bureau, Shaanxi Province; “Young Backborne Teacher” Cultivating Program of Xianyang Normal University (XSYGG201615); Special Scientific and Technological Research Program of Xianyang Normal University (XSYK17029)

收稿时间: 2018-09-19; 修改时间: 2018-10-12; 采用时间: 2018-10-30; csa 在线出版时间: 2019-03-28

数据初始化是任何一个应用系统运行其业务逻辑的基础. 初始化的数据一般都是基础数据, 由当前应用系统的需求分析而确定. 一部分数据需要自行定义, 一部分可能来自于和当前系统相关的其他业务系统. 初始化的数据通常在系统正式上线运行前或者运行时添加到数据库中, 是应用系统中其他业务数据赖以生存的前提, 因此每一个系统都必须执行数据初始化的操作. 目前, 常见的数据初始化操作方法有两种: 一种是为系统各模块提供数据添加操作, 通过用户手工方式逐条录入所需要的数据, 该方法适合规模较小、基础数据量少的系统[1]. 第二种方法是通过数据库第三方工具 (例如 MySQL 数据库的 SQLyog、Navicat 等) 将存储在 Excel、XML、HTML 等文件中或者是其他信息系统数据库中的相关数据批量导入到当前应用中, 该方法适合规模、数据量较大的系统, 但是在操作过程中也必须单表逐个导入. 无论采用上述哪种方法, 用户在操作时必须按照一定的顺序, 尤其是基础数据之间存在较复杂依赖关系的请求下, 无疑增加了用户操作的复杂度, 影响了用户的操作体验.

针对上述问题, 在研究建造者模式、策略模式以及单例模式的前提下, 提出一种新的数据初始化方法, 该方法在系统运行前自动读取存储在外部相关文件中的数据并保存到当前应用的数据库中, 避免了手动操作的繁琐. 以高等院校目标考核管理系统中的数据初始化操作为例, 分析了各部分的初始化数据以及顺序问题. 以 Excel 文件作为数据初始化前的载体, 利用 JXL、JDBC 和 Hibernate 作为混合数据持久化技术完成了数据的初始化操作, 详细阐述了上述各设计模式的具体实现过程, 为数据初始化操作提供了一定的参考.

1 相关技术

设计模式 (design pattern) 是面向对象技术中被精心编制的接口与类的组合, 在特定的应用场景中使用特定的模式可以获得结构清晰、易于扩展和理解的代码结构[2,3]. 建造者模式 (Builder Pattern) 属于创建型模式的一种, 主要用于将一个复杂对象的构建与它的表示分离, 向用户屏蔽复杂对象组成部分的创建细节[4]. 单例模式 (singleton pattern) 也属于创建型模式, 主要用于约束某个类的对象只能被实例化一次. 策略模式 (strategy pattern) 属于行为型模式, 该模式通常将某个行为的不同实现方式定义为某个单独的类, 允许调用端根据特定情况更换行为的实现方式[5].

基于 Java EE 平台的应用系统通常利用 JDBC 或 Hibernate 技术进行关系型数据库的访问. JDBC 通过 Connection、Statement、ResultSet 等接口执行原生 SQL 语句操作数据库[6]; Hibernate 是 ORM 协议的具体实现, 对 JDBC 进行了轻量级封装, 以映射文件或注解为基础实现对象模型与数据库表的映射, 并且自动管理对象与对象之间的关联关系[7]. Hibernate 提供了 Configuration、SessionFactory、Session、Transaction 和 Query 和 Criteria 等 6 个核心接口对持久化对象进行存储和事务控制, 交互过程如图 1 所示. 其中, Configuration 负责读取配置文件并创建 SessionFactory; SessionFactory 是重量级组件采用了单例设计模式, 在启动时创建全局唯一对象并常驻内容, 主要用来管理 Session 对象和部分缓存数据. Session 对象代表了一条与数据库的连接, 用来创建 Query 或 Criteria 对象. Query 或 Criteria 负责执行 HQL 语句, 在 Transaction 的管理下利用映射文件实现持久化对象与数据库的交互[8].

