AIMMS User's Guide - Page Management Tools

AIMMS 4

This documentation was typeset by AIMMS B.V. using LATEX and the Lucida font family.

Chapter 12

Page Management Tools

This chapter

When your decision support system grows larger, with possibly several people developing it, its maintainability aspects become of the utmost importance. To support you and your co-workers in this task, AIMMS offers several advanced tools. As discussed in Chapters 4 and 7, the **Model Explorer** combined with the **Identifier Selector** and **View Manager**, provide you with various useful views of the model's source code. In this chapter, the specialized AIMMS tools that will help you set up an advanced end-user interface in an easily maintainable manner will be introduced.

12.1 The Page Manager

Page navigation

In large decision support systems with many pages, navigating your end-users in a consistent manner through all the end-user screens is an important part of setting up your application. One can think of several organizational structures for all the available end-user screens in your application that would help your end-users maintain a good overview of their position. To help you set up, and modify, a clear navigational organization quickly and easily, AIMMS provides a tool called the **Page Manager**.

The Page Manager

With the **Page Manager** you can organize all the existing pages of an AIMMS application in a tree-like fashion, as illustrated in Figure 12.1. The tree in the **Page Manager** that holds all the pages of the main AIMMS project is called the main *page tree*. Relative to a particular page in this page tree, the positions of the other pages define common page relationships such as *parent* page, *child* page, *next* page or *previous* page.

Library page trees

In addition to the main page tree, each library project included in the main project can have its own tree of pages as illustrated in Figure 12.1. Section 12.1.1 discusses the facilities available in AIMMS that allow you to combine the page structures in all trees to construct a single navigational structure for the entire application.

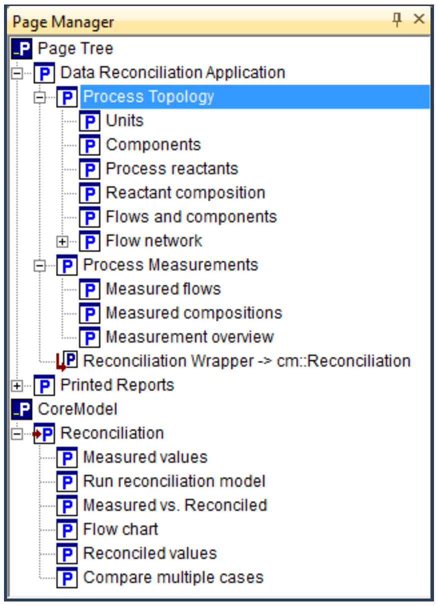

```
Page Manager
Page Tree
  Data Reconciliation Application
    Process Topology
      Units
      Components
      Process reactants
      Reactant composition
      Flows and components
      Flow network
    Process Measurements
      Measured flows
      Measured compositions
      Measurement overview
    Reconciliation Wrapper -> cm::Reconciliation
  Printed Reports
CoreModel
  Reconciliation
    Measured values
    Run reconciliation model
    Measured vs. Reconciled
    Flow chart
    Reconciled values
    Compare multiple cases
```

Figure 12.1: The **Page Manager**

Navigational controls

The page relationships defined by the page tree can be used in several navigational interface components that can be added to a page or end-user menu. These components include

- *navigation objects*,
- *navigation menus*,
- *button actions*, and
- *tabbed pages*.

These allow you to add dynamic navigation to the parent, child, next or previous pages with respect to the position of either

- the current page, or
- a fixed page in the page tree.

Section 12.1.2 explains in detail how to set up such automatic navigation aids.

Aimed at ease of maintenance

The strength of the **Page Manager** tool lies in the fact that it allows you to quickly add pages to the page tree, delete pages from it, or modify the order of navigation without the need to make modifications to hard-coded page links on the pages themselves. Thus, when a model extension requires a new section of pages, you only need to construct these pages, and store them at the appropriate position in the page tree. With the appropriate navigational interface components added to the parent page, the new page section will be available to the end-user immediately without any modification of existing pages.

12.1.1 Pages in library projects

Pages in library projects

AIMMS allows you to develop a separate page tree for every library project included in an AIMMS application. This is an important feature of library projects because

- it allows a developer to implement a fully functional end-user interface associated with a specific sub-project of the overall application completely independently of the main project, and
- pages defined inside a library project can refer to all the identifiers declared in that library, whereas pages defined in the main project (or in any other library) can only refer to the public identifiers in the interface of that library (see Section 3.2).

