Client Billing Application [1,2]

by David Scuse and Christopher Batty

Department of Computer Science, University of Manitoba, Winnipeg, Manitoba, Canada

Last revised: October 23, 2003

Overview:

In this document, we describe a simple application that keeps track of payments made to client accounts.

Setup:

The current version of the application is configured to use an HSQLDB database (although connections to a MySQL database and an MS Access database are also included in comments at the beginning of ClientBillingDB.java).

```
//Load the hsqldb driver
Class.forName("org.hsqldb.jdbcDriver");
url = "jdbc:hsqldb:Clients"; // stored on disk mode
//Open the connection
conn = DriverManager.getConnection(url, "SA", "");
System.out.println("Opened HSQLDB database.");

//Load the MySQL driver
//Class.forName("com.mysql.jdbc.Driver");
//url = "jdbc:mysql://localhost/clients";
//Open the connection
//conn = DriverManager.getConnection(url, "root", "");
//System.out.println("Opened MYSQL database.");

//Load the MS Access drivers
//Remember that in Windows 2000 and XP, ODBC connections are defined
// in the Control Panel --> Admin Tools --> Data Sources
//Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
//url = "jdbc:odbc:Clients";
//Open the connection
//conn = DriverManager.getConnection(url, "", "");
//System.out.println("Opened ACCESS database.");
```

To use an HSQLDB, download the HSQLDB system from http://hsqldb.sourceforge.net/ and copy hsqldb.jar into the JRE lib/ext folder (this jar file is also included in the HSQLDB folder). A sample database (clients.properties and clients.script) has been included in the project folder.

To use a MYSQL database, download the MYSQL server from http://www.mysql.com/, install the server and create the database. A script that creates the database has been included in the MYSQL folder in the project folder. If MYSQL is used, it is worthwhile downloading the MYSQL Control Center which provides GUI access to the databases.

To use an MS Access database under Windows, an ODBC connection that links the database, "clients.mdb", and the logical database name used in the system, Clients, must be defined. ODBC connections are defined in the control panel (ODBC) on earlier versions of Windows and are defined in "DataSources (ODBC)" in the Administrative Tools section of the Control Panel in more recent versions of Windows.

Running the System

Once the database connection has been made, the system can be run. Currently, there is not a lot of functionality, the system is intended merely to illustrate the use of various SWT controls in a working system.

