

Time Tracker Application^{1, 2}



by Raphael Enns and David Hadaller

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

Last revised: March 31, 2004

Overview:

In this tutorial, we describe a simple application that keeps track of employees, projects that the employees work on, and any transactions an employee may have while working on a project. This application is designed similar to the Client Billing application from the *Client Billing Application* tutorial, but with extra complexity and features.

Setup:

The current version of the Time Tracker application is configured to connect to a Microsoft Access database. Connections to an HSQLDB database and a MySQL database are also included in comments at the beginning of TimeTrackerDB.java.

To use a Microsoft Access database under Windows, an ODBC connection that links the database, "timetracker.mdb", and the logical database name used in the application, TimeTracker, must be defined. You can create an ODBC connection under Windows 2000 and Windows XP in "Data Sources (ODBC)" in the Administrative Tools section of the Control Panel. In previous versions of Windows, you can create an ODBC connection in the "ODBC" section of the Control Panel. To create the ODBC connection, add a new data source using the Microsoft Access Driver. Point the database to the "timetracker.mdb" file and specify the name as "TimeTracker". The database should now open when you run the Time Tracker application.

To use an HSQLDB database, download the HSQLDB system from <http://hsqldb.sourceforge.net> and copy hsqldb.jar into the JRE lib/ext folder.

¹ This work was funded by an IBM Eclipse Innovation Grant.

² © Raphael Enns, David Hadaller, and David Scuse

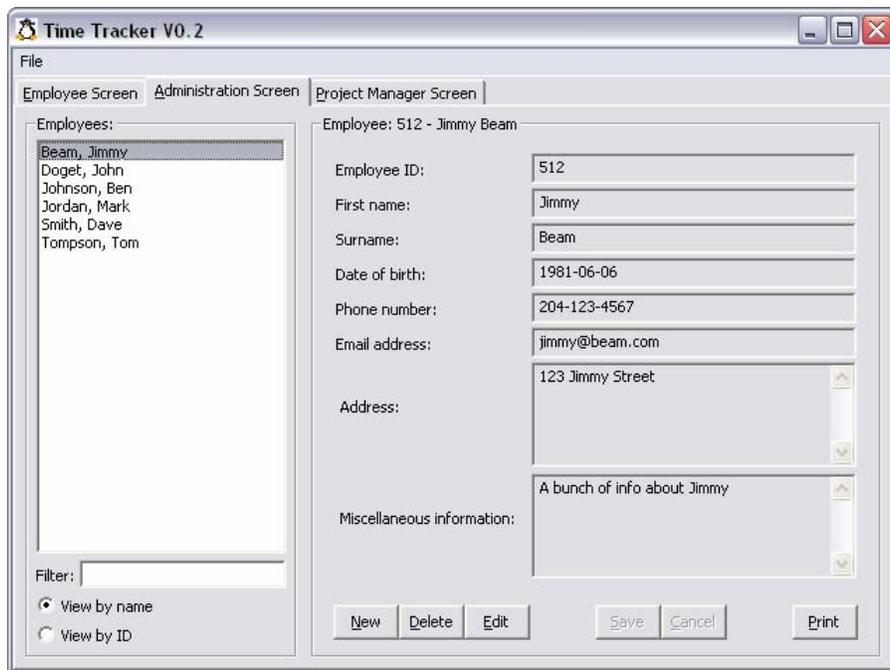
To use a MYSQL database, download the MYSQL server from <http://www.mysql.com>, install the server and create the database. If MYSQL is used, it is worthwhile downloading the MYSQL Control Center which provides GUI access to the databases.

The GUI of the Time Tracker application uses SWT. To be able to compile and run the application you must have swt.jar in your build path. For more information on how to do this, see the *Installing Eclipse* tutorial.