Duplicate names may occur

While AIMMS requires that the names of pages within a single (library) project be unique, page names need not be unique across library projects. To ensure global uniqueness of page names in the overall application, AIMMS internally prefixes the names of all the pages contained within a library with its associated library prefix (see Section 3.1). When you want to open an end-user page programmatically, for instance through the `PageOpen` function, you need to provide the full page name including its library prefix. Without a library prefix, AIMMS will only search for the page in the main page tree.

Separate page trees

The page trees associated with the main project and with all of its library projects are initially completely separate. That is, any navigational control (see Section 12.1.2) that refers to parent, child, next or previous pages can never navigate to a page that is not part of the tree in which the originating page was included.

All pages are accessible

Other than for the identifier declarations in a libray, AIMMS puts no restriction on which pages in the library can and cannot be shown from within the main application, or from within other libraries. Stated differently, the page tree of a library does not currently have a public interface.

Creating an application GUI

If an AIMMS project is composed of multiple libraries, then each of these libraries contains its own separate page tree, which may be combined to form the total end-user interface of the overall application. The navigational controls offered by AIMMS, however, can only reach the pages in the same tree in which an originating page is included.

Jumping to library pages

Without further measures, pages from different libraries would, therefore, only be accessible through a unidirectional direct link, which is very undesirable from an end-user perspective. After following such a link moving to a parent, next or previous page may give completely unexpected results, and getting back to the originating page may be virtually impossible. For both developers and end-users a situation in which all relevant pages can be reached from within a single navigational structure is much more convenient.

Page tree references

To address this problem, AIMMS offers a linkage mechanism called *page tree references*. Through a page tree reference, you can *virtually* move a subtree of pages from a library project to another location in either the main project or any other library project included in the AIMMS application. While physically the pages remain at their original location, the navigational controls in AIMMS will act as if the tree of pages has been moved to the destination location of the page tree reference. At the original location AIMMS' navigational controls will completely disregard the linked page tree.

Creating a page tree reference

You can create a page tree reference by inserting a page tree reference node at the destination location through the **Edit-New-Page Tree Reference** menu. In figure 12.1 the *Reconciliation Wrapper* node illustrates an example of a page tree reference node. It is linked to the tree of pages starting at the *Reconciliation* page in the *CoreModel* library. Note that AIMMS uses a special overlay of the page icon to visualize that a page is linked to a page tree reference node, and hence, at its original location, is not reachable anymore through AIMMS' navigational controls.

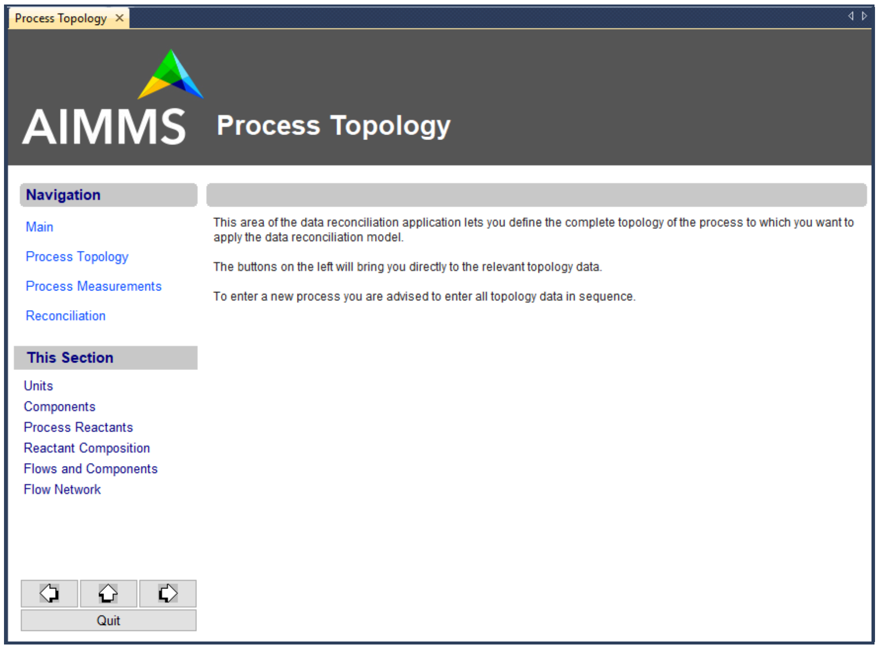
Linking a page tree reference

To create a link between a page tree reference node and a subtree of pages anywhere else in your application you have to select both the page tree reference node and the node that is at the root of the subtree that you want to link to, and select the **Edit-Page Tree Reference-Link** menu. You can unlink an existing link through the **Edit-Page Tree Reference-Unlink** menu.

