Utilities and Excel Interface.

1 Summary

The Utilities and Excel interface libraries provide generic functionality for .Net applications:

- Support for Filtering, Sorting and Editing of data,
- Segregation of Duty,
- Column Design,
- Multi-language support. The active language is User dependent,
- User Logging,
- User Blocking after periods of inactivity,
- Timer initiated termination of an Application,
- Semaphore support for fast updating.
- Flexible connection support for switching between Live and Test databases.
- Filtered and sorted data export to Excel.
- Help textbox in every form.

Project Number		**Distribution**	
Software / File		**Communication**	Robin Baines Schansdijk 6 8434 ND WASKEMEER E-mail: robin@baines.nl

Rev	Date	Modification	By		Approval
A	20100231		RPB		
B	20100831	Include testing.	RPB		
C	20110118	Improvements added when aligning all .Net applications and removed first version of the user management.	RPB		
D	20110218	Added New user procedure and Auditing of Users and Groups	RPB		
E	20120711	Examples and figures removed which contained Proprietary information.	RPB		
F	20141230	Logging of changes to user and group-form tables.	RPB		
G	20200704	Documented Semaphore support for fast updating	RPB		
H	20221111	Help Text available in every form.	RPB		
I	20251025	Added Performance Logging			

UTILITIES AND EXCEL INTERFACE....1

1 SUMMARY....1

2 INTRODUCTION....3

3 AUDITING OF USERS AND GROUPS....3

4 FILTERING AND SORTING IN A DATA GRID....3

4.1 Sorting....3
4.2 Filtering....4
4.3 Editing....4
5 MULTI LANGUAGE SUPPORT, SEGREGATION OF DUTY AND COLUMN DESIGN....4
5.1 Languages....4
5.2 Segregation of Duty....5
5.3 Column Design....7
5.4 Finding Names of Forms, Columns etc....7
5.5 Column And Table adjustment in context....9
6 HELP TEXT....9
7 LOGGING CHANGES TO GROUPS AND USERS....10
8 USER LOG AND USER BLOCKING....10
8.1 The User Log....10
8.2 User Blocking....10
9 THE OPTIONS FORM AND THE PARAMETER TABLE....12
10 AUTOMATIC TERMINATION OF APPLICATION....13
11 CONNECTION STRINGS AND TEST DATABASE....14
12 THE EXCEL INTERFACE....14
13 NEW USER PROCEDURE....14
14 DISTRIBUTION OF NEW VERSIONS....15
15 USING UTILITIES.DLL....15
16 SEMAPHORE SUPPORT FOR FAST UPDATING....15
17 PERFORMANCE LOGGING....16

2 Introduction

The Utilities provide generic functionality for .Net applications

- Enhanced DataGrids for Filtering, Sorting and Editing of data,
- Segregation of Duty by assigning a user to a group and by defining which forms are active for each group.
- Column Design.
- Switch columns on and off in forms and Excel export.
- Define sequence and width of columns in tables.
- Multi-language support. The active language is User dependent.
- User log for recording times when a user starts or stops the application and average length of time they use it.
- User Blocking after periods of inactivity,
- Timer initiated termination of an Application,
- Semaphore support for fast updating,
- Flexible connection support for switch between Live and Test databases,
- An Options form to adjust parameters such as whether automatic ending is active, how long the application runs before automatic ending, whether user blocking is active or not.
- Filtered and sorted data export to Excel.
- Help texts for every form.

The Segregation of Duty allows super users to adjust which groups of users may use forms. In many cases a group of users will correspond to a department or users with a particular function. Also forms can be thought of as transactions which result in alterations to the database and/or production of reports.

The Segregation of Duty prevents application users from accessing forms or from altering data if they have access to a form. It does not prevent users from altering the data directly in the database using other tools. Correct setting of the database security is therefore essential in addition to configuring of the segregation of duty described here. It is also necessary that user names are added to and removed from any User groups defined on the database independently of the Segregation of Duty functions.

This ‘double’ approach to security has been implemented because the Segregation of Duty is a practical way of matching functionality to users; and can be managed by super users.

3 Auditing of Users and Groups

The Segregation of Duty functionality provides 2 reports which can be used for auditing purposes.

1. The Group report (Groups.xls). In many cases a Group will correspond to a department or to a group of users who execute the same function.

2. The users and then coupled to a group.

4 Filtering and sorting in a Data Grid.

All tables/data-grids in forms are derived from a grid which handles filtering, editing and sorting and the design of each column.

4.1 Sorting

The tables load with a default field ordering.
Sort the records using a column by single clicking on the column header. Click the same column again to change sorting from ascending to descending.

4.2 Filtering

When a string is entered in a text box above a column, the data in the grid is filtered using a 'where clause'. For example if the 'm' is entered in the text box above the column 'User' in table v_usr_log in form 'User Log' only those users with names beginning with m will be shown.

The 'where clause' is as follows ***where user like 'm*'*** where * is the wild card character.
If for example, 'a' is entered in the text box above column Application then the filter will include both the columns and the where clause would be ***where user like 'm*' and application like 'a*'*.**

If the text box entry is '*in', then the match will be made on user names ending with 'in'. The where clause is ***where user like '*in'*.**

Double clicking on a value in a field sets the text box filter to that value plus the wild card character '*' at the end of the string. Where agreed the text will propagate to other forms with tables having the same field name.
The check boxes filter using 3-states: checked, unchecked and both.
Real dates do not work well with filtering and so where relevant a field formatted as yyyymmdd (ISO 8601) is included in a view. This format is more suited to sorting and filtering.
The Print to Excel exports only those records shown in a table, in other words 'what you filter and sort is what you get'.
Use the Reset menu entry to reset the filters values in the text boxes above columns.

4.3 Editing

Read only fields are shown with a grey background while editable fields are shown with a white background. Whether a field is read-only or can be edited is a design time decision and cannot be altered using the column design functionality described below.

The Update button may be visible above the form and can be used to write data to the database. However default behaviour is that a write operation is made if a modification has been made to a field in a row and the cursor moves off the row.
The Refresh button above the form may be used to re-read data from the database.
A complete form may be defined as Read Only for a group of users as described below.