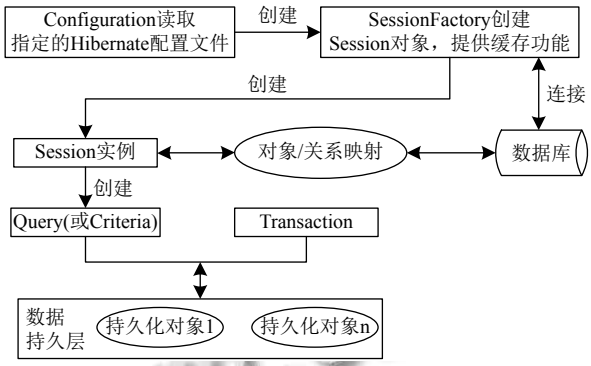

图 1 Hibernate 核心接口交互图

2 数据初始化

2.1 数据初始化的顺序

Java EE 应用的初始化过程中会涉及众多的对象, 他们之间必然存在复杂的关联关系. 通过 ORM 将这些对象保存到数据库中, 并维持之间的外键关系, 就必须考虑初始化的顺序问题. 例如在采用 BRAC 模型实现的权限管理系统中, 权限、角色和用户三者存在着关联关系[9]. 角色关联权限, 用户关联角色, 首先应初始化权限数据, 再初始化角色数据, 最后初始化用户数据. 因此, 当对象的关联关系之间具有先后顺序时, 需要先保存被关联的对象数据.

2.2 基于设计模式的数据初始化方法

综合应用单例、建造者、策略设计模式的数据初始化模型类图关系如图 2 所示. 其中, SysData、SysData

Builder、SysDataExcelBuilder 和 SysDataDirector 构成建造者模式. SysData 表示产品, 定义所有要初始化的数据; SysDataBuilder 表示抽象建造者, SysDataExcelBuilder 表示具体建造者; SysDataDirector 表示指挥者, 其中的 construct() 方法控制初始化数据的顺序. JDBCUtil 为单例设计模式, 该类的主要作用有两个, ① 封装加载数据库驱动、创建数据库连接和释放资源; ② 创建和删除数据库. SysInit、InitStrategy 和 ConcreteStrategy 构成策略模式. SysInit 为环境类、InitStrategy 为策略接口, ConcreteStrategy 为具体初始化策略, 包括开发阶段的初始化策略和系统部署上线运行阶段的策略.