12.1.2 Navigational interface components

Navigational control

The page tree can be used to directly control the navigational structure within an AIMMS-based end-user application. This can be accomplished either by special button actions or through the navigation object and menus. As an example, Figure 12.2 illustrates the *Process Topology* page contained in the page tree of Figure 12.1. In the lower left corner, the page contains three navigational buttons that are linked, from left to right, to the previous, parent and next page. Above this, the page contains a navigation object which, in this instance, automatically displays a list of buttons that corresponds exactly to the set of direct child nodes of the *Process Topology* page in the page tree.


Process Topology

AIMMS Process Topology

Navigation

Main

Process Topology

Process Measurements

Reconciliation

This Section

Units

Components

Process Reactants

Reactant Composition

Flows and Components

Flow Network

Quit

This area of the data reconciliation application lets you define the complete topology of the process to which you want to apply the data reconciliation model.

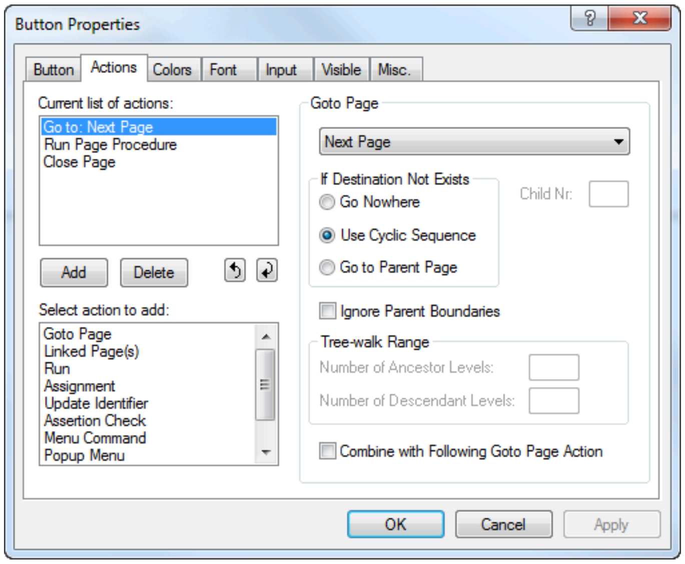
The buttons on the left will bring you directly to the relevant topology data.

To enter a new process you are advised to enter all topology data in sequence.

Figure 12.2: Page containing navigation buttons and a navigation object

Button actions

To add a page tree-based navigational control to a button, you only need to add a **Goto Page** action to the **Actions** tab in the button **Properties** dialog box, as illustrated in Figure 12.3. You can request AIMMS to open the previous,


Button Properties

Button | Actions | Colors | Font | Input | Visible | Misc.

Current list of actions:

Go to: Next Page

Run Page Procedure

Close Page

Add | Delete

Select action to add:

Goto Page

Linked Page(s)

Run

Assignment

Update Identifier

Assertion Check

Menu Command

Popup Menu

Goto Page

Next Page

If Destination Not Exists

Go Nowhere

[✓ Use Cyclic Sequence]

Go to Parent Page

Child Nr:

Ignore Parent Boundaries

Tree-walk Range

Number of Ancestor Levels:

Number of Descendant Levels:

Combine with Following Goto Page Action

OK | Cancel | Apply

Figure 12.3: Adding navigational control to a button

next, parent or (first) child page relative to the position of the current page in the page tree. If you want the current page to be closed after opening the new page, you should additionally insert a **Close Page** action as in Figure 12.3.

Cycling

When there is no longer a next or previous page to open in a particular branch of a page tree, AIMMS will cycle to the first or last page within that branch, respectively. You can further modify the result of a previous or next page action by placing special *separator* nodes into the page tree, given that AIMMS will never jump past a separator node. You will find the full details of separator nodes in the online help on the **Page Manager**.

