AIMMS User's Guide - Page and Page Object Properties

This file contains only one chapter of the book. For a free download of the complete book in pdf format, please visit www.aimms.com.

AIMMS B.V.
Diakenhuisweg 29-35
2033 AP Haarlem
The Netherlands
Tel.: +31 23 5511512

AIMMS Inc.
11711 SE 8th Street
Suite 303
Bellevue, WA 98005
USA
Tel.: +1 425 458 4024

AIMMS Pte. Ltd.
55 Market Street #10-00
Singapore 048941
Tel.: +65 6521 2827

AIMMS
SOHO Fuxing Plaza No.388
Building D-71, Level 3
Madang Road, Huangpu District
Shanghai 200025
China
Tel.: ++86 21 5309 8733

Email: info@aimms.com
WWW: www.aimms.com

Chapter 11

Page and Page Object Properties

This chapter

After you have created a page with one or more data objects on it, AIMMS allows you to modify the display properties of these objects. This chapter illustrates the available tools for placing and ordering page objects, and how to modify properties of both pages and page objects. It also provides a brief description of the available properties.

11.1 Selecting and rearranging page objects

Selecting an object

Before you can modify the properties of a page object, you must select the object. This can be accomplished as follows:

- make sure that the page is opened in edit mode (see Section 10.3),
- press the **Select Object** button on the page toolbar, if it is not already pressed, and
- click on the page object.

The selected object(s) on a page are marked with a small dark square on each of its corners. This is illustrated in Figure 11.1.

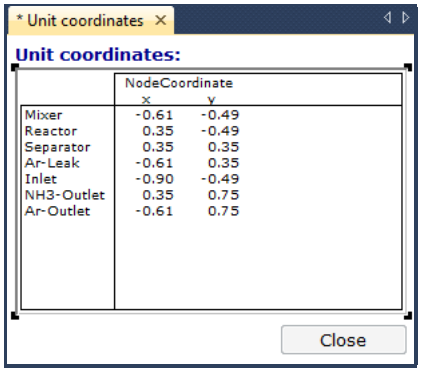

Figure 11.1: A selected page object

No template objects

When a page depends on one or more templates (see also Section 12.2), AIMMS will only let you select those objects that were placed on the page itself, and not those which are contained in any of its templates. Template objects can only be edited in the template page on which they are defined.

Selecting overlapping objects

When two or more objects are overlapping, clicking on the overlapping region will result in any one of the overlapping objects being selected. By holding the **Shift** key down during clicking, AIMMS will cycle through all the overlapping objects, allowing you to select the object of your choice. Alternatively, you can press the **Tab** key repeatedly to browse through all selectable objects on the page.

Selecting multiple objects

In addition to selecting a single page object, AIMMS also allows you to select multiple objects. You can do this by dragging a *select region* on the page, after which AIMMS will mark all objects contained in that region as selected. Alternatively, you can add or remove objects to form a selection by clicking on the objects while holding down the **Shift** key.

Object alignment

With the **Edit-Alignment** menu of a page in edit mode, you can correct the placement and sizes of all page objects that are currently selected. The **Alignment** menu lets you perform actions such as:

- give all selected objects the same height or width, i.e. the height or width of the largest object,
- align all selected objects with the top, bottom, left or rightmost selected object,
- center the selected objects horizontally or vertically, and
- spread all selected objects equally between the top and bottommost objects or between the left and rightmost objects.

An alternative method of alignment is to define a grid on the page (see Section 10.3), and align the borders of all objects with the grid.

Drawing order

With the **Drawing Order** item of the **Edit** menu, you can alter the order in which overlapping objects are drawn. When applied to a selected object, you can specify that the object at hand must be drawn as either the top or bottommost object. Modifying the drawing order only makes sense for drawing objects such as the text, rectangle, line, circle and picture objects.

Specifying the tab order

When there is a natural order in which an end-user has to enter data on a particular page, you can use the **Tab Order** item from the **Edit** menu, to specify this order. The **Tab Order** menu opens a dialog box as illustrated in Figure 11.2. In this dialog box all page objects are displayed in a list which determines the (cyclic) order in which AIMMS will select the next object for editing when the user leaves another object on the page through the **Tab** or **Enter** keys.