A collection of clients has been predefined in each database. The following screenshots show the contents of the Access database. Clients can be selected clicking on the client name in the Current Clients panel on the left of the screen.
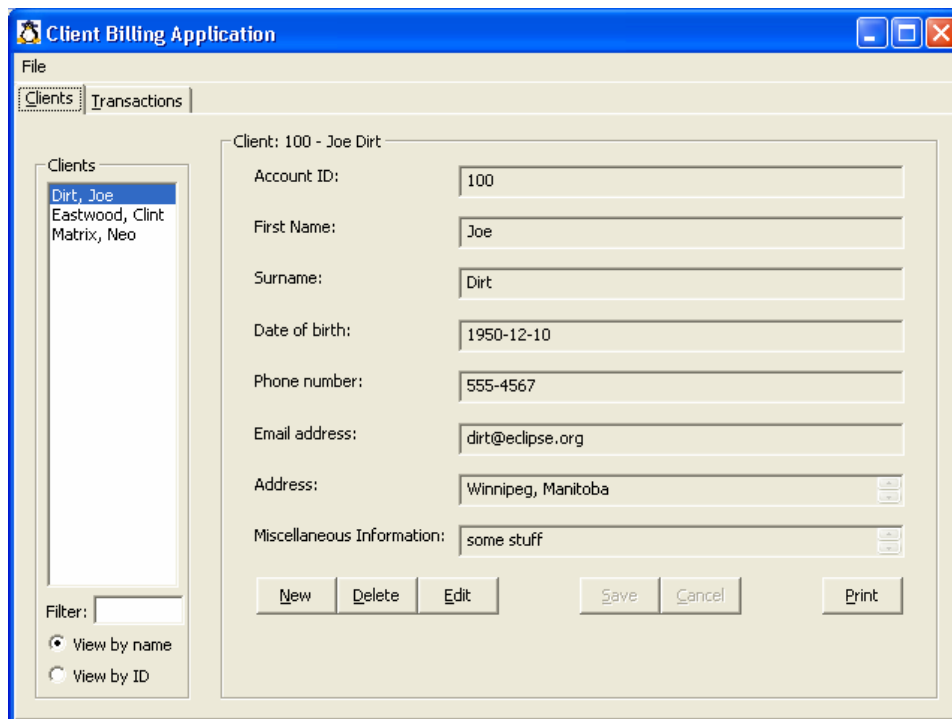
To access clients by the client identifier, click the View by ID button under the Current Clients panel.
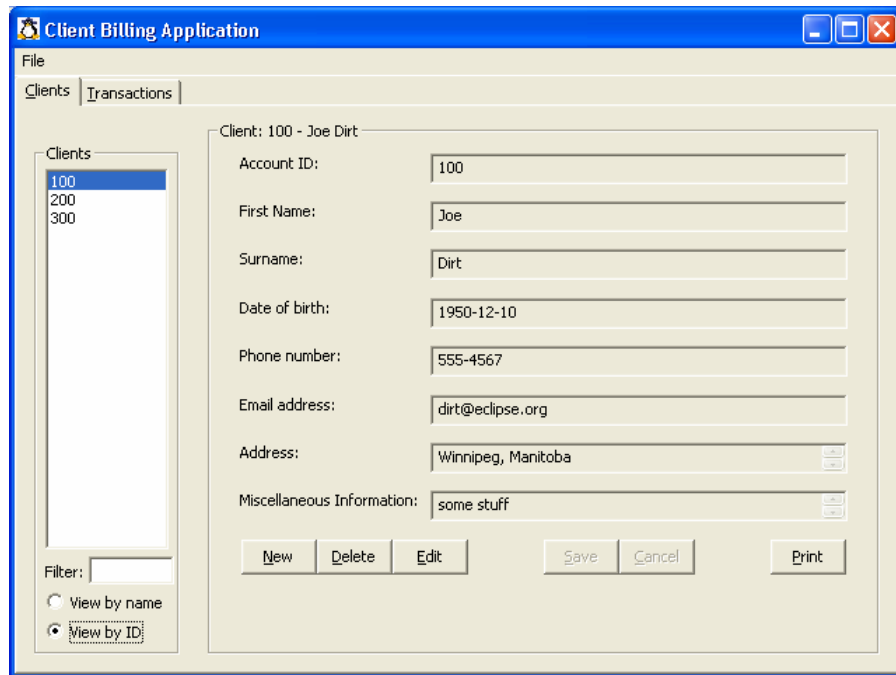
To locate a specific client (or clients), type any part of the client's name (or ID when the ID view is used) into the filter box and then type "enter". Note that after the filter has been applied, the client must still be selected by clicking on the client's name (or ID).
