基于数据驱动的图形界面开发方案

彭顺顺,周传生
(沈阳师范大学 软件学院,辽宁 沈阳 110034)

摘 要:基于数据驱动方法的图形界面开发方案以界面控件为对象,使用图形界面配置数据描述功能逻辑,通过图形界面配置数据的变动驱使系统的运行,实现功能逻辑。实现了图形界面配置数据和功能逻辑的分离,当需求发生变动时,只需要修改图形界面配置数据,功能逻辑即可实现。这不仅提高了图形界面的复用性,还降低了后期维护成本。

关键词:图形界面;数据驱动;配置数据;功能逻辑;重复利用性

中图分类号:TP311.52 **文献标识码**:A **文章编号**:1674-7720(2013)19-0001-03

User interface development program based on data-driven methodology

Peng Shunshun, Zhou Chuansheng
(Software College, Shenyang Normal University, Shengyang 110034, China)

Abstract: The coupling factor between user-interface and functional code had a great hindrance for the user-interface reusability so that reducing work efficiency. A program of user-interface development based on data-driven methodology, targeted at controls, described functional logic by the configured data of user-interface, ran the system by the modification of the configured data of user-interface, was to achieve functional logic. The program separated the configured data of user-interface from functional logic, when customer's demand was changed, functional logic came true after altering the configured data of user-interface. This not only enhances the reusability of user-interface but also makes a great saving on the maintaining-cost for user-interface.

Key words: user-interface; data-driven; configured data; functional logic; reusability

图形界面是连接用户和计算机的纽带,用户不仅要求图形界面功能丰富、操作简单,还在感官层次上要求操作的舒适感和愉快感[1],因此,在较短时间内开发出满足用户需求的图形界面成为企业竞争的焦点。目前,图形界面的开发是在集成开发环境下采用某种编程语言编写功能代码实现的[2]。这种开发方式存在以下问题:(1)图形界面与功能代码的高耦合[3],图形界面需求的改变会导致功能代码重复繁琐的修改,影响了开发效率,延迟软件发布时间;(2)图形界面重复利用性差[4],已经发布的图形界面不能被其他软件使用,即使是功能相似的软件,图形界面的重复使用也会导致开发人员根据客户的需求重新设定图形界面的布局和控件的功能,增加了开发人员的劳动量[5]。

随着数据库的潜力被无限发掘,越来越多的软件以数据为中心完成更多的任务,进入了数据驱动时代[6-7]。数据驱动方法借助数据交换工具(例如 XML)以数据为中心,根据数据的变动推动系统的运行[8-9]。数据驱动方法已经应用到软件测试的设计中,这种方法使测试脚本和测试数据分离,通过测试数据的变动驱使测试的运行,有效地缩短了软件开发周期,减少了开发人员的劳动量。

为解决图形界面开发存在的问题,把开发人员从繁琐的编码中解放出来,提出了基于数据驱动的图形界面开发方案。开发方案以 XML 形式描述图形界面的信息,根据 XML 提供的数据对界面控件进行布局和响应,组装满足需求的图形界面。

1 开发方案

1.1 开发流程

基于数据驱动的图形界面开发方案是以界面控件为对象,使用 XML 描述图形界面信息,由一个可重复利用的开发平台对图形界面数据进行转换、提取、分析、处理等操作,推动系统的运行,从而实现描述的功能。

根据数据驱动的原理，基于数据驱动的图形界面开发方案有以下几个步骤：

（1）根据图形界面的动态需求，基于数据驱动的图形界面开发方案利用现有的界面转换工具把图形界面控件信息编写到图形界面文件中，图形界面文件以XML形式描述；

（2）综合、抽象图形界面控件信息，根据图形界面控件信息初始化界面控件库；

（3）读取并分析图形界面文件中界面控件的外观、布局、行为等数据；

（4）根据获得的数据，调用界面控件库里的控件，设计图形界面控件的功能和布局，组合图形界面。

1.2 功能模块

从基于数据驱动的图形界面开发方案的步骤可以看出，基于数据驱动的图形界面开发方案由界面规范、功能界面、界面转换工具、图形界面文件、界面控件库、数据处理模块、运行环境几个模块组成，如图1所示。

图1 基于数据驱动的图形界面开发方案的总体框架

界面规范是描述操作的规定和要求，对界面输入、界面信息转换、界面数据的处理和界面布局等操作具有指导作用。

功能界面是由多个简单的界面控件组成，每个界面控件都有其基本属性。界面控件的属性包括大小、颜色、位置、字体类型等，这些属性可以是静态的，也可以是动态的，体现了界面控件的外观、布局和行为。