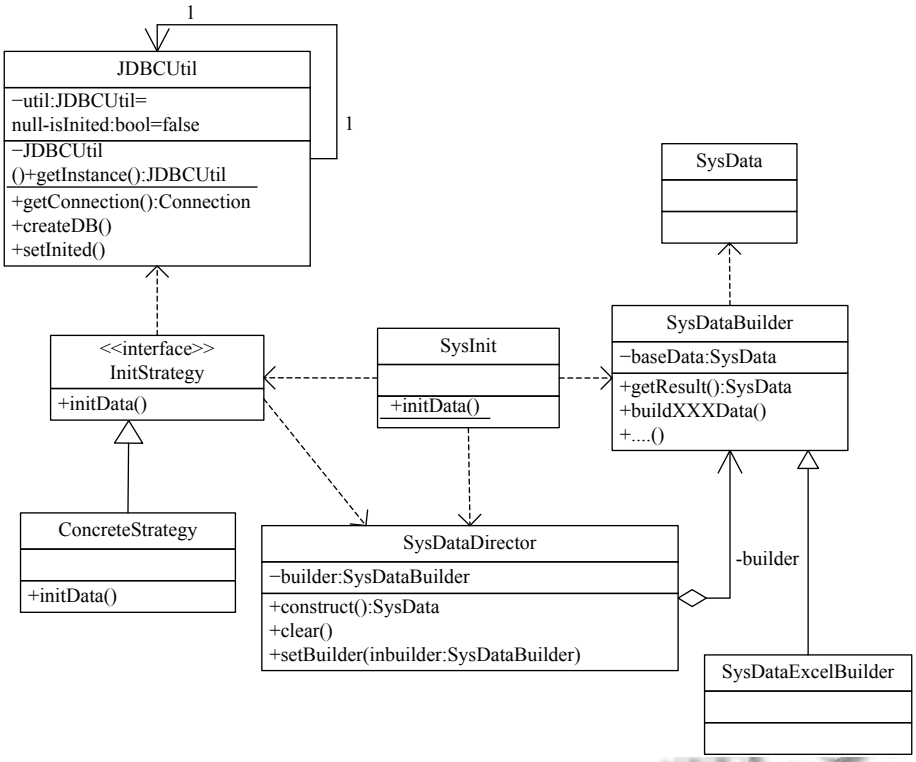

图 2 数据初始化方法类图

3 在高等院校目标考核系统中的应用

目标考核系统是采用计算机与网络技术所实现的对高校各部门年度目标任务进行信息化管理的 Web 系统. 系统基于 Java EE 平台, 采用 MVC 模式设计, 由 Struts2+Spring+Hibernate 轻量级框架构建[10]. 核心功能包括目标任务的编制、任务的下达、任务完成数据的上传、任务完成数据的审核、量化打分以及汇总等功能. 通过该系统实现了目标考核工作的数字化管理, 提高了考核过程的透明度、结果的公平性和公正性, 对全校教职工的工作具有良好的引导和激励作用, 为学校领导层作决策提供了数据支撑.

3.1 目标考核系统中的数据初始化顺序

由于目标考核工作涉及较多的职能部门 (发展规划处、人事处、科技处、学生工作处、教务处、学科办、团委、党委组织部、党委宣传部等)、各二级教学单位和全体教职工. 考核目标覆盖了教育教学、科学研究、师资队伍建设、学生工作、党风廉政建设等各个方面, 因此系统运行所需要的基础数据较多. 按照考核工作的职责、对象和任务, 将基础数据划分为三大部分, 包括: 用户数据、权限数据和指标数据. 其中用户数据包含技术职务、党政职务、党政职务级别、岗位级别、部门、教工基本信息、部门账户和科研类别. 权限数据包括权限和角色. 指标包括综合指标和业务指标两部分, 每部分又细分为一级指标、二级指标和三级指标. 其中一个一级指标包含多个二级指标, 一个二级指标包含多个三级指标. 上述各项数据的具体初始化先后顺序依次是: 技术职务、党政职务、党政职务级别、岗位级别、部门、权限、角色、教工基本信息、部门账户、

科研级别、一级综合指标、二级综合指标、三级综合指标、业绩一级指标、二级业绩指标、三级业绩指标.

3.2 建造者设计模式的应用

建造者设计模式包括抽象建造者、具体建造者、指挥者和产品 4 个角色, 如图 3 所示. 将目标考核系统的所有要初始化的数据看作一个整体, 充当其中的产品角色, 每个部分的数据作为产品的一个成员, 用 List 集合表示, 因此产品是一个较复杂的 POJO 对象. 抽象建造者角色关联一个受保护的产品对象, 定义初始化各部分数据的抽象接口, 其中每一个 buildXXX() 方法创建一类要初始化的数据, 通过 getResult() 方法得到所关联的产品对象. 具体建造者继承抽象建造者实现每一个 buildXXX() 方法, 定义每一类数据的初始化逻辑.

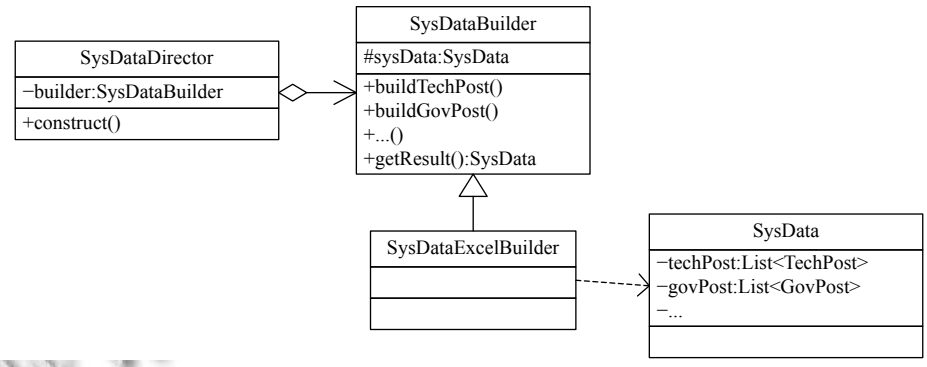

图 3 建造者设计模式类图

本文所实现的目标考核系统的初始化数据存储在 Excel 文件中, 所以定义 SysDataExcelBuilder 类作为具体的建造者. 其中包含每一类初始化数据对应的 Service 组件作为属性, 其对象通过@Resource 注解从 Spring IOC 容器获取. Service 对象通过@Resource 注解关联的 Dao 对象完成初始化操作, Dao 对象封装了 Hibernate API 具体负责初始化数据的增、删、改、查操作. 具体建造者通过读取 applicationContext.xml 文件创建 Spring IOC 容器, 用 ApplicationContext 对象表示. 另外还包含一个 InputStream 对象用作读取 Excel 文件的流对象, 一个 Workbook 对象用来表示 Excel 的工作表, 通过 JXL 完成对 Excel 工作表和单元格数据的访问[11].