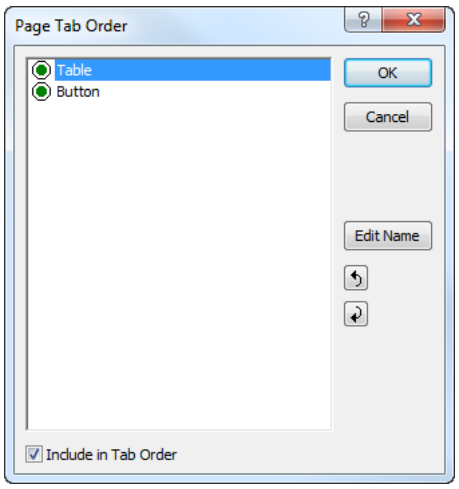

Figure 11.2: The **Tab Order** dialog box

Tabular objects

In tabular objects, the **Tab** and **Enter** keys can also be used to move to the next table entry to the right or below, respectively. In such cases, AIMMS will only go to the next object in the tab order, if further movement to the right or below within the object is no longer possible.

Disabling tab order

In addition to modifying the tab order, you can also use dialog box of Figure 11.2 to select the page objects that should not be included in the tab order. Alternatively, you can remove a page object from the tab order in the **Properties** dialog box of that object as explained in the next section. Objects excluded from the tab order are not accessible on the page by pressing the **Tab** or **Enter** keys, but can still be selected using the mouse.

11.2 Modifying page and object properties

Object properties

In addition to modifying the display properties of groups of objects on a page, AIMMS also allows you to modify the visual appearance of a page itself and of all of its individual page objects. When the page is in edit mode, you can open the **Properties** dialog box of either a page or a page object by simply double clicking on it, or by selecting **Properties** from the right-mouse pop-up menu. This will display a dialog box as illustrated in Figure 11.3. The dialog box contains tabs for all visual aspects that are relevant to that object, and initially displays the current settings of these visual aspects.

Properties of multiple objects

You can also modify properties of multiple objects at the same time by first selecting a group of objects and then selecting the **Edit-Properties** menu, or selecting **Properties** from the right-mouse pop-up menu. This will invoke a

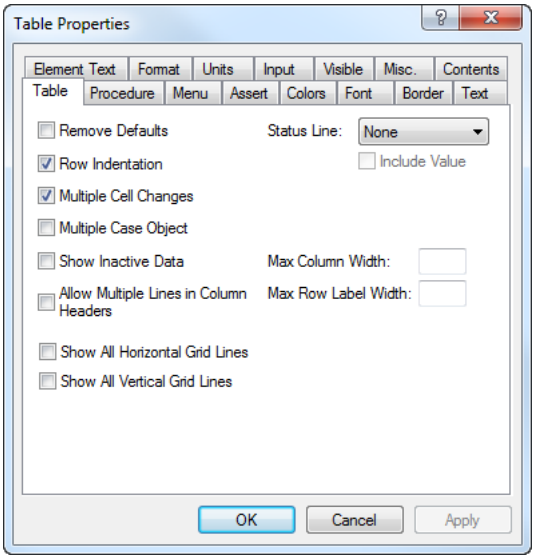

Figure 11.3: The **Properties** dialog box

Properties dialog box containing only those tabs that are common to all the selected objects. AIMMS will not display an initial value for the corresponding properties, as each property may hold different initial values for the various objects. Only the properties that you change are applied to the selected objects.

Property types

Through the tabs in the **Properties** dialog box, AIMMS lets you modify the various properties of pages and page objects. The following paragraphs provide a brief overview of the modifiable properties. A full explanation of the various properties of all the available objects can be found in the help file accompanying the AIMMS system.

The Contents tab

With the **Contents** tab you can add or remove identifiers from the list of identifiers that are displayed in the object. With this tab you can specify, for instance, that a table is to display the values of two or more identifiers. To modify the contents, AIMMS will open the common **Identifier Selection** dialog box as explained in Section 10.3.

Applying changes

Before you can make changes to the **Contents** tab, AIMMS requires that you apply any changes you have made to the other object properties before entering the **Contents** tab. You can apply these changes using the **Apply** button. Similarly, after you have made changes to the **Contents** tab, AIMMS requires that you apply these changes before you can go on to modify other object properties.