Running the Application:

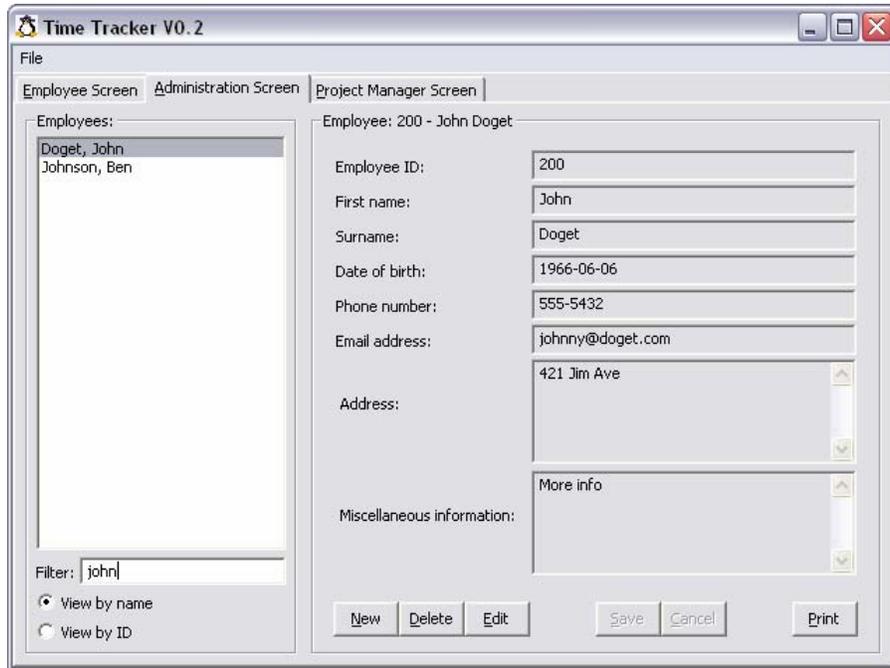
Once the database connection has been made, the application can be run. Although the Time Tracker application has more functionality than the Client Billing application, the Time Tracker application is still fairly simple. Like the Client Billing application, it is intended to illustrate the use of various SWT controls in a working application.

A collection of employees has been predefined in the database. You can add, delete, and edit employees from the Administration Screen shown below.

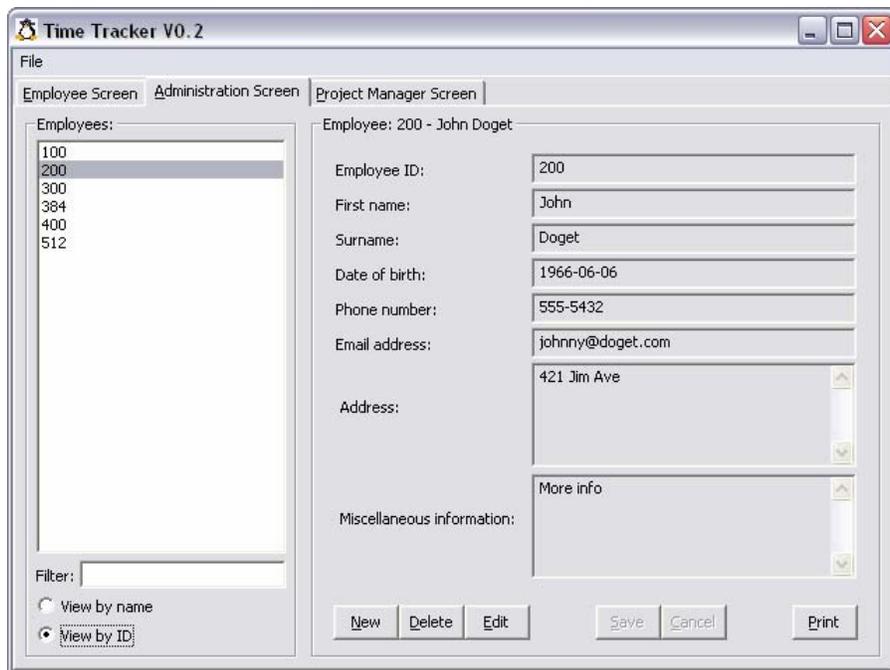


A list of employees is shown on the left with information on the selected employee on the right.