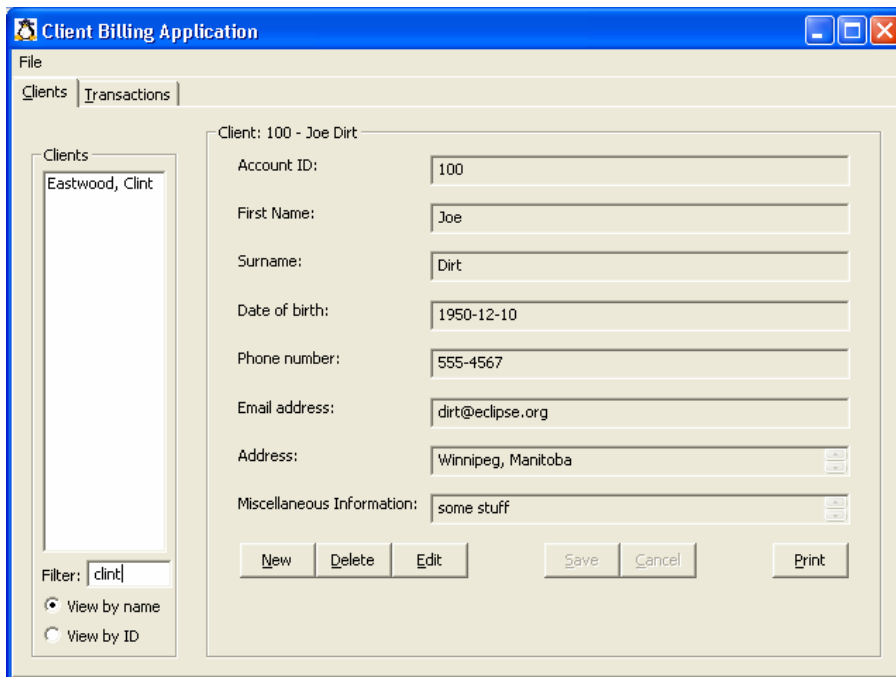
To return to the entire list of clients, erase the characters in the filter box and refresh the view either by typing enter or by clicking on the View by name or View by ID button.
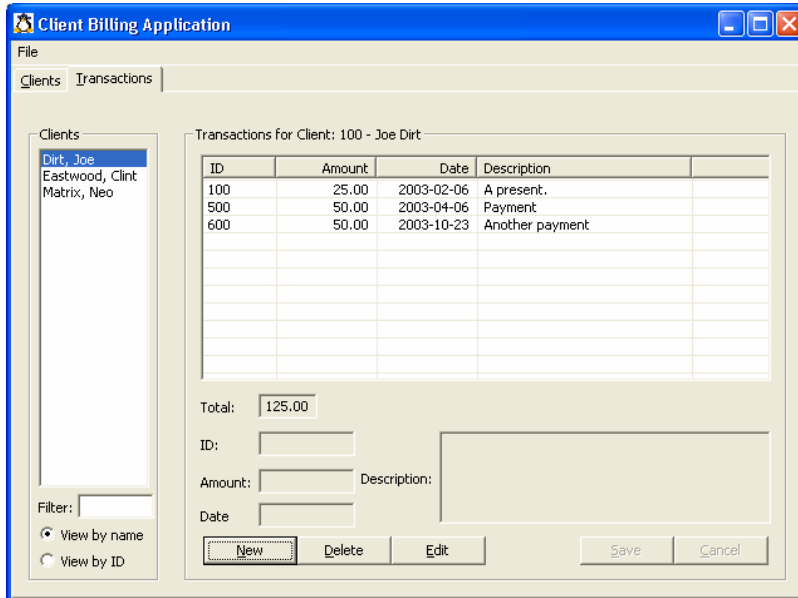
For each client, a list of payments that have been made against the client's account can be viewed on the Transactions tab.
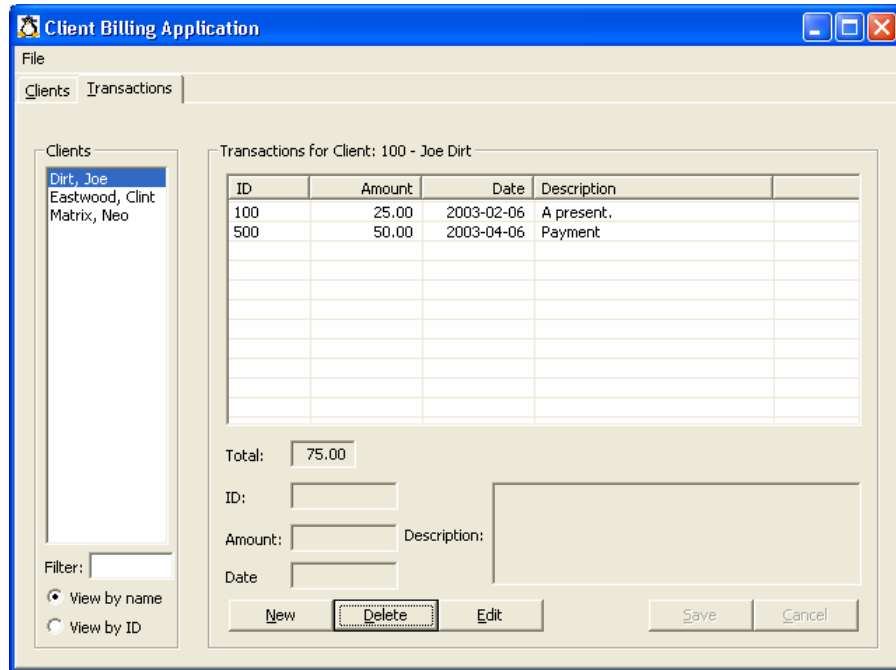
To enter a new transaction, click the New button and then fill in the necessary information about the transaction.
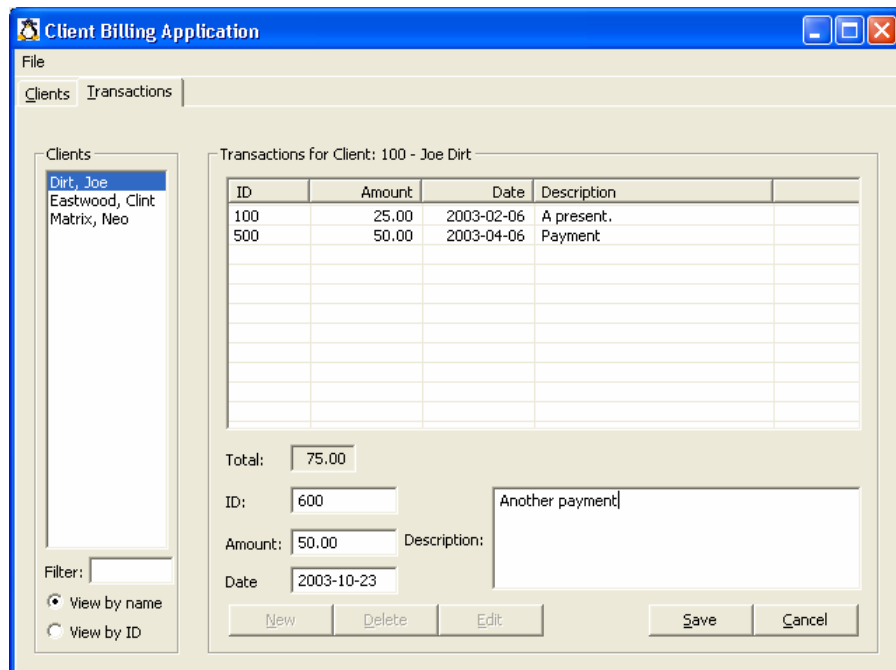
Click Save and the transaction is added to the window.