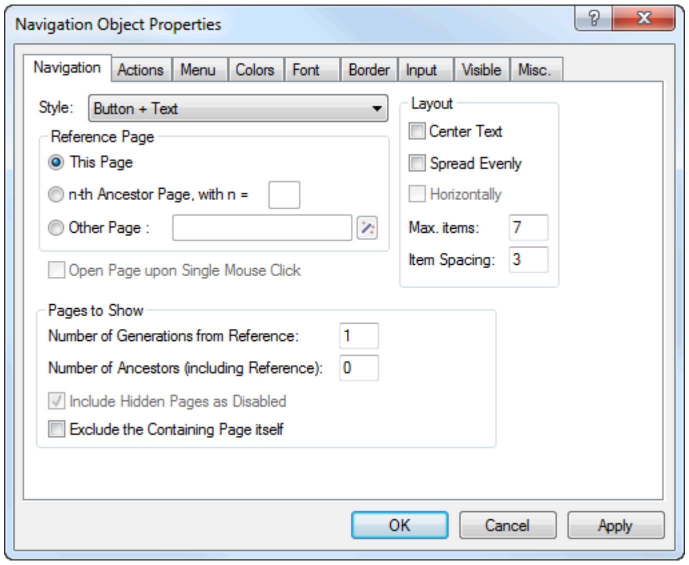
Navigation object

The second way to include a navigational control in an end-user page is by means of a custom *navigation* object. A navigation object can display a subtree of the entire page tree in several formats, such as:

- a list of buttons containing the page titles (as in Figure 12.2),
- a list of buttons accompanied by the page titles,
- a list of clickable or non-clickable page titles without buttons, or
- a tree display similar to the page tree itself.

Object properties

After adding a navigation object to a page, you must specify the subtree to be displayed through the **Properties** dialog box as displayed in Figure 12.4. What is displayed in the navigation object is completely determined by the



Style: Button + Text

Reference Page
- [● This Page]
- ○ n-th Ancestor Page, with n =
- ○ Other Page :
- ☐ Open Page upon Single Mouse Click

Layout
- ☐ Center Text
- ☐ Spread Evenly
- ☐ Horizontally
- Max. items: 7
- Item Spacing: 3

Pages to Show
- Number of Generations from Reference: 1
- Number of Ancestors (including Reference): 0
- ☑ Include Hidden Pages as Disabled
- ☐ Exclude the Containing Page itself

Figure 12.4: Navigation object **Properties** dialog box

reference page, together with the number of ancestor (parent) and child generations specified in this dialog box.

Display only

If you set a navigation object to read-only using the **Input** tab of the **Properties** dialog box, then you can use the navigation object for display- only purposes. Thus, you can use it to display the current page title as a page header, or the title of one or more parent pages in the header or footer area of the page. The "*Process Topology*" page header of the end-user page displayed in Figure 12.2 is an example of a display-only navigation object.

Navigation menus

Finally, you can add special navigation (sub)menus to your application in which the menu items and submenus represent a subtree structure of the page tree. Figure 12.5 illustrates an example of a navigation menu linked to the page tree displayed in Figure 12.1.

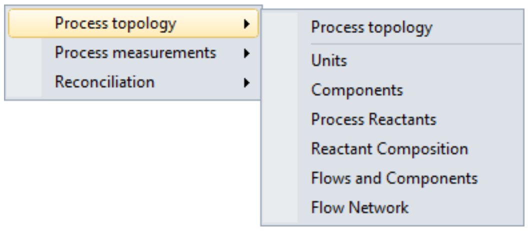


Figure 12.5: Example of a navigation menu

Adding navigation menus

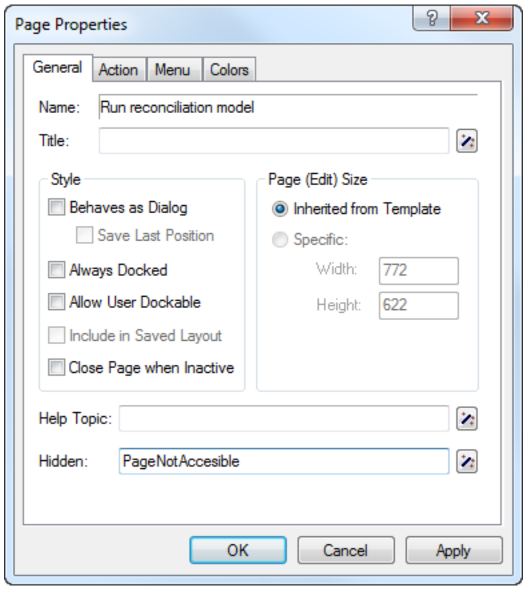
You can add a navigation menu to any menu in the **Menu Builder** tool (see Section 12.3). For each navigation menu you must specify a reference page and the scope of the subtree to be displayed in a similar fashion to that illustrated for the navigation object in Figure 12.4.