5 Multi Language Support, Segregation of Duty and Column Design

5.1 Languages

An application may support any number of different languages. Texts such as column headers and menu and button captions can be translated into the languages. The language used depends on the Windows user name of the person logged into Windows when the application is started.
The languages which the application needs to support are defined in Security.Languages form, see Figure 1.

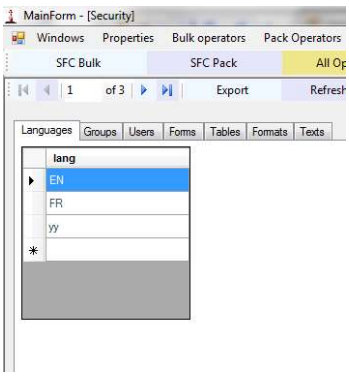


Figure 1 Language definition.

The Users are defined in the Security.Users form, see Figure 2. Figure 2 User log in the Security Users form.The Windows User name (gebruiker) must be defined in this table. In addition extra columns for the real name of the user, email address and telephone number are available.
Use these extra fields to identify the user as the Windows User name may be based on a number in the future and it will not be possible to work out the real name of a user from the Windows name.

The language the User will use and the group the user belongs to are selected here.
The grid shown lower right has the modifications made to the selected user name.

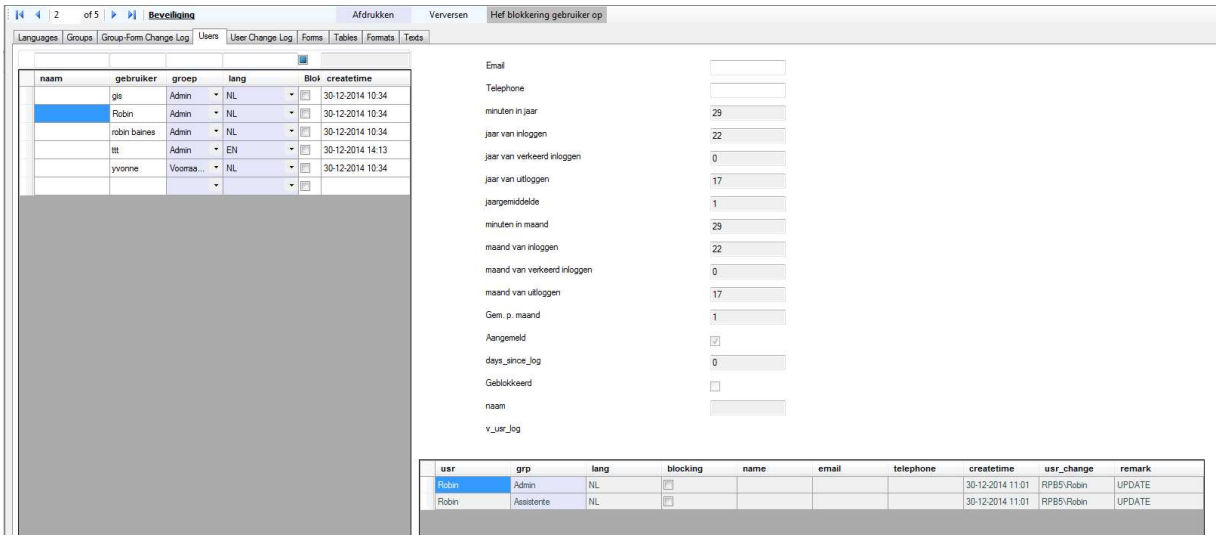


Figure 2 User log in the Security Users form.

5.2 Segregation of Duty

Segregation of duty works by defining Groups, see Security.Groups and Figure 3. A User belongs to a Group and the duties a Group may execute are defined by selecting the Forms which are assigned to the Group and defining whether the Form is read only or not.
A User may belong to only 1 Group.

The lower right hand grid shows any modifications made to the selected Group.

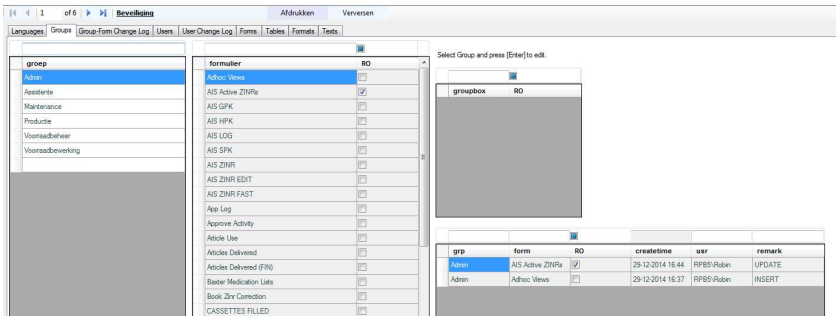
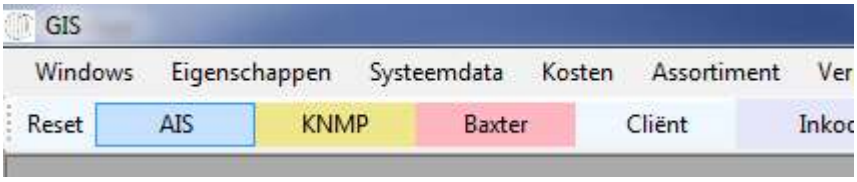


Figure 3 The Groups.

Forms are launched from a **main menu** which is on the 3rd row of the application and from a **sub menu** on the 2nd row of the application. The latter tend to be used for master data.



Both menu types can be used to launch a form directly or to show a list of forms in a drop down.
All forms in the **main menu** are coupled by name to a Group. In the **sub menu** only the top level buttons are coupled. This avoids having to couple up what may be many small tables in a **sub menu** drop down.
The Forms are defined in the Security.Forms form, see Figure 4.
The center grid at the bottom of the form shows which Group makes use of the Form.
The lower right hand grid shows any modifications made to the Form selected in the left hand grid and the Group selected in the center grid at the bottom.