The Procedure tab

With the **Procedure** tab you can specify the particular procedures that must be executed upon user inputs such as a data change or selecting a particular value in a data object. The use of procedures linked to data objects is mostly to perform error checks or update other identifiers based on a single data change.

The Action tabs

With the **Action** tab, the counterpart of the **Procedure** tab for pages, buttons and navigational controls, you can specify the particular actions that must be executed upon opening a page, pressing a button, or making a selection in a navigational control. Such actions typically can be a sequence of running a procedure within the model, executing predefined AIMMS menu actions, or checking assertions.

The Menu tab

The **Menu** tab lets you specify which menu bar, toolbar, and right-mouse pop-up menu should be active on top of either a page or an object on a page. The menus themselves, as well as the actions linked to the menus, can be created in the **Menu Builder** tool. The **Menu Builder** tool is explained in full detail in Chapter 12.

Double-click actions

An action type that is used quite frequently, is the double-click action. You can specify a double-click action either in the **Action** tab, or through the **Menu** tab. The following rules apply.

- If a **Double-Click** procedure is specified on the **Action** tab, AIMMS will execute that procedure.
- If no **Double-Click** procedure has been specified, but a pop-up menu associated with the page object has a default item, AIMMS will execute the default menu item.
- If neither of the above apply, and the object is a table displaying a set, the double-click action will toggle set membership of the set element which currently has the focus.
- In all other cases, double-clicking will be ignored.

The Assert tab

Through the **Assert** tab you can indicate which assertions already declared in your model are to be checked upon end-user data changes to a particular identifier in a data object. AIMMS can perform the assertion immediately upon every data change, or delay the verification until the end-user presses a button on the page. Once an immediate assertion fails, the assertion text will be displayed to the user and the original value will be restored.

The Colors tab

With the **Colors** tab you can not only specify the colors that are to be used for the foreground and background of a page or page object, but also the color for the user-selected values in a page object. In addition, you can specify a model-defined (indexed) color parameter to define the foreground color that will be used for each identifier in a data object. With such a parameter you can, for instance, color *individual* values of an object depending on a certain

threshold. The necessary computations for this individual coloring need to be made inside the model underlying the end-user interface. You will find more details about assigning color parameters in Section 11.4.

The Font tab

The **Font** tab lets you define the font that is to be used for a particular object. You can choose the font from a list of user-defined font descriptions as illustrated in Figure 11.4. To add a new font name to the list, you should press the

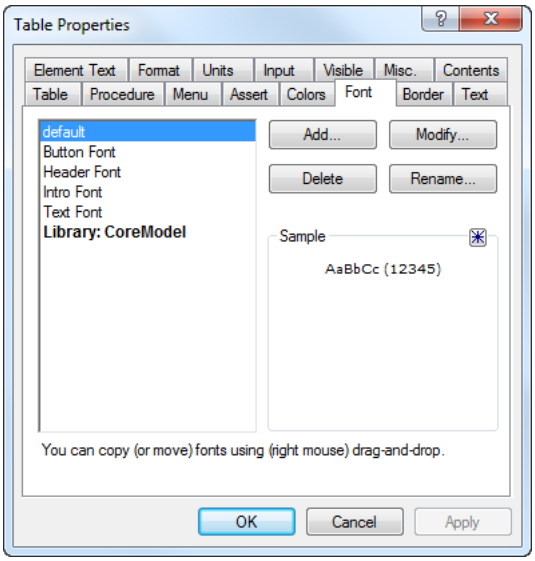

Figure 11.4: The **Font** tab of a **Properties** dialog box

Add button. This will open the standard Windows font selection dialog box, allowing you to define a new AIMMS font based on the list of available Windows fonts, font styles and sizes. Once you have made a selection, you will be requested to provide a description for the newly selected font.

Choose functional font names

It is strongly recommended that you choose functional names for AIMMS fonts (i.e. describing their intended use) instead of merely describing the choices you made. For instance, naming a new font "Button font" instead of "Arial Regular, 8 pt" will help tremendously in preventing mistakes when selecting a font for a button.

Fonts in library projects

AIMMS also allows you to store fonts within a library project. The list of fonts shown in Figure 11.4 displays a single font *Small Table Font* associated with the library *CoreModel*. You can manage the list of fonts associated with a library by pressing the buttons on the right-hand side of the dialog box, while the selection in the listbox on the left-hand side is in the area associated with the library.