Hiding pages

Pages can be statically or dynamically hidden using the page **Properties** dialog box (see also Section 11.2), as illustrated in Figure 12.6. In the **Hidden** field, you must enter a scalar value, identifier or identifier slice. Whenever the property assumes a nonzero value the page is hidden, and automatically removed from any navigational interface component in which it would otherwise be included.

Authorizing access

For larger applications, end-users can usually be divided into groups of users with different levels of authorization within the application. Disabling pages based on the level of authorization of the user (explained in Chapter 19) then provides a perfect means of preventing users from accessing those data to which they should not have access. You can still open a hidden page via a hard-coded page link.



Name: Run reconciliation model
Title:
Style
[ ] Behaves as Dialog
[ ] Save Last Position
[ ] Always Docked
[ ] Allow User Dockable
[ ] Include in Saved Layout
[ ] Close Page when Inactive
Page (Edit) Size
[✓ Inherited from Template]
Specific:
Width: 772
Height: 622
Help Topic:
Hidden: PageNotAccesible

Figure 12.6: Hiding a page

12.2 The Template Manager

Consistent look-and-feel

Complementary to the **Page Manager** is the AIMMS **Template Manager**. Using the **Template Manager**, you can ensure that all pages are the same size and possess the same look-and-feel, simply by positioning all end-user pages in the template tree associated with a project. An example of a template tree containing both templates and end-user pages is displayed in Figure 12.7.

Hierarchical template structure

In addition to all the end-user pages, the template tree can contain a hierarchical structure of template pages. Within the template tree, template pages behave as ordinary pages, but they are not available to end-users. Through templates you can define common page objects that are shared by all the template *and* end-user pages positioned below a particular template in the template tree.

Duplicating page templates

When you want to use the same template page at two or more distinct positions in the template tree, AIMMS lets you duplicate, rather than copy, the template node containing that component. Changes made to the duplicated page template at any position in the template tree, are automatically propagated to all other occurrences. Duplicated templates can be recognized by the duplication symbol which is added to the icon of every duplicate template in the template tree.

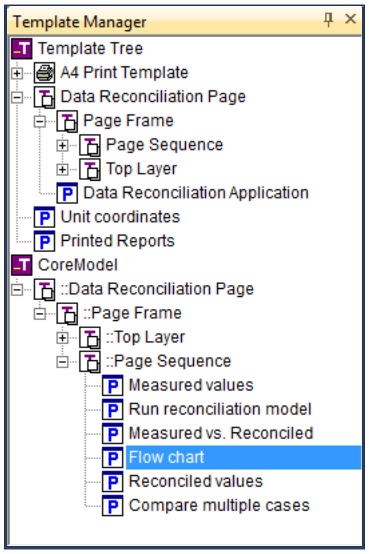


Figure 12.7: The **Template Manager**

End-user pages automatically added

Every new end-user page created in the **Page Manager**, is automatically added to the root node in the template tree. By moving the page around in the template tree, it will inherit the combined look-and-feel of all templates above it.

