TẠO LƯỚI HIỂN THỊ VÀ XỬ LÝ DỮ LIỆU CỦA CÁC BẢNG DỮ LIỆU TRONG JAVA

HÀ VIẾT HẢI
Trường Đại học Sư phạm - Đại học Huế

Tóm tắt: Java, không giống như VB và C#, không có lưới hiển thị dữ liệu và bảng dữ liệu (RecordSet trong VB, C# và ResultSet trong Java). Vì vậy, người lập trình thường gặp khó khăn khi tạo các ứng dụng quản lý dữ liệu. Bài báo này giới thiệu một lớp Java do tác giả xây dựng có thể giúp giảm bớt những khó khăn này

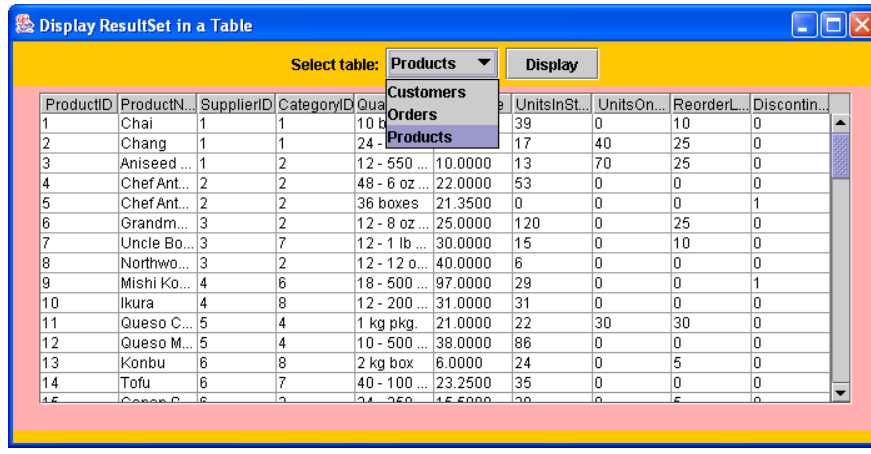
1. MỞ ĐẦU

Java hiện là một trong những công nghệ lập trình mạnh mẽ và được sử dụng rộng rãi nhất trên toàn thế giới. Ngôn ngữ này có khả năng đáp ứng những nhu cầu hiện tại để phát triển các ứng dụng từ đơn giản đến phức tạp, những ứng dụng chạy cục bộ trên máy đơn và những ứng dụng mạng ở các mức độ khác nhau. Về mặt truy cập cơ sở dữ liệu (CSDL), Java sử dụng các cầu nối ODBC, JDBC dạng cục bộ và mạng rất mạnh để kết nối tới các dạng CSDL khác nhau ở phạm vi kết nối khác nhau. Tuy nhiên, để hiển thị dữ liệu của một bảng dữ liệu (Data Table), Java không có sẵn công cụ thuận tiện để hiển thị dữ liệu ở dạng bảng. Điều này khác với VB, C#, khi sử dụng lớp lưới hiển thị dữ liệu (Data Grid) của chúng, ta có thể dễ dàng đưa dữ liệu của một bảng dữ liệu và hiển thị chúng trong các giao diện đồ hoạ một cách rất đơn giản và dễ dàng. Vì vậy, người lập trình thường gặp khó khăn và mất công sức khi lập trình Java để hiển thị và xử lý dữ liệu trong lưới. Bài báo này nhằm giới thiệu một lớp (class) Java có tên gọi là ResultSetTable do tác giả xây dựng để giảm bớt sự khó khăn này. Lớp này đã được các sinh viên Tin học sử dụng rộng rãi trong việc tạo lập các phần mềm quản trị CSDL bằng ngôn ngữ Java.

2. MỘT VÍ DỤ SỬ DỤNG LỚP ResultSetTable

Dưới đây là một ví dụ sử dụng lớp ResultSetTable. Ứng dụng kết nối với CSDL NorthWind trong bộ MicroSoft Office. Người sử dụng chọn tên bảng dữ liệu cần hiển thị trên combo box Select table, sau đó nhấn nút Display, dữ liệu của bảng được chọn sẽ được hiển thị ở trung tâm của giao diện. Đây là một ví dụ điển hình cho thấy sự cần thiết của việc hiển thị dữ liệu của một bảng dữ liệu (chứa trong một đối tượng kiểu ResultSet). Mã nguồn của ví dụ này được giới thiệu trong phần 5.