Use only in library pages

If you have defined fonts within a library project, you should ideally only use these fonts in pages that are also part of the library project. If you use the fonts in pages outside of the library, such pages may fail to display properly after you have removed the library project from the AIMMS project.

Font names must be unique

AIMMS requires that all font names be unique across the main project and all library projects that are included in the main project. If you include an existing library project, which contains a font name that is already present in the AIMMS project, AIMMS assumes that both fonts are the same and will ignore the second font definition.

The Border tab

With the **Border** tab you can stipulate the border settings for any particular data object on a page. A border can consist of merely a surrounding line, or provide an in- or out-of shadow effect.

The Text tab

With the **Text** tab you can specify for each identifier a single line of text that is displayed in a page object. With this line of text you can, for instance, provide descriptions for the data in a table containing one or more identifiers. In addition, the **Text** tab will let you define the element description for the (optional) status line associated with the object. The status line will display the currently selected value along with its element description. If the element description contains references to the indices over which the identifier at hand is defined, these references will be expanded to the currently selected element names.

The Element Text tab

By default, any set element in a data object will be displayed by its name in the model. If you want to display an alternative text for a set element, you can use the **Element Text** tab to specify a string parameter holding these alternative element descriptions. You can use this feature, for instance, to display set elements with their long description in the end-user interface, whereas the model itself, and perhaps paper reports, work with short element names.

The Format tab

The **Format** tab defines the numerical format in which the data of a particular identifier is displayed. This format can be specified on the spot, or can use a named format already predefined by you as the application developer. The display format specifies not only such properties as the width of a number field and its number of decimal places, but also their relative alignment, the use of a 1000-separator for large numbers, and the display of default values.

The Units tab

The AIMMS modeling language offers advanced support for defining units of measurement for each identifier in the model. In particular, AIMMS supports unit conventions which let you define a coherent set of units (e.g. Imperial or metric units) in a single declaration. In the end-user interface you can indicate in the **Units** tab whether you want units to be displayed for every identifier or

for every individual value contained in a particular data object. The displayed units are the units defined for the identifier at hand, unless the end-user has selected a current unit convention with alternative units. Figure 11.5 illustrates an end-user page in which identifier values are displayed along with their associated units of measurement.

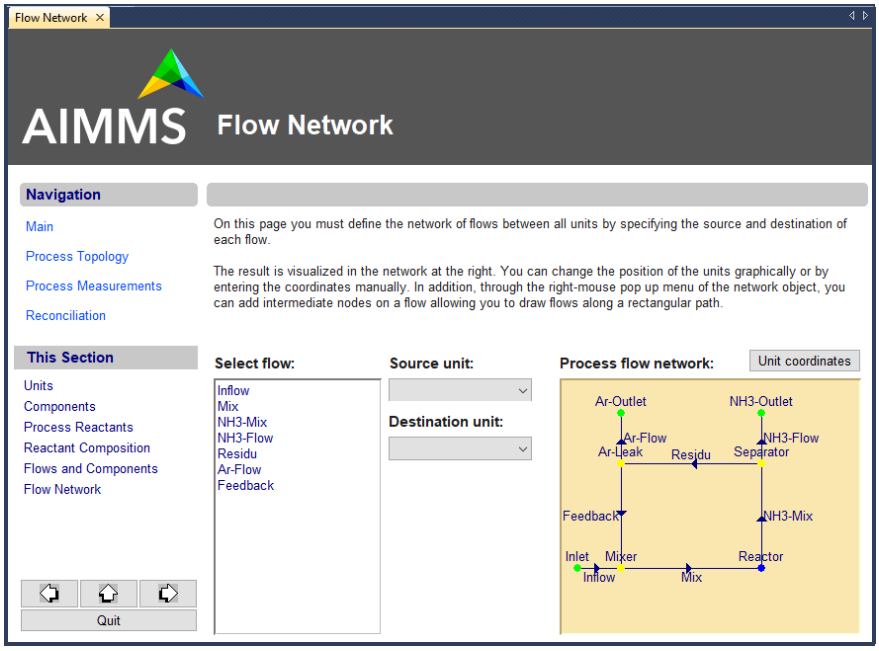

Figure 11.5: Use of units in a data object

The Input tab

With the **Input** tab you can specify the read-only properties of every identifier in a page object separately. The decision as to whether numbers are read-only can depend on (indexed) identifiers in your model. Thus, you can arrange it so that particular numbers in, for example, a table can be edited by the end-user, while other numbers associated with that same identifier are considered as read-only. In addition to the properties specified on this tab, the overall read-only behavior of identifiers is also influenced by the contents of the predefined identifier `CurrentInputs` (see Section 17.1).