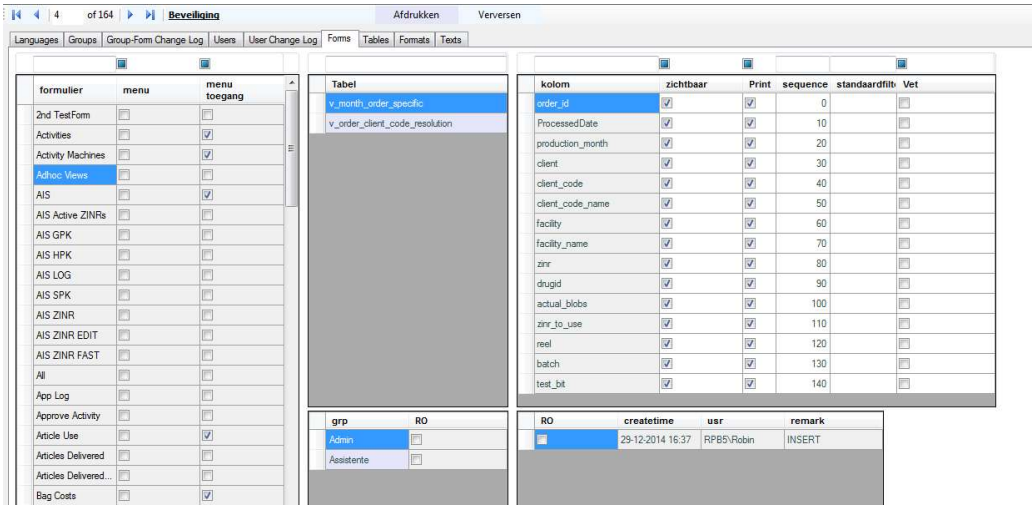


Figure 4 The Forms.

Forms are coupled to a Group by selecting the Group and pressing the enter key. The Form selection dialog appears as shown in Figure 5.

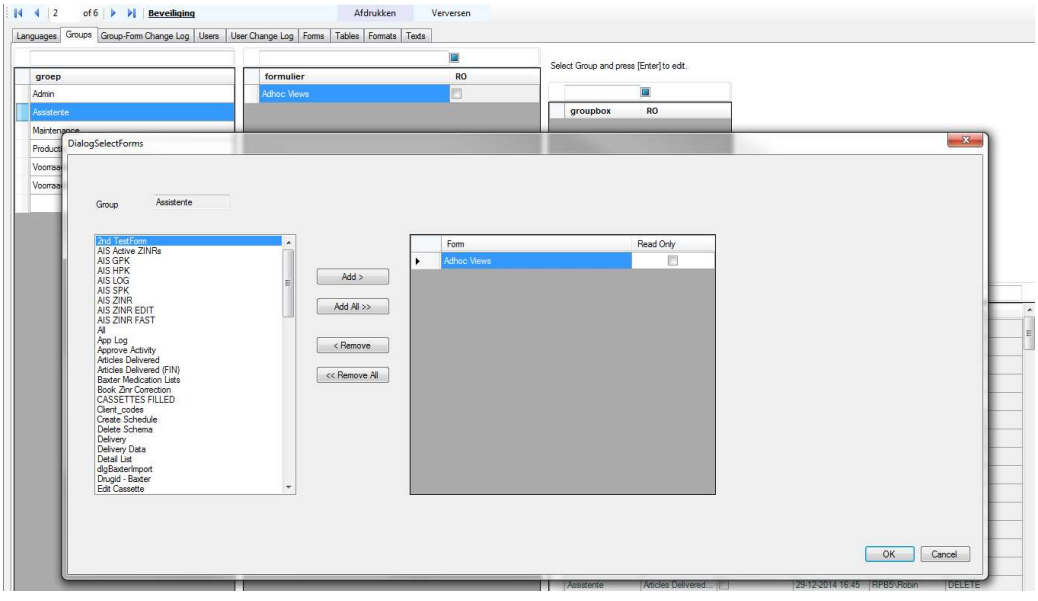


Figure 5 Coupling a Form to a Group in the Groups form.

5.3 Column Design

A form may use several Tables also known as data-grids.. Use that dialog to adjust

- whether the field is visible in the form,
- whether the field is visible in the Excel print out,
- the sequence in the table in the form,
- the default filter setting when the form is opened and
- whether the text is bold or not.

The same data table may be used in several forms. The above characteristics are characteristics of how the table is used in the form. However the column format, width and header text are characteristics of the fields of the data table and are therefore independent of the form where the table is used. These properties can be adjusted in Security.Tables, see Figure 6. Figure 6 shows the values for table v_SAP_AFKO. The 2nd table shows the columns of v_SAP_AFKO and the right hand side table shows the column header in French and in English for the selected column 'ABLAD'. If no translation text has been entered then the column name will be shown and will be used when the form is opened.

The Security tables also appear in the Forms and Tables forms and so the appearance and column visibility can be adjusted for the Security form. However be careful adjusting the column visibility as the result can be confusing!

The Security.Tables form also enables format adjustment for each column. The possible formats are defined in Figure 7 and apply only to numeric fields. 0.0\% will show a float field as a % with 1 digit behind the decimal. N1 will show a float field as a decimal with 1 digit behind the decimal.

The application will include a number of text strings which appear as button or menu captions. These are defined in Security.Texts, see Figure 8. The translation to the different languages can be entered.

Column design aspects can be tested after modification by closing the form of interest and re-opening it. Column headers can be tested in the same way but the Security form should also be closed before re-opening the form of interest. Texts of menus etc are only loaded if the application is stopped and re-started.

5.4 Finding Names of Forms, Columns etc.

In order to make the above modifications the original name of a form, a column, a text and the name of a table/data-grid in a form are needed. The Tooltip text is used for this purpose. Holding the mouse above a form button in the main menu or above a column heading in a data grid will show the original name of the form respectively the column. Holding the mouse above the top left hand box of a data grid will show the name of a table.