3.3 策略设计模式的应用

策略设计模式包括环境、抽象策略和具体策略三个角色, 如图 4 所示. 该模式可以根据环境或者条件的不同而选择不同的策略来完成某项任务. InitStrategy 为抽象策略接口, 其中 initData 方法定义数据初始化算法. DevelopmentStrategy 和 RunningStratey 是两个具体的初始化策略, DevelopmentStrategy 是系统开发阶段使用的策略, 具体的数据初始化算法是如果数据库存在先删除数据库, 再创建数据库并通过建造者模式的指挥者对象调用初始化操作. RunningStrategy 是系统部署上线运行时采用的策略, 具体的数据初始化算法是获取初始化完成标记信息, 判断是否已经执行过初始化操作, 如果已经执行则忽略; 如果没有初始化过, 那么先创建数据库, 通过建造者模式的指挥者对象调用初始化操作并设置初始化完成标记, 标记信息存储在特定数据库表中.

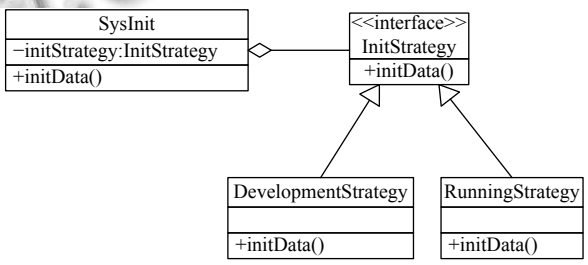

图 4 策略设计模式类图

3.4 单例设计模式的应用

单例设计模式能够确保某个类只能拥有自己的一个对象, 在某些工具类的定义中普遍应用单例模式. 这种工具类基本上贯穿程序始终, 必然会频繁调用. 如果每一次调用都要重新生成实例, 带来的就是内存堆空间的浪费, 所以单例模式能够提高程序的运行速度, 减

少资源消耗. 文献[12]分析了单例模式实现的 5 种方式, 总结了每种方式的优缺点和应用场景. 本初始化模型在操作数据库时采用 JDBC 与 Hibernate 相结合的混合持久化技术, 所以采用静态内部类方式作为单例模式的实现方法, 因为该方法具有调用效率高、线程安全、支持延迟加载等优点.

由于 Hibernate 框架能够创建并更改数据库表结构, 实现对象与表、对象之间关系与表之间关联的映射, 但无法自动创建数据库, 因此需要通过 JDBC 执行 SQL 自动创建数据库. 利用单例模式定义 JDBCUtil, 用来封装 JDBC 的加载数据库驱动、获取 Connection 对象和释放资源等 3 个操作, 这个单例对象作为 JDBC 操作的全局唯一入口. 同样为了简化 Hibernate 进行持久化操作代码的重复性, 利用单例模式定义 HibernateUtil, 该类完成对 Hibernate 有关代码的封装, 其中包含的主要操作有 getSessionFactory()、getSession() 和 closeSession(). 在各部分初始化数据对应的泛型 DAO 实现类中对 HibernateUtil 单例对象进行调用, 完成公共的 save()、delete()、update()、getById()、getByIds() 和 findAll() 方法的逻辑.

4 初始化方法应用测试

系统实施所采用的相关工具信息如表 1 所示.

表 1 系统相关工具信息表

名称	版本	作用
MySQL	5.5	数据库服务器
JDK	1.7	Java 开发工具集
Struts2	2.5	MVC 框架, 完成控制层功能
Spring	4.0	业务逻辑层容器
Hibernate	4.0	ORM 框架, 执行数据库操作
Tomcat	7.0	Java EE 服务器
JUnit	4.0	测试工具