Common page components

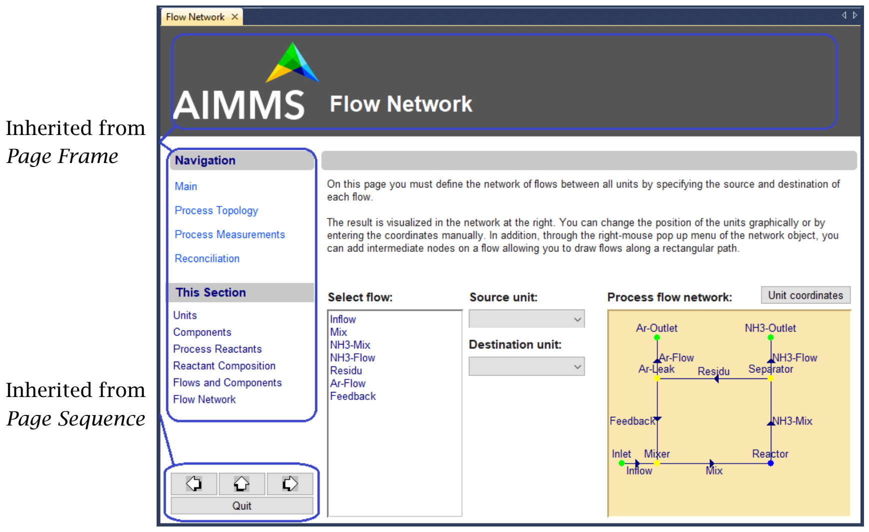
The hierarchical structure of the template tree lets you define layers of common objects on top of each other. Thus, a first template might globally define the page size and background color of all underlying pages, while a second template could define common components such as a uniformly shaped header and footer areas. As an example, Figure 12.8 illustrates a template for an end-user page from the template tree of Figure 12.7, in which the components defined in various templates are identified.

Modify look-and-feel

You can quickly modify the entire look-and-feel of your application, by moving a subtree of templates and end-user pages from one node in the template tree to another. Thus, the entire look-and-feel of page size, header and footer areas, background color and navigational area(s) of all pages in an AIMMS application can be changed by a single action.

Template objects not editable

When you open a template or end-user page in the template manager, it will be opened in edit mode by default, and inherit all the properties of, and all objects contained in, the templates above. On any template or end-user page you can only modify those objects or properties that are defined on the page itself. To modify objects defined on a template, you must go to that template and modify the objects there.



On this page you must define the network of flows between all units by specifying the source and destination of each flow.

The result is visualized in the network at the right. You can change the position of the units graphically or by entering the coordinates manually. In addition, through the right-mouse pop up menu of the network object, you can add intermediate nodes on a flow allowing you to draw flows along a rectangular path.

Figure 12.8: Example of an end-user page using templates

Combine with navigational components

You can achieve an exceptionally powerful combination by adding navigational components to a template page. If the reference page property of such a navigational component is expressed in terms of the current page, or one of its ancestor pages, then, in end-user mode, the current page will always refer to the particular end-user page which uses that template. Thus, given a well-structured page tree, you potentially only need a *single* template to add navigational control components to *all* end-user pages. This is particularly true for such common controls as **Previous** and **Next** buttons.

12.2.1 Templates in library projects

Templates in libraries

Each library project in AIMMS has a separate template tree, which is available as a separate root node in the **Template Manager**, as illustrated in Figure 12.7. Pages in a library must be positioned in the template tree of that library to obtain their look-and-feel. This allows the developer of a library project to define the look-and-feel of the pages in the library completely independent of the main project and other library projects.

Sharing templates with the main project

If you want the pages of an entire application to share a common look-and-feel across all library projects included in the application, AIMMS also allows you to duplicate template pages from the main project into a library project. Thus, any changes made to the templates in the main project are automatically inherited by the pages in the library that depend on the duplicates of these templates.

Sharing templates across multiple projects

Conversely, you can also use library projects to enable the end-user GUIs of multiple AIMMS project to share a common look-and-feel. By defining templates, fonts, and colors in a single library project, and including this library project into multiple AIMMS projects, the pages in these projects can depend on a single, shared, collection of page templates. Thus, changes in a single template library will propagate to all AIMMS projects that depend on it.

Moving pages to a library

If you move or copy pages from the main project to a library project (or between library projects), AIMMS will automatically duplicate the relevant template structure from the source project to the destination project. This ensures that the pages have the exact same look-and-feel at the destination location as they had at their source location.

Example

The automatic duplication behavior of AIMMS is illustrated by the page tree in Figure 12.1 and the template tree in Figure 12.7. These trees were created by moving the *Reconciliation* page and its child pages from the main project to the *CoreModel* library. Subsequently, AIMMS automatically duplicated the template structure in the *CoreModel* library to ensure the identical look-and-feel for the moved pages.