Tạp chí Khoa học và Giáo dục, Trường Đại học Sư phạm Huế
ISSN 1859-1612, Số 03(03)/2007: tr. 42-47



Hình 1. Giao diện của một ứng dụng sử dụng lớp ResultSetTable

3. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ LỚP ResultSetTable

Các yêu cầu chính về lưới hiển thị dữ liệu là

- Khả năng hiển thị dữ liệu của một bảng dữ liệu Java (ResultSet) trong lưới.
- Khả năng hiển thị lưới dữ liệu trong một khung có thanh trượt dọc.
- Một số tính năng khác để xử lý dữ liệu trong lưới.

3.1. Hiển thị dữ liệu của một bảng dữ liệu Java (ResultSet) trong lưới

Đây là yêu cầu cơ bản và quan trọng nhất. Java không có sẵn lớp nào thực hiện trực tiếp được điều này. Thông thường, bảng dữ liệu được hiển thị ở dạng lưới, với dòng đầu tiên chứa tên các trường, mỗi dòng tiếp theo chứa dữ liệu của một bản ghi.

Lớp gần nhất của Java để đáp ứng nhu cầu là JTable. Nó có khả năng hiển thị một lưới, tuy nhiên, dữ liệu phải đưa vào dưới dạng các Vector (một lớp của Java) chứ không phải phải dạng ResultSet. Trong số những cấu tử của JTable, cái thích hợp nhất cho việc hiển thị bảng dữ liệu là cấu tử có hai tham số, trong đó tham số thứ nhất là một vector (một chiều) chứa tên các trường dữ liệu, tham số thứ hai cũng là một vector (hai chiều) chứa dữ liệu của các bản ghi.

Do vậy, ta có thể lựa chọn phương án xây dựng lớp mới có chứa một bảng dạng JTable. Việc đưa dữ liệu từ ResultSet vào bảng được tiến hành qua hai bước. Bước thứ nhất, lần lượt đọc cấu trúc (tên các trường dữ liệu) và dữ liệu (các record) của ResultSet nạp chúng vào trong một vector (một chiều) chứa tên các trường và một vector (hai chiều) chứa dữ liệu. Bước thứ hai, gọi cấu tử của JTable với các tham số là hai vector của bước trên để khởi tạo bảng hiển thị dữ liệu có dòng tiêu đề chứa tên của các trường và các dòng nội dung chứa nội dung của các bản ghi.

Để lấy ra được cấu trúc của một ResultSet nói chung và tên của các trường của đối tượng này nói riêng, ta sử dụng phương thức getMetaData( ) của nó, phương thức này sẽ trả về một đối tượng dạng ResultSetMetaData. Từ đối tượng nhận được này, ta có thể lấy ra những thông tin cần thiết về cấu trúc của ResultSet như số trường, tên trường...

Việc lấy dữ liệu của các bản ghi từ một ResultSet thì đơn giản hơn nhờ sử dụng các phương thức có sẵn của lớp này. Để đưa dữ liệu của nó vào một vector, ta chỉ việc lần lượt đọc từng bản ghi của nó và nạp thêm vào vector cần tạo.

3.2. Hiển thị lưới dữ liệu trong một khung có thanh trượt dọc

Yêu cầu này là cần thiết trong trường hợp dữ liệu chứa quá nhiều dòng và không thể hiển thị toàn bộ cùng một lúc trên giao diện. Lúc này cần có thanh trượt để người sử dụng có thể duyệt tới phần dữ liệu mà mình quan tâm.

Lớp Java đáp ứng yêu cầu gần nhất là JScrollPane. Nó là một lớp chứa (container) có khả năng hiển thị một JTable ở bên trong và có thanh trượt dọc ở biên để duyệt qua nội dung của JTable này khi cần thiết. Do vậy phương án được chọn là khởi tạo một JSrcollPane với tham số là JTable chứa bảng dữ liệu được yêu cầu.

3.3. Một số yêu cầu bổ sung khác

Để thuận tiện hơn cho lập trình viên khi xử lý bảng dữ liệu, có thể thêm các phương thức khác để đáp ứng cho các nhu cầu thường gặp được yêu cầu trên lưới dữ liệu như bổ sung dòng, xoá dòng, chọn dòng...