The Visible tab

You can use the **Visible** tab to hide a particular page object in its entirety from a page. Whether or not a page object is visible may depend on a scalar identifier (slice) in your model. The ability to hide page objects comes in handy when, for instance,

- you want to hide a page object because a particular end-user has no right to modify its data, or

- a page contains two exactly overlapping page objects—e.g. one holding relative numbers, the other holding absolute numbers—and you want to display just the one based on the user's choice.

The Misc. tab

With the **Misc.** tab you can specify various miscellaneous settings such as

- whether a page object must be included in the page tab order to specify a natural navigation order on the page (see also Section 11.1),
- whether an object is actually printed or skipped during printing (only relevant for print pages, see also Chapter 14),
- which end-user help topic should be displayed for the page or page object at hand, or
- a tag name, which is used when you want to refer to the object from within the model (see Section 17.3.1).

Help file

Before adding end-user help to a particular page, page object, end-user menu or toolbar, you must add a help file to your project directory, and specify its name through the **Options** dialog box (see Section 20.1). All the available end-user help associated with your project must be contained in the specified project help file.

Help file formats

AIMMS supports several help file formats, allowing you to create a help file for your project using the tools you are most familiar with. They are:

- standard Windows help files (with the `.hlp` extension),
- compiled HTML help files (with the `.chm` extension), and
- PDF files (with the `.pdf` extension), which require that Acrobat Reader version 4.0 or higher is installed on your machine.

An executable Acrobat Reader installation can be downloaded from the Adobe website www.adobe.com.

Creating help files

To create a help file in any of the supported formats you will need an appropriate tool such as RoboHelp, Help & Manual or DocToHelp to create either a Windows or compiled HTML help file, or Adobe Acrobat to create a PDF file. To jump to a marked position inside the help file when providing help for a page, a page object, a menu or a button on a toolbar you should add:

- (so called) *K-keywords* to an ordinary Windows help file,
- *keywords* to a compiled HTML help file, or
- *named destinations* added to a PDF file.

All of the destinations that you added to the help in this way can serve as the **Help Topic** inside the **Misc.** tab of a page or page object.

Object-dependent properties

In addition to the tabs described above, which are common to most objects, the **Properties** dialog box also has a number of tabs where you can change properties that are very specific for a particular type of object. Through such object-dependent options you can specify, for instance, whether a table should display default values, what should be displayed along the axes in a graph or chart, or how the arcs and nodes in a network flow object should be drawn. The contents of these object-specific tabs are explained in full detail in the online AIMMS help file.

11.3 Using pages as dialog boxes

Use of dialog boxes

By default all end-user pages behave as normal windows, i.e. whenever you have multiple windows open, you can freely switch from window to window simply by clicking in the window that should become active. Sometimes, however, your application may contain sequential actions which require the user to make a certain choice or data change before moving on to the next action. In this case the page should behave as a dialog box instead of a normal window. While a dialog box is displayed on the screen, it is impossible to access other windows in the application without closing the dialog box first for example with an **OK** or **Cancel** button. By using dialog boxes you can force an end-user to follow a strict sequence of operations.