12.3 The Menu Builder

The Menu Builder

The last page-related design tool available in AIMMS is the **Menu Builder**. With the **Menu Builder** you can create customized menu bars, pop-up menus and toolbars that can be linked to either template pages or end-user pages in your application. The **Menu Builder** window is illustrated in Figure 12.9. In the **Menu Builder** window you can define menus and toolbars in a tree-like structure in a similar fashion to the other page-related tools. The menu tree closely resembles the natural hierarchical structure of menus, submenus and menu items.

Default menu bar and toolbar

As illustrated in Figure 12.9, the **Menu Builder** will always display four default nodes. Two nodes representing the standard end-user menu bar and toolbar. These bars are linked to all end-user pages by default. And two nodes representing the standard Data Manager menu bar and toolbar. Although non-editable, you can use these nodes to copy (or duplicate) standard end-user menus or submenus into your own customized menu bars and toolbars. The data manager items will allow you to build your own menu and toolbar for the data manager with the same functionality as the standard menu and toolbar for the data manager.

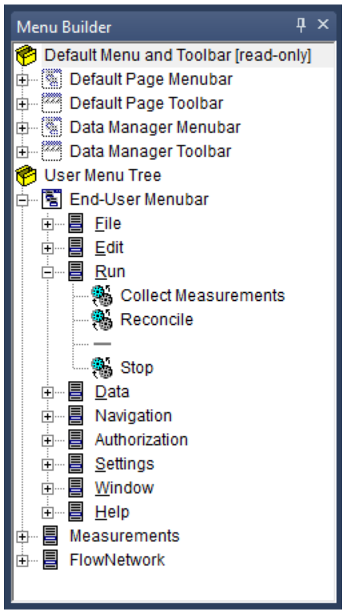


Figure 12.9: The **Menu Builder** window

Inserting new nodes

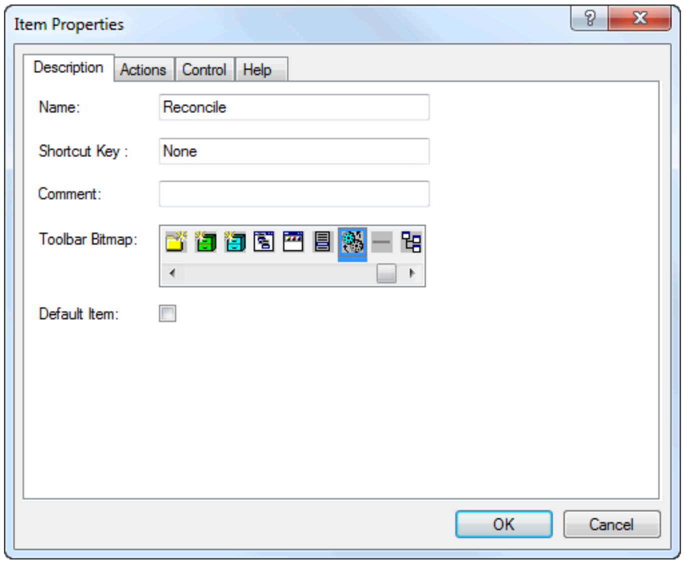
In the *User Menu Tree*, you can add nodes to represent menu bars, (sub)menus, menu items or toolbars in exactly the same manner as in other trees such as the model and page trees. Also, you can copy, duplicate or move existing nodes within the tree in the usual manner (see Section 4.3). The names given to menu and menu item nodes are the names that will be displayed in the end-user menus, unless you have provided a model-specific menu description in the menu **Properties** dialog box (e.g. to support multiple languages).

Menu item properties

For every node in the menu tree you can modify its properties through the **Properties** dialog box. In the **Properties** dialog box you can perform tasks such as linking end-user actions or model procedures to a menu item, provide shortcut keys, tooltips and help, or link a menu item to model identifiers that specify whether the item should be disabled within an end-user menu, or even be completely hidden from it. The **Properties** dialog box for a menu item is shown in Figure 12.10.

Adding menu actions

Through the **Actions** tab of the **Properties** dialog box, you can associate a list of actions with a menu item. Such actions can consist of executing menu items from system menus, navigational commands such as opening or closing pages, and also running procedures from your model, verifying assertions or updating identifiers.



Name: Reconcile

Shortcut Key : None

Comment:

Toolbar Bitmap:

Default Item: ☐

Figure 12.10: The menu item **Properties** dialog box

Hiding and disabling items

With the **Control** tab it is possible to provide control over a menu item from within your model. You can specify scalar 0-1 identifiers from within your model to determine whether a menu item or submenu should be disabled (grayed out) or completely hidden from the menu. Thus, you can prevent an end-user from performing tasks for which he is not authorized. In addition, you can couple a 0-1 identifier to a menu item in order to determine whether a menu item is checked, and which conversely toggles its value when an end-user checks or unchecks the item.