4. MÃ NGUỒN LỚP ResultSetTable

Phần này chỉ giới thiệu phương thức cơ bản nhất là cấu tử khởi tạo một đối tượng của lớp từ một ResultSet. Độc giả có thể liên hệ với ban biên tập hoặc tác giả (theo địa chỉ *HaVietHai@Google.com*) nếu muốn có mã nguồn chi tiết hơn.

```
/**
 * <p>Title: ResultSetTable</p>
 * <p>Description: A class helping to create a Table displaying the contents of a resultset</p>
 * <p>Copyright: Copyright (c) 2003 - 2004</p>
 * <p>Company: Hue University</p>
 * @author Ha Viet Hai
 * @version 2.1
 */
import javax.swing.*;
import java.util.*;
import java.sql.*;

public class ResultSetTable{
        private Vector vectorColNames; //vector contains field names
        private Vector vectorData;          //vector contains all records
        private JTable jTableData;          //table displays recordset
        private JScrollPane jScrollPaneTable;    //scrollPane contains the table displaying recordset
        //constructor 1 - create a ResultSetTable from a Resultset
```

```
        public ResultSetTable(ResultSet rs) throws SQLException{
                ResultSetMetaData rm = rs.getMetaData();
                //determing the number of columns in each row of the resultset
                int numberCols = rm.getColumnCount();

                //create the vector contains field names
                vectorColNames = new Vector();
                for(int i=1; i<=numberCols; ++i){
                        vectorColNames.addElement(rm.getColumnName(i));
                }

                //create the vector contains all the rows of resultset
                vectorData = new Vector();//vector contains all records
                Vector row;                     //vetor contains one record
                while(rs.next()){               //read the current record and add its data to a vector
                        row = new Vector();
                        for(int col=1; col<=numberCols; ++col){
                                row.addElement(rs.getString(col));
                        }
                        vectorData.addElement(row);         //add the current record to vector
contains all records
                }

                this.jTableData = (new JTable(vectorData, vectorColNames));
                this.jScrollPaneTable = new JScrollPane(this.jTableData);
        }//constructor 1

// The other methods of class here have been omitted here
}//class
```

5. MÃ NGUỒN MỘT VÍ DỤ SỬ DỤNG LỚP ResultSetTable

Phần này trình bày mã nguồn sử dụng lớp ResultSetTable có giao diện như trong phần II. Để chạy được ví dụ này, trước hết cần tạo một System DSN có tên NWind qua cầu nối ODBC để kết nối tới CSDL NorthWind (file Northwind.mdb) trong bộ MicroSoft Office. Độc giả có thể sử dụng trực tiếp mã nguồn trong bài báo này hoặc liên hệ với ban biên tập hoặc tác giả nếu muốn có mã nguồn chi tiết hơn.

Phần mã đầu của lớp nhằm tạo giao diện như hình 1. Mỗi khi nút Display được bấm, tên của bảng dữ liệu cần được hiển thị được lấy từ ComboBox Select table. Tên này được dùng để đọc ra một ResultSet tương ứng từ kết nối Nwind. Sau đó lớp ResultSetTable được sử dụng để khởi tạo một JScrollPane chứa dữ liệu của ResultSet vừa đọc được. JScrollPane được hiển thị lên ở giữa giao diện như ở hình 1.

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;

/**
 * <p>Title: </p>
 * <p>Description: A resultset displaying program</p>
 * <p>Copyright: Copyright (c) 2003-2004</p>
```

```
 * <p>Company: Hue University</p>
 * @author: Ha Viet Hai
 * @version 2.0

 //Note: Before run this program, set a DNS (ODBC) pointing to the NorthWind database of Microsoft

public class ResultSetDisplayFrame extends JFrame implements ActionListener{
  JPanel contentPane;
  BorderLayout borderLayout1 = new BorderLayout();
  JPanel jPanelNorth = new JPanel();
  JPanel jPanelCenter = new JPanel();
  JPanel jPanelSouth = new JPanel();
  JLabel jLabelSelectTable = new JLabel();
  JComboBox jComboBoxTableName = new JComboBox();
  JButton jButtonRun = new JButton();