Dialog pages

In AIMMS you can define that a page should behave like a dialog box by using the page **Properties** dialog box as illustrated in Figure 11.6. If such a dialog

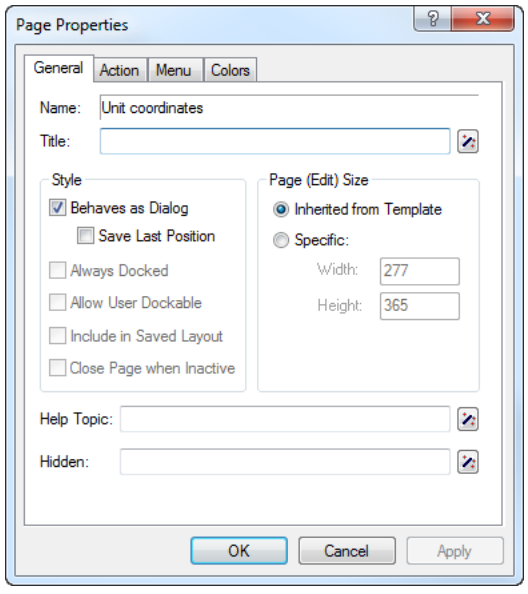

Figure 11.6: Creating a dialog page

page is opened using either a button, a menu, a navigation object or from within the model through a call to the PageOpen procedure, it will behave like a dialog box. *If, on the other hand, the dialog page is opened from within either the **Page Manager** or the **Template Manager**, the page will behave as an ordinary window.* This offers you the possibility of editing the contents and layout of the page.

Blocking execution

When a dialog page is called from within a procedure using PageOpen, the execution of the calling procedure will only continue after the dialog page has been closed by the end-user. In this way, any data supplied by the end-user in the dialog page will always be available during the remaining execution of the calling procedure.

Dialog box result

Note that dialog pages do not offer built-in support to determine whether an end-user has finished the dialog box for example by pressing the **OK** or **Cancel** button. However, such control can easily be modeled in the AIMMS language itself. Perhaps the most straightforward manner to accomplish this is by introducing

- a set DialogActions containing two elements 'OK' and 'Cancel',
- an associated global element parameter CurrentDialogAction, and
- procedures such as ButtonOK and ButtonCancel which set CurrentDialogAction equal to 'OK' or 'Cancel', respectively.

Linking to dialog box buttons

To obtain the result of a dialog page, you can simply add the execution of the procedures ButtonOK or ButtonCancel to the list of actions associated with the **OK** and **Cancel** buttons, respectively. In addition, you should link the functionality of the close icon for the dialog page to that of the **Cancel** button as illustrated in Figure 11.7.

Obtaining the result

To obtain the end-user choice in the dialog page after the return of the PageOpen procedure, you can simply check for the value of the element parameter CurrentDialogAction, as illustrated in the following code excerpt.

```
! Open the dialog page and stop processing when the user
! has pressed the 'Cancel' button.

OpenPage( "Supply input data" );
return 0 when CurrentDialogAction = 'Cancel';

! Otherwise perform further data processing based on the supplied input data.
```

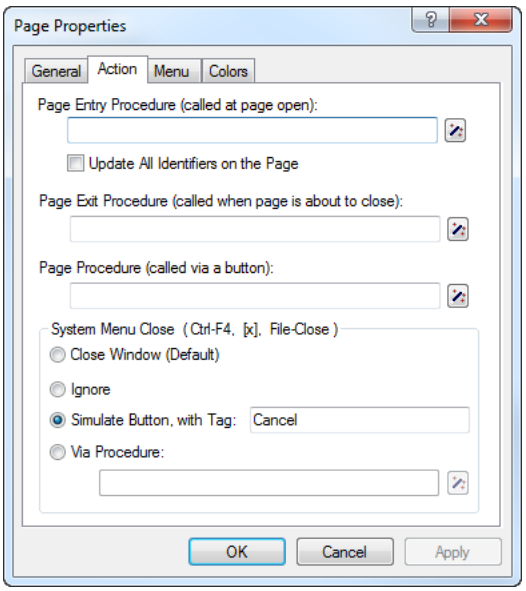

Figure 11.7: Linking dialog close to **Cancel**

Create a dialog page template

You may want to create a customized dialog page *template* (see also Section 12.2) to capture the end-user choices as described above. Based on such a dialog page template, you can quickly create as many dialog pages as necessary, all behaving in a similar fashion when opened in a procedure of your model.

11.4 Defining user colors

User colors

As already explained in the previous section, AIMMS allows you to define the color of particular objects in a graphical end-user interface from within the execution of your model. In this section you will see how you can define *user colors* which can be used within the model, and how you can use them to provide model-computed coloring of page objects.