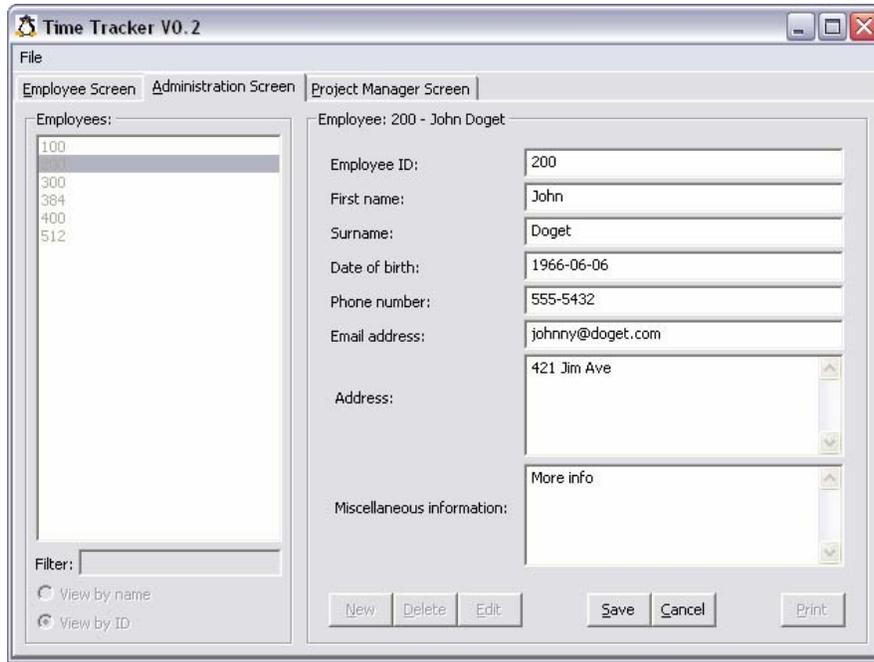
To locate a specific employee (or employees), type any part of the employee's name (or ID when you are viewing employees by ID) into the filter box and press Enter. Only the employees matching your filter will be displayed. To show all the employees again, clear the filter box and press Enter.



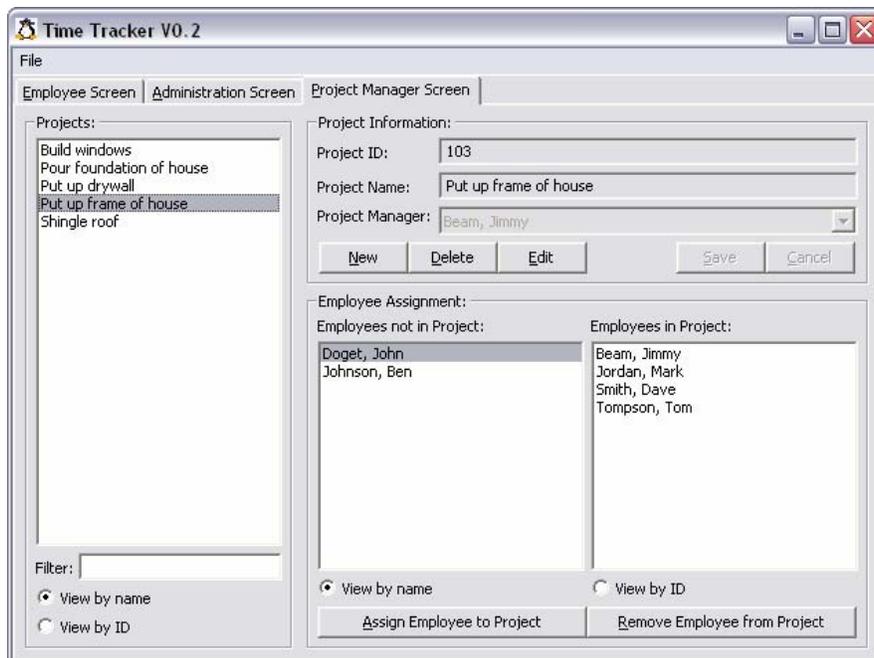
Employees can also be listed by their ID. If you click on the View by ID radio button, the Administration screen looks like the following:



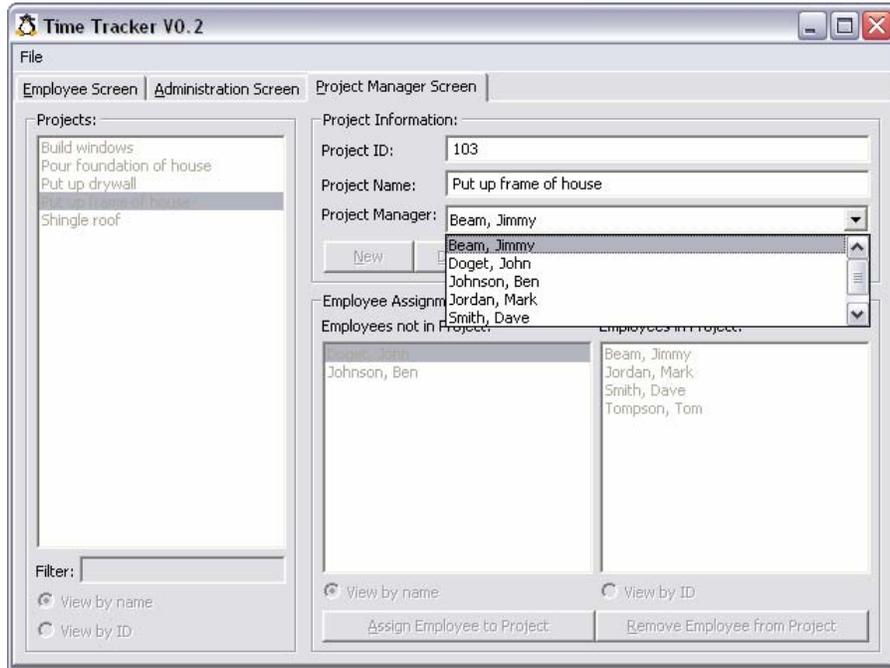
Clicking on the New or Edit buttons allows you to edit the text fields. Once you have filled in the correct information, click the Save button. Note that the employees list, filter box, view radio buttons and the new, delete and edit buttons are disabled during while modifying the employee information.



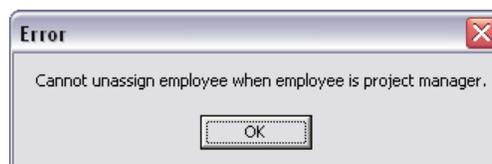
In addition to the collection of employees, the database also contains a collection of projects that employees can be part of. You can configure the projects from the Project Manager Screen.



Selecting projects is the same as selecting employees. The filter box and view radio buttons work the same as well. When you click on the New or Edit buttons, a combo list becomes available for Project Manager. The combo list displays all the employees and allows you to select one of them. This sets the project manager to one of the available employees.

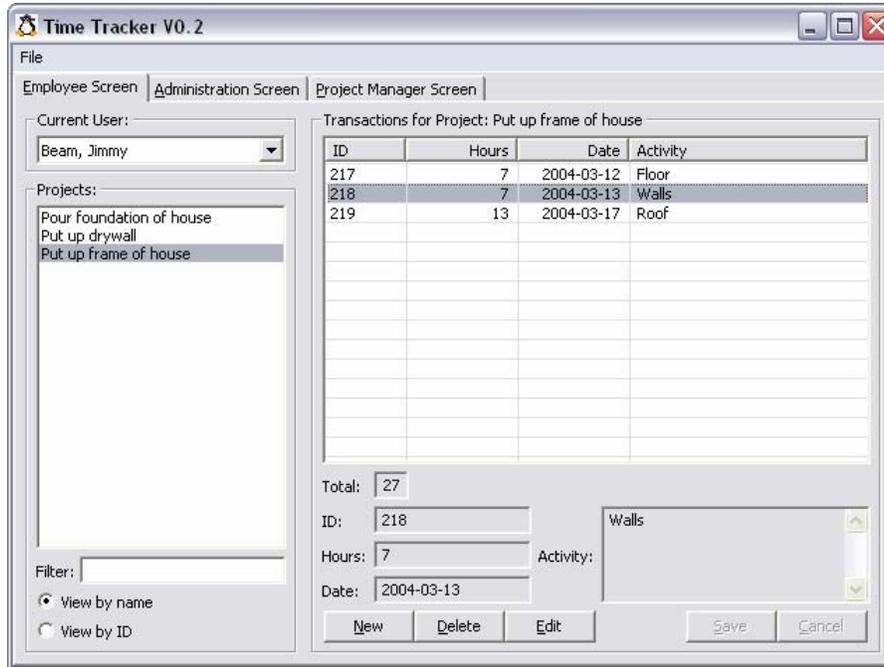


You can assign or remove employees from the selected project in the Employee Assignment group. To assign an employee to the project, select an employee in the left list and click Assign Employee to Project. To remove an employee from a project, select an employee in the right list and click Remove Employee from Project. The selected employee then moves from one list to the other. One thing to note is that you cannot remove an employee from a project if the employee is the project manager of that project. The application is set up so that every project must have a project manager. If you wish to remove an employee who is a project manager, you must first change the project manager of the project to a different employee or simply remove the project. You can change the project manager using the Project Manager combo list. If you try to remove the project manager from a project, the following message box appears:



Similar message boxes appear when a user tries to perform an operation not allowed by the application such as trying to delete an employee if no employee is selected.