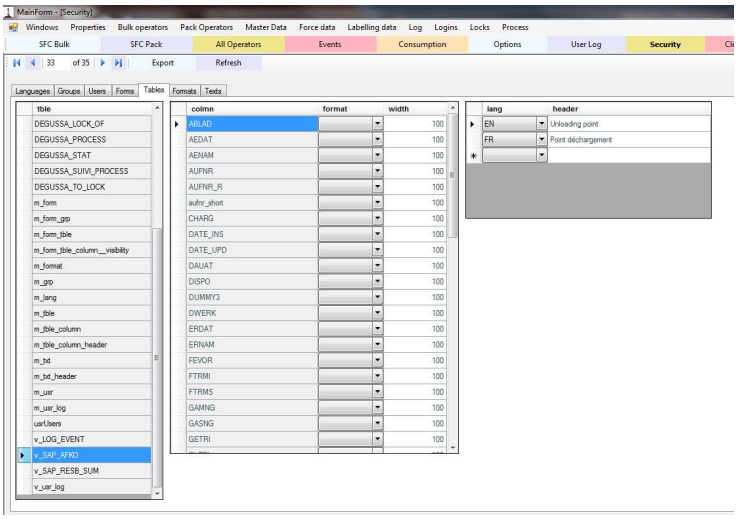


Figure 6 The Tables.

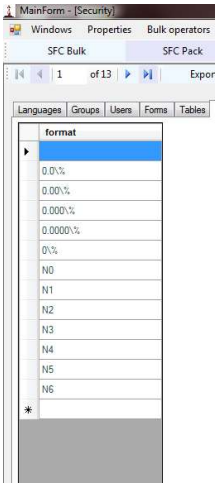


Figure 7 The Formats.

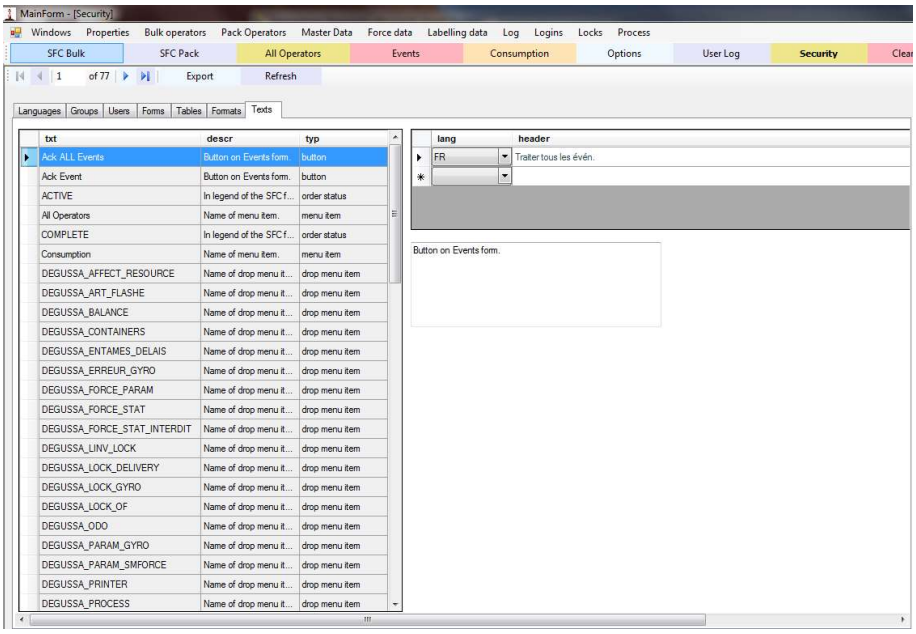


Figure 8 The Texts.

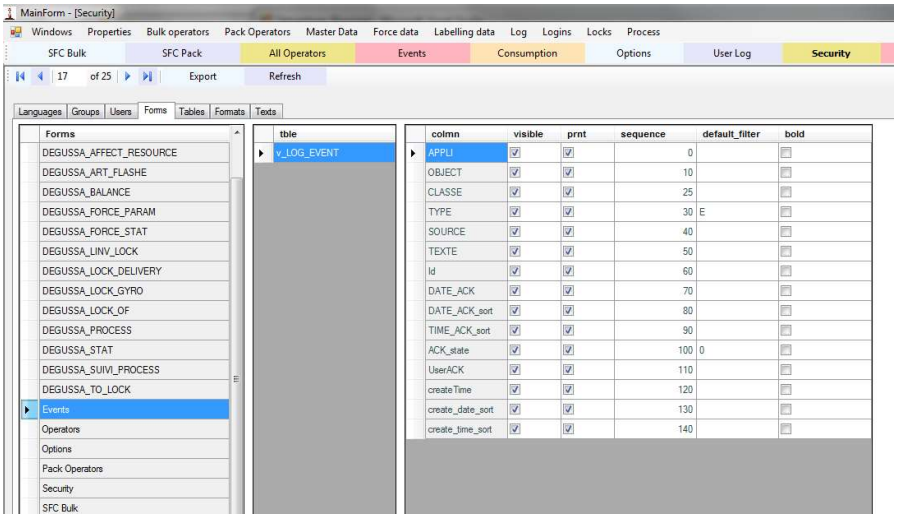
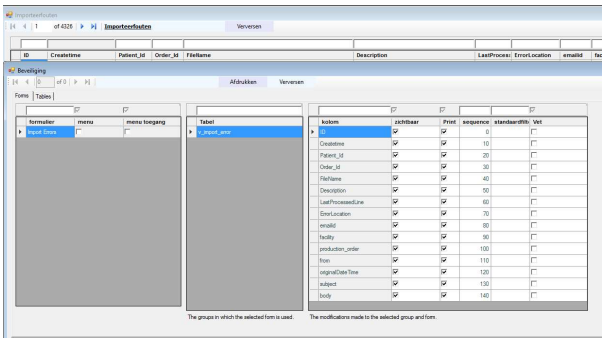


Figure 9 Security.Forms for Events form showing filter presets.