界面转换工具根据功能界面提供的信息编写图形界面文件。

图形界面文件是用来描述图形界面信息的，包括图形界面基本信息和界面控件信息，其文件结构如图2所示。

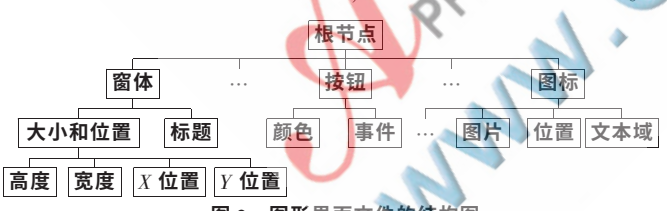

图2 图形界面文件的结构图

数据处理模块负责图形界面信息的读取、分析和显示。以下是数据处理模块完成的主要操作：

（1）采用工厂模式获取DOM解析器，这种模式将应用程序代码和DOM解析器彻底分离开来，实现更好的解耦；

（2）DOM解析器解析描述图形界面信息的XML文档，获取界面控件信息，并按照一定的机制将界面控件信息放入到迭代器中；

（3）数据处理模块识别界面控件信息，为了更好地识别界面控件类型，定义了界面控件库；

（4）当系统运行时，数据处理模块依次从迭代器中获得界面控件信息，根据界面控件信息设定界面控件的颜色、大小、位置等属性，形成图形界面。

界面控件库封装了控件集和图形库，在构造界面控件库时，首先分析界面控件的外观、功能等信息，抽象分离这些信息找出类似点，根据类似点将界面控件分类；其次对每类的界面控件来说，将界面控件具有的共同信息以编码形式保存下来，初始化界面控件库，通过编程语言的继承方式保存界面控件的共性；最后，设定界面控件信息的接口，当数据处理模块识别界面控件并对界面控件设定时，能保证界面控件之间的无缝连接，不需要重新设定界面控件信息。

基于数据驱动的图形界面开发方案将图形界面信息封装到XML描述文件中，把处理XML描述文件的操作以代码形式显示出来，这样不仅使图形界面与功能代码分离，便于图形界面的维护，还提高了图形界面的重复利用性。当需求发生变化时，图形界面文件随之发生改变，数据处理模块会重新处理图形界面数据，从而实现图形界面的功能，不需要手动修改功能代码。这种方法使开发人员的主要精力放在了数据的获取和分析上，减少了开发人员的劳动量，提高了开发效率。由于图形界面信息是用XML描述的，便于图形界面的扩展。

2 图形界面的设计

2.1 图形界面文件

XML是一种元标记语言，使用元标记来描述数据信息，具有良好的扩展性、自我描述性和层次嵌套性。图形界面的控件信息是多样的，并且控件之间具有层次嵌套性，图形界面的控件信息可用XML来描述。XML描述图形界面文件具有以下特点：

（1）图形界面文件的结构类似一棵树，是从根元素开始构造的；

（2）图形界面文件是对图形控件的描述，如果图形界面文件包括多个图形控件，就要对多个图形控件进行描述；

（3）图形控件的信息封装在一个自定义的标签内，在这个自定义的标签内，可以定义图形控件的类型、名称、外观、功能等子标签；

（4）图形界面文件不仅显示了界面控件的信息，还要显示界面控件之间的关系，以及定义界面控件响应的对象。

下面的XML文档是对按钮和标签的描述：

```
<? xml version='1.0' encoding='gb2312'? >
<page>
  <window>
    <window_title>图形界面</window_title>
    <window_visible>true</window_visible>
    <window_bounds>
      <window_height>300</window_height>
      <window_width>100</window_width>
```

```
    <window_LocationX>100</window_LocationX>
    <window_LocationY>100</window_LocationY>
    </window_bounds>
    ……
    <window_action>action.WindowAction
    </window_action>
  </window>
  <component type="button">
    <button_text>修改</button_text>
    <button_height>5</button_height>
    <button_width>10</button_width>
                              <button_background>red
    </button_background>
    …….
    <button_action>action.ButtonAction
    </button_action>
  </component>
  <component type="label">
    <label_text>图标</label_text>
    <label_icon>images/1.png</lable_icon>
    <label_locationX>10</label_locationX>
    <label_locationY>10</label_locationY>
    ……
    <label_action>action.LabelAction
    </label_action>
  </component>
</page>
```

2.2 数据处理模块

处理图形界面信息时，数据处理模块要解析XML图形界面文件，DOM（文档对象模型）定义了访问和操作XML图形界面文档的方式。获取DOM解析器并解析XML图形界面文件的步骤如下：