In the Employee Screen you can modify the transactions for each employee.



The Current User combo list is similar to the combo list in the Project Manger Screen. From this combo list you can select an employee from a list of all employees. All of the projects that the employee is in are displayed below the combo list. Selecting a project from the project list displays the corresponding transactions on the right side of the window in a table format.

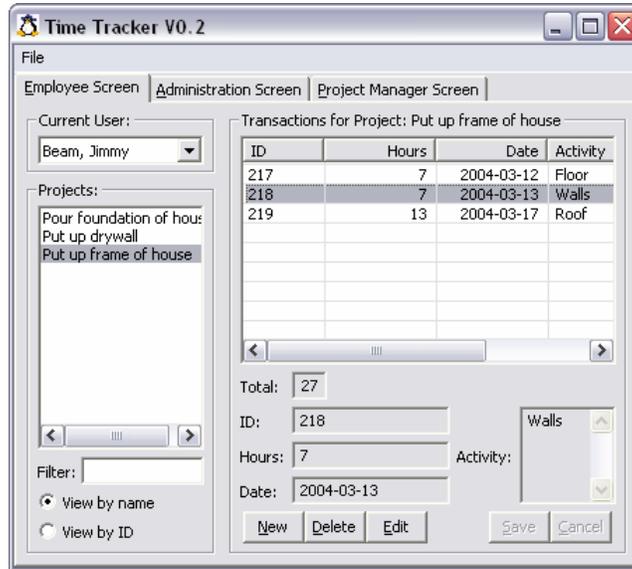
The filter box and view radio buttons work the same as in the other screens. Creating or editing transactions is also the same as creating or editing employees and projects.

Layout Managers:

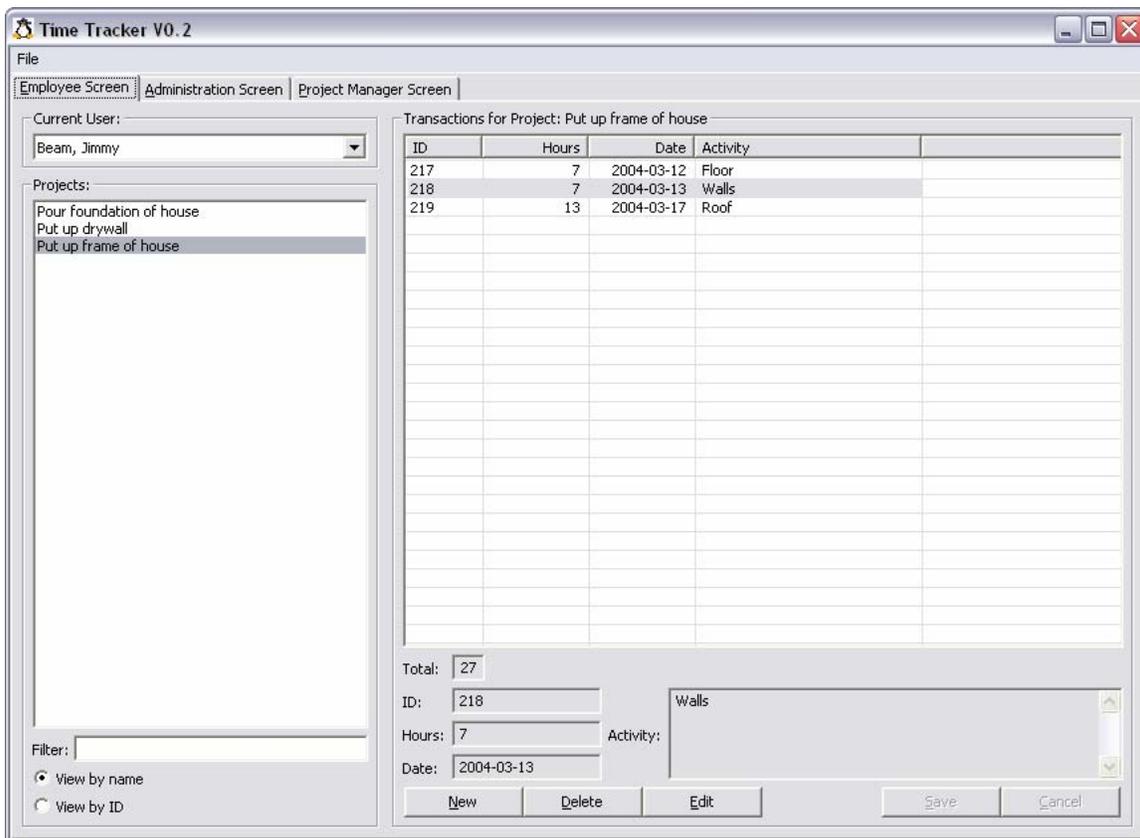
The Time Tracker UI was written using layout managers. Because layout managers were used, individual controls in the application are able to be resized “intelligently” when the application window is resized.