Defining persistent user colors

To define user colors that persist across multiple sessions of a project, you should open the **User Colors** dialog box as illustrated in Figure 11.8 from the **Tools** menu. By pressing the **Add** or **Change Color** button, AIMMS will display the standard Windows color selection dialog box, which you can use to create a new user color or modify an existing user color. After you have selected a color, AIMMS will request a name for the newly defined color for further usage within the model.

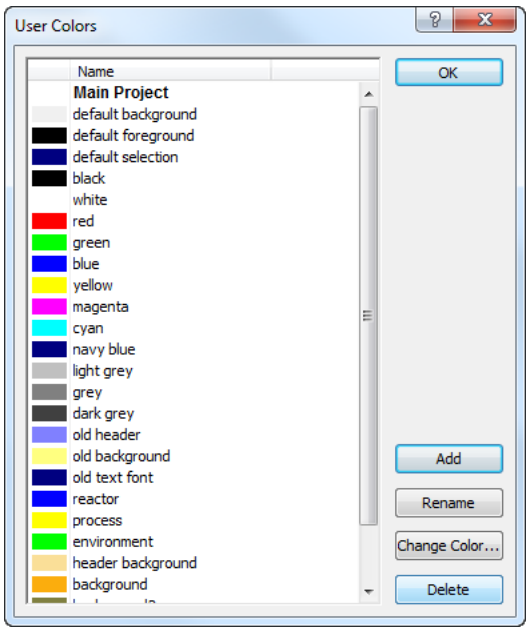

Figure 11.8: The **User Colors** dialog box

Functional color names

As with font names, you may prefer to choose functional color names rather than names describing user colors. For instance, colors named “Full tank color”, “Partially filled color” and “Empty tank color” may be a much better choice, from a maintenance point-of-view, than such simple names as “Red”, “Blue” and “Green”. In addition, choosing descriptive names may make the intention of any assignment to, or definition of, color parameters in your model much clearer.

User colors in library projects

Similar as with fonts, a library project can also contain its own set of user colors. The list of colors shown in Figure 11.8 displays the user colors defined within the main project. For each library included in the project the listbox contains a separate area displaying the user colors that are associated with that library. You can manage the list of user colors associated with a library by pressing the buttons on the right-hand side of the dialog box, while the selection in the listbox on the left-hand side is in the area associated with the library.

Use only in library pages

If you have defined user colors within a library project, you should ideally only use these user colors in pages that are also part of the library project. If you use the user colors in pages outside of the library, such pages may fail to display properly after you have removed the library project from the AIMMS project.

Color names must be unique

AIMMS requires that all user color names be unique across the main project and all library projects that are included in the main project. If you include an existing library project, which contains a user color name that is already present in the AIMMS project, AIMMS assumes that both user colors are the same and will ignore the second color definition.

Runtime user colors

Persistent user colors cannot be modified or deleted programmatically. However, you can add runtime colors (which only exist for the duration of a project session) programmatically from within your model using the function UserColorAdd. In the **User Colors** dialog box, runtime colors are shown under the header **Runtime colors**. You can modify or delete such runtime colors using the functions UserColorModify and UserColorDelete. These functions are discussed in full detail in Section 17.2.1.

The set AllColors

All persistent and non-persistent user colors are available in your model as elements of the predefined set AllColors. To work with colors in your model you can simply define scalar and/or indexed element parameters into the set AllColors. Through simple assignments or definitions to such parameters you can influence the coloring of identifiers or individual identifier values on an end-user page.

Example

Consider a set of Flows in a network with index f. If a mathematical program minimizes the errors in computed flows in respect to a set of measured flow values, then the following simple assignment to a color parameter FlowColor(f) marks all flows for which the error exceeds a certain threshold with an appropriate color.

```
FlowColor(f) := if ( FlowError(f) >= ErrorThreshold ) then
    'Red' else 'Black' endif;
```

Use in interface

With the above assignment, any graphical display of Flows can be colored individually according to the above assignment by specifying that the color of the individual numbers or flows in the **Colors** dialog box of the object be given by the value of the color parameter FlowColor(f). Figure 11.5 (on page 139) illustrates an example of an end-user page where the flows in the network flow object, as well as the individual entries in the tables and lists, are colored individually with respect to the parameter FlowColor(f) (the colors are only visible in the electronic version of this book).