5.5 Column And Table adjustment in context.

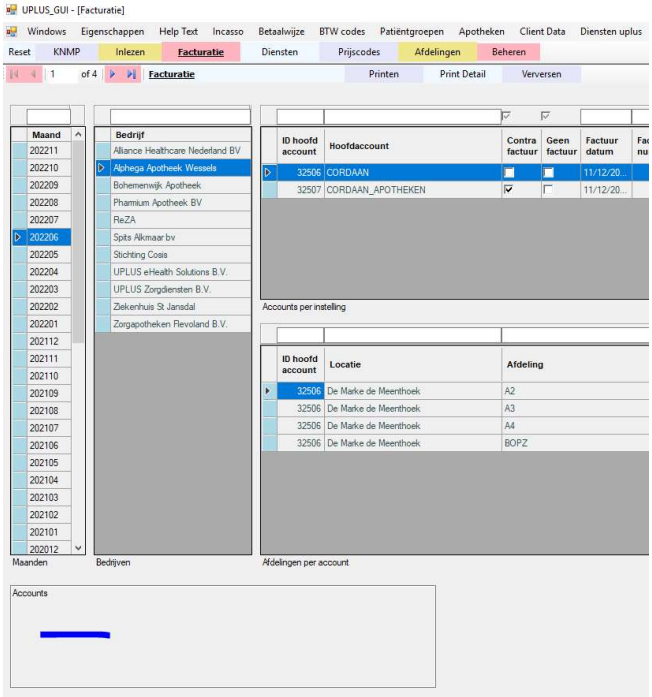
If a user has rights to the Security forms the Columns and tables used in the forms may be accessed from a form by pressing F9. By cascading the windows it is easy to adjust exactly which columns should appear, the width, format and sequence.



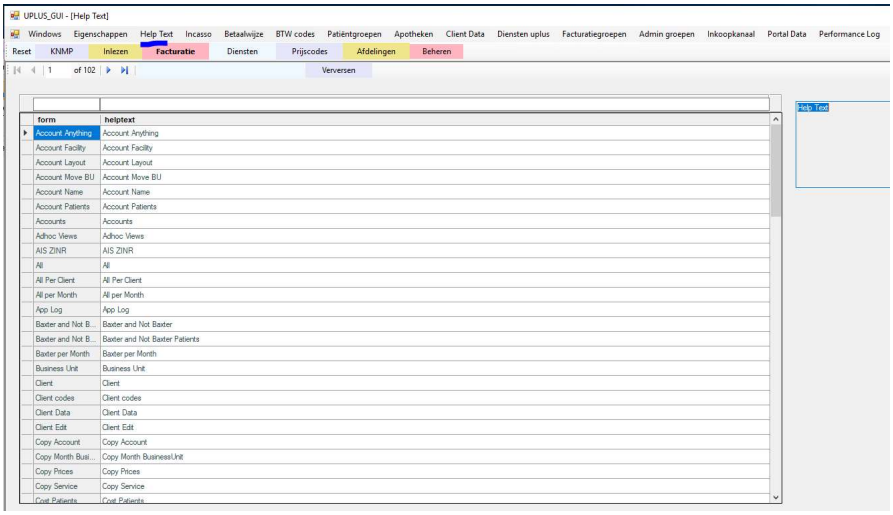
Using F9 to show form columns and tables.

6 Help Text

The utilities library includes a help textbox in every form.



The text in the textbox may be altered using the Help Text from the Master menu on the 2nd line of the application. It also includes a help text!
If the help text is modified it will be shown to all users. It is updated when the relevant form is closed and re-opened.



7 Logging Changes to Groups and Users.

Groups and Forms described above, show filtered versions of the log of changes to groups. The Users form shows a filtered version of the modifications made to the Users.
The unfiltered versions of these changes can be viewed from ‘Group-Form Change Log’ and ‘User Change Log’.
The most recent change is shown at the top of the grids.

8 User Log and User blocking.

8.1 The User Log

The User log records

- all attempts to start an application,
- the total length of time in minutes the user used the application in the last month and the last year,
- all windows users who attempt to start the application even if they were not defined in Users table,
- the time when the user stopped using the application and
- whether the user is currently logged in.

The user log is viewed in the User Log form, see Figure 10. It shows the list of users who attempted to use the application on the left hand side and the log activity on the right hand side. The list of users includes Users defined in Security.Users who have not yet attempted to use the application and also ‘unauthorized’ users who attempted to use the application but who are not defined in the Users table.

The left hand view of the Users Log form is also shown in the Security.Users form but because this form uses the Users table it will not show the ‘unauthorized’ attempts to use the application.

8.2 User Blocking

Only users who are defined in the User table may use the application. It is also possible to activate functionality which prevents/blocks use of the application after a period of inactivity. User blocking can be activated or de-activated for all users or per user.

If the ‘block_after_days_inactivity’ parameter is defined and is not 0 in the Options form then a user will be blocked if the user last used the application more than that number of days ago. If this is the case the user may start the application but will have no rights to open forms. A blocked user will see a message box, see Figure 11 and on acknowledgement of the message, the application without menu entries, see Figure 12.

A user can be ‘unblocked’ by using the ‘Unblock User’ button in the Security Users form. This is implemented by adding a login message to the log; meaning that the user will be unblocked from that moment and until there has been no activity for ‘block_after_days_inactivity’ days. (Technical detail: This type of unblocking log message has a ‘WindowIdentity’ field equal to the user who did the unblocking and a User field corresponding to the user who was unblocked. These 2 fields and an unblocking log message are highlighted in Figure 10.)

A User, for example the Super User who is responsible for managing the users, can be removed from the blocking scheme by un-checking the ‘blocking’ checkbox in the Security.Users form. When this checkbox is unchecked the user will never be blocked.

The above blocking functionality blocks a user who has rights after a period of inactivity. It is also possible to block a user by using the Security. For example define a ‘Blocked’ group with no forms and then assign the User who should be blocked to this group. If this user should start the application then they will see Figure 12. Unblocking this user will require assigning them to a group with appropriate rights and may require resetting the blocked check box if they have also been inactive.

A relevant use case is a user who attempts to use the application before they have been added to the security; in other words before they have been authorized. Subsequently the User is added to the security to the list of authorized users. For this reason the blocked checkbox is set only if the user has been added to the security and has authorized activity in the usr_log.

If the delete_usr_log_days parameter is defined and is not 0 in the Options form then old log messages will be deleted and they will no longer be included in the statistics.