Three types of layout managers were used in this application. The most used was the FormLayout. This layout manager positions edges of controls a fixed offset away from percentage points or other controls. Another layout manager used was the GridLayout, which lays out controls in a grid fashion. The FillLayout layout manager was used for the button bars.

The FormLayout and GridLayout layout managers allow you to specify which controls will resize when the application window is resized and by how much. Shown below is the application resized to smaller than its initial size.



Shown below is the application resized to larger than its initial size.



As you can see in the above two images of the application, some controls stay the same size and stay close together while other controls change their size. If we had a lot of transactions, it would be useful for the size of the table to grow vertically when we increase the vertical size of the window, but it does not make sense to increase the size of the text fields below the table. Therefore we make the table grow vertically but keep the size of the text fields constant. The same thing is done with other controls in the application.

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.

Printing:

To facilitate the ability to print the information stored in the database, a Print button was included on the Administration Screen. Clicking the Print button generates a printed copy of the current employee's information on the user's default printer. The source statements for this facility can be found in the `setupAdministrationPane()` method in `TimeTrackerUI.java`. The printing is performed using a `StyledText` widget.

The following information on Jimmy Bean would be printed:

```
Employee Report - Jimmy Bean
-----
Account ID: 512
First Name: Jimmy
Last Name: Beam
Date of birth: 1981-06-06
Phone Number: 204-123-4567
Email: jimmy@beam.com
Address:
123 Jimmy Street
Miscellaneous information:
A bunch of info about Jimmy

Transactions
-----

ID      Project Hours      Date      Activity
--      -
101     105      6      2004-02-27  Step 1
102     105     17      2004-03-31  Step 2
217     103      7      2004-03-12  Floor
218     103      7      2004-03-13  Walls
219     103     13      2004-03-17  Roof
321     101      5      2000-04-04  Step 1
322     101     18      2000-04-05  Step 2
3221    101     21      2003-12-23  Extra Work on Project

Total hours: 94
```

Missing Functionality:

This application is well on its way to becoming a completely functional application. However, in order to make this a finished product, several extensions would have to be included.

Being able to filter transactions by date would be really helpful in a realistic application.

Many checks on input would have to be added. Negative numbers should not be allowed in the hours field of transactions or as an ID for employees, projects, or transactions.

Currently the class that handles all interaction with the database, `TimeTrackerDB.java`, handles all SQL errors by simply printing a stack trace to standard output. A more user friendly alternative would be to have checks in place in the UI that makes sure that input from the user is in the proper format, and if not, displays a helpful message to the user. Two examples of this kind of error are having an incorrect date format or including an apostrophe in a string to be placed in the database.

The printing facility could be expanded to permit the user to select more than one employee, as well as filter out unwanted projects / transactions (for example, only print transactions for the current month or between two user-specified dates).

Extra Notes:

This application performs some domain processing in the UI class. This is not coded according to the MVC pattern. To conform to the MVC pattern, any processing not directly related to the UI should be delegated to another class. This allows the UI to be easily replaced or modified.

The majority of the GUI of the Time Tracker application was designed by hand. In order to ease the completion of the GUI, the SWT Designer was used to extend and modify the previously existing GUI. The SWT Designer was able to read in the hand-coded GUI and modify it using the style already in place. Only small changes were made to make the code generated by the SWT Designer conform to the existing style that was used to code the GUI. For more information on using the SWT Designer to design GUIs, see the tutorial, *An Eclipse GUI Builder*.