一种导出复杂 Excel 报表的设计与实现①

阳瑞发, 冯 彬

(中国工程物理研究院 计算机应用研究所, 绵阳, 621900)

摘 要: 在管理信息系统中, 将报表导出到 Excel 是一个常见的功能. 通过分析报表的常见组成结构, 设计了一个动态模板, 并基于该模板利用 Java Excel API 将数据写入 Excel 文件, 实现复杂报表的导出.

关键词: Excel; jxl; 复杂报表; 动态模板; 工厂模式

Design and Implementation of Exporting Complicated Excel Report

YANG Rui-Fa, FENG Bin

(Institute of Computer Application, China Academy of Engineering Physics, Mianyang 621900, China)

Abstract: It's a common feature in Management Information System that to export report to an Excel file. By analyzing the general structure of report, this paper shows an approach to export complicated report to an Excel file through Java Excel API based on a dynamic template.

Key words: Excel; jxl; complicated report; dynamic template; factory pattern

日常工作中, 从管理信息系统(MIS)中将数据导出到 Excel 是一个很常见的功能[1], 如: 财务管理系统中需要经常导出复杂的财务统计报表, 采购管理系统中需要生成采购统计报表, 项目管理系统中需要统计与项目相关的各类报表, 等等. 而在基于 Java EE 的 MIS 系统中, 通常使用 Java Excel API(简称 jxl)将数据导出到 Excel 文件, jxl 是一个广泛使用的用于读写 Excel 文件的优秀的开源工具. 使用 jxl 导出数据时, 通常需要重写 servlet 的 service 方法, 在 service 方法中根据不同的使用场合定制 Excel 报表的格式, 并填充报表数据, 同时还需要在 web.xml 文件中对 servlet 进行配置. 然而, 这种实现方式缺乏灵活性, 一个 servlet 通常只能用于一种场合, 这在一定程度上造成了代码的重复冗余. 同时, 当报表格式发生变化时, 就需要修改相应的后台代码, 程序缺乏可重用性. 以上弊端导致实际应用中导出报表时, 开发效率低下, 后期维护更新困难等问题.

基于上述问题, 笔者通过分析报表的常见组成结构, 设计了一个动态模板[2], 并利用 jxl 实现了复杂报表的导出, 解决了上述的代码冗余问题, 提高了程序的可重用性, 能更好的支持后期的维护和更新.

1 动态模板设计

常见报表的组成包括四部分: 标题、表头、表体、表尾, 各部分与报表之间的关系如图 1 所示.

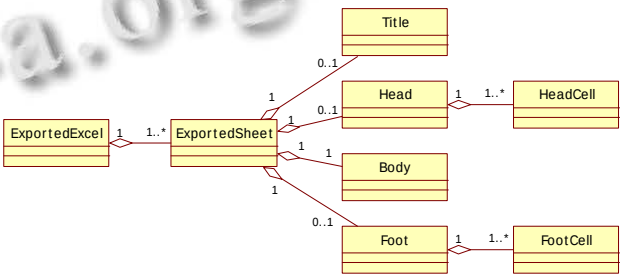

图 1 报表各组成部分的类图

其中, ExportedExcel 表示一个工作簿, 它至少包含一个工作表(ExportedSheet), 工作表也可称为报表,

① 收稿时间:2012-07-19;收到修改稿时间:2012-09-03

报表包含标题(Title)、表头(Head)、表体(Body)、表尾(Foot)四部分，表体是必需的，其余三部分是可选项. 表头由多个字段(HeadCell，也可称为列)组成，每个字段可能横跨或纵跨 Excel 的多个单元格，而表尾也类似的由多个 FootCell 组成.

基于该设计的动态模板有两种形式，一是使用 XML 配置文件，二是使用 JavaBean 在程序中动态配置. 如下所示是一个使用 XML 配置文件的示例.

```
<?xml version="1.0" encoding="UTF-8"?>
<ExportedSheet>
  <Title>
    <Static name="" height="" />
    <!-- <Dynamic method="" /> -->
  </Title>
  <Head>
    <Static>
      <HeadCell label="" x="" y="" width="" height="" />
      <HeadCell label="" x="" y="" width="" height="" />
      <HeadCell label="" x="" y="" width="" height="" />
    </Static>
    <!-- <Dynamic method="" /> -->
  </Head>
  <Body>
    <Dynamic method="" />
  </Body>
  <Foot>
    <Static>
      <FootCell label="" x="" y="" width="" height="" isSum="" sumMinY="" sumMaxY="" />
      <FootCell label="" x="" y="" width="" height="" isSum="" sumMinY="" sumMaxY="" />
      <FootCell label="" x="" y="" width="" height="" isSum="" sumMinY="" sumMaxY="" />
    </Static>
    <!-- <Dynamic method="" /> -->
  </Foot>
</ExportedSheet>
```