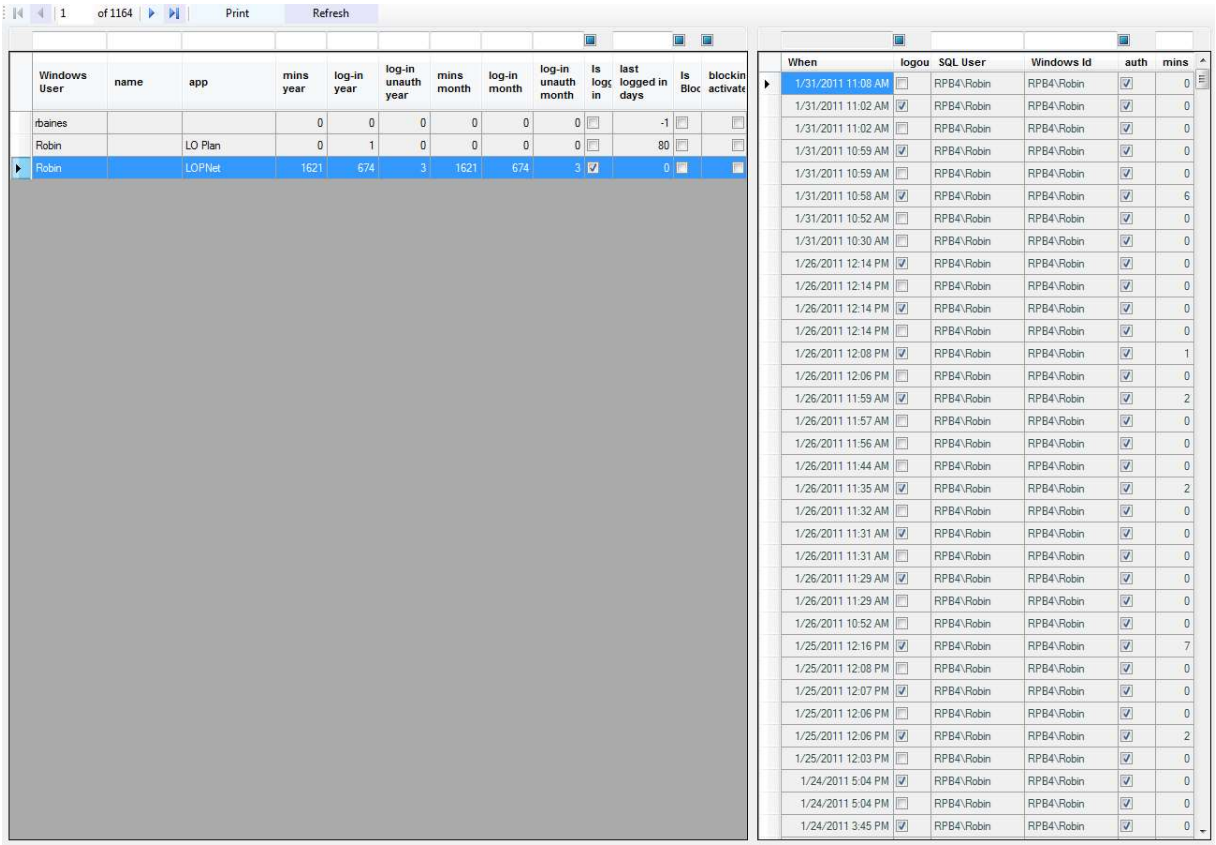


Figure 10 User Log Form.

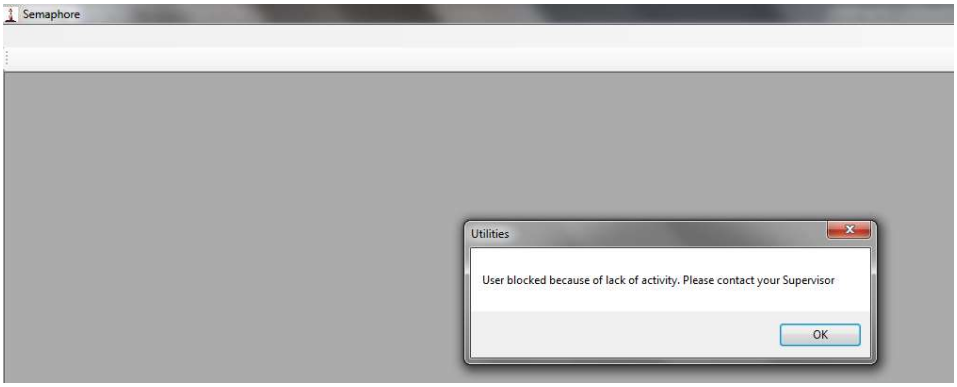


Figure 11 Message box when Blocked User starts the application

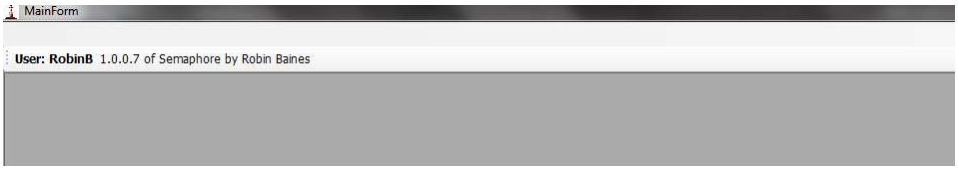


Figure 12 What a Blocked User sees.

9 The Options form and the Parameter Table.

Application parameters will be stored in the b_app_parameter table by the application Developer.
Application colour parameters will be stored in the b_app_color table.
These parameters may be viewed and edited using the Options form, see Figure 13.
If parameters are being used the Options button will be shown in the main Toolstrip.