Transactions can be deleted by selecting a transaction and then clicking the Delete button. Similarly, transactions can be edited by selecting a transaction, clicking the Edit button, making the changes, and then clicking the Save button.

Layout Managers:

The controls used in the development of the GUI for this application were initially given "hard-coded" locations and sizes (via the setSize() and setLocation() methods). (The source for the GUI is in the class ClientBillingUI.java.) If the size of the window in which the application is run is made smaller by the user, some controls will no longer be visible. Note that when the window is made smaller, scroll bars are not available.
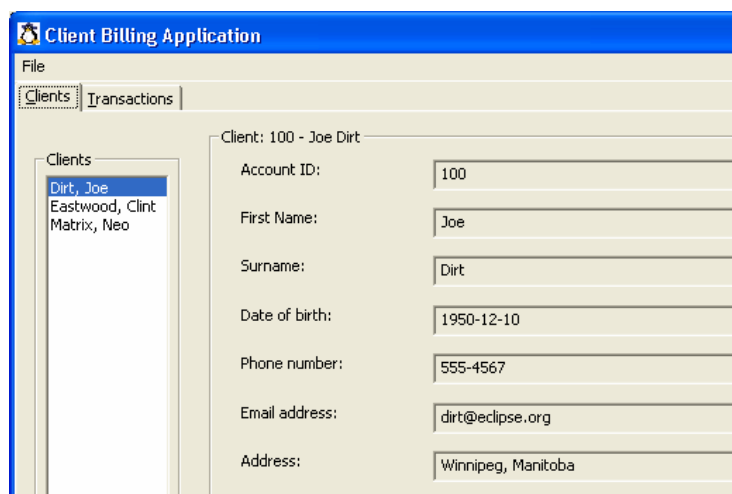
For many applications, this behaviour is not a problem since the window is unlikely to be resized. However, for some users and applications, resizing windows is a frequent occurrence. The appropriate use of a layout manager can made the resizing of a window more convenient since the components are resized as the window is resized.

We rewrote the GUI for the client billing application to use layout managers. Note that only ClientBillingUI.java had to be modified, the other classes remained unchanged.