该配置文件指出了 Excel 报表的各组成部分的结构，在各组成部分中，Static 节点下的配置表明该组成部分的结构由 XML 进行配置，Dynamic 节点表明该组成部分的结构由程序动态实现，调用方法由 method 属性指定，实际应用中，Static 适用于事先已知报表结构的情况，而 Dynamic 适用于在程序运行时动态决定报表结构的场合，两种配置方式二选一即可. 此外，一个 HeadCell 节点表示表头的一个字段，其各属性中，label 表示名称，x、y 表示相对于表头左上角的坐标位置，width 和 height 分别表示本字段横向和纵向所占的 Excel 单元格的数目. 同样的，组成表尾的各 FootCell 中，各属性的含义与 HeadCell 相同，不同的是，此时，x、y 表示相对于表尾左上角的坐标位置，isSum 表示是否对表体的数据进行合计，合计数据在表体中的位置由 x、width、sumMinY、sumMaxY 共同决定.

有了该动态模板，即可基于该模板创建符合实际需要的各种样式的报表.

2　实现机制

通过工厂模式[3]的使用，可有效解决 servlet 冗余，程序可重用性差的问题，相关核心类的类图如图 2 所示.

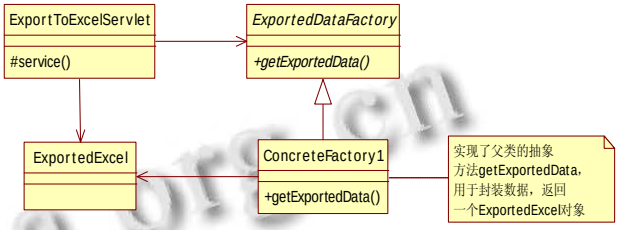

图 2　导出数据核心类的类图

其中，ExportToExcelServlet 是一个 servlet，用于处理数据导出的请求，并根据不同的场合调用对应的工厂类 ExportedDataFactory 的实例，由工厂类的实例创建所需要的产品——ExportedExcel——一个包含导出报表的结构及其数据的工作薄对象，之后在 servlet 中利用 jxl 按照 ExportedExcel 中指定的结构将数据写入 Excel 文件，并输出到客户端，从而实现报表的导出. 在 servlet 的 service 方法中，响应用户请求，并导出 Excel 的部分代码如下所示.

```
protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
```

```
        try {
          // 获取请求参数
          Map<String, String> params = getParameters(request);
          // 创建工厂类的实例
          ExportedDataFactory factory = Exported Data Factory.getInstance(params.get("clazz"),
        Boolean.valueOf(params.get("default")));
        ExportedExcel excel = factory. getExported Data(params);
          // 导出数据到 Excel
          generateExcelFile(excel, request, response);
        }
        catch (Exception e) {
          e.printStackTrace();
        }
      }
```

3 扩展应用

在上述设计的基础上, 使用 jxl 导出数据到 Excel 时, 只需要完成以下几步操作即可: 首先创建 Web 页面, 将导出请求提交到 ExportToExcelServlet, 然后创建符合实际需要的动态模板, 最后创建 Exported DataFactory 的实现类, 实现其抽象方法 getExportedData, 并根据动态模板封装数据, 创建 ExportedExcel 对象即可, 至于之后的操作, 开发人员无需再关注.

4 结语

通过动态模板的设计, 以及工厂模式的使用, 并利用 Java Excel API 实现了复杂报表的导出. 其中, 动态模板的使用提高了程序的灵活性, 在报表格式发生变化时也无需修改程序代码, 这有利于后期的维护和更新, 而工厂模式的引入则有效的解决了代码冗余的问题, 提高了程序的可重用性. 在该设计的基础上做导出 Excel 报表的应用时, 只需要实现三个步骤的扩展, 提高了开发效率. 目前, 该设计已在中国工程物理研究院的部分系统中得到了实际应用, 并取得了较好的应用效果.

参考文献

1 冼进.基于 Web 的智能报表设计研究.计算机与现代化,2011,191(7):161–164.
2 梁文斌,孙涌,张书奎.基于 Web 的通用报表系统设计与实现.计算机与现代化,2005,116(4):107–110.
3 宋文鹏,刘杰,贺红卫,董丽.基于工厂模式的构件获取技术.计算机工程与设计,2008,28(20):4865–4867.