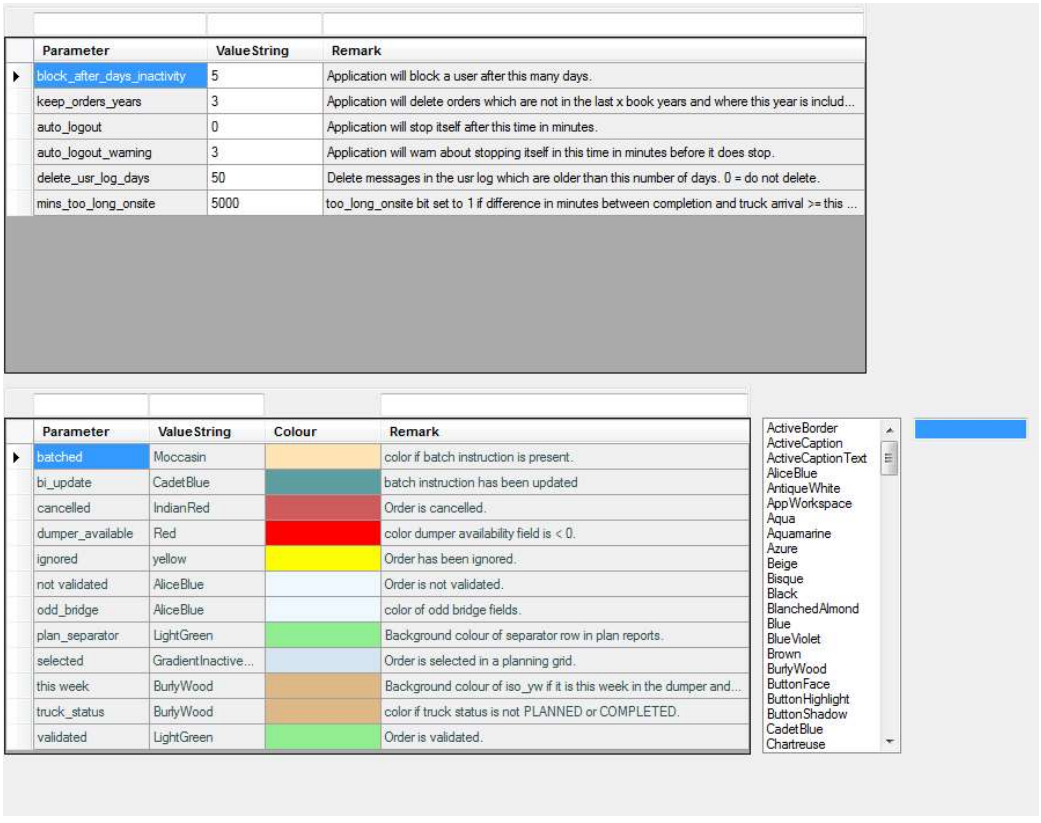


Figure 13 the Options form showing Colors defined for the Semaphore application.

The following parameters are used by utilities:

Parameter	ValueString	Remark
auto_logout	600	Application will stop itself after this time in minutes.
auto_logout_warning	2	Application will warn about stopping itself in this time in minutes before it does stop.
block_after_days_inactivity	5	Application will block a user after this many days.
delete_usr_log_days	50	Delete messages in the usr log which are older than this number of days. 0 = do not delete.

10 Automatic Termination of Application

The application may be setup to terminate a number of minutes after it has been started.

If the auto_logout parameter is defined using the Options form a timer is started in the Main Form. The time in minutes before the application will end is shown on the status bar bottom left, see Figure 14. The time decrements to 0 and then the application will end.

If the auto_logout_warning parameter is defined in the b_app_parameter table then when the time before the application will end is less than this parameter in minutes all open forms are closed and a warning that the application will end is shown, see Figure 14. A button labelled 'Press to prevent auto end' is shown. It allows the user to reset the auto_logout timer to its original value.

If changed, these parameters will have immediate effect. The auto_logout timer is restarted when this occurs.

Setting auto_logout to 0 stops the timer and disables this functionality.

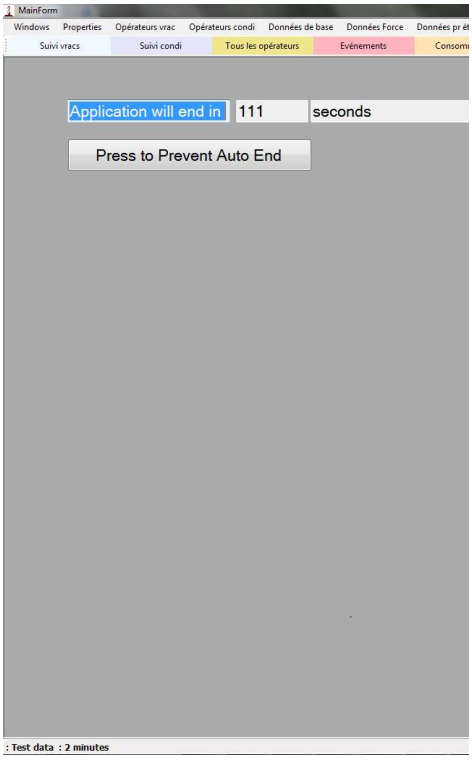


Figure 14 Warning before automatic termination of the application.

11 Connection Strings and Test Database

The Application will have connections to the development and live databases and may also have a connection to a test database.

The DataSource and Catalogs are defined in the 'Application'.exe.config file in the following variables.

- DataSourceDevelopment, CatalogDevelopment
- DataSourceLive and CatalogLive
- And DataSourceTest and CatalogTest.

The connection to the development database is generally setup automatically when computer name equals the DevelopmentDataSource.

The Utilities library supports switching between live and test databases using a setting called TestDatabase which is also stored in the Application.exe.config file. This can be set/reset from the Properties form from where it is saved, see Figure 15.

When the switch is made all open forms are closed and the automatic termination timer is reset to the value set in the new database.

'Live Data' is shown in the Main Toolstrip above right if the live database is active and 'Test Data' in bold if the test database is active.

All the user settings are computer dependent meaning that one user can be using the Test Database while another is working on the live one.

The Properties form also includes an Enable Audio checkbox. Where relevant this checkbox can be used in the code to disable/enable audio.

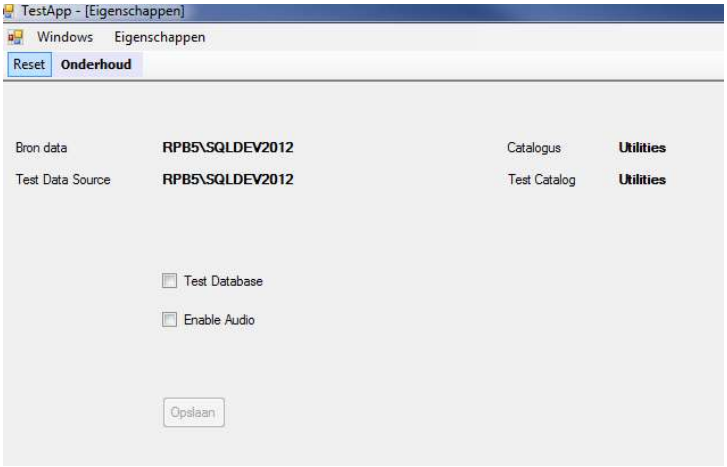


Figure 15 Properties form.