Tooltips and help

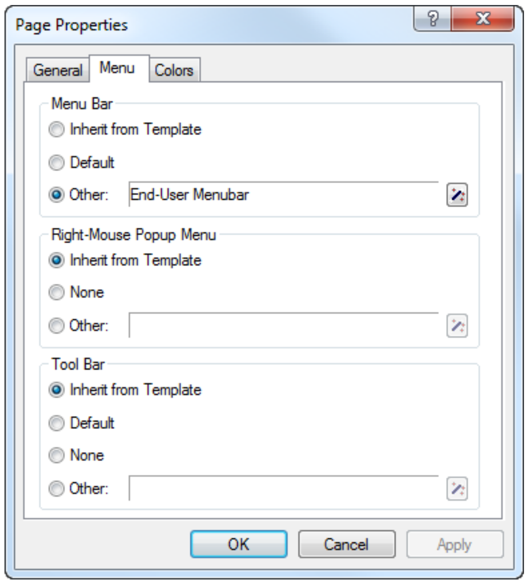
In the **Help** tab of the **Properties** dialog box, you can provide a description and help describing the functionality of a menu command. It lets you specify such things as the tooltips to be displayed for buttons on the button bar, a descriptive text for to be shown in the status bar, and a link to a help item in the project related help file.

Navigation menus

Navigation menus are a special type of menu that can be added to the menu tree. Navigation menus expand to a number of items in the current menu, or to one or more submenus, according to the structure of a particular subtree of the page tree as specified by you. Through navigation menus you can quickly and easily create menus that help an end-user navigate through your application. For example, you could create a menu item which links to the first child page, or to the parent page, of any page to which the menu is linked. The details of how to specify which pages are displayed in a navigation menu can be found in Section 12.1.2.

Linking to pages and objects

You can link a single menu bar, toolbar and pop-up menu to any end-user or template page in your project through the **Menu** tab of the page **Properties** dialog box, as illustrated in Figure 12.11 For every field in the dialog box,


Page Properties

General | Menu | Colors

Menu Bar
- Inherit from Template
- Default
- [✓ Other:] End-User Menubar

Right-Mouse Popup Menu
- [✓ Inherit from Template]
- None
- Other:

Tool Bar
- [✓ Inherit from Template]
- Default
- None
- Other:

Figure 12.11: Linking menus to pages

Aimms lets you select an existing node in the menu tree. If you do not specify a menu bar or toolbar, Aimms will automatically open the default end-user menu bar and toolbar.

Inherited menus

When you add a menu bar or toolbar to a page template, these bars are automatically inherited by all pages that use that template. In this manner, you can quickly add your own customized end-user menu to all, or groups of, pages in your application. All new end-user pages will, by default, inherit their menu bar and toolbar from their templates.

12.3.1 Menus in library projects

Menus in library projects

In addition to the main *User Menu Tree* in the **Menu Builder**, each library project in Aimms has a separate menu tree, as illustrated in Figure 12.9. In this menu tree, you can create the menus and toolbars that are specific for the pages defined in the library at hand.

Accessing private identifiers

When you are specifying the actions associated with a menu item in the menu tree of a library, you have access to all the identifiers and procedures declared in the library module of that library. For menu items in all other menu trees, you can only access the identifiers in the interface of the library.

Creating menus

When creating menus and toolbars in a library, you can duplicate menus and menu items from any other menu tree in the **Menu Builder**. Likewise, you can duplicate menus and menu items from a library project into the menu tree of the main project. This enables you to compose global menu- and toolbars that can be used in the overall application, yet containing library-specific submenus and menu items to dispatch specific tasks to the appropriate libraries.

Assigning menus to pages

When you want to assign a menu to a page or template, AIMMS allows you to choose a user menu of either the main project or of any of its library projects. You should note, however, that choosing a menu from a different library project creates an implicit dependency on that project which is not immediately apparent in the page or template tree. If you copy or move pages with a user menu or toolbar from one project to another, AIMMS will not duplicate that menu or toolbar, but still refer to their original locations as expected.