  Connection con;
  Statement stm;
  ResultSet rs;
  ResultSetTable resultSetTableRS;
  JScrollPane scp;

  //Construct the interface – A frame
  public ResultSetDisplayFrame(String title){
    super(title);
    contentPane = (JPanel) this.getContentPane();
    contentPane.setLayout(borderLayout1);
    jPanelSouth.setBackground(Color.orange);
    jPanelCenter.setBackground(Color.pink);
    jPanelNorth.setBackground(Color.orange);
    jLabelSelectTable.setText("Select table:");

    jComboBoxTableName.addItem("Customers");
    jComboBoxTableName.addItem("Orders");
    jComboBoxTableName.addItem("Products");

    jButtonRun.setText("Display");
    jButtonRun.addActionListener(this);
    jButtonRun.setActionCommand("Display");
    jComboBoxTableName.addActionListener(this);
    jComboBoxTableName.setActionCommand("Select Table");

    contentPane.add(jPanelNorth,  BorderLayout.NORTH);
    jPanelNorth.add(jLabelSelectTable);
    jPanelNorth.add(jComboBoxTableName);
    jPanelNorth.add(jButtonRun);
    contentPane.add(jPanelCenter, BorderLayout.CENTER);
    contentPane.add(jPanelSouth, BorderLayout.SOUTH);

    //Connect to the database
            try{
               Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
               con = DriverManager.getConnection("jdbc:odbc:NWind");
               stm = con.createStatement();
            }catch(Exception e){System.out.println(e);}
  }  //constructor
  //---------------------------------------------
  public void actionPerformed(ActionEvent e) {
```

```
   if(e.getActionCommand().equals("Display")){
                    String query = "Select * From " + jComboBoxTableName.getSelectedItem().toString();
                    try{
                              rs = stm.executeQuery(query);
                              resultSetTableRS = new ResultSetTable(rs);
                              scp = resultSetTableRS.getJScrollPane();
                              scp.setPreferredSize(new Dimension(650, 250));
                              jPanelCenter.removeAll();
                              jPanelCenter.add(scp);
                              this.validate();
                    }catch(Exception ex){System.out.println(ex);}
          }//Display
  }
  //---------------------------------------------
  public static void main(String args[]){
                    ResultSetDisplayFrame frame1 = new ResultSetDisplayFrame("Display ResultSet in a
Table");
                    frame1.setSize(new Dimension(700,400));
                    frame1.show();
  } //main()
}  //class
```

6. KẾT LUẬN

Khả năng đáp ứng cho việc lập trình quản trị CSDL của Java là rất mạnh. Tuy nhiên, tại một số điểm, ta cần phải viết thêm những tiện ích bổ sung để phát triển khả năng của nó. Lớp ResultSetTable là một sản phẩm đi theo hướng này. Hy vọng nó cung cấp thêm cho những ai quan tâm đến mảng lập trình quản trị CSDL nói riêng và lập trình trên Java nói chung một chút ý tưởng để cho công việc của mình được tiến hành một cách nhanh chóng và thuận tiện hơn. Độc giả có thể tiếp tục phát triển lớp này để nó thể có được những tính năng tương tự như của lớp DataGrid của VB và C#.

TÀI LIỆU THAM KHẢO

[1] Nguyễn Tiến - Nguyễn Văn Tâm - Nguyễn Văn Hoài (2001), *Java, lập trình cơ sở dữ liệu*, NXB Thống kê.

[2] Aptech World Wide (2002), *Advanced Java*, Giáo trình giảng dạy Java.

[3] Sun MicroSystems Inc., *Java Tutorial*, version 1.4.1.

[4] Sun MicroSystems Inc., *JavaTM 2 SDK*, Standard Edition Documentation, version 1.4.1.

Title: CREATING A GRID FOR DATA DISPLAYING AND PROCCESSING OF DATA TABLES IN JAVA

Abstract: In Java, not like in VB and C#, there isn't the grid that can display directly the data of a data table (RecordSet in VB, C# and ResultSet in Java) so programmers often have many difficulties when creating the data management applications. This paper refers to present an author's class that helps this work becomes easier.

ThS. HÀ VIẾT HẢI
GV Khoa Tin học, Trường Đại học Sư phạm - Đại học Huế.