（1）调用DOM解析器工厂类的newInstance()方法获取解析器工厂对象；

（2）调用DOM解析器工厂的newDocumentBuilder()方法获取DOM解析器，将XML数据转换为常驻内存的树状结构；

（3）将XML文档解析成Document对象，读取XML图形界面文件里的数据；

（4）根据getDocumentElement()方法获得XML文件的根节点；

（5）根据树状结构的层次嵌套结构，结合递归方式依次建立节点对象，并将节点对象存放在迭代器内；

（6）通过循环方式访问节点对象，获得节点数据，即界面控件属性。

DOM解析XML图形界面文件流程图如图3所示，得到界面控件信息后，数据处理模块根据界面控件信息判断界面控件类型，为界面控件创建界面控件对象，通过界面控件对象把界面控件信息设置到功能代码中，运行功能代码，实现图形界面。

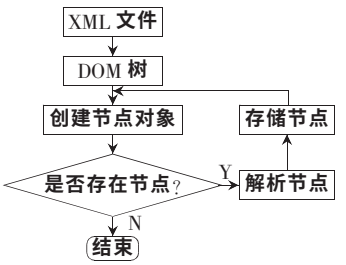

图3 DOM解析流程图

2.3 界面控件类

界面控件库是由描述界面控件外观、类型、行为等的界面控件和管理这些界面控件的控件组成。界面控件的外观决定了图形界面带给用户的愉悦感和可操作性，界面控件的类型决定了图形界面功能的多样性，界面控件的行为决定了图形界面的动态响应，图形界面不仅仅是静态的，也可以是动态的，例如：鼠标在按钮上时，按钮的颜色是红色的，鼠标离开按钮后，按钮的颜色就是蓝色的。

图形界面的控件无非是窗体、面板、标签控件与图标、按钮控件、文本控件、列表控件、进度条等类型，界面控件的属性也就是name、text、size、height、width等性质，界面控件库可以包括这些基本信息。构造界面控件库的步骤如下：

（1）为图形界面的每种类型的界面控件创建一个类，并且设置这些类继承的基类，将这些类放到一个文件夹中；

（2）在类的函数中添加相应的响应事件，例如键盘，keyPressed()是键盘上按键按下时触发的事件，keyReleased()是键盘上按键释放时触发的事件。

当应用程序运行时，数据处理模块读取图形界面文件的数据，通过反射机制调用界面控件库中的组件，设置组件的属性，完成图形界面的开发。

图形界面开发已经成为软件开发的重要组成部分，对于目前图形界面开发存在的问题，如何提高图形界面重用性和降低图形界面与功能代码的耦合度对于企业很重要。基于数据驱动的图形界面开发方案将界面控件信息以XML文件形式描述，围绕对界面控件信息的解析、处理等操作驱使系统运行。这种方式实现了以数据驱动软件运行，数据与功能代码的分离，当需求发生变化时，不需要修改源代码，减小了开发人员的劳动量。同时，XML具有较高的扩展性，用XML描述图形界面信息，便于后期的扩展。

参考文献

[1] 言金刚，樊东平，刘又诚.基于XML的统一用户界面描述[J].计算机工程，2002，28(6)：83-85.

[2] 杭欣静，蒋泽军，王丽芳.基于XML的用户界面动态生成框架的设计与实现[J].科学技术与工程，2009，9(9)：2492-2495.

[3] 张敦华.一种界面自动生成技术[J].计算机与数字工程，2004，32(5)：20-24.

[4] 李松.一种人机界面动态调度技术[J].微处理机，2012，

33(1):63-66.
[5] 靳京,桑楠,刘一.基于 XML 的嵌入式 Linux 系统用户界面定制[J].电子科技大学学报,2007,36(3):510-513.
[6] 侯忠生,许建新.数据驱动控制理论及方法的回顾和展望[J].自动化学报,2009,35(6):650-667.
[7] REDMOND M, BAVEJA A. A data-driven software tool for enabling cooperative information sharing among police departments[J]. European Journal of Operational Research, 2002,141(3):660-678.
[8] 梁玮. 数据驱动设计模式的研究及在 OTA 开发工具中的应用[D].北京:北京邮电大学,2009.
[9] 徐陋.数据驱动设计模式的研究及应用[D].广州:暨南大学,2006.

(收稿日期:2013-06-07)

作者简介:

彭顺顺,女,1987 年生,硕士研究生,主要研究方向:数据驱动方式,软件工程。

周传生,男,1966 年生,教授,主要研究方向:软件工程,分布式对象技术及其应用。