要导入到系统中的初始化数据存放在 Excel 文件中, 共包括 16 张 worksheet, 其中技术职务表有数据 52 条、党政职务类别表有数据 6 条、党政职务名称表 28 条、岗位类别表 4 条、部门信息表 48 条、权限数据表 126 条、角色表有 27 条、部门账户表有 60 条、教工信息表 1248 条、科研级别表 47 条、一级指标表 5 条、二级指标表 30 条、三级指标表 109 条、业绩一级指标表 6 条、业绩二级指标表 23 条、业绩三级指标表 67 条, 文件总大小为 680 KB. 进行开发阶段的策略测试时, 在 JUnit 单元测试中调用初始化数据的代码, 如下代码所示:

```
//创建策略对象
InitStrategy initStrategy=new RunningStrategy();
//创建单例对象
JDBCUtil util=JDBCUtil.getInstance();
//创建建造者模式对象
SystemBaseDataDirector director=new SystemBaseDataDirector();
SystemBaseDataBuilder builder=new ExcelSystemBaseDataBuilder();
director.setBuilder(builder);
//执行初始化操作
initStrategy.initData(util, director);
```

执行后发现在 MySQL 中生成了对应的数据库以及相应的表, 每个表中都插入了对应 Excel 工作表中的数据. 第二次再执行会先删除上次创建的数据库, 再重新生成数据库及对应的表. 多次测试后平均完成初始化操作的时间为 195s. 针对运行时策略的测试, 由于运行时策略在系统正式部署时使用, 所以定义了初始化监听器执行上述代码, 将具体策略换成 RunningStrategy. 初始化监听器在 web.xml 中的配置位置位于 Spring IOC 容器的配置后, 因为只有 Spring 容器初始化后, 才能通过容器对象得到各部分初始化数据对应的 Service 组件. 经过测试, 初始化操作会在 Web 服务器启动后自动执行, 以后再次启动 Web 服务器初始化操作不再执行.

5 结论与展望

设计模式是面向对象程序设计中通用问题的一种可复用解决方案. 将建造者、策略和单例模式应用到基础数据的初始化设计中, 简化了初始化操作. 同时, 应用系统如需增加其他初始化数据, 只需更改产品、抽象建造者和具体建造者角色; 如果需要从其他存储方式中读取初始化数据, 只需重新定义具体建造者角色. 因此, 所提出的数据初始化方法具有较高的复用性、维护性和扩展性.

参考文献

1 肖东, 黄烟波, 索剑. 一种数据库的数据初始化方法. 惠州学院学报 (自然科学版), 2005, 25(6): 47–51.

2 陈烽, 陈蓉, 王跟成, 等. 设计模式在区域综合管网中的应用研究. 计算机技术与发展, 2015, 25(4): 193–196.
3 程朝晖, 朱杰. 基于观察者模式的气象卫星数据接收与预处理调度机制研究. 电子测量技术, 2017, 40(12): 22–28.
4 欧阳宏基, 葛萌, 陈伟. 一种改进的建造者设计模式. 咸阳师范学院学报, 2014, 29(6): 43–46. [doi: 10.3969/j.issn.1672-2914.2014.06.013]
5 段群, 吴粉侠, 欧阳宏基. 软件设计模式在目标考核系统中的应用. 自动化技术与应用, 2017, 36(9): 60–63. [doi: 10.3969/j.issn.1003-7241.2017.09.015]
6 韩兵, 江燕敏, 方英兰. 基于 JDBC 的数据访问优化技术. 计算机工程与设计, 2017, 38(8): 1991–1996, 2031.
7 宋松, 何得平, 张少华, 等. 基于 REST 构架的油田设备管理系统软件的设计. 计算机测量与控制, 2016, 24(9): 170–171, 175.
8 李杰. 基于 ORM 的轻量级数据持久化技术研究及应用. 计算机科学, 2010, 37(9): 190–193. [doi: 10.3969/j.issn.1002-137X.2010.09.046]
9 白晋国, 胡泽明, 孙红胜. 基于 RBAC 模型多级角色的 SQLite3 安全访问控制. 计算机系统应用, 2015, 24(5): 177–182.
10 欧阳宏基, 葛萌. 基于 S2SH 框架的煤炭企业生产统计管理系统. 计算技术与自动化, 2015, 34(3): 118–122. [doi: 10.3969/j.issn.1003-6199.2015.03.026]
11 阳瑞发, 冯彬. 一种导出复杂 Excel 报表的设计与实现. 计算机系统应用, 2013, 22(2): 155–157. [doi: 10.3969/j.issn.1003-3254.2013.02.039]
12 葛萌, 欧阳宏基, 陈伟. 单例设计模式的研究与实现. 微型电脑应用, 2017, 33(9): 68–70, 74. [doi: 10.3969/j.issn.1007-757X.2017.09.022]