12 The Excel Interface

The Excel interface is for Excel 2003 and above.
Double clicking the top left field of the datagrid will export the contents of the grid to Excel. This means that filtering and sorting can be used to select the data.

13 NEW USER PROCEDURE

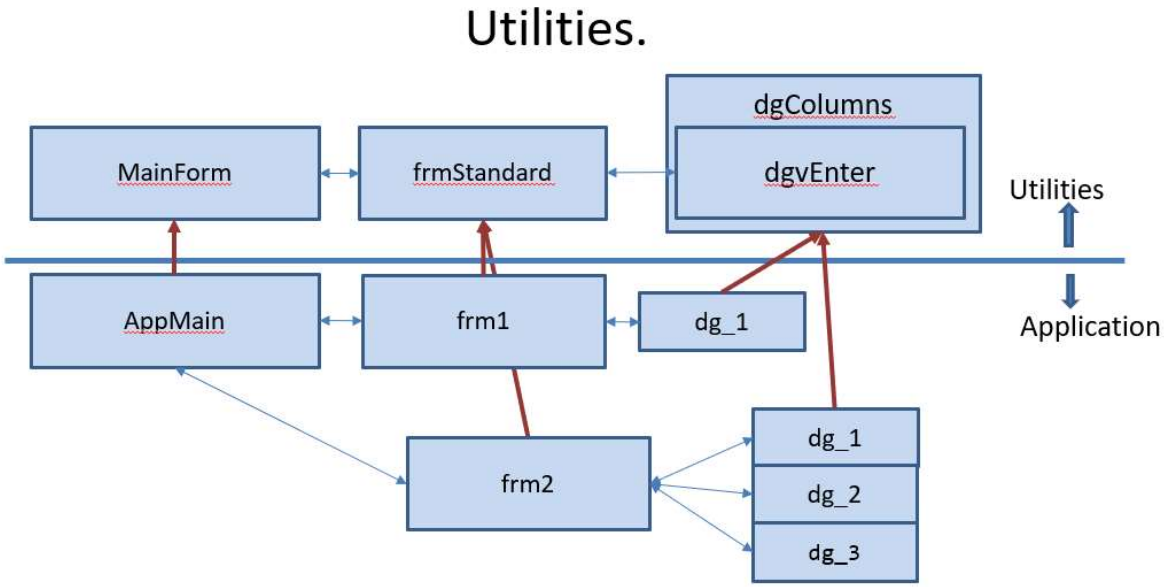
See application documentation.

14 Distribution of New Versions

See application documentation.

15 Using Utilities.dll

The diagram shows how the Utilities.dll is used to build an application. The application main form is derived from the Utilities.Mainform, the forms are derived from the Utilities.frmStandard and the datagridviews are derived from the Utilities.dgvEnter datagridview. Derivation is shown with the red arrows.
Overridable suroutines and functions are available to use timer and semaphore functionality. These are shown as blue arrows.
The idea is illustrated with the Semaphore functionality.



16 Semaphore support for fast updating

Events may take place in applications, services or the database which need to change the contents of forms in the Application. The Semaphore function is available to avoid complications of inter process communication when reaction time of 4 or 5 seconds is acceptable.
A table called b_semaphore is defined in the database as follows.

```
CREATE TABLE [dbo].[b_semaphore](
        [app] [nvarchar](50) NOT NULL,
        [tble] [nvarchar](50) NOT NULL,
        [semaphore] [int] NULL,
PRIMARY KEY ([app],[tble])
)
```

Utilities.MainForm reads this table every 5 seconds. If a semaphore field has changed since the previous read an overridable sub called RefreshAll is called with the above 3 fields as parameters.
The default RefreshAll calls each open form which has been derived from frmStandard using the overridable RefreshTheForm sub with the above 3 fields as parameters.
Typically a form will override RefreshTheForm to refresh the contents of the datagrids in the form.

The following is an example.
When 2 separate instances of an application are used to process orders it may be necessary to update one application if the other is used to change the status of an order.

Use SQL Procedures to update the status of ‘orders’ and increment the semaphore either in a trigger or explicitly in the procedure:

```
UPDATE b_semaphore SET semaphore = semaphore + 1 WHERE app = 'App' AND tble = 'ScheduleBoard'
```

And return the current value of the semaphore.

```
SET @semaphore = ISNULL((SELECT semaphore FROM b_semaphore WHERE app = 'App' AND tble = 'ScheduleBoard'), 0)
```

The calling form stores the value of the semaphore.
Several seconds later the overridden RefreshTheForm sub receives the changed semaphore value. If it is the same as the value returned from the Proc then there is no need for further action however if it is changed then a refresh of order status may occur because the other instance has just been used to change the status of an order.

17 Performance Logging.

The static clLogging class is used to log the time taken to execute a process.
For example

```
clLogging.LogStart() ‘starts a timer

Me.V_order__scheduledTableAdapter.FillByCanProduce(Me.ScheduleDataSet.v_order__scheduled,
iWeek, idow)

‘stop the timer and write the time taken and the message to a log file.
 clLogging.LogStop("frmProduction.V_order__scheduledTableAdapter.FillByCanProduce.")
```

The following is written to the log file.

```
24-10-2025 20:22:08 - 8712 msec. frmProduction.V_order__scheduledTableAdapter.FillByCanProduce
```

The logfile is in a folder called LogFiles in the executable folder. The default file name is
‘Logfile_’ %SessionId% ‘_’ ddMMyyyy ‘.txt’.

For example Logfile_1_24102025.txt.
The SessionId is included so that a unique file is created when running an executable in Citrix.

Property Public Shared Property Logfile() As String
is available to define the name of the log file. If the extension is not in the filename paramter .txt is added.
clLogging.Logfile = "test.md" in the constructorresults in a log file name: test.md

Property Public Shared Property InFix() As String
is available to define a string which is inserted into the log file name.
clLogging.InFix = "test" in the constructor results in a log file name: Logfile_1_test_24102025.txt