When the application is run, it has the following initial appearance.
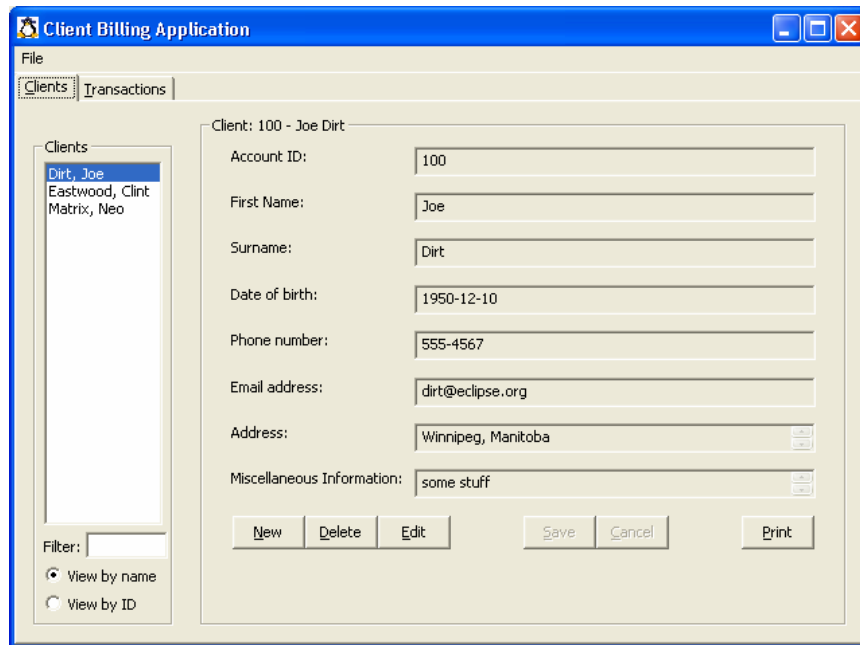
Now if the window is resized, controls do not simply disappear. Although, some information is truncated if the window is resized significantly.
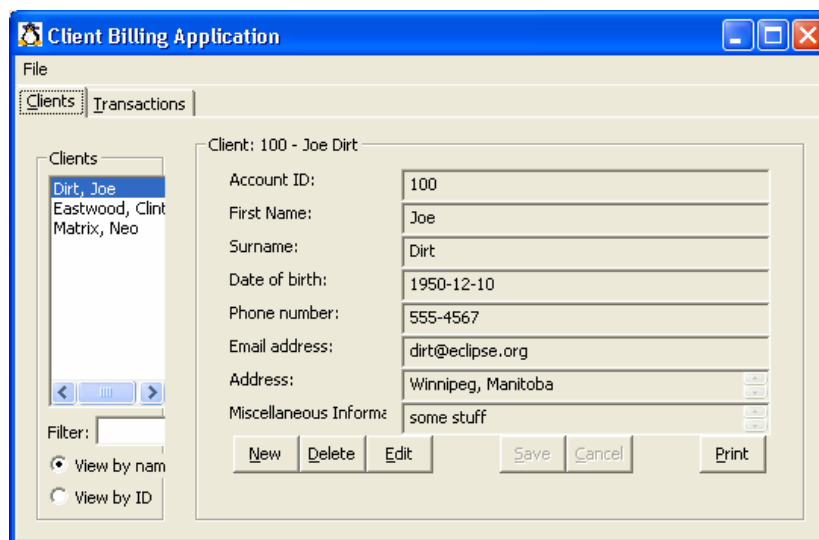
The number of lines of code for both the original (without layouts) version and the layouts version are approximately the same. However, the layouts version has significantly less complexity since absolute locations and sizes are not used. Instead, with the form layout, the position is specified as a percentage of the available space. This permits designer to specify more or less where the control is to be placed and permits the layout manager to position controls in a more "intelligent" manner.

Another design point to note is that the windows are subdivided into composites (groups) and controls are added to the appropriate composite/group. This organization provides more flexibility than if controls are added directly onto the underlying shell or tab folder.

Printing:

Since most billing systems require the ability to print invoices or statements, a Print facility was included on the Clients tab. Clicking the Print button generates a printed copy of the current client's information on the user's default printer. The source statements for this facility can be found in the `printTransButton` in ClientBillingUI.java. The printing is performed using a StyledText widget.

The following information on Joe Dirt would be printed.

```
Client Report - Joe Dirt
------------------------
Account ID: 100
First Name: Joe
Last Name: Dirt
Date of birth: 1950-12-10
Phone Number: 555-4567
Email: dirt@eclipse.org
Address:
Winnipeg, Manitoba
Miscellaneous information:
some stuff

Transactions
------------
100 25.00 2003-02-06 A present.
500 50.00 2003-04-06 Payment
600 50.00 2003-10-23 Another payment

Total balance: 125.00
```

Missing Functionality:

This system is not quite complete, there are various extensions that would have to be included before the system could actually be used.

A negative value may be entered as a transaction amount but credits and debits would have to be improved to prevent confusion on the part of the user.

Having buttons that filter the transactions by year and month would also be helpful in a realistic system.

The printing facility could be expanded to permit the user to select either an individual client or all clients and also to permit the user to print all transactions or selected transactions (for example, transactions for the current month or between two user-specified dates).

Bugs:

When the description field of a transaction is edited, sometimes the change is applied to all transactions with the same transaction ID, even though the client number is not the same. ????